Universidad de Valladolid

FACULTAD DE CIENCIAS

TRABAJO FIN DE GRADO

Grado en Estadística

Influencia del variador en la detección de fallos en rodamientos en motores de inducción a través de métodos avanzados de clasificación supervisada

Autor: Alejandro Pérez de la Fuente
Tutor: Miguel Alejandro Fernández Temprano
2023

Resumen

Los motores eléctricos de inducción son los más utilizados en la industria. En esta, los fallos incontrolados suponen costes a veces multimillonarios. Por ello, es necesario llevar a cabo un diagnóstico temprano de los fallos sin penalizar el rendimiento del motor en funcionamiento. En este contexto, surgen múltiples técnicas de diagnóstico no intrusivas, es decir, que no alteran el funcionamiento del motor: el análisis de la corriente eléctrica que alimenta los motores, el análisis del sonido que producen en funcionamiento y las vibraciones que generan, entre otras.

En este trabajo se lleva a cabo un problema de clasificación de motores de inducción en función de su estado. Para ello, se utilizan estadísticos resumen de las ondas de corriente, sonido y vibraciones de los motores en funcionamiento. También se estudia la influencia de los factores método de alimentación empleado y carga a la que el motor esta sometido.

La metodología empleada para el análisis de los datos se basa en la utilización de técnicas de Boosting. De este modo, se alcanzan unas tasas de acierto en la clasificación considerablemente buenas así como conclusiones de interés industrial acerca de la influencia de los distintos factores en la clasificación.

Abstract

Induction electric motors are the most widely used in industry. Uncontrolled failures in this area can result in multi-million dollar costs. Therefore, it is necessary to carry out early diagnosis of faults without penalizing the motor's performance during operation. In this context, multiple non-intrusive diagnostic thchniques arise, meaning they do not alterate the motor's operation. These techniques include analyzing the electrical current that powers the motors, analyzing the sound they produce during operation and studying the vibrations they generate, among others.

This work addresses the classification problem of induction motors based on their condition. To achieve this, summary statistics of the current, sound, and vibration waveforms of the motors during operation are used. The influence of factors such as the feeding method employed and the load to which the motor is subjected is also studied.

The methodology used for data analysis is based on the application of Boosting techniques. In this way, considerably good classification accuracy rates are achieved, as well as conclusions of industrial interest about the influence of different factors on the classification.

Agradecimientos

A mis padres y a mi hermana Ana Isabel, Juan José y Lydia por su apoyo y ánimo incondicional.

A mi tutor Miguel Alejandro por su tiempo, ayuda y guía durante el desarrollo del trabajo.

A todos los profesores del grado por transmitirme sus conocimientos.

A mis amigos por creer en mí.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Descripción del problema

Los motores de inducción también conocidos como motores asíncronos, son una clase de motores que convierten la energía eléctrica en mecánica. Este tipo de motores es utilizado ampliamente por la industria debido a su simplicidad, fiabilidad y bajo coste. Tanto es así que la Comisión Europea estima que hay unos 8 billones de motores eléctricos consumiendo alrededor del 50 % de electricidad que se produce dentro de la Unión Europea [1].

Partiendo de las partes esenciales de un motor de inducción que se ven en la figura 1.1, su funcionamiento se basa en hacer girar el rotor por medio de inducción electromagnética. Para ello, en primer lugar, se genera un campo magnético en el estator gracias a la corriente eléctrica con la que se alimenta el motor. Este campo magnético induce una corriente eléctrica en el rotor creando así este su propio campo magnético. Por último, estos dos campos magnéticos interaccionan entre si haciendo que el rotor gire.

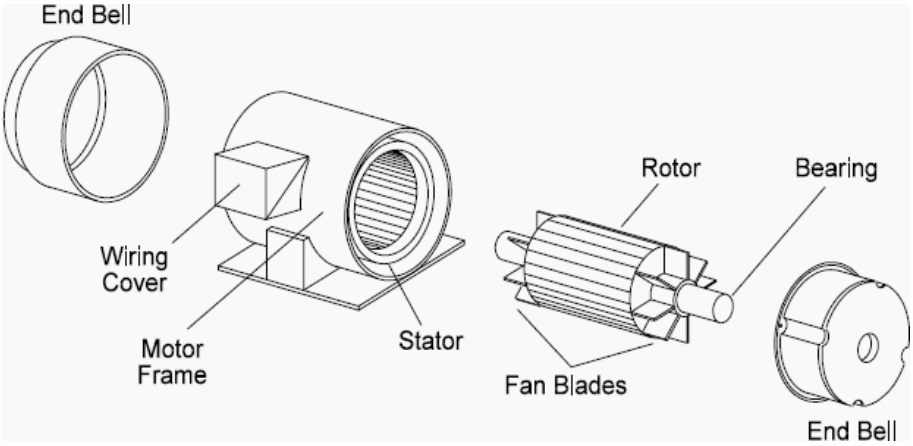

Figura 1.1: Esquema de un motor de inducción [2]

Puesto que estos motores se utilizan de forma masiva en la industria donde un fallo

puede suponer perdidas de grandes cuantías de dinero, en la actualidad se buscan métodos de detección de fallos basados en distintos tipos de medidas. Algunos de estos, se encuentran documentados en artículos científicos como: métodos basados en la monitorización de la corriente del estator del motor en funcionamiento [3]-[6], métodos basados en el sonido y vibraciones que produce el motor en funcionamiento [7]-[11] o incluso métodos basados en imágenes térmicas del motor en funcionamiento [12].

1.2. Objetivos

En este trabajo, se realiza un estudio cuya finalidad es el diagnóstico de fallos en rodamientos de motores de inducción. Para ello se dispone de una base de datos suministrada por profesores del departamento de Ingeniería Eléctrica de la Universidad de Valladolid expertos en este tópico como demuestran, entre otras, sus publicaciones [13]-[15]. Esta base de datos recoge observaciones de diferentes fuentes de información que, como se ha mencionado, se utilizan en la literatura para diagnosticar problemas en los motores de inducción. Concretamente se dispone de datos sobre el sonido, la corriente y las vibraciones del motor en los tres ejes de coordenadas, así como de factores como la alimentación empleada y el estado de carga del motor.

El objetivo global es encontrar los modelos que mejor capacidad predictiva tengan para predecir el estado de los motores de inducción en función de las distintas variables y factores considerados. Con este objetivo en mente, se abordan los siguientes objetivos específicos:

- Establecer una metodología que permita la obtención de modelos sin sobreajuste.

  Hay que notar que, como se especificará más adelante, el tamaño de la base de datos no es grande en relación al número de variables disponibles. En consecuencia, el riesgo de sobreajuste es alto si no se controla el número de variables predictoras que se incluyen en los procedimientos.

- Estudiar la **capacidad de predicción de cada tipo de variables**: corriente, sonido y vibraciones.

  Este objetivo es de gran interés desde el punto de vista industrial, puesto que los datos de corriente se obtienen de una forma mucho más automática y menos invasiva que los de sonido y vibraciones, cuya correcta obtención puede depender fuertemente de la pericia de los operarios encargados de su recogida.

- Estudiar la **influencia del método de alimentación** del motor en la detección del fallo.

Este es uno de los objetivos fundamentales del proyecto. En la base de datos se dispone de observaciones recogidas de un motor o bien alimentado directamente desde la red, o bien alimentado mediante distintos variadores (en este caso AB, ABB y WEG) que regulan la velocidad de giro del motor. En este sentido será interesante observar si los procedimientos diagnósticos son homogéneos para todas las formas de alimentación o si, por el contrario, hay dependencia entre el procedimiento de diagnóstico y el tipo de alimentación con lo que deberían establecerse reglas de diagnóstico específicas para cada método de alimentación.

- Estudiar la **influencia del tipo de carga** del motor en la detección del fallo.

  También esta cuestión tiene interés desde el punto de vista industrial. Los motores habitualmente funcionan a carga alta. Si se determinara que es más fácil detectar los fallos cuando el motor funciona a carga baja sería conveniente establecer ciclos de bajada de la carga del motor para aumentar la eficiencia en la detección de los fallos, lo que podría conllevar un incremento en los costes de producción.

1.3. Estructura de la memoria

Esta memoria se desarrolla en los siguientes capítulos.

- **Capítulo 1: Introducción.** En este capítulo se introduce el problema a tratar, su interés y los principales objetivos que se abordan.

- **Capítulo 2: Marco teórico.** En este capítulo se explican los métodos y modelos estadísticos que se utilizan para abordar el problema.

- **Capítulo 3: Datos.** En este capitulo se describe el conjunto de datos así como el preprocesado llevado a cabo en el mismo. Entre ellos, la detección de valores atípicos.

- **Capítulo 4: Metodología.** En este capítulo se explica la metodología desarrollada para el análisis del conjunto de datos.

- **Capítulo 5: Resultados.** En este capítulo se muestran los resultados de aplicar la metodología a subconjuntos de datos y variables así como su interpretación.

- **Capítulo 6: Conclusiones y futuras líneas de investigación.** En este capítulo se enumeran las conclusiones obtenidas del trabajo así como las posibles ampliaciones del mismo.

1.4. Asignaturas del grado relacionadas

Las siguientes asignaturas de los Grados en Estadística e Informática han sido especialmente relevantes en el desarrollo de este trabajo.

- Análisis de datos (Cod. 47093), Análisis multivariante (Cod. 47097) y Análisis de datos categóricos (Cod. 47102) donde se estudian las bases del problema de clasificación así como técnicas de *Boosting*.

- Regresión y ANOVA (Cod. 47092) donde se estudian técnicas de análisis de la varianza.

- Técnicas de aprendizaje automático (Cod. 46932) y Minería de datos (Cod. 46970) donde se estudian métodos de clasificación y *Boosting* con Python.

- Fundamentos de programación (Cod. 46904), Paradigmas de programación (Cod. 46909), Estructuras de datos y algoritmos (Cod. 46913) y Programación orientada a objetos (Cod. 46914) donde se adquieren competencias básicas para la programación.

Capítulo 2

Marco teórico

En este capítulo se introducen los principales métodos y modelos que se utilizan para el análisis del conjunto de datos.

2.1. *Gradient Boosting*

Gradient Boosting es el modelo que se utiliza para llevar a cabo la clasificación de los datos. Es una técnica de aprendizaje automático basado en realizar un ensemble de clasificadores débiles que habitualmente son árboles de decisión sencillos. Construye el ensemble de clasificadores añadiendo uno a uno en el caso de clasificación binaria de forma aditiva tal y como se aprecia en la figura 2.1.

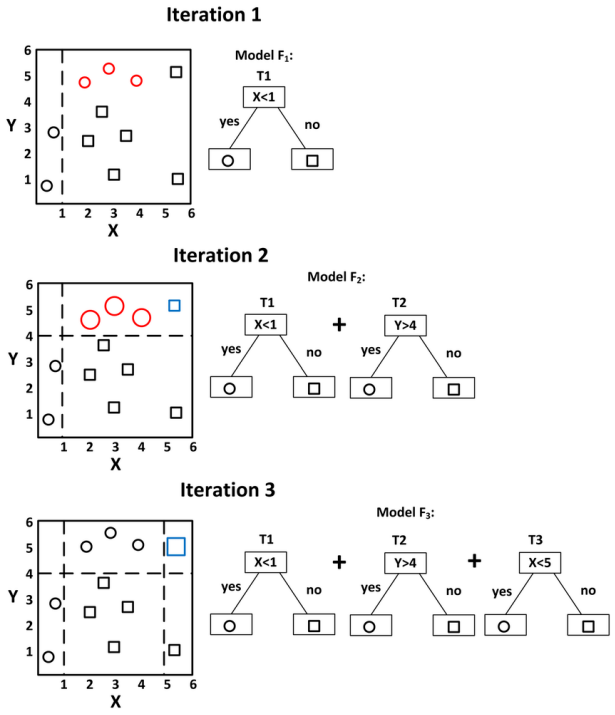

Figura 2.1: Ensemble *Gradient Boosting* en clasificación binaria [16]

Estos árboles se van escogiendo de tal modo que disminuya el error, para ello, *Gradient Boosting* utiliza una ecuación de actualización del modelo:

$$\hat{y}_m = \hat{y}_{m-1} + \eta(-\Delta L(y, \hat{y}_{m-1}))$$

Esta ecuación lleva a cabo una actualización de las predicciones en la dirección negativa del gradiente de la función de pérdida (L) proporcional a la tasa de aprendizaje (η) del modelo. De este modo, cada árbol que introduce reduce el error total del ensemble acercándose a un mejor modelo en cada iteración.

Este modelo, tiene una serie de hiperparámetros que han de ser optimizados para no caer en un modelo sobreajustado entre los cuales se destacan los siguientes:

Hiperparámetro
Mínimo número de observaciones para dividir un nodo
Mínimo número de observaciones en nodo hoja
Máximo de profundidad del árbol
Máximo número de nodos hoja
Tasa de aprendizaje (η)
Número de árboles del ensemble
Porcentaje de muestra para entrenar cada árbol

Tabla 2.1: Hiperparámetros de *Gradient Boosting*

De los hiperparámetros mostrados en la tabla 2.1, los 4 primeros son relativos a la forma de cada uno de los árboles que compondrá el ensemble mientras que los 3 últimos son relativos al ensemble. Todos ellos son importantes ya que nos dicen la forma del ensemble: si son árboles sencillos o complejos, si son muchos o pocos... En específico, algunos hiperparámetros son de especial interes por como influyen en el modelo [17], [18]:

- **Mínimo número de observaciones para dividir un nodo:** Este hiperparámetro del modelo nos ayuda a controlar el sobreajuste ya que con valores pequeños conseguimos que el modelo se adapte mejor a la muestra (pudiendo sobreajustar el modelo) mientras que con valores grades reducimos la varianza de predicción en cada nodo hoja.

- **Máximo de profundidad del árbol:** Este parámetro controla la complejidad de los árboles que forman el ensemble. Escogiendo árboles con mayor profundidad, estaremos ante árboles más complejos los cuales se ajustarán mejor a nuestros datos (pudiendo sobreajustar el modelo) mientras que con árboles de menor profundidad, tendremos árboles más sencillos que faciliten la generalidad del modelo.

- **Tasa de aprendizaje (η):** Este parámetro controla la velocidad de entrenamiento del modelo en su ecuación de actualización. Con valores pequeños (<0,1) se llega a modelos que generalizan mejor respecto a no variando la regla de actualización (= 1) a costa de un mayor costo computacional.

- **Número de árboles del ensemble:** Este parámetro controla el número de árboles del ensemble. A un mayor número de árboles mayor sobreajuste sobre la muestra se genera.

- **Porcentaje de muestra para entrenar cada árbol:** Este parámetro controla el número de observaciones con las que se entrena cada árbol. No utilizando el 100 % de los datos para entrenar cada árbol introduce aleatoriedad en la creación de los mismos que resulta en una mejor generalización del ensemble.

La interpretabilidad del modelo resultante es limitada ya que es un cúmulo de muchos árboles de decisión evaluados de forma aditiva para llegar al clasificador. No obstante, dado que son árboles de decisión se puede calcular la importancia de cada una de las variables del modelo como la fracción de observaciones que atraviesan un nodo que se divide por dicha variable y promediar dichas fracciones de todos los árboles. Obteniéndose valores entre 0 y 1 que indican la importancia de cada una de las variables en el ensemble.

2.2. Validación de resultados

En esta sección se enuncian los métodos de validación que se utilizan para comprobar la calidad de los modelos obtenidos. Se utiliza como medida de calidad de los modelos la tasa de acierto (*accuracy*) que mide la proporción de coincidencias entre la respuesta dada por el modelo y el valor real:

$$accuracy = \frac{\#(\hat{y} = y)}{\#y}$$

Donde $\#(\hat{y} = y)$ es el número de observaciones predichas correctamente y $\#y$ es el número total de observaciones del conjunto de datos.

2.2.1. *Hold-out*

Este método de validación [19] divide el conjunto de datos en dos subconjuntos aleatorios: uno de entrenamiento (*train*) y otro de validación o prueba (*test*) tal y como se aprecia en la figura 2.2. El primero de estos, el de entrenamiento, se utiliza para entrenar el modelo, ya sea sus coeficientes y/o sus hiperparámetros. Mientras que el segundo de estos, el de prueba, se utiliza para comprobar la calidad del modelo.

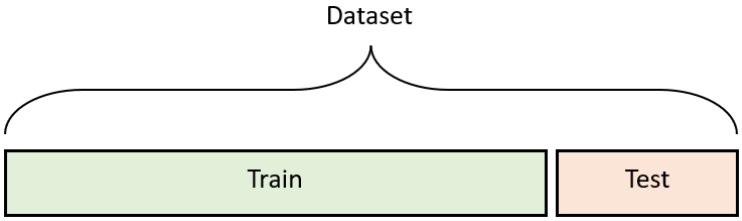

Figura 2.2: Partición del conjunto de datos por *Hold-out*

La ventaja de este método de validación es que el modelo se entrena con unos datos diferentes a los que se van a utilizar para medir su calidad. Por tanto, no estamos sesgando la medida de calidad del mismo por usar los mismos datos para entrenar que para comprobar su calidad.

No obstante, tiene sus inconvenientes, entre ellos cabe destacar que tanto el entrenamiento del modelo como la estimación de la calidad del mismo se condiciona fuertemente a la partición concreta de los datos llevada a cabo.

Para solventar parte de este problema se lleva a cabo una estratificación de las particiones, lo que quiere decir, que se mantienen las mismas proporciones de datos que hay de cada clase respuesta en ambos subconjuntos que en el conjunto de datos completo. No obstante, seguimos teniendo el problema de la partición concreta escogida.

2.2.2. Validación cruzada

Este método de validación [19] surge de intentar evitar el problema enunciado anteriormente en la sección 2.2.1 acerca del sesgo de la medida de calidad debido a la partición concreta de los datos escogida. En este caso, el conjunto de datos total se divide en un número concreto de subconjuntos aleatorios K tal y como se aprecia en la figura 2.3 de tal modo que uno de ellos sea el conjunto de prueba (*test*) y el resto sean los de entrenamiento (*train*).

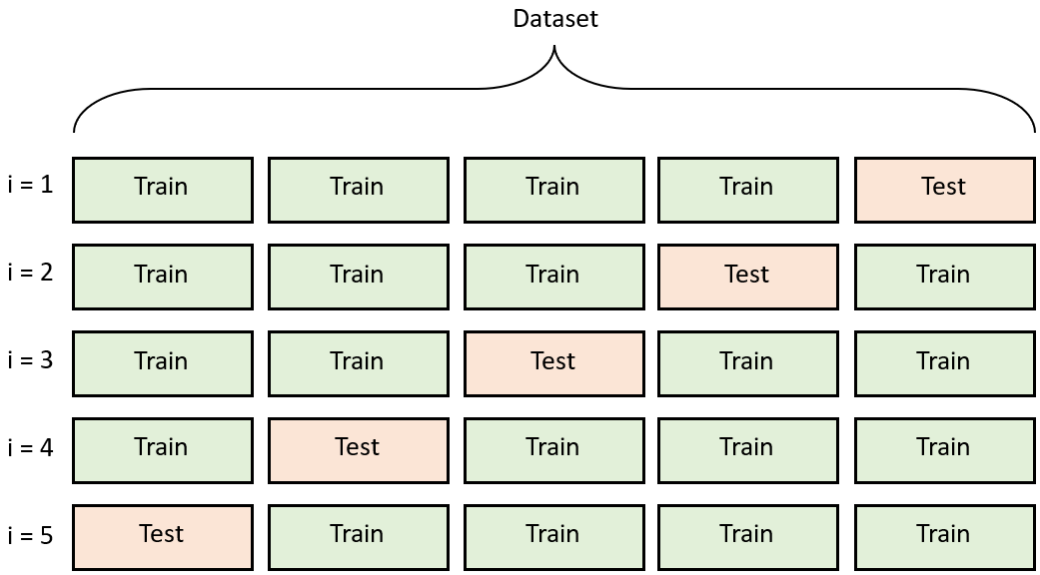

Figura 2.3: Partición del conjunto de datos por validación cruzada (K = 5)

Para obtener la medida de la calidad del modelo, se realizan K iteraciones teniendo en cada una de ellas uno de los K subconjuntos como prueba y el resto como entrenamiento. Se obtiene una medida global de la calidad del modelo promediando las K obtenidas en cada una de las iteraciones.

$$accuracy = \frac{\sum_{i=1}^{K} accuracy_i}{K}$$

La principal ventaja de este método de validación es que se reduce la varianza del estimador de la calidad del modelo gracias a las K iteraciones llevadas a cabo solventando el problema de *Hold-out*. Además, permite utilizar un mayor número de datos para entrenar el modelo por lo que los coeficientes estimados del mismo y/o los hiperparámetros escogidos serán mejores.

No obstante, hemos generado un nuevo problema, estamos promediando estimadores que no son independientes puesto que hay solapamiento en los conjuntos de entrenamiento entre las distintas iteraciones.

2.3. *Bootstrap*

Bootstrap es una técnica estadística para realizar inferencias en una población llevando a cabo un remuestreo de la muestra de la población disponible. Aunque existen muchos tipos de *Bootstrap* adaptados a diferentes contextos, a grandes rasgos, el proceso puede hacerse llevando a cabo los siguientes pasos [20]:

1. En primer lugar, se escoge el número de veces que se llevará a cabo el remuestreo

(N). El número de repeticiones ha de ser elevado (habitualmente más de 20 o 30).

2. En segundo lugar, en cada repetición se escoge una muestra aleatoria con reemplazamiento de un tamaño concreto habitualmente representado como una proporción de la muestra total. En aprendizaje automático o si se tiene un número reducido de observaciones se suele escoger como tamaño el 100 % de la muestra total.

3. En tercer lugar, para cada muestra escogida, se ajusta el modelo y se obtienen los estadísticos estimados $\hat{\theta}_i$ (ej. tasa de acierto).

4. Por último, se estiman los estadísticos promediando los obtenidos en cada repetición así como la varianza de los mismos como la cuasivarianza muestral obtenida de los estadísticos estimados en cada repetición. Esto permite llevar a cabo intervalos de confianza de los mismos y/o contrastes de hipótesis.

$$\hat{\theta} = \frac{\sum_{i=1}^{N} \hat{\theta}_i}{N}$$
$$\hat{S}^2 = \frac{\sum_{i=1}^{N} (\hat{\theta}_i - \hat{\theta})^2}{N-1}$$

Puesto que los estadísticos que estamos estimando son medias provenientes de un gran número repeticiones independientes, el Teorema Central del Límite establece que la distribución del estimador de la media seguirá asintóticamente una distribución normal.

2.4. Análisis de la varianza (ANOVA)

Es una técnica desarrollada por Ronald Fisher [21] que sirve para estudiar el efecto que tiene uno o más factores sobre la media de una variable continua en diferentes niveles de los factores. La forma en que se lleva a cabo el estudio es un contraste de hipótesis. En específico, las hipótesis que se contrastan sobre las medias en cada grupo en un ANOVA de un factor son:

$$\begin{cases} H_0 : \forall\ i, j\ \ \mu_i = \mu_j \\ H_1 : \exists\ i, j\ \ \mu_i \neq \mu_j \end{cases}$$

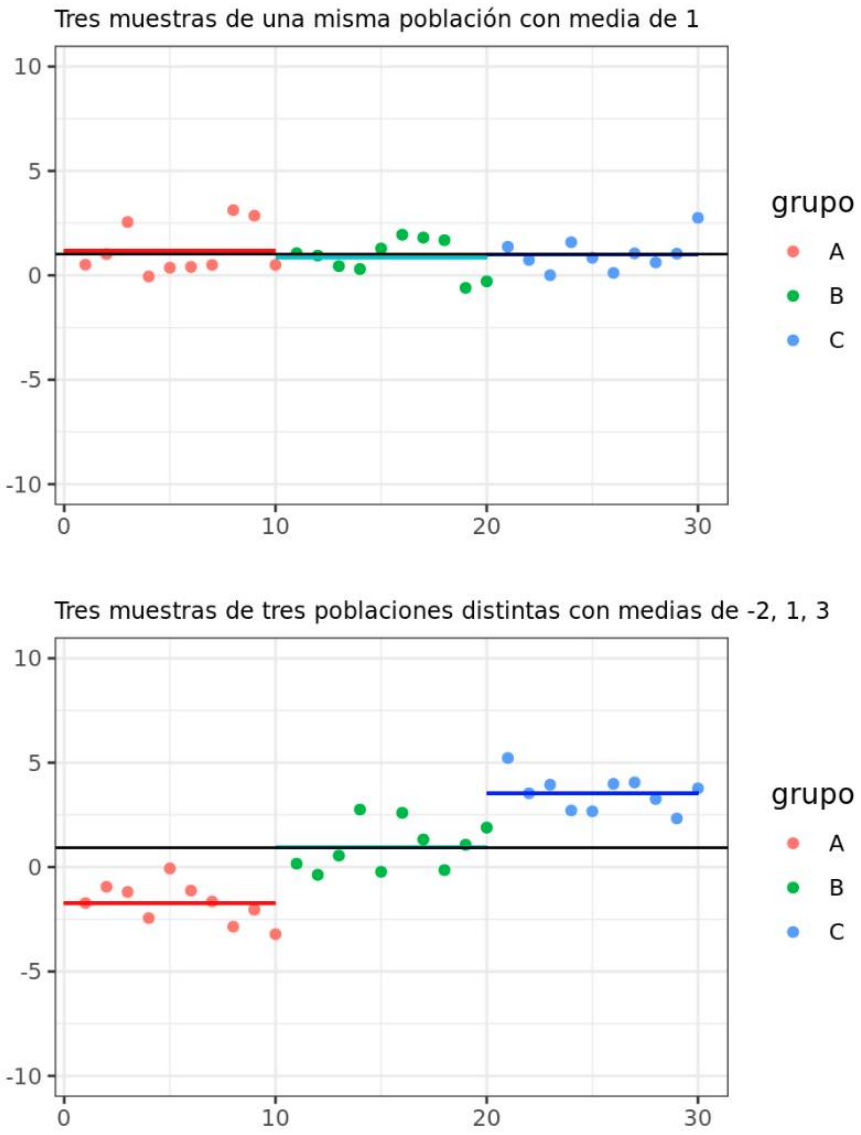

Figura 2.4: ANOVA de un factor con tres niveles [21]

El estadístico de contraste para el test ANOVA de un factor es el F_{ratio} el cual se basa en comparar la varianza de las medias de cada grupo con la varianza dentro de cada grupo:

$$F_{ratio} = \frac{n\frac{\sum_{j=1}^{k}(\bar{y_j}-\bar{y})^2}{k-1}}{\frac{\sum_{j=1}^{k}\sum_{i=1}^{n}(y_{ij}-\bar{y_j})^2}{N-k}} \sim F_{k-1,\ N-k}$$

donde N es el número total de observaciones, k es el número de niveles del factor, n es el número de observaciones de cada grupo (suponiendo grupos equilibrados), $\bar{y}$ es la media total de los datos y $\bar{y_j}$ es la media de los datos correspondientes al nivel j del factor.

Bajo la hipótesis nula de igualdad de medias en cada nivel del factor, el estadístico sigue una distribución $F_{k-1,\ N-k}$ lo que nos permite contrastar la verosimilitud de la hi-

pótesis nula.

Para poder llevar a cabo el contraste de hipótesis se asume una serie de hipótesis:

- **Normalidad.** La variable respuesta numérica para cada nivel del factor ha de seguir una distribución normal. Aunque el test es robusto ante la no normalidad.
- **Homocedasticidad.** La variable respuesta numérica para cada nivel del factor ha de tener la misma varianza. Aunque el test es robusto a la falta de homocedasticidad si los tamaños de muestra para cada nivel del factor son balanceados.
- **Independencia.** Las observaciones han de ser independientes.

También esta definido el análisis de la varianza (ANOVA) con dos factores. Este se realiza de manera análoga al de un factor salvo por las hipótesis que se contrastan. En este caso se combinan 3 contrastes de hipótesis [21]:

$$\begin{cases} H_0 : \forall\ i_1, i_2, j\ \mu_{i_1 j} = \mu_{i_2 j} \\ H_1 : \exists\ i_1, i_2, j\ \mu_{i_1 j} \neq \mu_{i_2 j} \end{cases}$$

$$\begin{cases} H_0 : \forall\ i, j_1, j_2\ \mu_{i j_1} = \mu_{i j_2} \\ H_1 : \exists\ i, j_1, j_2\ \mu_{i j_1} \neq \mu_{i j_2} \end{cases}$$

$$\begin{cases} H_0 : \forall\ i, j\ (\forall\ k\ \mu_{ik} * \forall\ t\ \mu_{tj})_{ij} = 0 \\ H_1 : \exists\ i, j\ (\forall\ k\ \mu_{ik} * \forall\ t\ \mu_{tj})_{ij} \neq 0 \end{cases}$$

donde μ_{ij} es la media de la variable respuesta en el nivel i del primer factor y j del segundo.

La primera de las hipótesis significa que las medias agrupadas por el nivel j del segundo factor son iguales para todos los niveles del primero. La segunda de las hipótesis significa que las medias agrupadas por el nivel i del primer factor son iguales para todos los niveles del segundo. Por último, la última hipótesis significa la no interacción entre ambos factores.

2.4.1. Análisis *post-hoc* (test de Tukey)

Una vez se encuentran diferencias significativas como resultado de un ANOVA, se necesita realizar las comparaciones 2 a 2 de todos los niveles del factor para ver cuales son los que son significativamente diferentes. En este contexto, el test desarrollado por John Tukey (test de Tukey) es uno de los más populares [22].

Para poder aplicarse, se han de cumplir las mismas hipótesis del análisis de la varianza (ANOVA): normalidad de la variable respuesta para cada nivel del factor, homocedasti-

cidad de la variable respuesta para cada nivel del factor e independencia de las observaciones.

En el caso de un factor, el test contrasta las siguientes hipótesis [23], [24] de igualdad de medias para cada par de niveles del factor:

$$\begin{cases} H_0 : \forall\ i, j\ \mu_i = \mu_j \\ H_1 : \exists\ i, j\ \mu_i \neq \mu_j \end{cases}$$

Para ello, en el caso de grupos balanceados, el test utiliza el estadístico rango estudentizado:

$$Q = \frac{\bar{y}_{max} - \bar{y}_{min}}{\sqrt{\frac{MSE}{n}}}$$

donde $\bar{y}_{max}$ e $\bar{y}_{min}$ son la mayor y menor media encontrada.

Este estadístico bajo las hipótesis nulas sigue una distribución $F_{k,n-k}$ siendo n el número de datos y k el número niveles del factor. Rechazándose la hipótesis nula de igualdad de medias si:

$$MSE < 2Q^{-2}(\alpha, k, n-k)(\bar{y}_i - \bar{y}_j)^2 \frac{n_i n_j}{n_i + n_j}$$

donde $Q(\alpha, k, n-k)$ el el valor crítico a nivel α del estadístico de rango estudentizado Q, $\bar{y}_i$ y $\bar{y}_j$ son las medias para los dos grupos que se están comparando y, n_i y n_j los tamaños de cada grupo.

Capítulo 3

Datos

En este capítulo se describe el conjunto de datos a utilizar así como los todos los preprocesados del mismo que permiten su análisis.

3.1. Descripción de los datos

El conjunto de datos que se utiliza ha sido facilitado por el departamento de Ingeniería Eléctrica de la Universidad de Valladolid y contiene información sobre fallos en los rodamientos de motores industriales de inducción.

Los motores se clasifican en dos estados {1, 2} dependiendo de si se encuentran en buenas condiciones o dañados. Además, también se tiene información sobre 3 características principales de los motores: la corriente eléctrica que los alimenta, el sonido que producen en funcionamiento y las vibraciones que generan en cada eje (x, y, z).

El conjunto de datos se compone de 20 observaciones para cada una de las combinaciones de los factores considerados. De estas 20 observaciones, 10 corresponden a situaciones en las que el motor que se encuentra en buenas condiciones y mientras que en las otras 10 el motor está dañado. Los factores considerados son los siguientes. En primer lugar, el tipo de alimentación que se ha utilizado: {AB, ABB, Red, WEG} donde AB, ABB y WEG son distintos tipos de variadores de frecuencia y Red la propia red eléctrica. En segundo lugar, la frecuencia de trabajo del mismo {35, 50, 65} Hz a excepción de los alimentados por Red de los que solo se tienen datos a 50 Hz. Por último, el tipo de carga al que esta siendo sometido el motor {CA, CB} siendo estas carga alta y carga baja respectivamente. Obteniéndose un total de 400 observaciones.

3.1.1. Corriente eléctrica

Para cada observación, se miden los datos de corriente en tres fases (f1, f2, f3). Estos datos son estadísticos resumen sobre la onda de corriente de alimentación del motor. Específicamente en la tabla 3.1, apoyándonos en el trabajo [25] donde ya se realizaron las transformaciones de las variables originales, para cada una de las fases se miden los estadísticos resumen de onda:

Nombre	**Fórmula**	**Descripción**
c_1, m_1	$\frac{1}{n}\sum_{i=1}^{n} x_i$	Cumulante de primer orden (media)
m_2	$\frac{1}{n}\sum_{i=1}^{n} x_i^2$	Momento de segundo orden (varianza)
m_3	$\frac{1}{n}\sum_{i=1}^{n} x_i^3$	Momento de tercer orden
m_4	$\frac{1}{n}\sum_{i=1}^{n} x_i^4$	Momento de cuarto orden
c_2	$m_2 - m_1^2$	Cumulante de segundo orden
c_3	$m_3 - 3m_2m_1 + 2m_1^3$	Cumulante de tercer orden
c_4	$m_4 - 4m_3m_1 - 3m_2^2 + 12m_2m_1^2 - 6m_1^4$	Cumulante de cuarto orden
skew	$\frac{c_3}{\sqrt{c_2}^3}$	Coeficiente de asimetría
kurt	$\frac{c_4+3c_2^2}{\sqrt{c_2}^4}$	Coeficiente de apuntamiento
x_p	$max\|x_i\|$	Máximo valor absoluto
x_r	$(\frac{1}{n}\sum_{i=1}^{n}\sqrt{\|x_i\|})^2$	Valor cuadrático medio
cf	$\frac{x_p}{\sqrt{m_2}}$	Factor de cresta
sf	$\frac{\sqrt{m_2}}{\frac{1}{n}\sum_{i=1}^{n}\|x_i\|}$	Factor de forma

Tabla 3.1: Lista de estadísticos resumen

De este modo, se obtiene un total de 39 variables explicativas relacionadas con la corriente eléctrica que alimenta el motor, 13 para cada una de las fases.

3.1.2. Sonido

En cuanto a los datos de sonido, se miden las mismas variables mencionadas en la tabla 3.1 sobre las ondas de sonido grabadas con micrófono que producen los motores en funcionamiento.

En este caso, al no existir distintas fases, son únicamente 13 variables explicativas.

3.1.3. Vibraciones

En cuanto a los datos de vibraciones, se recogen los datos sobre la onda de vibración del motor en funcionamiento con un acelerómetro. Son las mismas variables mencionadas

en la tabla 3.1 en cada uno de los ejes (x, y, z).

En este caso, se obtiene un total de 39 variables explicativas, 13 en cada uno de los ejes.

3.1.4. Conjunto de datos total

Juntando todas estas variables explicativas, se obtiene un total de 91 variables numéricas: 39 correspondientes a los datos eléctricos, 13 a los de sonido y 39 a los de vibraciones. Además, se tienen las 3 variables categóricas mencionadas al principio de la sección 3.1 correspondientes al tipo de alimentación del motor, la frecuencia de trabajo y la carga del motor. Esto sumado a la variable respuesta estado del motor (*resp*) y una variable mas que identifica cada observación (*id*) para facilitar el procesado de los datos, nos da un conjunto de datos de 400 observaciones y 96 variables para cada una de estas.

3.2. Procesado de los datos

En esta sección se explican todas las transformaciones realizadas a los datos antes de su análisis.

Para la lectura de los datos y su procesado se utiliza el lenguaje de programación Python con las librerías Pandas, Sklearn y Numpy entre otras. Todo el código desarrollado para la lectura y procesado de los datos se encuentra en el anexo B.

3.2.1. Codificación de las variables categóricas

Para el análisis de los datos con el software que utilizamos, es necesario que estos tengan un correcto formato. Es por ello que las variables categóricas se codifican mediante una codificación *One-Hot* transformándose en un conjunto de variables ficticias o indicador $\{1, 0\}$ que indican si la observación pertenece a cada clase o no de una variable categórica. Mediante este método, pasamos de 3 variables categóricas a 8 indicadoras una por cada categoría a excepción de carga del motor puesto que al solo haber 2 categorías ({CA, CB}) con una sola variable indicadora (en este caso, para carga baja) es suficiente. La figura 3.1 es un ejemplo de este tipo de codificación.

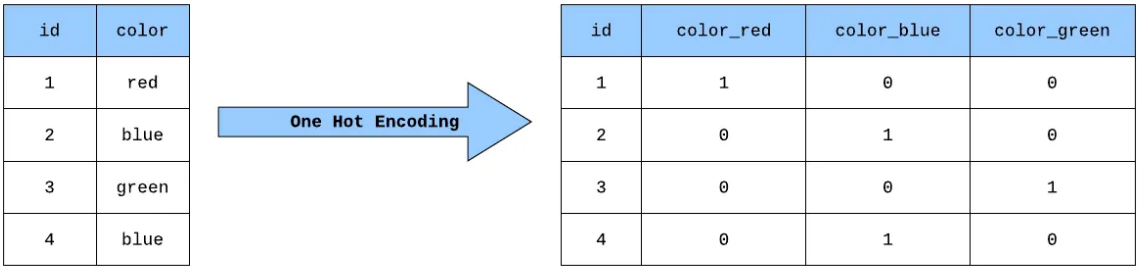

Figura 3.1: Codificación *One-Hot* de las variables categóricas [26]

3.2.2. Detección de valores atípicos

Puesto que se aplican métodos de clasificación, es interesante detectar los posibles valores atípicos ya que en caso de ignorarlos, estos podrían influenciar fuertemente los modelos obtenidos llevándonos a conclusiones erróneas o de baja calidad.

En primer lugar, se reliza una serie de matrices de gráficos de dispersión de todas las variables para detectar posibles agrupaciones de valores atípicos. Para poder realizar las matrices con variables categóricas, el software que se utiliza necesita que estas estén codificadas numéricamente. Por ello, en estos gráficos, las variables categóricas se codificarán siguiendo el siguiente mapeado de correspondencias:

- **Alimentación:** {AB:1, ABB:2, Red:3, WEG:4}
- **Frecuencia:** {35:1, 50:2, 65:3}
- **Carga:** {CA:1, CB:2}

Las matrices de gráficos de dispersión realizadas se encuentran en el anexo A.

En la figura 3.2 que es una ampliación de una de las matrices de dispersión contenida en la figura A.1, se aprecia como hay al menos dos posibles conjuntos de valores atípicos: los que tienen un coeficiente de apuntamiento de 0 y los que tienen el coeficiente de apuntamiento superior a 4.

Empezamos estudiando los que tienen el coeficiente de apuntamiento de 0, en total, 24 observaciones algunas de ellas representadas en la tabla 3.2. Estas corresponden a motores alimentados por variadores ABB con carga de trabajo alta. Apreciamos que, para estas observaciones todos los estadísticos resumen que se miden están a 0. Por tanto, decidimos verificar con el departamento de Ingeniería Eléctrica que había sucedido.

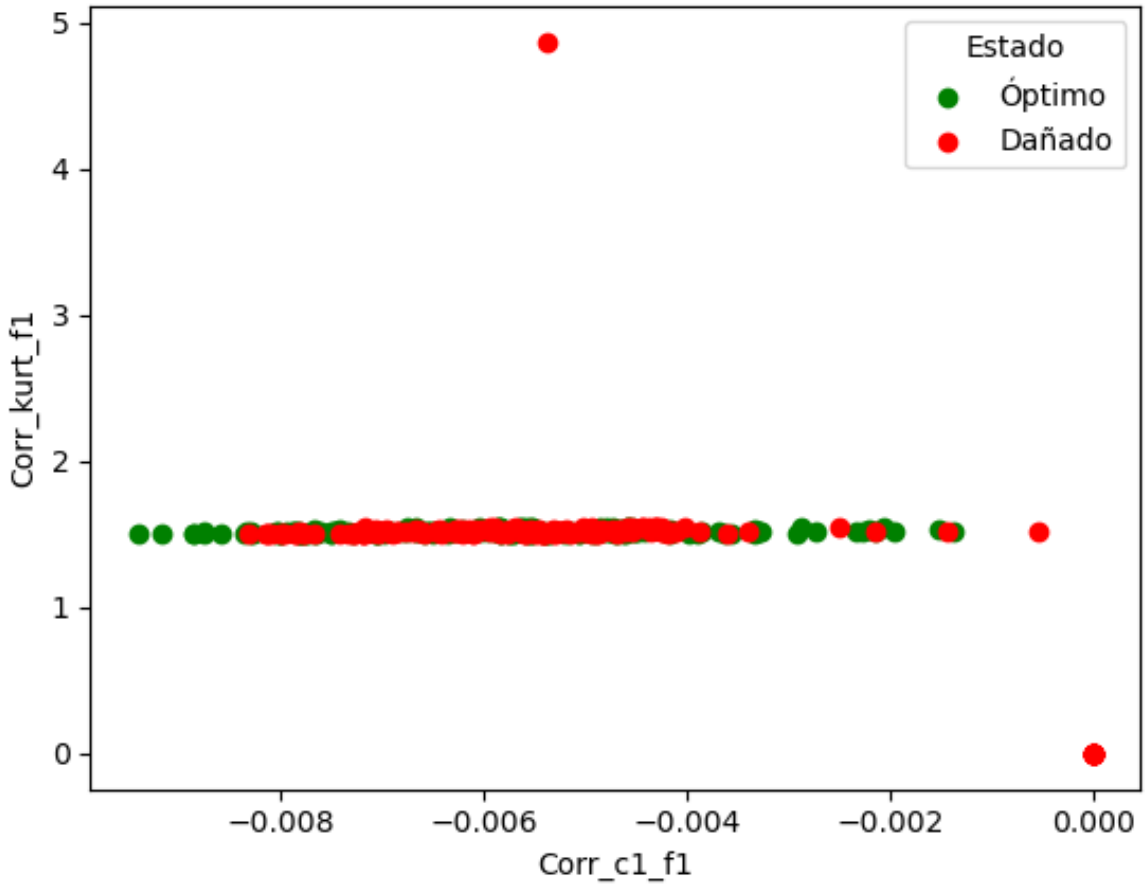

Figura 3.2: Coeficiente de apuntamiento vs. Cumulante de orden 1 (datos de corriente fase 1)

Resulta que estos datos corresponden a un problema que se produjo y por tanto no se han de tener en cuenta y han de eliminarse del conjunto de datos. La lista de identificadores de las 24 observaciones afectadas es: del 130 al 139, del 176 al 179 y del 210 al 219.

id	**resp**	**alimentacion**	**frecuencia**	**carga**	**Corr__c1__f1**	**...**	**Vibrac__sf__z**
130	2	ABB	35	CA	0	...	0
131	2	ABB	35	CA	0	...	0
...	...	...	...	...	...	...	...
219	2	ABB	65	CA	0	...	0

Tabla 3.2: *Outliers*

En cuanto a los que tienen coeficiente de apuntamiento mayor a 4, solamente es una observación que, corresponde también a una observación alimentada por un variador ABB en carga de trabajo alta, en este caso, a 50 Hz. Verificando con el departamento de Ingeniería Eléctrica que había sucedido en este caso, resulta que es justo el momento en que se produjo el fallo y por eso se dieron unos valores atípicos en esta observación antes de que

se volvieran todos nulos. Es por ello que este motor tampoco se debería tener en cuenta. Su identificador es: 175.

Finalmente, después de haber eliminado los 25 valores atípicos anteriormente mencionados, se procede a rehacer los gráficos de dispersión, encontrándose estos en el anexo A. Como se puede apreciar en estos gráficos de dispersión, ya no hay ningún indicio de valores atípicos. Vemos en la figura 3.3 la misma ampliación que antes del gráfico de la matriz A.2 pero después de haber eliminado los valores atípicos.

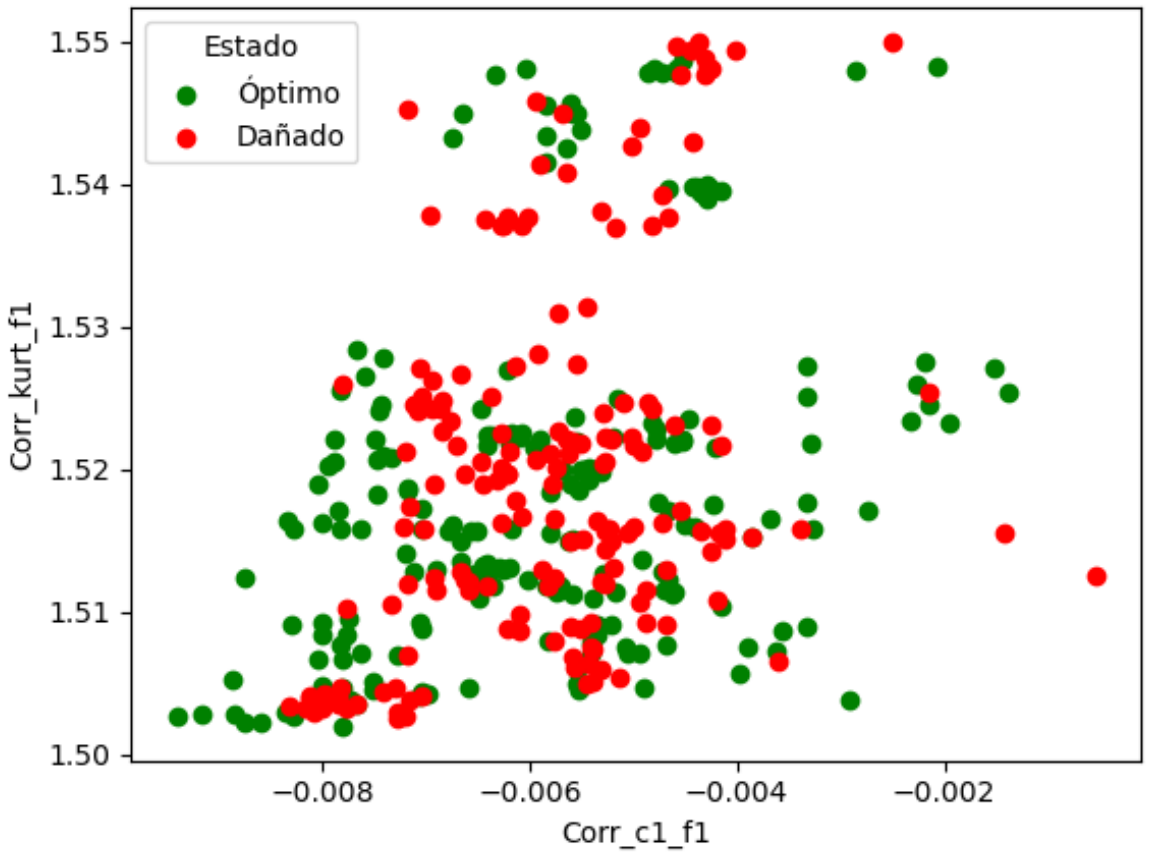

Figura 3.3: Coeficiente de apuntamiento vs. Cumulante de orden 1 (datos de corriente fase 1) después de eliminar los valores atípicos

Se aprecia como definitivamente no parece haber más atípicos. Debido a la poca cantidad de datos que tenemos finalmente sobre variadores ABB, puede que fuera necesario prescindir de estos a la espera de un mayor número de datos ya que los resultados obtenidos para estos variadores pueden estar sesgados a las observaciones restantes que quedan concretamente.

Por último, fijándonos en la figura 3.4 extraída de la figura A.3 del anexo A, se aprecia como los motores alimentados por Red son un conjunto de valores atípicos respecto al resto

para las variables de sonido. Comentándolo con el departamento de Ingeniería Eléctrica, resulta ser algo debido al método de alimentación por lo que se tratará como tal.

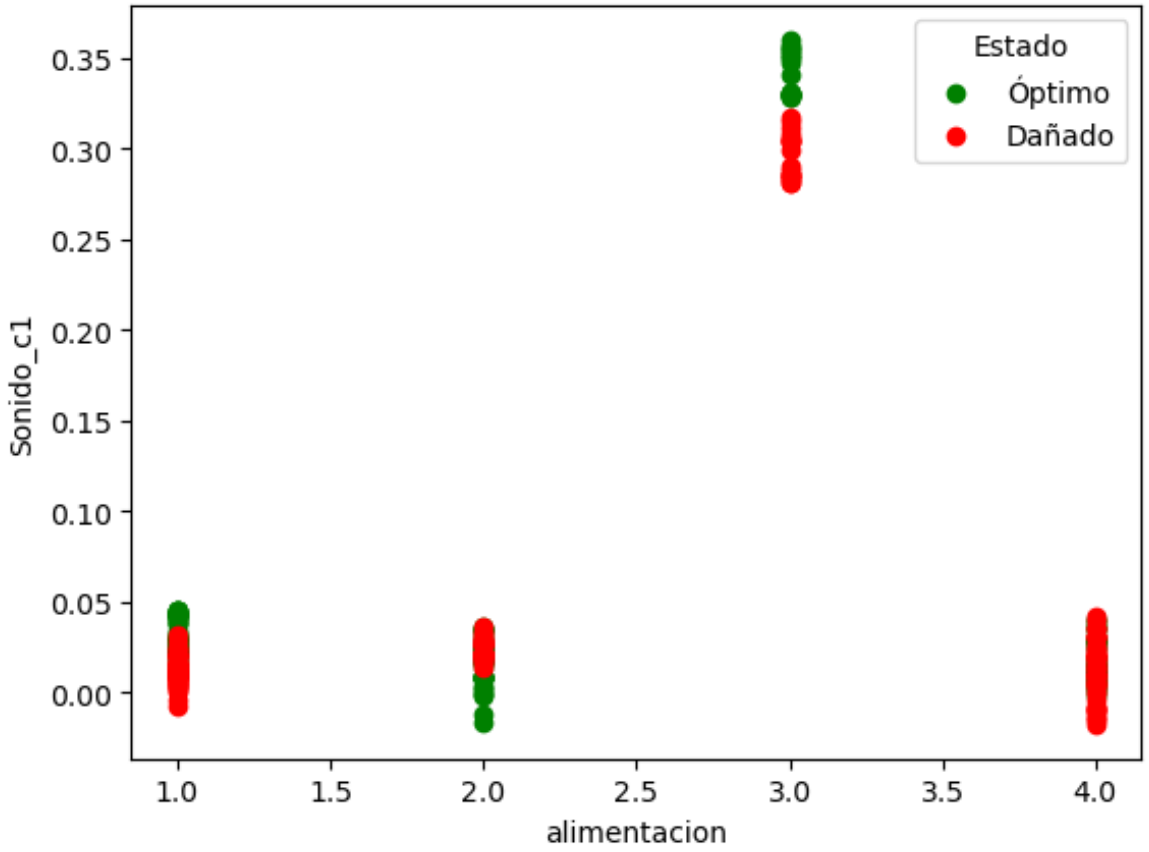

Figura 3.4: Cumulante de orden 1 vs. alimentación (datos de sonido)

3.2.3. Conjunto de datos resultante

Después de eliminar todos los valores atípicos anteriormente mencionados, la composición del conjunto de datos resultante es la que se encuentra en la tabla 3.3.

Estado del motor			**Buenas condiciones (1)**	**Dañado (2)**	**TOTAL**
Alimentación	**Frecuencia**	**Carga**			
AB	35	CA	10	10	20
		CB	10	10	20
	50	CA	10	10	20
		CB	10	10	20
	65	CA	10	10	20
		CB	10	10	20
ABB	35	CA	10	0	10
		CB	10	10	20
	50	CA	10	5	15
		CB	10	10	20
	65	CA	10	0	10
		CB	10	10	20
Red	50	CA	10	10	20
		CB	10	10	20
WEG	35	CA	10	10	20
		CB	10	10	20
	50	CA	10	10	20
		CB	10	10	20
	65	CA	10	10	20
		CB	10	10	20
TOTAL:			200 (53,33 %)	175 (46,67 %)	375 (100 %)

Tabla 3.3: Frecuencias del conjunto de datos por combinación de factores

Capítulo 4

Metodología de análisis

En este capítulo se introduce la metodología empleada para el análisis del conjunto de datos.

4.1. Diseño del experimento

El experimento base es encontrar el mejor modelo de diagnóstico posible dado un subconjunto de todas las variables explicativas mencionadas en el capítulo 3. Con este pretexto, definimos el mejor modelo posible como aquel que con el menor número de variables obtiene la mayor tasa de acierto. Se utiliza Gradient Boosting debido a los buenos resultados obtenidos en trabajos pasados [25].

Para ello, se realiza el siguiente procedimiento dado un subconjunto de observaciones y variables explicativas:

4.1.1. Modelo inicial e hiperparámetros óptimos

Para evitar el último problema mencionado en la sección 2.2.2, se lleva a cabo un método de validación híbrido que se compone tanto de *Hold-out* como de validación cruzada. De este modo, se evita el solapamiento entre el conjunto de entrenamiento y test que había en la validación cruzada pero mantenemos las ventajas de la misma a la hora de entrenar el modelo.

Método de búsqueda de hiperparámetros óptimos:

1. Se divide el subconjunto de datos mediante el método *Hold-out* estratificado. De este modo, se obtiene una partición de validación (*test*) y otra de entrenamiento (*train*) con el 20 % y 80 % respectivamente del total del subconjunto de datos. Gracias a la estratificación, mantendremos la misma proporción de motores estropeados en ambos conjuntos (*train* y *test*) asegurándonos de que las estimaciones del error

sean lo menos sesgadas en ambas clases (motores en buenas condiciones y dañados).

En lo que sigue, la partición de validación no se utilizará en el entrenamiento del modelo, de este modo, se mantiene independiente de estos datos que no se utilizarán hasta el final.

2. Para llevar a cabo el entrenamiento del modelo, en concreto la búsqueda de los hiperparámetros óptimos mencionados en la sección 2.1, se realiza validación cruzada estratificada de 5 particiones sobre el conjunto de entrenamiento. De este modo, se reduce la posibilidad de escoger un hiperparámetro peor que otro debido a una partición concreta del conjunto de entrenamiento que lo beneficiase.

   La selección de hiperparámetros óptimos se realiza con una búsqueda hacia adelante sobre una malla de posibles valores. Dado que es una búsqueda adelante, se ha de definir un orden en que se irán evaluando los posibles candidatos, en nuestro caso, usaremos el orden establecido en la tabla 2.1. El punto de partida serán los hiperparámetros por defecto de la función que se esta utilizando [17].

3. Con los mejores hiperparámetros obtenidos del paso anterior, se utiliza todo el conjunto de datos de entrenamiento para entrenar el modelo y se utiliza para predecir las observaciones de la partición de prueba. Obteniendo así una estimación del error totalmente independiente de los datos de entrenamiento. Los resultados obtenidos se muestran en tablas como la 5.1 o 5.5.

Este proceso mencionado, sirve para buscar los hiperparámetros óptimos así como una primera aproximación de la calidad del modelo con todas las variables. No obstante, como ya se menciono en el capítulo 3, nos encontramos con datos de alta dimensionalidad por lo que es pertinente llevar a cabo una reducción del número variables explicativas del modelo. De este modo, evitamos el sobreajuste que podemos tener por estar utilizando todas las variables disponibles.

4.1.2. Selección de variables

Para realizar la selección de variables, debido a que realizar una búsqueda exhaustiva de todas las posibles combinaciones de variables es computacionalmente muy costoso además del sobreajuste que estaríamos generando. Partiendo de las importancias obtenidas por las variables explicativas del modelo anteriormente obtenido con los hiperparámetros óptimos en la sección anterior, se lleva a cabo el siguiente método de selección de variables:

1. Se ordenan de mayor a menor las importancias de las variables.

2. Se ajustan modelos añadiendo las variables una a una con el orden determinado en el primer paso y, se estima su tasa de acierto por validación cruzada estratificada de 5 particiones.

3. Para reducir la variabilidad de los resultados, se lleva a cabo un remuestreo *Bootstrap* de 10 repeticiones para cada uno de los modelos.

4. Con todas las repeticiones *Bootstrap* se obtiene los estimadores de las tasas de acierto de cada uno de los modelos y se obtienen gráficos del tipo "Tasa de acierto vs. Número de variables explicativas" como el de la figura 4.1.

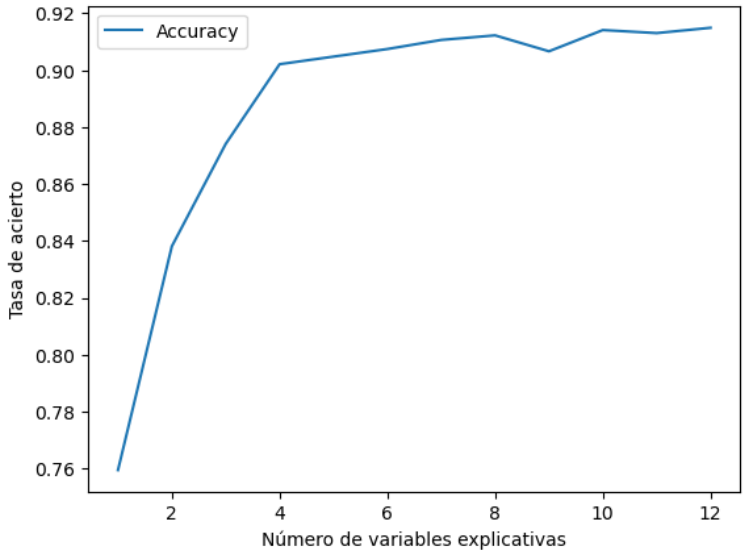

Figura 4.1: Ejemplo de Tasa de acierto vs. Número de variables explicativas

Con estos gráficos, se escoge el modelo que obtenga una tasa de acierto semejable al modelo con todas las variables pero con menor número de estas. En el ejemplo de la figura 4.1 el número de variables escogidas sería 4.

4.1.3. Estudio de la estabilidad del modelo

Partiendo del modelo seleccionado en la sección anterior, se realizan 50 repeticiones *Bootstrap* del modelo escogido variando las particiones de entrenamiento y prueba. Esto permite verificar la variabilidad de las tasas de acierto del modelo dependiendo del conjunto de datos de entrenamiento y prueba escogidos. De este modo, si se obtiene una gran variabilidad de las tasas de acierto podemos estar ante una falta de estabilidad en el modelo. La tabla 4.1 es un ejemplo de este tipo de estudio.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
c1, c3, kurt y sf	50	0,9029	0,0285

Tabla 4.1: Ejemplo de estudio de la estabilidad de un modelo

4.2. Comparación de modelos

Con las muestras *Bootstrap* obtenidas se lleva a cabo un análisis de la varianza (ANOVA) para comparar distintos modelos. Como se ha comentado en la sección 2.4, para poder aplicar este análisis se ha de cumplir una serie de hipótesis: normalidad, homocedasticidad e independencia de los datos.

Teniendo en cuenta el diseño del experimento llevado a cabo, la hipótesis de independencia se cumple puesto que se remuestrean los datos de forma aleatoria. En cuanto a la normalidad y homocedasticidad de los datos, se comprueba gráficamente con gráficos cuantil-cuantil y de cajas y bigotes respectivamente (un ejemplo de estos gráficos se encuentra en la figura 5.2). No obstante, en caso de no cumplirse la hipótesis de homocedasticidad dado que nuestro diseño del experimento genera 50 repeticiones *Bootstrap* de la tasa de acierto para cada modelo, nos encontramos con grupos balanceados por lo que la falta de homocedasticidad no es un problema grave para el ANOVA [21]. De modo similar, la falta de normalidad tampoco es un problema importante para el ANOVA [21]. Los resultados del ANOVA se guardan en tablas como la 5.4.

Si el ANOVA saliera significativo, se lleva a cabo un análisis *post-hoc* con el test de Tukey para realizar las comparaciones 2 a 2 de los niveles del factor viendo así que niveles son significativamente diferentes. En el caso de más de un factor, se aplica la misma metodología para la comparación teniendo en cuenta que a mayores entra en juego la interacción de ambos factores.

En caso de que la interacción fuese significativa (tabla 5.16 y figura 5.7), se realiza un nuevo factor producto cartesiano de los dos y se analiza este como uno solo. En caso de no ser significativa la interacción, se estudian los factores por separado siguiendo la metodología ya vista.

Capítulo 5

Resultados

En este capítulo se aborda el análisis del conjunto de datos resultante de aplicar el procesado de datos indicado en la sección 3.2 con la metodología enunciada en el capítulo 4. Se utiliza el lenguaje de programación Python con las librerías Pandas, Sklearn y Numpy entre otras. Todo el código desarrollado para el análisis de los datos se encuentra en el anexo C.

5.1. Análisis por fuente de datos

5.1.1. Análisis solo con los datos de corriente

Todas las variables

En primer lugar, se realiza un modelo inicial con todas las variables de corriente disponibles en la sección 3.1.1 llevando a cabo la búsqueda de hiperparámetros óptimos mencionada en la sección 4.1.1. De este modo, se obtiene la siguiente mejora (tabla 5.1).

Modelo	Tasa de acierto
Modelo inicial	0,8667
Modelo optimizado	0,9067

Tabla 5.1: Resultados de aplicar *Gradient Boosting* a todos los datos de corriente

Debido a la gran cantidad de variables explicativas que se esta utilizando podría indicar sobreajuste. Por ello, se decide junto con el departamento de Ingeniería Eléctrica de la Universidad de Valladolid promediar las fases de corriente reduciéndose las 39 variables a tan solo 13.

Todas las variables promediadas

Modelo	Tasa de acierto
Modelo inicial	0,8533
Modelo optimizado	0,9067

Tabla 5.2: Resultados de aplicar *Gradient Boosting* a todos los datos de corriente promediados

En este caso (tabla 5.2), el punto de partida es peor, no obstante, gracias a la optimización de los hiperparámetros del modelo se llega a alcanzar la misma tasa de acierto que con todas las variables sin promediar por lo que ambos modelos parecen ser igual de buenos.

Reducción del número de variables

Tomando como punto de partida el modelo anterior, se intenta llevar a cabo una reducción del número de variables explicativas del modelo que no penalice la tasa de acierto. Para ello, se lleva a cabo el método enunciado en la sección 4.1.2.

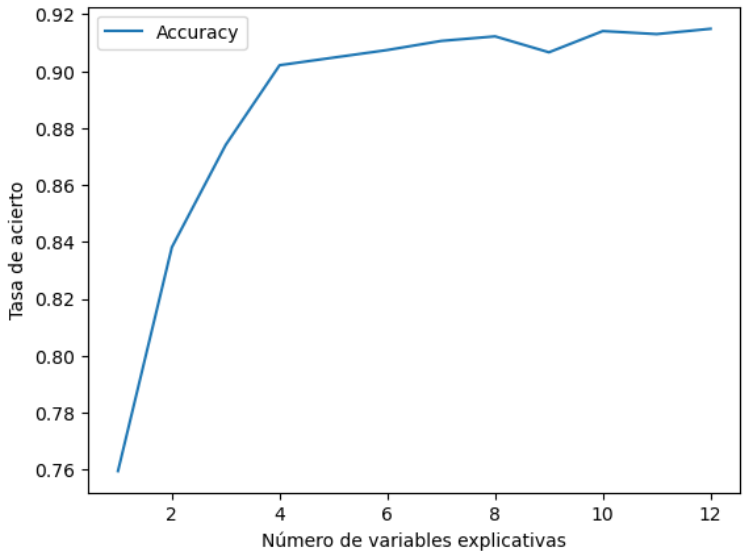

Figura 5.1: Tasa de acierto vs. número de variables explicativas de corriente promediadas

Como se aprecia en la figura 5.1, a partir de 4 variables explicativas la tasa de acierto apenas mejora por lo que nos quedaremos con dicho modelo por ser el más sencillo que explica los datos con la mayor tasa de acierto posible. Se lleva a cabo repeticiones del modelo remuestreando las particiones de entrenamiento y prueba como se menciona en la sección 4.1.3 (tabla 5.3).

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
c_1, c_3, kurt y sf	50	0,9437	0,0263

Tabla 5.3: Resumen de la estabilidad del modelo obtenido con variables de corriente promediadas

Modelo obtenido vs. Modelo todas las variables sin promediar

Con las tasas de acierto obtenidas con las repeticiones *Bootstrap* se realiza un análisis de la varianza como el enunciado en la sección 4.2. En primer lugar, se verifica la hipótesis de normalidad en el gráfico cuantil-cuantil y la de homocedasticidad en la amplitud del diagrama de cajas y bigotes de la figura 5.2.

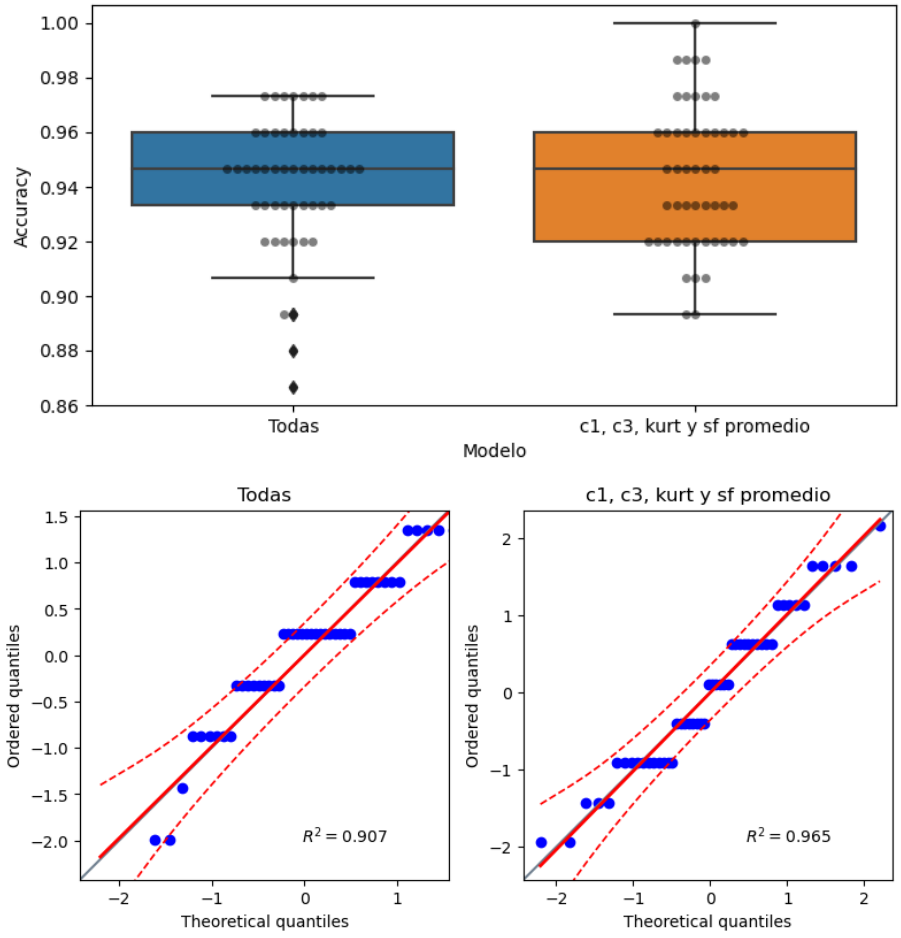

Figura 5.2: Comprobación de las hipótesis del ANOVA para comparar los modelos con datos de corriente

Como las hipótesis de ANOVA se verifican se lleva a cabo la comparación de los modelos no encontrándose diferencias significativas entre estos (tabla 5.4).

Source	Sum of Squares	Degrees of Freedom	Mean Squares	F Value	p
Modelo	0,000178	1	0,000178	0,277809	0,599332
Error	0,062713	98	0,000640		
Total	0,062891				

Tabla 5.4: Tabla ANOVA de corriente

5.1.2. Análisis solo con los datos de sonido

Todas las variables

En primer lugar, se realiza un modelo con todas las variables de sonido disponibles en la sección 3.1.2. Llevando a cabo la búsqueda de mejores hiperparámetros para el modelo, obtenemos un modelo con 100 % de tasa de acierto. De esta selección de hiperparámetros óptimos que encontramos en la tabla 5.5 cabe destacar el número de estimadores (árboles) que componen el modelo (1) y la sencillez de este dado que es de profundidad 1.

Hiperparámetro	Valor
Mínimo de observaciones para dividir un nodo	2
Mínimo de observaciones en nodo hoja	1
Máximo de profundidad del árbol	1
Máximo de nodos hoja	2
Tasa de aprendizaje	0,1
Número de árboles	1
Porcentaje de muestra para entrenar cada árbol	90 %

Tabla 5.5: Hiperparámetros del modelo optimizado con variables de sonido

Reducción del número de variables

Dados los resultados de la sección anterior, se decide realizar una reducción del número de variables del modelo. Para ello, manteniendo los hiperparámetros óptimos encontrados en la tabla 5.5, se lleva a cabo un método de introducción de variables paso a paso con validación cruzada estratificada de 5 particiones, introduciendo primero las de mayor importancia.

En este caso, se obtiene que el modelo sólo con la variable c_2 ya se obtiene una tasa de acierto del 100 %.

Estudio de la variable c_2

Dado el resultado obtenido en la sección anterior, se decide ver que esta sucediendo realmente con la variable c_2. Para ello, se realiza un gráfico de dispersión de esta variable respecto al estado del motor:

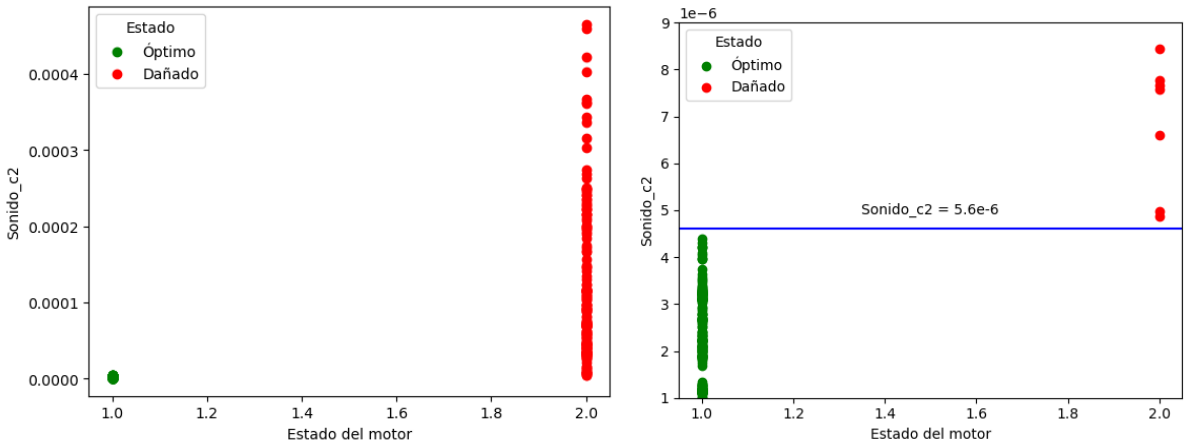

Figura 5.3: Sonido_c2 vs. estado del motor

Como se aprecia en la recta azul del gráfico de la derecha de la figura 5.3 que es una ampliación del de la izquierda, el estado de los motores se separa linealmente por la variable de sonido c_2.

Se lleva a cabo repeticiones del modelo remuestreando las particiones de entrenamiento y prueba como se menciona en la sección 4.1.3 (tabla 5.6).

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
c_2	50	0,9989	0,0045

Tabla 5.6: Resumen de la estabilidad del modelo obtenido con variables de sonido

5.1.3. Análisis solo con los datos de vibraciones

Todas las variables

En primer lugar, se realiza un modelo con todas las variables de vibraciones disponibles obteniendo una tasa de acierto del 100 %. Como nos ocurría con los datos de corriente, utilizando todas las variables de vibraciones estamos utilizando un gran número de variables explicativas (39) respecto al número de observaciones (375). Por tanto, se decide junto con el departamento de Ingeniería Eléctrica de la Universidad de Valladolid que tiene sentido utilizar los datos de vibraciones utilizando cada eje de manera independiente. Reduciéndose así el número de variables explicativas a 13 en cada caso.

Eje X

Todas las variables

En primer lugar, se realiza un modelo con todas las variables de vibraciones del eje X. En este caso, se obtiene una tasa de acierto del 98,67 % la cuál se mejora con la búsqueda de hiperparámetros óptimos hasta el 100 %.

Reducción del número de variables

Se lleva a cabo una reducción del número de variables usando el método enunciado en la sección 4.1.2 obteniéndose los siguientes resultados (figura 5.4).

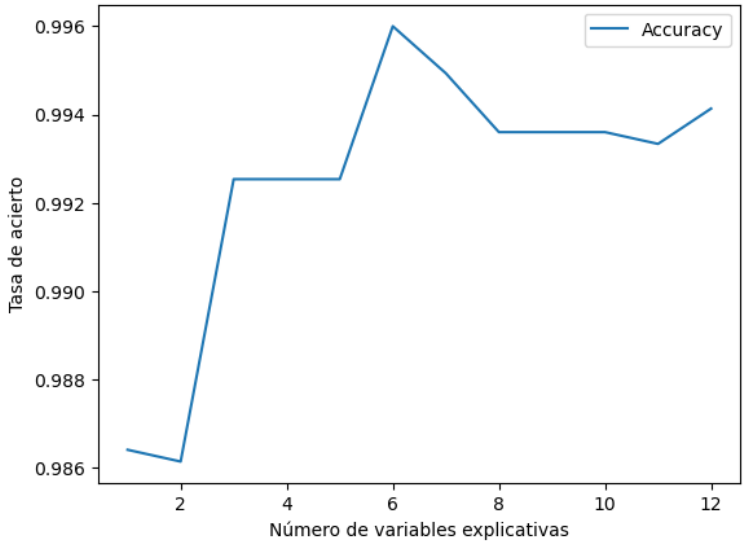

Figura 5.4: Tasa de acierto vs. número de variables explicativas de vibraciones (eje X)

Del gráfico de la figura 5.4 cabe destacar que las tasas de acierto obtenidas oscilan mínimamente. Saliendo mejor parado el modelo con las variables: m_2, c_2 y x_r.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
m_2, c_2 y x_r	50	0,9936	0,0105

Tabla 5.7: Resumen de la estabilidad del modelo obtenido con variables de vibraciones (eje X)

Eje Y

Todas las variables

En primer lugar, se realiza un modelo con todas las variables disponibles obteniendo una tasa de acierto del 100 % por lo que se decide optimizar los hiperparámetros del modelo en busca de uno más sencillo (tabla 5.8).

Hiperparámetro	Valor
Mínimo de observaciones para dividir un nodo	2
Mínimo de observaciones en nodo hoja	1
Máximo de profundidad del árbol	3
Máximo de nodos hoja	4
Tasa de aprendizaje	0,3
Número de árboles	1
Porcentaje de muestra para entrenar cada árbol	80 %

Tabla 5.8: Hiperparámetros del modelo optimizado con variables de vibraciones (eje Y)

Fijándonos en los hiperparámetros óptimos encontrados podemos apreciar de que se trata de un único árbol sencillo. Por lo que se lleva a cabo una reducción agresiva del número de variables explicativas del modelo encontrando que con tan sólo la variable c_1 se tiene una tasa de acierto del 100 %.

Estudio de la variable c_1

Dado que el modelo obtenido es un único árbol sencillo y con una sola variable es capaz de alcanzar el 100 % de tasa de acierto, se realiza un estudio de esta variable.

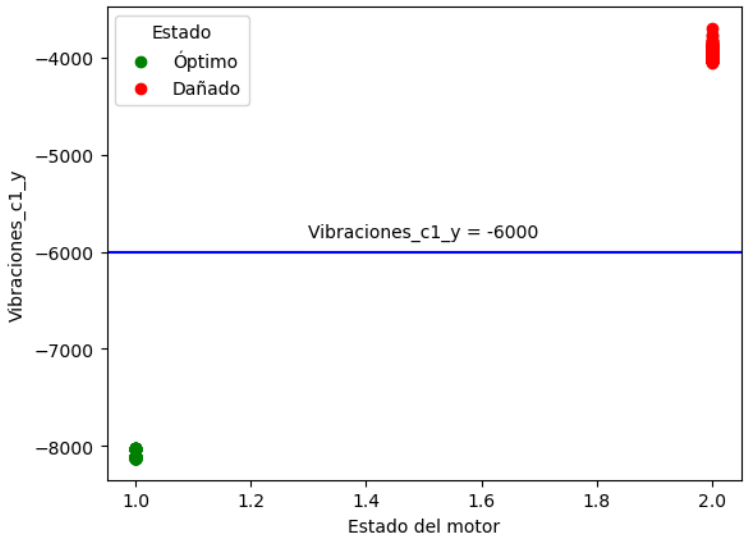

Figura 5.5: Vibraciones_c1_y vs. estado del motor

Como se aprecia en la recta azul del gráfico de la figura 5.5, el estado de los motores

se separa linealmente por la variable c_1 de vibraciones en el eje Y.

Se lleva a cabo repeticiones del modelo remuestreando las particiones de entrenamiento y prueba como se menciona en la sección 4.1.3 (tabla 5.9).

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
c_1	50	1	0

Tabla 5.9: Resumen de la estabilidad del modelo obtenido con variables de vibraciones (eje Y)

Eje Z

Todas las variables

En primer lugar, se realiza un modelo con todas las variables disponibles obteniendo una tasa de acierto del 98,67 %. Por ello, se lleva a cabo una optimización de los hiperparámetros del modelo en búsqueda de una mejora. No obstante, no se consigue mejorar la tasa de acierto.

Reducción del número de variables

Utilizando el método enunciado en la sección 4.1.2 se obtienen las siguientes tasas de acierto en función del número de variables del modelo.

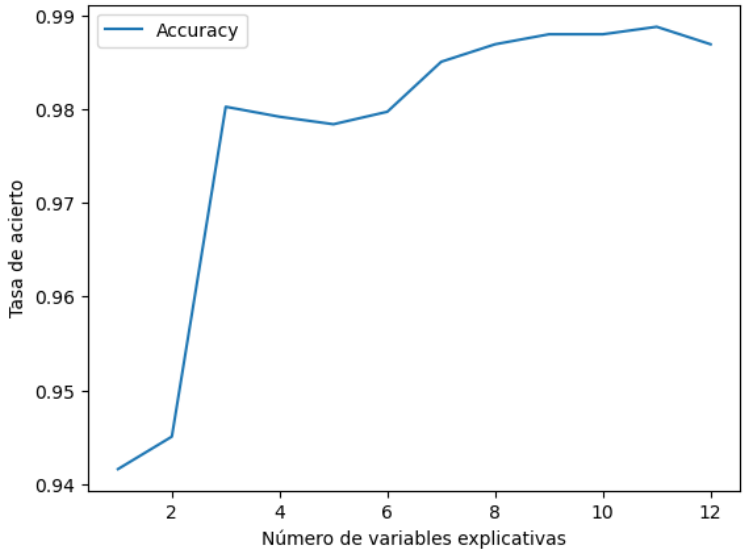

Figura 5.6: Tasa de acierto vs. número de variables explicativas de vibraciones (eje Z)

Como se aprecia en la figura 5.6, podemos mantener la tasa de acierto del modelo con todas las variables seleccionando 3 variables. Siendo m_2, c_2 y c_1 el modelo más sencillo

que explica los datos (tabla 5.10).

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
m_2, c_2 y c_1	50	0,9875	0,0122

Tabla 5.10: Resumen de la estabilidad del modelo obtenido con variables de vibraciones (eje Z)

5.1.4. Comparación de las fuentes de datos

En resumen, los mejores modelos obtenidos en las secciones anteriores para cada tipo de datos son los que se encuentran en la figura 5.11.

Tipo de variables	Variables	Tasa de acierto	Desviación típica
Corriente promediadas	c_1, c_3, kurt y sf	0,9437	0,0263
Sonido	c_2	0,9989	0,0045
Vibraciones (eje X)	m_2, c_2 y x_r	0,9936	0,0105
Vibraciones (eje Y)	c_1	1	0
Vibraciones (eje Z)	m_2, c_2 y c_1	0,9875	0,0122

Tabla 5.11: Mejores modelos obtenidos con cada tipo de variable

Teniendo en cuenta las replicaciones *Bootrstrap* de cada uno de estos modelos se lleva a cabo una comparación ANOVA. El resultado del ANOVA se encuentra en la tabla 5.12, en esta, se rechaza la hipótesis nula de igualdad de medias por lo que los modelos son significativamente distintos.

Source	Sum of Squares	Degrees of Freedom	Mean Squares	F Value	p
Modelo	0,110090	4	0,027522	141,760465	1,54e-62
Error	0,047566	245	0,000194		
Total	0,157656				

Tabla 5.12: Tabla ANOVA de comparación por fuente de datos

En vista a los resultados del ANOVA se realiza un análisis *post-hoc* utilizando el test de comparaciones múltiples de Tukey para comparar los distintos modelos 2 a 2. Los p-valores de los contrastes de hipótesis de este análisis se encuentran recogidos en las tabla 5.14. También se muestra un resumen de los niveles del factor en la tabla 5.13.

Group Levels	Count	Mean
Corriente promediadas	50	0,943733
Sonido	50	0,998933
Vibraciones (eje X)	50	0,993600
Vibraciones (eje Y)	50	1,000000
Vibraciones (eje Z)	50	0,987467

Tabla 5.13: Tabla resumen de los niveles del factor fuente de datos

Contrast	p-value	Difference
Corriente promediadas - Sonido	0,001000	-0,055200
Corriente promediadas - Vibraciones (eje X)	0,001000	-0,049867
Corriente promediadas - Vibraciones (eje Y)	0,001000	-0,056267
Corriente promediadas - Vibraciones (eje Z)	0,001000	-0,043733
Sonido - Vibraciones (eje X)	0,312747	0,005333
Sonido - Vibraciones (eje Y)	0,900000	-0,001067
Sonido - Vibraciones (eje Z)	0,001000	0,011467
Vibraciones (eje X) - Vibraciones (eje Y)	0,149440	-0,006400
Vibraciones (eje X) - Vibraciones (eje Z)	0,182701	0,006133
Vibraciones (eje Y) - Vibraciones (eje Z)	0,001000	0,012533

Tabla 5.14: Tabla *post-hoc* de Tukey de comparación por fuente de datos

En las tabla 5.14 vemos como la fuente de datos corriente es significativamente distinta de todas las demás. Además, teniendo en cuenta que la diferencia de tasas de acierto es negativa, es significativamente peor que el resto de fuentes de datos.

Por otro lado, la fuente de datos sonido es significativamente mejor que la fuente de datos vibraciones (eje Z). No obstante es solo por un 1 % de diferencia en la tasa de acierto.

Y por último, en cuanto a las fuentes de datos de vibraciones, el eje X no es significativamente diferente a los demás y el Y es significativamente mejor que el Z.

5.2. Análisis de los datos por tipo de alimentación

En esta sección se realiza un análisis similar al de la sección anterior teniendo en cuenta a mayores el factor tipo de alimentación. La obtención de los mejores modelos para cada fuente de datos y tipo de alimentación que se ven en la tabla 5.15 se realiza de manera análoga al procedimiento visto en la sección anterior. Por ello, el proceso detallado de la obtención se encuentra en el anexo D.

Tipo alimentación	**Tipo de variables**	**Variables**	**Tasa de acierto**	**Desviación típica**
AB	**Corriente promediadas**	c_1, c_3, kurt, x_r, cf y sf	0,9175	0,0568
	Sonido	c_2	1	0
	Vibraciones (eje X)	c_2, c_4, m_2, m_4, x_r y sf	0,9933	0,0176
	Vibraciones (eje Y)	c_1	1	0
	Vibraciones (eje Z)	c_1	1	0
ABB	**Corriente promediadas**	c_1, c_4 y sf	0,9579	0,0521
	Sonido	c_2	1	0
	Vibraciones (eje X)	m_4	1	0
	Vibraciones (eje Y)	c_1	1	0
	Vibraciones (eje Z)	m_4	1	0
Red	**Corriente promediadas**	c_1	1	0
	Sonido	c_2	0,9925	0,0300
	Vibraciones (eje X)	m_4	0,99	0,0343
	Vibraciones (eje Y)	c_1	1	0
	Vibraciones (eje Z)	cf	0,9825	0,0506
WEG	**Corriente promediadas**	c_1, c_3, m_2, m_3, skew, cf y sf	0,9533	0,0481
	Sonido	c_2	1	0
	Vibraciones (eje X)	m_2	1	0
	Vibraciones (eje Y)	c_1	1	0
	Vibraciones (eje Z)	m_2	1	0

Tabla 5.15: Mejores modelos obtenidos con cada tipo de variable según alimentación

Haciendo uso de las replicaciones *Bootstrap* de cada uno de los modelos contenidos en la tabla 5.15, se lleva a cabo una comparación ANOVA de dos factores: el factor fuente de datos y el factor tipo de alimentación.

Source	Sum of Squares	Degrees of Freedom	Mean Squares	F Value	p
Tipo de dato	0,261322	4	0,065331	99,068024	8e-71
Tipo de alimentación	0,017906	3	0,005969	9,051025	6e-6
Tipo de dato * Tipo de alimentación	0,170606	12	0,014217	21,559041	1e-42
Error	0,646263	980	0,000659		
Total	1,096097				

Tabla 5.16: Tabla ANOVA de comparación por fuente de datos y tipo de alimentación

En la tabla 5.16 se muestra como tanto el factor tipo de dato como el factor tipo de alimentación y su interacción son significativas. Para realizar las comparaciones 2 a 2 puesto que la interacción de los factores es significativa, se realiza un nuevo factor siendo este el producto cartesiano de "Tipo de dato" * "Tipo de alimentación" sobre el que se aplica el test de Tukey contenido en la tabla 5.17. También se muestra un resumen de los niveles del factor en la tabla 5.18. Debido a la gran cantidad de comparaciones 2 a 2 que hay (5 tipos de datos * 4 tipos de alimentación = 20), solamente se mencionarán aquellas que sean significativamente distintas. También se muestra el gráfico de la interacción de los factores en la figura 5.7 para facilitar la comprensión de los resultados.

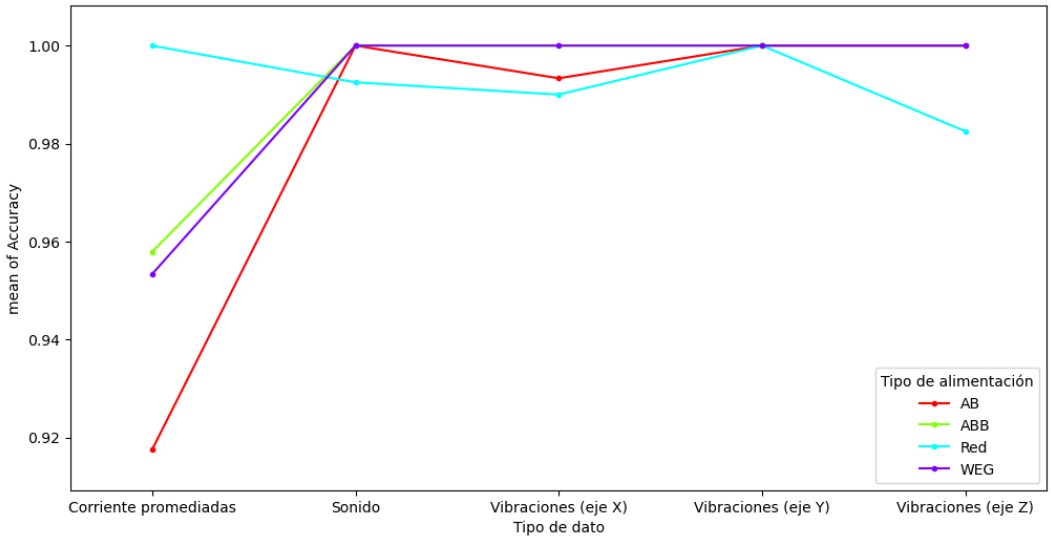

Figura 5.7: Interacción Tipo de dato * Tipo de alimentación

Contrast	p-value	Difference
Corriente promediadas x AB - Corriente promediadas x ABB	0,001000	-0,040395
Corriente promediadas x AB - Corriente promediadas x Red	0,001000	-0,082500
Corriente promediadas x AB - Corriente promediadas x WEG	0,001000	-0,035833
Corriente promediadas x ABB - Corriente promediadas x Red	0,001000	-0,042105
Corriente promediadas x ABB - Corriente promediadas x WEG	0,900000	0,004561
Corriente promediadas x Red - Corriente promediadas x WEG	0,001000	0,046667
Sonido x AB - Sonido x ABB	0,900000	0,000000
Sonido x AB - Sonido x Red	0,900000	0,007500
Sonido x AB - Sonido x WEG	0,900000	0,000000
Sonido x ABB - Sonido x Red	0,900000	0,007500
Sonido x ABB - Sonido x WEG	0,900000	0,000000
Sonido x Red - Sonido x WEG	0,900000	-0,007500
Vibraciones (eje X) x AB - Vibraciones (eje X) x ABB	0,900000	-0,006667
Vibraciones (eje X) x AB - Vibraciones (eje X) x Red	0,900000	0,003333
Vibraciones (eje X) x AB - Vibraciones (eje X) x WEG	0,900000	-0,006667
Vibraciones (eje X) x ABB - Vibraciones (eje X) x Red	0,900000	0,010000
Vibraciones (eje X) x ABB - Vibraciones (eje X) x WEG	0,900000	0,000000
Vibraciones (eje X) x Red - Vibraciones (eje X) x WEG	0,900000	-0,010000
Vibraciones (eje Y) x AB - Vibraciones (eje Y) x ABB	0,900000	0,000000
Vibraciones (eje Y) x AB - Vibraciones (eje Y) x Red	0,900000	0,000000
Vibraciones (eje Y) x AB - Vibraciones (eje Y) x WEG	0,900000	0,000000
Vibraciones (eje Y) x ABB - Vibraciones (eje Y) x Red	0,900000	0,000000
Vibraciones (eje Y) x ABB - Vibraciones (eje Y) x WEG	0,900000	0,000000
Vibraciones (eje Y) x Red - Vibraciones (eje Y) x WEG	0,900000	0,000000
Vibraciones (eje Z) x AB - Vibraciones (eje Z) x ABB	0,900000	0,000000
Vibraciones (eje Z) x AB - Vibraciones (eje Z) x Red	0,079590	0,017500
Vibraciones (eje Z) x AB - Vibraciones (eje Z) x WEG	0,900000	0,000000
Vibraciones (eje Z) x ABB - Vibraciones (eje Z) x Red	0,079590	0,017500
Vibraciones (eje Z) x ABB - Vibraciones (eje Z) x WEG	0,900000	0,000000
Vibraciones (eje Z) x Red - Vibraciones (eje Z) x WEG	0,079590	-0,017500

Tabla 5.17: Tabla *post-hoc* de Tukey de comparación por tipo de alimentación

Group Levels	Count	Mean
Corriente promediadas x AB	50	0,917500
Corriente promediadas x ABB	50	0,957895
Corriente promediadas x Red	50	1,000000
Corriente promediadas x WEG	50	0,953333
Sonido x AB	50	1,000000
Sonido x ABB	50	1,000000
Sonido x Red	50	0,992500
Sonido x WEG	50	1,000000
Vibraciones (eje X) x AB	50	0,993333
Vibraciones (eje X) x ABB	50	1,000000
Vibraciones (eje X) x Red	50	0,990000
Vibraciones (eje X) x WEG	50	1,000000
Vibraciones (eje Y) x AB	50	1,000000
Vibraciones (eje Y) x ABB	50	1,000000
Vibraciones (eje Y) x Red	50	1,000000
Vibraciones (eje Y) x WEG	50	1,000000
Vibraciones (eje Z) x AB	50	1,000000
Vibraciones (eje Z) x ABB	50	1,000000
Vibraciones (eje Z) x Red	50	0,982500
Vibraciones (eje Z) x WEG	50	1,000000

Tabla 5.18: Tabla resumen de los niveles del factor interacción tipo de dato * tipo de alimentación

Resultados del test *post-hoc* de Tuckey para las comparaciones 2 a 2 de la interacción Tipo de dato * Tipo de alimentación (tabla 5.17):

- En primer lugar, para la fuente de datos corriente, los únicos tipos de alimentación que no son significativamente diferentes son los variadores ABB y WEG. Entre el resto, se aprecia como el tipo de dato corriente funciona significativamente mejor para los motores alimentados por Red, luego para los alimentados por variador ABB y WEG, y, por último, para los motores alimentados por variador AB.

- Para la fuente de datos sonido, no se obtiene ningún p-valor que rechace la hipótesis nula de igualdad de tasas de acierto. Por ello, para este tipo de dato, sonido, no hay diferencias significativas dependiendo del tipo de alimentación del motor. Esto también sucede con la fuente de datos vibraciones (eje X) y vibraciones (eje Y).

- Por último, para la fuente de datos vibraciones (eje Z), aunque no se aprecian

diferencias significativas a nivel 0,05, si a nivel 0,1 entre los tipos de alimentación Red y el resto. Para los cuales podemos decir que el tipo de dato vibraciones (eje Z) funciona significativamente peor para los motores alimentados por Red que para los alimentados por los variadores.

Como resumen final de esta sección podemos decir que, para las observaciones disponibles, el uso de los datos de vibraciones del eje Y arrojan una clasificación perfecta tanto para la alimentación directa de red como para todos los variadores considerados en este estudio. Además esta clasificación se obtiene en todos los casos con la variable c_1 que es, como se definió en la Tabla 3.1 la media de la señal recogida.

Además los datos de vibraciones del eje Z dan también resultado perfecto, excepto para el caso de la alimentación directa de red, con una única variable, aunque esta variable no es siempre la misma ya que aparece c_1 para el variador AB, m_4 para el variador ABB y m_2 para WEG.

En cuanto a los datos de sonido se obtiene también el mejor resultado posible para todos los casos excepto en la alimentación directa de red y en todos los casos se llega al resultado con una única variable que siempre es c_2, el cumulante de segundo orden.

Finalmente, también es de interés observar que en lo que se refiere a los datos de corriente, que son de especial interés puesto que es la fuente de datos menos invasiva y menos sensible a errores de medida, solo se obtiene el mejor resultado posible para la alimentación directa de Red mediante el uso de la variable c_1. Cuando hay variadores presentes los datos de red arrojan resultados significativamente peores que los obtenidos mediante otras fuentes de datos.

Cabe señalar también que, aunque estaba previsto en un principio, no ha habido necesidad de implementar procedimientos diagnósticos en los que se combinen diferentes fuentes de información ya que se han obtenido resultados perfectos tanto para la alimentación en red como para todos los variadores con una única fuente de información.

5.3. Análisis de los datos por tipo de carga

En esta sección se realiza un análisis similar al de la sección 5.1 teniendo en cuenta a mayores el factor tipo de carga del motor. La obtención de los mejores modelos para cada fuente de datos y tipo de carga que se ven en la tabla 5.19 se realiza de manera análoga al procedimiento visto en la sección 5.1. Por ello, el proceso detallado de la obtención se encuentra en el anexo E.

Tipo carga	Tipo de variables	Variables	Tasa de acierto	Desviación típica
Baja	**Corriente promediadas**	c_1, m_3, x_r, cf y sf	0,947	0,0415
	Sonido	c_2	0,9995	0,0035
	Vibraciones (eje X)	c_2, m_2 y x_r	0,9975	0,0076
	Vibraciones (eje Y)	c_1	1	0
	Vibraciones (eje Z)	c_2, x_r y cf	0,988	0,0184
Alta	**Corriente promediadas**	c_1, c_3, skew, kurt, x_r y sf	0,9497	0,0430
	Sonido	c_2	0,9977	0,0078
	Vibraciones (eje X)	m_2	0,9874	0,0175
	Vibraciones (eje Y)	c_1	1	0
	Vibraciones (eje Z)	c_1 y x_r	0,9903	0,0188

Tabla 5.19: Mejores modelos obtenidos con cada tipo de variable según carga

Haciendo uso de las replicaciones *Bootstrap* de cada uno de los modelos contenidos en la tabla 5.19, se lleva a cabo una comparación ANOVA de dos factores: el factor fuente de datos y el factor tipo de carga.

Source	Sum of Squares	Degrees of Freedom	Mean Squares	F Value	p
Tipo de dato	0,182318	4	0,045580	96,886497	1e-60
Tipo de carga	0,000235	1	0,000235	0,499747	0,4799
Tipo de dato * Tipo de carga	0,002695	4	0,000674	1,432298	0,2221
Error	0,230517	490	0,000470		
Total	0,415765				

Tabla 5.20: Tabla ANOVA de comparación por fuente de datos y tipo de carga

En la tabla 5.20 se muestra como el factor interacción entre el tipo de dato y carga del motor no es significativo, por ello, se decide eliminarlo del análisis ANOVA y se realiza de nuevo este contenido en la tabla 5.21.

Source	Sum of Squares	Degrees of Freedom	Mean Squares	F Value	p
Tipo de dato	0,182318	4	0,0455795	96,55	0
Tipo de carga	0,000235	1	0,000235	0,5	0,4807
Error	0,233212	494	0,000472		
Total	0,415765				

Tabla 5.21: Tabla ANOVA de comparación por fuente de datos y tipo de carga sin interacción

En las tabla 5.21 se muestra como el factor tipo de dato es significativo. No obstante, el factor carga no lo es. Es decir, el tipo de carga del motor no influye significativamente en los modelos de diagnóstico de fallos de los motores. Por ello, no se realizan tests *post-hoc* ni para el factor de carga ni para la interacción. En cuanto a los tests *post-hoc* para el factor tipo de dato, ya se realizaron en la sección 5.1.4. En la figura 5.8 se puede ver la interacción entre los factores tipo de dato y tipo de carga del motor y, si bien es cierto que para el tipo de dato vibraciones (eje X) se aprecia una ligero peor funcionamiento para motores con carga alta de trabajo, las diferencias no son significativas como se ve en la tabla 5.20.

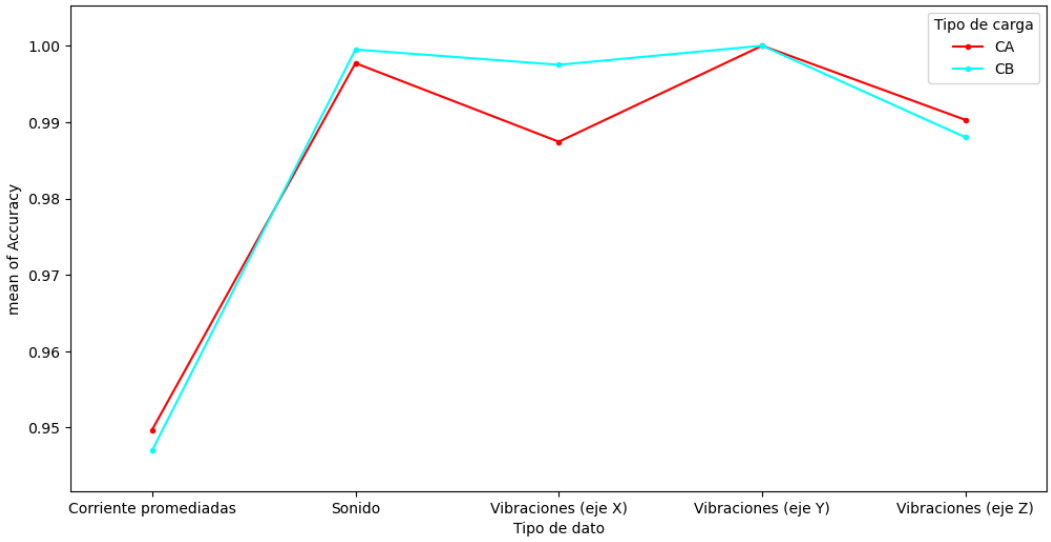

Figura 5.8: Interacción Tipo de dato * Tipo de carga

La observación de que el tipo de carga no es relevante en la calidad de los procedimientos diagnósticos es muy relevante desde el punto de vista industrial, puesto que lo habitual es que los motores estén funcionando a carga alta y esta observación dice que no es necesario bajar la carga del motor para detectar los fallos. Más aún, en la tabla 5.19 puede verse que para todos los variadores, es decir excepto en la alimentación directa a corriente, los

procedimientos diagnósticos en carga alta tienen a lo sumo el mismo número de variables implicadas que en carga baja. Por ejemplo, en carga baja con vibraciones en el eje X se utilizan las variables c_2, m_2 y x_r mientras que en carga alta solo se considera la variable m_2 y en vibraciones eje Z se utilizan c_2, x_r y cf en carga baja mientras que en carga alta son suficientes c_1 y x_r.

Capítulo 6

Conclusiones y futuras líneas de investigación

6.1. Conclusiones

A partir de los resultados obtenidos en el capítulo 5 se obtienen una serie de conclusiones que se pueden resumir como sigue:

- Los resultados de la sección 5.1.1 confirman la opinión dada por los expertos del departamento de Ingeniería Eléctrica de la Universidad de Valladolid. Es decir, a la hora de establecer reglas diagnósticas para detectar fallos en rodamientos de motores de inducción, no se observan diferencias significativas entre utilizar los datos de corriente promediados y utilizar los datos de corriente de cada una de las tres fases sin promediar. Por este motivo será preferible el uso de los datos promediados ya que se consigue una reducción de dimensión en el problema que conlleva reducir el riesgo de sobreajuste en los modelos obtenidos.

- En base a los resultados de la sección 5.1.4, cuando se consideran los datos globalmente sin tener en cuenta ni la carga del motor ni el tipo de alimentación, los datos de corriente promediados son significativamente peores para el diagnóstico de los fallos en motores. Además, los datos de sonido y vibraciones (eje Y) funcionan significativamente mejor que los de vibraciones (eje Z), por lo que estos tipos de datos, sonido y vibraciones (eje Y) son preferibles para el diagnóstico de los fallos. Más aún, los datos de vibraciones (eje Y) han obtenido el mejor resultado posible en todos los casos con una única variable (la media de la señal) que es además la más sencilla de obtener en el análisis de los datos. Esta observación ha hecho innecesario el análisis del desempeño de la combinación de diferentes fuentes de información en la clasificación que sí tiene interés en otras situaciones [25].

- Como se mencionó en los objetivos de este trabajo en la sección 1.2, es de especial

interés el análisis del desempeño de los datos de corriente puesto que son los que pueden obtenerse de forma más automática y menos invasiva. Según la conlusión previa este tipo de datos ha funcionado de forma significativamente peor que otros cuando no se han tenido en cuenta otros factores como el tipo de alimentación o la carga del motor. Sin embargo, según los resultados de la sección 5.2, el método de alimentación influye en el diagnóstico de los motores cuando se utiliza el tipo de datos corriente. En este caso, el tipo de datos corriente funciona significativamente mejor en motores alimentados por Red pero no funciona al nivel deseado cuando la alimentación se efectúa mediante variadores. De hecho cuando los variadores están presentes también hay diferencias significativas entre ellos, siendo el AB aquel para el que los datos de corriente funcionan peor. En cualquier caso, es interesante resaltar que cuando hay alimentación directa a la red los datos de corriente funcionan de manera óptima siendo una sola variable (de nuevo la media) suficiente para obtener el resultado.

- Otro objetivo, mencionado también en la sección 1.2, muy relevante desde el punto de vista industrial al que se ha dado respuesta en la sección 5.3 es el relativo a la influencia de la carga del motor en el diagnóstico de los motores. En base a esos resultados se puede afirmar que la carga del motor no influye en la capacidad de diagnóstico y que incluso en carga alta se puede obtener un buen diagnóstico con menos variables que en carga baja. Esto conlleva que no es necesario bajar la carga de trabajo del motor para diagnosticarlo correctamente lo que permite no perturbar el esquema de producción para efectuar los diagnósticos.

6.2. Futuras líneas de investigación

Este trabajo abre una serie de líneas de investigación que se enumeran a continuación.

- Obtención de más datos para mejorar la fiabilidad de los resultados, específicamente, de las partes del análisis de la influencia del tipo de alimentación y el tipo de carga del motor.

  El número de observaciones disponible para este trabajo ha sido bastante limitado por lo que sería conveniente incrementar su número para poder tener más confianza en las soluciones obtenidas y reducir más aún el riesgo de sobreajuste en los modelos.

- Mayor tiempo de entrenamiento para la obtención de mejores hiperparámetros óptimos.

  El incremento del número de observaciones propuesto en el punto anterior debería conllevar un incremento del tiempo de entrenamiento para obtener posibles valores de los hiperparámetros que mejoren las soluciones. El tiempo de computación de

los modelos considerados en este trabajo no ha sido grande por lo que tampoco es de esperar que el tiempo de entrenamiento aumente de forma excesiva.

- Uso de otros modelos para contrastar los resultados obtenidos.

  Hay que notar que en este trabajo se han considerado exclusivamente métodos boosting. No se han considerado otros métodos de amplio uso en la actualidad como las redes neuronales por dos motivos: el escaso número de observaciones y el hecho de que no han funcionado mejor que los métodos boosting en este contexto (véase [25]) pero podría ser buena idea darles una segunda oportunidad si puede incrementarse el número de observaciones.

  Por otro lado, vistos los resultados obtenidos en este trabajo, también puede ser una línea interesante de investigación el uso de modelos más simples como el análisis discriminante lineal o los árboles de decisión que permiten una interpretación más sencilla de las reglas diagnósticas que puedan obtenerse.

- Análisis conjunto de los tres factores: tipo de dato, tipo de alimentación y carga del motor.

  Habría sido interesante llevar a cabo este análisis global de los tres factores en este trabajo. Sin embargo, de nuevo el limitado número de observaciones no lo ha hecho posible puesto que había un número demasiado pequeño de datos en cada uno de los cruces de los factores. Es de esperar que este estudio pueda abordarse si se consigue incrementar el número de observaciones.

Bibliografía

[1] E. Commission. «Electric motors and variable speed drives.» (2019), dirección: https://commission.europa.eu/energy-climate-change-environment/standards-tools-and-labels/products-labelling-rules-and-requirements/energy-label-and-ecodesign/energy-efficient-products/electric-motors-and-variable-speed-drives_en. (último acceso: 03/06/2023).

[2] M. Eslamian-Koupaie, *Technical Language for Power Electric Technician Students*. oct. de 2016. DOI: 10.13140/RG.2.2.17658.70084.

[3] R. Schoen, B. Lin, T. Habetler, J. Schlag y S. Farag, «An unsupervised, on-line system for induction motor fault detection using stator current monitoring,» *IEEE Transactions on Industry Applications*, vol. 31, n.º 6, págs. 1280-1286, 1995. DOI: 10.1109/28.475698.

[4] N. Mehla y R. Dahiya, «An Approach of Condition Monitoring of Induction Motor Using MCSA,» *International Journal of Systems Applications, Engineering & Development*, vol. 1, págs. 13-17, 2007.

[5] G. B. Kliman y J. Stein, «Methods of Motor Current Signature Analysis,» *Electric Machines & Power Systems*, vol. 20, n.º 5, págs. 463-474, 1992. DOI: 10.1080/07313569208909609.

[6] R. C. Kryter y H. D. Haynes, «Condition monitoring of machinery using motor current signature analysis,» *Sound Vibrations*, vol. 23, n.º 9, págs. 14-21, 1989.

[7] Y.-S. Lee, J. Nelson, H. Scarton, D. Teng y S. Azizi-Ghannad, «An acoustic diagnostic technique for use with electric machine insulation,» *IEEE Transactions on Dielectrics and Electrical Insulation*, vol. 1, n.º 6, págs. 1186-1193, 1994. DOI: 10.1109/94.368645.

[8] A. Chiba, T. Fukao y M. A. Rahman, «Vibration Suppression of a Flexible Shaft with a Simplified Bearingless Induction Motor Drive,» en *Conference Record of the 2006 IEEE Industry Applications Conference Forty-First IAS Annual Meeting*, vol. 2, 2006, págs. 836-842. DOI: 10.1109/IAS.2006.256622.

[9] S. Daley, J. Hätönen y K. Tammi, «Instantaneous harmonic vibration control of a flexible rotor,» en *Proceedings of the 2006 International Symposium on Active Control of Sound and Vibration*, 2006.

[10] P. A. Delgado-Arredondo, D. Morinigo-Sotelo, R. A. Osornio-Rios, J. G. Avina-Cervantes, H. Rostro-Gonzalez y R. de Jesus Romero-Troncoso, «Methodology for fault detection in induction motors via sound and vibration signals,» *Mechanical Systems and Signal Processing*, vol. 83, págs. 568-589, 2017, ISSN: 0888-3270. DOI: https://doi.org/10.1016/j.ymssp.2016.06.032. dirección: https://www.sciencedirect.com/science/article/pii/S0888327016302151.

[11] P. Konar y P. Chattopadhyay, «Bearing fault detection of induction motor using wavelet and Support Vector Machines (SVMs),» *Applied Soft Computing*, vol. 11, n.º 6, págs. 4203-4211, 2011, ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2011.03.014. dirección: https://www.sciencedirect.com/science/article/pii/S1568494611001104.

[12] A. K. Al-Musawi, F. Anayi y M. Packianather, «Three-phase induction motor fault detection based on thermal image segmentation,» *Infrared Physics and Technology*, vol. 104, pág. 103 140, 2020, ISSN: 1350-4495. DOI: https://doi.org/10.1016/j.infrared.2019.103140. dirección: https://www.sciencedirect.com/science/article/pii/S1350449519304207.

[13] V. Fernandez-Cavero, J. Pons-Llinares, O. Duque-Perez y D. Morinigo-Sotelo, «Detection and quantification of bar breakage harmonics evolutions in inverter-fed motors through the dragon transform,» *ISA Transactions*, vol. 109, págs. 352-367, 2021.

[14] O. Duque, M. Perez y D. Morinigo, «Detection of Bearing Faults in Cage Induction Motors Fed by Frequency Converter using Spectral Analysis of Line Current,» en *IEEE International Conference on Electric Machines and Drives, 2005.*, 2005, págs. 17-22. DOI: 10.1109/IEMDC.2005.195695.

[15] O. Duque-Perez, C. Del Pozo-Gallego, D. Morinigo-Sotelo y W. Fontes Godoy, «Condition Monitoring of Bearing Faults Using the Stator Current and Shrinkage Methods,» *Energies*, vol. 12, n.º 17, 2019. dirección: https://www.mdpi.com/1996-1073/12/17/3392.

[16] J. A. y Jose Miguel Bolarín. «Gradient Boosting.» (2023), dirección: https://centicmurcia.github.io/curso-ciencia-datos/2.1-tree/3-gradient-boosting/. (último acceso: 01/06/2023).

[17] scikit-learn developers. «sklearn.ensemble.GradientBoostingClassifier.» (2023), dirección: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html. (último acceso: 17/05/2023).

[18] colaboradores de Wikipedia. «Gradient boosting.» (2023), dirección: https://en.wikipedia.org/w/index.php?title=Gradient_boosting&oldid=1156127757. (último acceso: 01/06/2023).

[19] T. Hastie, R. Tibshirani y J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2009.

[20] J. Brownlee. «A Gentle Introduction to the Bootstrap Method.» (2018), dirección: https://machinelearningmastery.com/a-gentle-introduction-to-the-bootstrap-method/. (último acceso: 01/06/2023).

[21] J. A. Rodrigo. «Análisis de varianza (ANOVA) con Python.» (2021), dirección: https://www.cienciadedatos.net/documentos/pystats09-analisis-de-varianza-anova-python. (último acceso: 02/06/2023).

[22] A. Nanda, B. B. Mohapatra, A. P. K. Mahapatra, A. P. K. Mahapatra y A. Mahapatra, «Multiple comparison test by Tukey's honestly significant difference (HSD): Do the confident level control type I error,» *International Journal of Statistics and Applied Mathematics*, 2021.

[23] V. Gurvich y M. Naumova, «Logical Contradictions in the One-Way ANOVA and Tukey–Kramer Multiple Comparisons Tests with More Than Two Groups of Observations,» *Symmetry*, vol. 13, n.º 8, 2021, ISSN: 2073-8994. DOI: 10.3390/sym13081387. dirección: https://www.mdpi.com/2073-8994/13/8/1387.

[24] J. W. Tukey, «Comparing individual means in the analysis of variance.,» *Biometrics*, vol. 5 2, págs. 99-114, 1949.

[25] M. Astorgano, «Diagnóstico de fallos de rodamientos en motores de inducción en estado estacionario mediante técnicas de boosting y redes neuronales,» Universidad de Valladolid, 2022. dirección: https://uvadoc.uva.es/handle/10324/57935.

[26] G. Novack. «Building a One Hot Encoding Layer with TensorFlow.» (2020), dirección: https://towardsdatascience.com/building-a-one-hot-encoding-layer-with-tensorflow-f907d686bf39. (último acceso: 01/06/2023).

Apéndice A

Gráficos de dispersión

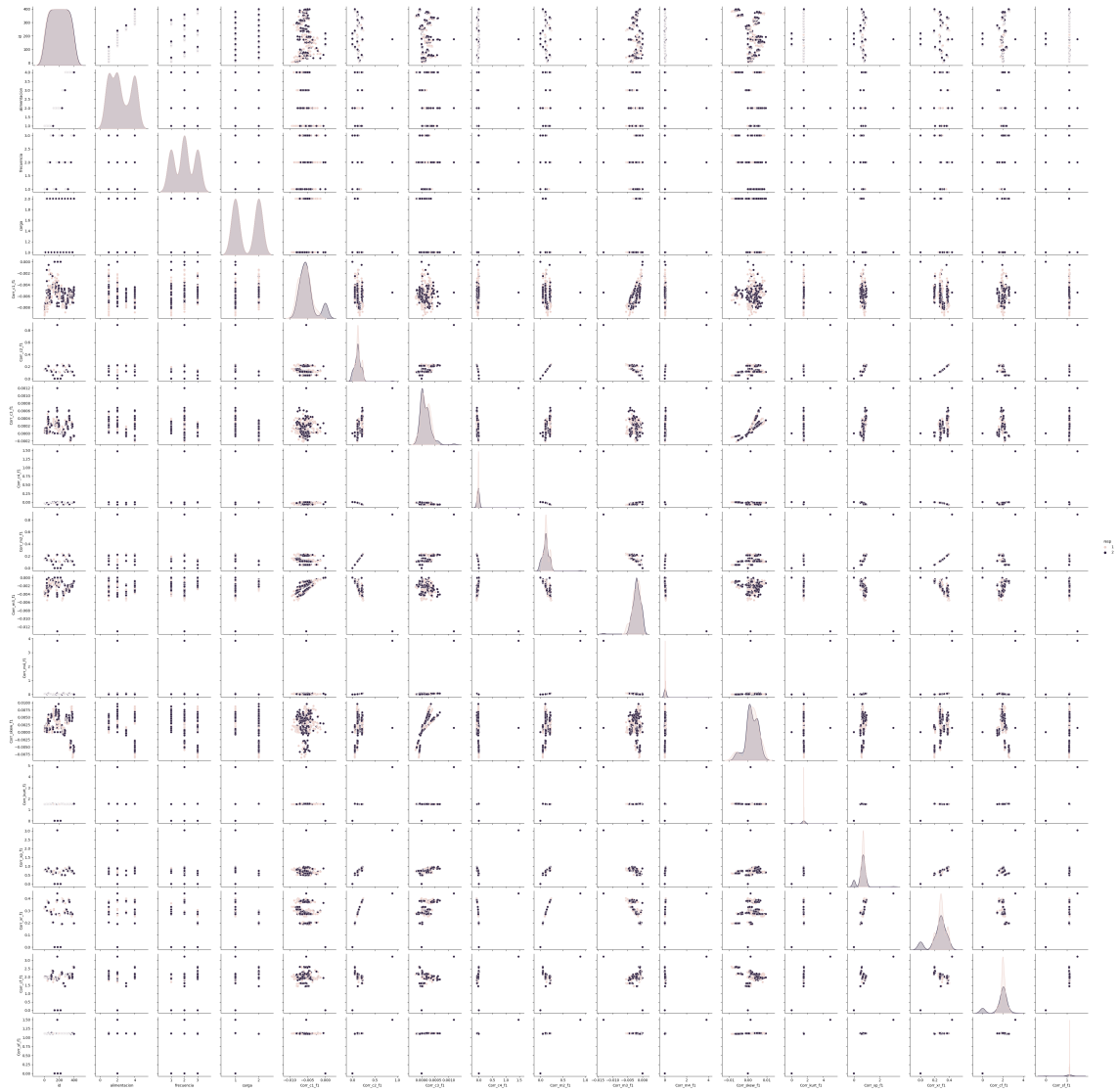

Figura A.1: Matriz de gráficos de dispersión de datos de corriente en fase 1

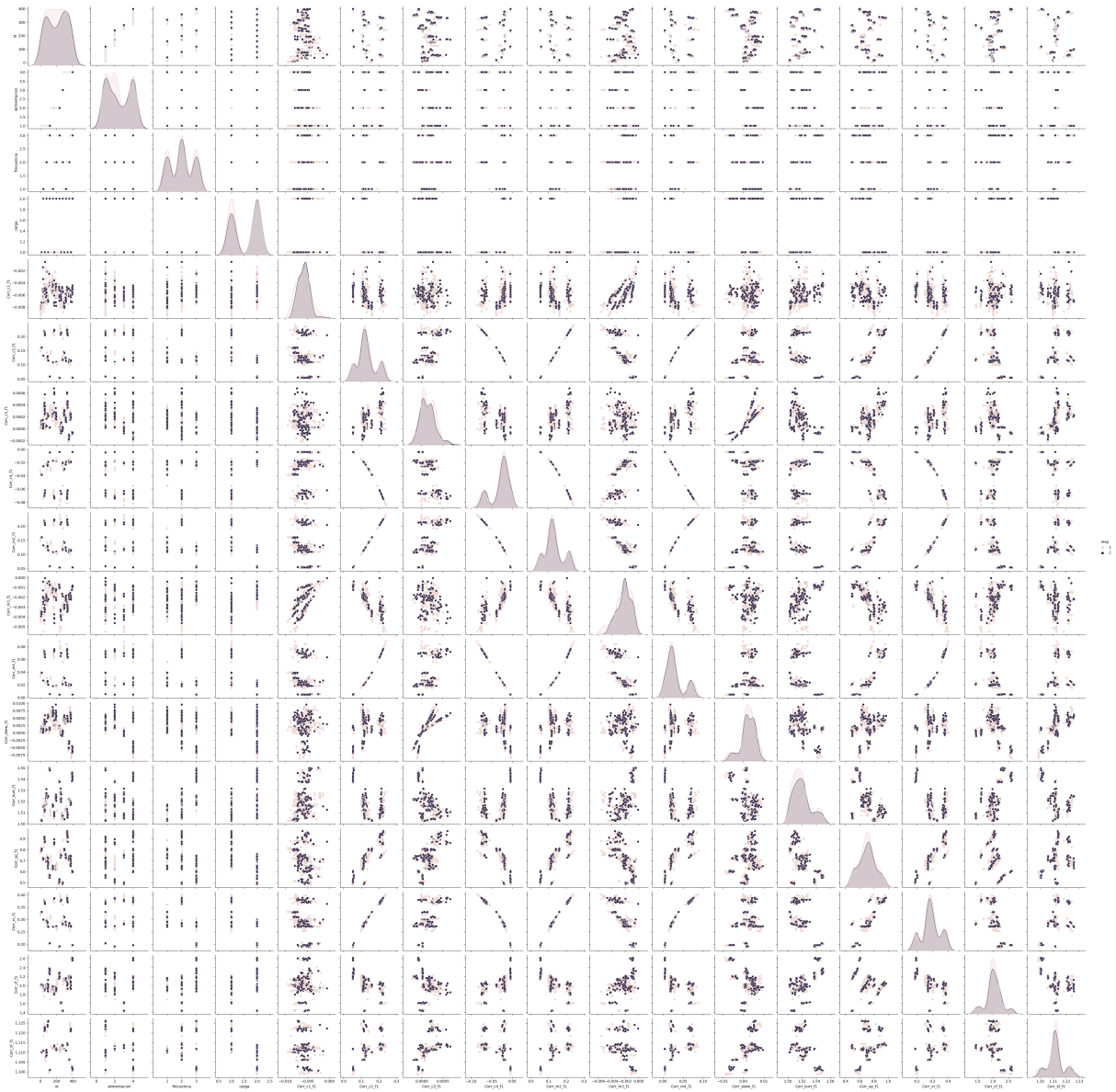

Figura A.2: Matriz de gráficos de dispersión de datos de corriente en fase 1 después del procesado

Figura A.3: Matriz de gráficos de dispersión de datos de sonido después del procesado

Apéndice B

Código del procesado de los datos

B.1. Código de lectura de datos

```
1  import sys
2  import os
3  import yaml
4  from yaml.loader import SafeLoader
5  from lib.util import util as u
6  import pandas as pd
7  import numpy as np
8  from functools import reduce
9  from sklearn.preprocessing import LabelBinarizer
10
11 # Lectura de la configuración
12 config = yaml.load(open("config/config.yml", 'r'), Loader=
       SafeLoader)
13
14 # Configuración en config.yml
15 DATA_PATH = config["data"]["path"]
16 FILETYPE = config["data"]["file"]["type"]
17 FILENAME_STRUCTURE = config["data"]["file"]["name_structure"]
18 DATA_TYPES = list(config["data"]["file"]["name_structure"].keys
       ())
19 CATEGORICAL_VARS = config["vars"]["categorical"]
20 NUMERIC_VARS = config["vars"]["numeric"]
21
22 # Obtención de los nombres de los ficheros de datos
```

```
23 DATA_FILES = [filename for dirname, dirs, files in os.walk(
       DATA_PATH) for filename in files if filename.endswith(
       FILETYPE)]
24
25 # Lectura de los ficheros de datos
26 def parse_name(filename: str) -> dict:
27     """Dado el nombre de un fichero extrae las variables
           categóricas de los datos que contiene
28     en su nombre.
29
30     Args:
31         filename (str): Nombre del fichero.
32
33     Returns:
34         dict: Variables y valores correspondientes extraidos.
35     """
36
37     detected_vars = {}
38
39     # Detecta [Corr, Sonido, Vibraciones]
40     detected_vars["datatype"] = u.substring_match(filename,
           FILENAME_STRUCTURE.keys())
41
42     structure = FILENAME_STRUCTURE[detected_vars["datatype"]]
43     for e in structure:
44         # Detecta variable categorica
45         var = u.substring_match(e, CATEGORICAL_VARS.keys())
46
47         if e == "frecuencia" and detected_vars["alimentacion"]
               == "Red": # Datos red 50 Hz
48             detected_vars["frecuencia"] = str(50)
49         elif var:
50             # Detecta valor de la variable categorica
51             detected_vars[var] = u.substring_match(filename,
                   CATEGORICAL_VARS[var])
52
53             filename = filename[len(detected_vars[var]):]
54             var = None
55         else:
```

```
56                filename = filename[len(e):]
57
58        return detected_vars
59
60    def extract_filename(metadata_list, datatype, fase=None, eje=
          None):
61        for (metadata, filename) in metadata_list:
62            if(metadata["datatype"] == datatype):
63                if(metadata["datatype"] == DATA_TYPES[0] and
                      metadata["fase"] == fase):
64                    return filename
65                if(metadata["datatype"] == DATA_TYPES[1]):
66                    return filename
67                if(metadata["datatype"] == DATA_TYPES[2] and
                      metadata["eje"] == eje):
68                    return filename
69
70    def load_data() -> pd.DataFrame:
71        """Carga los datos localizados en el directorio configurado
              en config.yml
72
73        Raises:
74            Exception: Si hay errores en el formato del directorio
                  o en el formato de los datos.
75
76        Returns:
77            pd.DataFrame: Data frame con las observaciones.
78        """
79
80        # Creación de las variables numéricas
81        # Corriente
82        numerical_vars_corr = []
83        for f in CATEGORICAL_VARS["fase"]:
84            for i in NUMERIC_VARS:
85                numerical_vars_corr.append("_".join([DATA_TYPES[0],
                      i, f]))
86        # Sonido
87        numerical_vars_sonido = []
88        for i in NUMERIC_VARS:
```

```
89              numerical_vars_sonido.append("_".join([DATA_TYPES
                    [1], i]))
90      # Vibraciones
91      numerical_vars_vibraciones = []
92      for a in CATEGORICAL_VARS["eje"]:
93          for i in NUMERIC_VARS:
94              numerical_vars_vibraciones.append("_".join([
                    DATA_TYPES[2], i, a]))
95
96      df = pd.DataFrame(columns=["id", "resp", "alimentacion", "
            frecuencia", "carga"] + numerical_vars_corr +
            numerical_vars_sonido + numerical_vars_vibraciones)
97      df.drop(columns=["Corr_mc4_f1", "Corr_m1_f1", "Corr_m6_f1",
             "Corr_am_f1", "Corr_mc4_f2", "Corr_m1_f2", "Corr_m6_f2"
            , "Corr_am_f2", "Corr_mc4_f3", "Corr_m1_f3", "Corr_m6_f3
            ", "Corr_am_f3"], inplace=True)
98      df.drop(columns=["Sonido_mc4", "Sonido_m1", "Sonido_m6", "
            Sonido_am"], inplace=True)
99      df.drop(columns=["Vibraciones_mc4_x", "Vibraciones_m1_x", "
            Vibraciones_m6_x", "Vibraciones_am_x", "
            Vibraciones_mc4_y", "Vibraciones_m1_y", "
            Vibraciones_m6_y", "Vibraciones_am_y", "
            Vibraciones_mc4_z", "Vibraciones_m1_z", "
            Vibraciones_m6_z", "Vibraciones_am_z"], inplace=True)
100
101     n_obs = 0
102     for alimentacion in CATEGORICAL_VARS["alimentacion"]:
103         for frecuencia in CATEGORICAL_VARS["frecuencia"]:
104             for carga in CATEGORICAL_VARS["carga"]:
105                 same_motors = list(filter(lambda metadata :
106                     metadata[0]["alimentacion"] == alimentacion
                            and
107                     metadata[0]["frecuencia"] == frecuencia and
108                     metadata[0]["carga"] == carga,
109                         zip(map(parse_name, DATA_FILES),
                                DATA_FILES)))
110
111                 if(len(same_motors) == 0):
112                     break
```

```
113                 if(len(same_motors) != 7):
114                     raise Exception("Datos incompletos.")
115
116                 # Lectura de archivos
117                 data_raw = []
118                 # Corriente
119                 fases = CATEGORICAL_VARS["fase"]
120                 data_raw.append(pd.read_excel(DATA_PATH + '/' +
                        alimentacion + '/' + extract_filename(
                    same_motors, DATA_TYPES[0], fase=fases[0]),
                    header=None, names=["_".join([DATA_TYPES[0],
                     i, fases[0]]) for i in NUMERIC_VARS] + ["
                    resp"]))
121                 data_raw.append(pd.read_excel(DATA_PATH + '/' +
                        alimentacion + '/' + extract_filename(
                    same_motors, DATA_TYPES[0], fase=fases[1]),
                    header=None, names=["_".join([DATA_TYPES[0],
                     i, fases[1]]) for i in NUMERIC_VARS] + ["
                    resp"]))
122                 data_raw.append(pd.read_excel(DATA_PATH + '/' +
                        alimentacion + '/' + extract_filename(
                    same_motors, DATA_TYPES[0], fase=fases[2]),
                    header=None, names=["_".join([DATA_TYPES[0],
                     i, fases[2]]) for i in NUMERIC_VARS] + ["
                    resp"]))
123
124                 # Sonido
125                 data_raw.append(pd.read_excel(DATA_PATH + '/' +
                        alimentacion + '/' + extract_filename(
                    same_motors, DATA_TYPES[1]), header=None,
                    names=["_".join([DATA_TYPES[1], i]) for i in
                     NUMERIC_VARS] + ["resp"]))
126
127                 # Vibraciones
128                 ejes = CATEGORICAL_VARS["eje"]
129                 data_raw.append(pd.read_excel(DATA_PATH + '/' +
                        alimentacion + '/' + extract_filename(
                    same_motors, DATA_TYPES[2], eje=ejes[0]),
                    header=None, names=["_".join([DATA_TYPES[2],
```

```
                i, ejes[0]]) for i in NUMERIC_VARS] + ["
                resp"]))
130             data_raw.append(pd.read_excel(DATA_PATH + '/' +
                alimentacion + '/' + extract_filename(
                same_motors, DATA_TYPES[2], eje=ejes[1]),
                header=None, names=["_".join([DATA_TYPES[2],
                i, ejes[1]]) for i in NUMERIC_VARS] + ["
                resp"]))
131             data_raw.append(pd.read_excel(DATA_PATH + '/' +
                alimentacion + '/' + extract_filename(
                same_motors, DATA_TYPES[2], eje=ejes[2]),
                header=None, names=["_".join([DATA_TYPES[2],
                i, ejes[2]]) for i in NUMERIC_VARS] + ["
                resp"]))
132
133             # Me quedo solo con dos valores respuesta
134             if(alimentacion == "Red"):
135                 new_data_raw = []
136                 for data in data_raw:
137                     d = data[(data["resp"] == 1) | (data["
                        resp"] == 6)]
138                     d["resp"] = d["resp"].map({1:1, 6:2})
139                     d["fakeid"] = range(20)
140                     d = d.set_index("fakeid")
141                     new_data_raw.append(d)
142                 data_raw = new_data_raw
143
144             response = data_raw[0]["resp"]
145             list(map(lambda x : x.drop(columns=x.columns
                [-1], inplace=True), data_raw))
146
147             # Elimina mc4 m1 m6 y am
148             list(map(lambda x : x.drop(columns=x.columns
                [[3, 5, 9, 12]], inplace=True), data_raw))
149
150             data_raw = pd.concat(data_raw, axis=1)
151
152             n = data_raw.shape[0]
153             common = pd.DataFrame()
```

```
154                     common["id"] = range(n_obs, n_obs + n)
155                     common["alimentacion"] = n*[alimentacion]
156                     common["frecuencia"] = n*[frecuencia]
157                     common["carga"] = n*[carga]
158                     common["resp"] = list(response)
159
160                     new_data = pd.concat([common, data_raw], axis
                            =1)
161
162                     df = pd.concat([df, new_data])
163
164                     n_obs += n
165
166     df.set_index("id")
167
168     return df
169
170 # Lectura de datos
171 df = load_data()
172 pretty_df = df.copy() # Save on separate dataframe
173 df2 = df.copy()
174
175 # Fases promediadas - df2
176 CORR_VAR_IDS = pd.Series(list(df2.columns[range(5, 44)]))
177 n_vars = len(CORR_VAR_IDS) // 3
178 CORR_VAR_IDS_F1 = CORR_VAR_IDS[0:n_vars]
179 CORR_VAR_IDS_F2 = CORR_VAR_IDS[n_vars:2*n_vars]
180 CORR_VAR_IDS_F3 = CORR_VAR_IDS[2*n_vars:3*n_vars]
181 new_corr_colnames = list(map(lambda x:x[:-3], list(
        CORR_VAR_IDS_F1)))
182 means = (np.array(df2[CORR_VAR_IDS_F1]) + np.array(df2[
        CORR_VAR_IDS_F2]) + np.array(df2[CORR_VAR_IDS_F3])) / 3
183 df2[new_corr_colnames] = means
184 df2 = df2.drop(columns=CORR_VAR_IDS)
185 df2["id"] = df2["id"].astype("int32")
186 df2["resp"] = df2["resp"].astype("int32")
187 alimentacion_encoder = LabelBinarizer().fit(df2["alimentacion"
        ])
```

```
188 alimentacion_types = list(map(lambda x:"alimentacion_"+x,
        alimentacion_encoder.classes_))
189 alimentacion = pd.DataFrame(alimentacion_encoder.transform(df2[
        "alimentacion"]), columns=alimentacion_types)
190 df2 = df2.drop(columns=["alimentacion"])
191 i = 2
192 for a in alimentacion_types:
193     df2.insert(i, a, alimentacion[a].array)
194     i += 1
195 frecuencia_encoder = LabelBinarizer().fit(df2["frecuencia"])
196 frecuencia_types = list(map(lambda x:"frecuencia_"+x,
        frecuencia_encoder.classes_))
197 frecuencia = pd.DataFrame(frecuencia_encoder.transform(df2["
        frecuencia"]), columns=frecuencia_types)
198 df2 = df2.drop(columns=["frecuencia"])
199 i = 6
200 for f in frecuencia_types:
201     df2.insert(i, f, frecuencia[f].array)
202     i += 1
203 carga_encoder = LabelBinarizer().fit(df2["carga"])
204 carga_types = list(map(lambda x:"carga_"+x, carga_encoder.
        classes_))
205 df2.insert(9, carga_types[1], carga_encoder.transform(df2["
        carga"]))
206 df2 = df2.drop(columns=["carga"])
207
208 # Cast de cada dato a su tipo (para aplicar modelos) - df
209 df["id"] = df["id"].astype("int32")
210 df["resp"] = df["resp"].astype("int32")
211 alimentacion_encoder = LabelBinarizer().fit(df["alimentacion"])
212 alimentacion_types = list(map(lambda x:"alimentacion_"+x,
        alimentacion_encoder.classes_))
213 alimentacion = pd.DataFrame(alimentacion_encoder.transform(df["
        alimentacion"]), columns=alimentacion_types)
214 df = df.drop(columns=["alimentacion"])
215 i = 2
216 for a in alimentacion_types:
217     df.insert(i, a, alimentacion[a].array)
218     i += 1
```

```
219 frecuencia_encoder = LabelBinarizer().fit(df["frecuencia"])
220 frecuencia_types = list(map(lambda x:"frecuencia_"+x,
      frecuencia_encoder.classes_))
221 frecuencia = pd.DataFrame(frecuencia_encoder.transform(df["
      frecuencia"]), columns=frecuencia_types)
222 df = df.drop(columns=["frecuencia"])
223 i = 6
224 for f in frecuencia_types:
225     df.insert(i, f, frecuencia[f].array)
226     i += 1
227 carga_encoder = LabelBinarizer().fit(df["carga"])
228 carga_types = list(map(lambda x:"carga_"+x, carga_encoder.
      classes_))
229 df.insert(9, carga_types[1], carga_encoder.transform(df["carga"
      ]))
230 df = df.drop(columns=["carga"])
```

B.2. Código de búsqueda de valores atípicos

```
1  import matplotlib.pyplot as plt
2  import seaborn as sns
3
4  execfile("src/data_reader/data_load_with_outliers.py")
5
6  common_vars = list(pretty_df.columns[range(5)])
7  CORR_VAR_IDS = pd.Series(common_vars + list(pretty_df.columns[
       range(5, 44)]))
8  SONIDO_VAR_IDS = pd.Series(common_vars + list(pretty_df.columns
       [range(44, 57)]))
9
10 corr_data_encoded = pretty_df[CORR_VAR_IDS].copy()
11 corr_data_encoded["alimentacion"] = corr_data_encoded["
       alimentacion"].map({"AB":1, "ABB":2, "Red":3, "WEG":4})
12 corr_data_encoded["frecuencia"] = corr_data_encoded["frecuencia
       "].map({"35":1, "50":2, "65":3, -1:-1})
13 corr_data_encoded["carga"] = corr_data_encoded["carga"].map({"
       CA":1, "CB":2})
14
```

```
15 # Fase 1 por separado
16 f1 = corr_data_encoded[list(CORR_VAR_IDS[list(range(5))]) +
       list(CORR_VAR_IDS[list(range(5, 18))])]
17 sns.pairplot(f1, hue="resp")
18
19 # Plot de outliers
20 plt.scatter(f1[f1["resp"] == 1]["Corr_c1_f1"], f1[f1["resp"] ==
       1]["Corr_kurt_f1"], c="green")
21 plt.scatter(f1[f1["resp"] == 2]["Corr_c1_f1"], f1[f1["resp"] ==
       2]["Corr_kurt_f1"], c="red")
22 plt.xlabel("Corr_c1_f1")
23 plt.ylabel("Corr_kurt_f1")
24 plt.legend(["Óptimo", "Dañado"], title="Estado")
25 plt.show()
26
27 # Identificadores de los outliers
28 f1[f1["Corr_kurt_f1"] < 1]["id"].sort_values()
29 f1[f1["Corr_kurt_f1"] > 4]["id"]
30
31 # Eliminación de los outliers
32 outliers = list(f1[f1["Corr_kurt_f1"] < 1]["id"]) + list(f1[f1[
       "Corr_kurt_f1"] > 4]["id"])
33 f1 = f1.loc[~f1["id"].isin(outliers)]
34 sonido_data_encoded = sonido_data_encoded.loc[~
       sonido_data_encoded["id"].isin(outliers)]
35
36 # Plot sin outliers
37 plt.scatter(f1[f1["resp"] == 1]["Corr_c1_f1"], f1[f1["resp"] ==
       1]["Corr_kurt_f1"], c="green")
38 plt.scatter(f1[f1["resp"] == 2]["Corr_c1_f1"], f1[f1["resp"] ==
       2]["Corr_kurt_f1"], c="red")
39 plt.xlabel("Corr_c1_f1")
40 plt.ylabel("Corr_kurt_f1")
41 plt.legend(["Óptimo", "Dañado"], title="Estado")
42 plt.show()
43
44 sonido_data_encoded = pretty_df[SONIDO_VAR_IDS].copy()
45 sonido_data_encoded["alimentacion"] = sonido_data_encoded["
       alimentacion"].map({"AB":1, "ABB":2, "Red":3, "WEG":4})
```

```
46 sonido_data_encoded["frecuencia"] = sonido_data_encoded["
       frecuencia"].map({"35":1, "50":2, "65":3, -1:-1})
47 sonido_data_encoded["carga"] = sonido_data_encoded["carga"].map
       ({"CA":1, "CB":2})
48 sns.pairplot(sonido_data_encoded, hue="resp")
49
50 # Plot sonido
51 plt.scatter(sonido_data_encoded[sonido_data_encoded["resp"] ==
       1]["alimentacion"], sonido_data_encoded[sonido_data_encoded[
       "resp"] == 1]["Sonido_c1"], c="green")
52 plt.scatter(sonido_data_encoded[sonido_data_encoded["resp"] ==
       2]["alimentacion"], sonido_data_encoded[sonido_data_encoded[
       "resp"] == 2]["Sonido_c1"], c="red")
53 plt.xlabel("alimentacion")
54 plt.ylabel("Sonido_c1")
55 plt.legend(["Óptimo", "Dañado"], title="Estado")
56 plt.show()
```

Apéndice C

Código del análisis de los datos

En este anexo se muestra el código utilizado para el análisis de un subconjunto de datos y variables explicativas concreto. Para todos los que aparecen en resultados el código es el mismo pero variando los datos X e y de las primeras líneas de cada sección.

C.1. Librerías utilizadas para el análisis de los datos

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import pingouin as pg
from sklearn.model_selection import StratifiedKFold,
    train_test_split
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.metrics import accuracy_score
from sklearn.utils import resample
import scikit_posthocs as sp
from statsmodels.graphics.factorplots import interaction_plot
```

C.2. Validación cruzada sobre un modelo

```
RANDOM_SEED = 2023

# Función para hacer validación cruzada sobre modelos
def apply_model(X, y, model, kwargs, random_seed=RANDOM_SEED):
```

```
    results = pd.DataFrame(columns=["fold", "accuracy"] + list(
        X.columns))
    for i, (train_index, test_index) in enumerate(
        StratifiedKFold(n_splits=5, shuffle=True, random_state=
        random_seed).split(X, y)):
        X_train = X.iloc[train_index]
        X_test = X.iloc[test_index]
        y_train = y.iloc[train_index]
        y_test = y.iloc[test_index]

        trained_model = model(random_state=random_seed, **
            kwargs).fit(X_train, y_train)
        y_test_pred = trained_model.predict(X_test)

        results.loc[i] = [i+1, accuracy_score(y_test,
            y_test_pred)] + list(trained_model.
            feature_importances_)

    return results.mean()
```

C.3. Búsqueda de hiperparámetros óptimos

```
# Datos a utilizar por el modelo
X = df.drop(columns=["id", "resp"])
CORR_INDICES = X.columns[8:47]
X = X[CORR_INDICES]
y = df["resp"]

# Hold-out train/test split
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=RANDOM_SEED, stratify=y)

# Hiperparámetros a optimizar (por defecto)
best_params = {
    # Tree parameters
    "min_samples_split": 2,
    "min_samples_leaf": 1,
    "max_depth": 3,
```

```
    "max_leaf_nodes": None,
    # Boosting parameters
    "learning_rate": 1,
    "n_estimators": 100,
    "subsample": 1
}

# Modelo inicial
model = GradientBoostingClassifier(random_state=RANDOM_SEED, **
    best_params).fit(X_train, y_train)
y_pred = model.predict(X_test)
print("Accuracy modelo inicial:", accuracy_score(y_test, y_pred
    ))

# Hiperparámetros a probar
param_candidates = {
    # Tree parameters
    "min_samples_split": list(range(2, 5, 1)),
    "min_samples_leaf": range(1, 5, 1),
    "max_depth": range(1, 10, 1),
    "max_leaf_nodes": range(2, 20, 2),
    # Boosting parameters
    "learning_rate": np.arange(0.1, 1, 0.1),
    "n_estimators": range(50, 300, 50),
    "subsample": np.arange(0.1, 1.1, 0.1)
}

# Tunning del modelo
for param in best_params:
    results = pd.DataFrame(columns=["accuracy"])
    for i, candidate in enumerate(param_candidates[param]):
        best_params[param] = candidate

        accuracy = apply_model(X_train, y_train,
            GradientBoostingClassifier, best_params)["accuracy"]

        results.loc[i] = [accuracy]

```

```
52      best_params[param] = param_candidates[param][results.idxmax
            ()["accuracy"]]
53
54  # Modelo resultante
55  model = GradientBoostingClassifier(random_state=RANDOM_SEED, **
        best_params).fit(X_train, y_train)
56  y_pred = model.predict(X_test)
57  print("Accuracy modelo tuneado:", accuracy_score(y_test, y_pred
        ))
```

C.4. Obtención de importancias del modelo ordenadas

```
1  importances = pd.DataFrame(columns=["Var", "Importance"])
2  importances["Var"] = model.feature_names_in_
3  importances["Importance"] = model.feature_importances_
4  var_candidates = importances.sort_values(by="Importance")["Var"
       ]
5  importances.sort_values(by="Importance")
```

C.5. Reducción de variables

```
1  X = df2.drop(columns=["id", "resp"])
2  CORR_INDICES = X.columns[60:]
3  y = df2["resp"]
4  X = X[CORR_INDICES]
5  CORR_VARS = list(X.columns)
6
7  N_REP = 10
8
9  global_results = pd.DataFrame(columns=["Accuracy", "Vars", "
       N_vars"])
10 for j, var in enumerate(var_candidates[:-1]):
11     CORR_VARS.remove(var)
12
13     # Datos a utilizar por el modelo
```

```
    X = X[CORR_VARS]

    accuracy_sum = 0
    for i in range(N_REP):
        # 5-XV
        results = apply_model(X, y, GradientBoostingClassifier,
            best_params, RANDOM_SEED+j+i)
        accuracy_sum += results["accuracy"]
    accuracy_mean = accuracy_sum / N_REP

    global_results.loc[j] = [accuracy_mean, CORR_VARS.copy(),
        len(CORR_VARS)]
global_results.sort_values(by="Accuracy", ascending=False)

global_results.plot.line(x="N_vars", y="Accuracy", xlabel="
    Número de variables explicativas", ylabel="Tasa de acierto")
```

C.6. Repeticiones *Bootstrap* del modelo

```
# Datos a utilizar por el modelo
X = df2.drop(columns=["id", "resp"])
SELECTED_VARS = ["Corr_c1", "Corr_c3", "Corr_kurt", "Corr_sf"]
X2 = X[SELECTED_VARS]
y2 = df2["resp"]

N_REP = 50

results = pd.DataFrame(columns=["Rep", "Accuracy"])
for i in range(N_REP):
    # Remuestreo Bootstrap con remplazamiento
    X, y = resample(X2, y2, random_state=RANDOM_SEED+i)

    # Hold-out train/test split
    X_train, X_test, y_train, y_test = train_test_split(
        X, y, test_size=0.2, random_state=RANDOM_SEED+i,
            stratify=y)

    # Modelo inicial
```

```
19      model = GradientBoostingClassifier(random_state=RANDOM_SEED
            +i, **best_params).fit(X_train, y_train)
20      y_pred = model.predict(X_test)
21
22      results.loc[i] = [i, accuracy_score(y_test, y_pred)]
```

C.7. Gráficos de dispersión

```
1  plt.scatter(pretty_df[df2["resp"] == 1]["resp"], df2[df2["resp
       "] == 1]["Sonido_c2"], c="green")
2  plt.scatter(pretty_df[df2["resp"] == 2]["resp"], df2[df2["resp
       "] == 2]["Sonido_c2"], c="red")
3  plt.xlabel("Estado del motor")
4  plt.ylabel("Sonido_c2")
5  plt.legend(["Óptimo", "Dañado"], title="Estado")
6  plt.show()
```

C.8. Comparación de modelos con ANOVA de 1 factor

Se ha de llamar a la función con una lista de tuplas que contenga el *dataset results* obtenidos de las repeticiones *Bootstrap* y el nivel del factor de dicho *dataset*.

```
1  def anova_comparison(model_list) -> None:
2      data, model_name = model_list[0]
3
4      # Data preparation
5      concat = data.copy()
6      concat["Modelo"] = model_name
7      concat = concat.drop("Rep", axis=1)
8
9      for (data, model_name) in model_list[1:]:
10         # Data preparation
11         test1 = data.copy()
12         test1["Modelo"] = model_name
13         test1 = test1.drop("Rep", axis=1)
14
```

```
        # fig, axs = plt.subplots(1, 2, figsize=(8, 7))
        # pg.qqplot(concat.loc[concat.Modelo==var_name1, 'Accuracy'], dist='norm', ax=axs[0])
        # axs[0].set_title(var_name1)
        # pg.qqplot(concat.loc[concat.Modelo==var_name2, 'Accuracy'], dist='norm', ax=axs[1])
        # axs[1].set_title(var_name2)
        # plt.tight_layout()

        concat = pd.concat([concat, test1])

    fig, ax = plt.subplots(1, 1, figsize=(12, 6))
    sns.boxplot(x="Modelo", y="Accuracy", data=concat, ax=ax)
    sns.swarmplot(x="Modelo", y="Accuracy", data=concat, color='black', alpha = 0.5, ax=ax)

    print(pg.anova(data=concat, dv='Accuracy', between='Modelo', detailed=True))

    return concat
```

C.9. Comparación de modelos con ANOVA de 2 factores

Se ha de llamar a la función con una lista de tuplas que contenga el *dataset results* obtenidos de las repeticiones *Bootstrap*, el nivel del factor 1 de dicho *dataset* y el factor 2 de dicho *dataset*. Además, se introducen también los nombres de cada factor como argumentos.

```
def anova2_comparison(model_list, factor1_name, factor2_name) -> None:
    data, factor1, factor2 = model_list[0]

    # Data preparation
    concat = data.copy()
    concat[factor1_name] = factor1
    concat[factor2_name] = factor2
    concat = concat.drop("Rep", axis=1)
```

```
9
10      for (data, factor1, factor2) in model_list[1:]:
11          # Data preparation
12          test1 = data.copy()
13          test1[factor1_name] = factor1
14          test1[factor2_name] = factor2
15          test1 = test1.drop("Rep", axis=1)
16
17          concat = pd.concat([concat, test1])
18
19      print(pg.anova(data=concat, dv='Accuracy', between=[
            factor1_name, factor2_name], detailed=True))
20
21      return concat
```

C.10. Análisis *post-hoc* utilizando el test de Tukey

Partiendo de los *datasets* resultado de los ANOVA, se utiliza la siguiente orden indicando en *group_col* el nombre del factor.

```
1 sp.posthoc_tukey(result_df, val_col="Accuracy", group_col="
    Modelo")
```

Si es necesario crear el factor interacción se utiliza el siguiente código:

```
1 result_df["Interaccion"] = result_df["Nombre␣factor␣1"] + "␣x␣"
    + result_df["Nombre␣Factor␣2"]
```

C.11. Gráfico de interacción

Usando el *dataset* resultado de aplicar la función ANOVA, se usa el siguiente código:

```
1 fig, ax = plt.subplots(figsize=(12, 6))
2 fig = interaction_plot(
3     x        = result_df["Nombre␣Factor␣1"],
4     trace    = result_df["Nombre␣Factor␣2"],
5     response = result_df["Accuracy"],
6     ax       = ax,
7 )
```

C.12. Tablas *post-hoc*

```
def to_pretty_table(anova_result, post_hoc_result, factor_name):
    group_levels = anova_result[factor_name].unique()
    count = np.array(anova_result.groupby(factor_name).count()["Accuracy"])
    means = np.array(anova_result.groupby(factor_name).mean())
    p_values = np.array(post_hoc_result)

    summary1 = pd.DataFrame(columns=["Group Levels", "Count", "Mean"])
    summary1["Group Levels"] = group_levels
    summary1["Count"] = count
    summary1["Mean"] = means

    diff_matrix = []
    for i in range(len(group_levels)):
        row = []
        for j in range(len(group_levels)):
            row.append((means[i] - means[j])[0])
        diff_matrix.append(row)
    diff_matrix = np.array(diff_matrix)

    summary2 = pd.DataFrame(columns=["Contrast", "p-value", "Difference"])
    row_index = 0
    for i in range(len(group_levels)):
        for j in range(i+1, len(group_levels)):
            summary2.loc[row_index] = [group_levels[i] + " - "
                + group_levels[j],
                                       p_values[i, j],
                                       diff_matrix[i, j]]
            row_index = row_index + 1

    return (summary1, summary2)
```

Apéndice D

Obtención de los modelos por tipo de alimentación

D.1. Análisis solo con los datos de corriente

D.1.1. Modelos separados para cada tipo de alimentación

Se decide realizar modelos separados para las observaciones con distinto tipo de alimentación para ver si realmente puede haber diferencias o no. Cabe mencionar que los resultados de esta sección son de menor fiabilidad debido a la baja cantidad de observaciones que hay para cada uno de los tipos de alimentación que va desde 40 observaciones para las alimentadas por Red hasta las 120 para las alimentados por variador AB.

Alimentación AB

En primer lugar, se realiza un ajuste solo con las observaciones alimentadas por variador AB. Y, seguidamente, se lleva a cabo un procedimiento de reducción de variables cuyo resultado se encuentra en la figura D.1. En este caso, se puede reducir el número de variables a 6 manteniendo la tasa de acierto igual de buena que con más. El modelo resultante sería sólo con las variables: c_1, c_3, *kurt*, x_r, *cf* y *sf* de corriente promediadas.

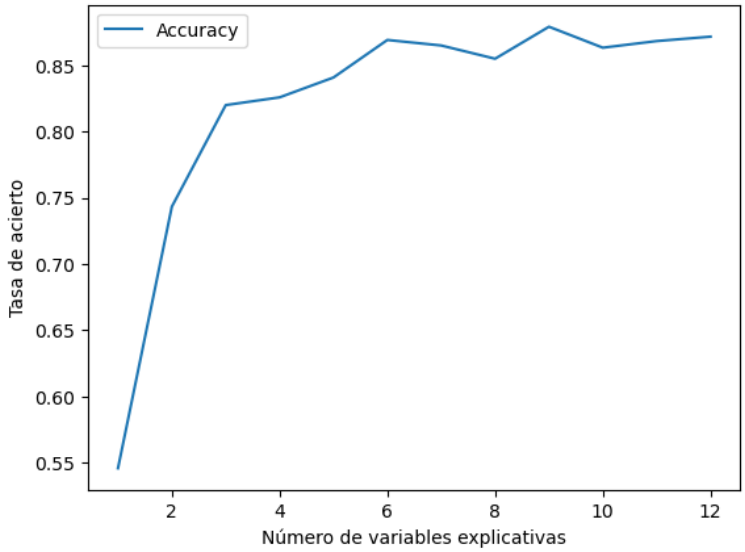

Figura D.1: Tasa de acierto vs. número de variables explicativas de corriente promediadas en observaciones alimentadas por variador AB

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla D.1.

Modelo	**Repeticiones**	**Tasa de acierto promedio**	**Desviación típica**
c_1, c_3, kurt, x_r, cf y sf	50	0,9175	0,0568

Tabla D.1: Resumen de la estabilidad del modelo obtenido con variables de corriente promediadas en observaciones alimentadas por variador AB

Alimentación ABB

Ahora, se lleva a cabo un modelo solo con las observaciones alimentadas por variadores ABB. Y, seguidamente, se lleva a cabo una reducción de variables siguiendo el mismo procedimiento que ya se ha utilizado cuyo resultado se encuentra en la figura D.2. Como podemos apreciar en los resultados, podemos reducir el número de variables a 3: c_1, c_4 y sf manteniendo la tasa de acierto elevada.

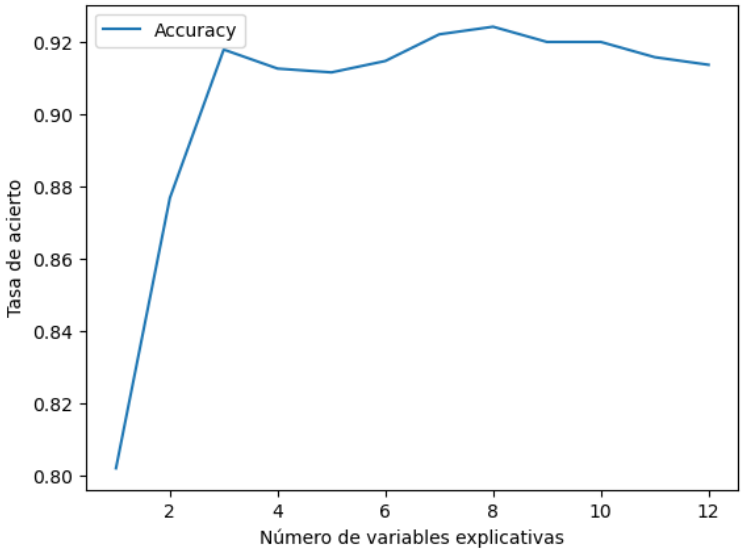

Figura D.2: Tasa de acierto vs. número de variables explicativas de corriente promediadas en observaciones alimentadas por variador ABB

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla D.2.

Modelo	**Repeticiones**	**Tasa de acierto promedio**	**Desviación típica**
c_1, c_4 y sf	50	0,9579	0,0521

Tabla D.2: Resumen de la estabilidad del modelo obtenido con variables de corriente promediadas en observaciones alimentadas por variador ABB

Alimentación Red

Ahora, se lleva a cabo un modelo solo con las observaciones alimentadas por Red obteniéndose un 100 % de tasa de acierto. Puesto que obtenemos un 100 % de tasa de acierto, realizamos una búsqueda de hiperparámetros en busca de un modelo más sencillo que lo mantenga. Los hiperparámetros óptimos encontrados se encuentran en la tabla D.3.

Hiperparámetro	Valor
Mínimo de observaciones para dividir un nodo	1
Mínimo de observaciones en nodo hoja	1
Máximo de profundidad del árbol	1
Máximo de nodos hoja	2
Tasa de aprendizaje	0.1
Número de árboles	1
Porcentaje de muestra para entrenar cada árbol	50 %

Tabla D.3: Hiperparámetros del modelo optimizado con variables de corriente promediadas en observaciones alimentadas por Red

En vista a los resultados en la tabla D.3, podemos apreciar como el modelo es un único árbol sencillo. Por ello, se decide realizar una reducción de variables en la que se encuentra que solo con la variable c_1 ya se alcanza una tasa de acierto del 100 %. Llevando a cabo un estudio de esta variable, se descubre que separa linealmente las observaciones alimentadas por Red tal y como se aprecia en la figura D.3.

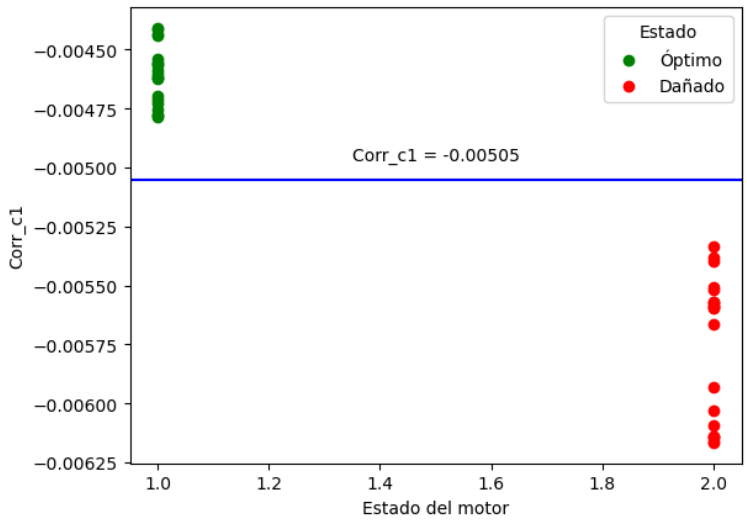

Figura D.3: Corr_c1 vs. Estado del motor (observaciones alimentadas por Red)

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla D.4.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
c_1	50	1	0

Tabla D.4: Resumen de la estabilidad del modelo obtenido con variables de corriente promediadas en observaciones alimentadas por Red

Alimentación WEG

Por último, se lleva a cabo un modelo solo con las observaciones alimentadas por variadores WEG. Y, seguidamente, se lleva a cabo una reducción de variables siguiendo el mismo procedimiento que ya se ha utilizado.

Como podemos apreciar en la figura D.4, se pueden reducir hasta un modelo con 7 variables sin perder tasa de acierto. El modelo resultante sería manteniendo las variables: c_1, c_3, m_2, m_3, *skew*, *cf* y *sf*.

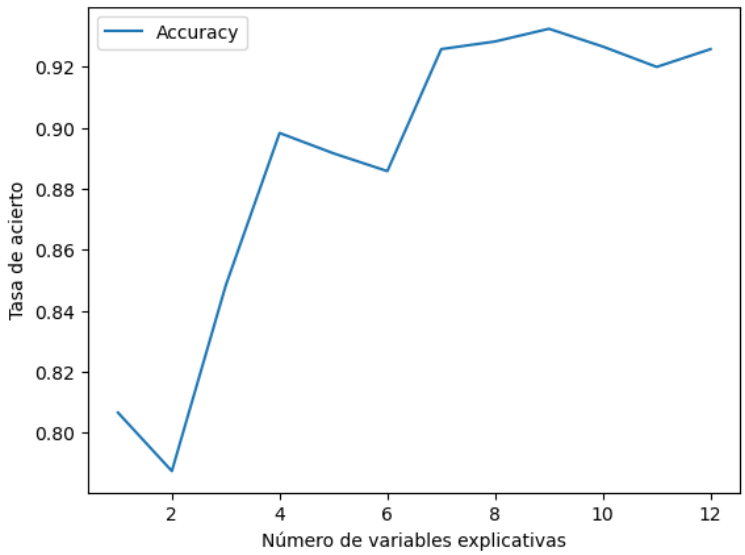

Figura D.4: Tasa de acierto vs. número de variables explicativas de corriente promediadas en observaciones alimentadas por variador WEG

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla D.5.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
c_1, c_3, m_2, m_3, skew, cf y sf	50	0,9533	0,0481

Tabla D.5: Resumen de la estabilidad del modelo obtenido con variables de corriente promediadas en observaciones alimentadas por variador WEG

D.2. Análisis solo con los datos de sonido

Puesto que al llevar a cabo el análisis sin diferenciar por tipo de alimentación ya se encontró una variable que separaba perfectamente ambos conjuntos, los modelos que se aplicarán para cada tipo de alimentación sera con esa variable (c_2).

D.2.1. Modelos separados para cada tipo de alimentación

Alimentación AB

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla D.6.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
c_2	50	1	0

Tabla D.6: Resumen de la estabilidad del modelo obtenido con variables de sonido en observaciones alimentadas por variador AB

Alimentación ABB

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla D.7.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
c_2	50	1	0

Tabla D.7: Resumen de la estabilidad del modelo obtenido con variables de sonido en observaciones alimentadas por variador ABB

Alimentación Red

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla D.8.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
c_2	50	0,9925	0,0300

Tabla D.8: Resumen de la estabilidad del modelo obtenido con variables de sonido en observaciones alimentadas por Red

Alimentación WEG

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla D.9.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
c_2	50	1	0

Tabla D.9: Resumen de la estabilidad del modelo obtenido con variables de sonido en observaciones alimentadas por variador WEG

D.3. Análisis solo con los datos de vibraciones

Para cada uno de los ejes (X, Y, Z) se lleva a cabo modelos separados para cada tipo de alimentación.

D.3.1. Eje X

Alimentación AB

En primer lugar, se lleva a cabo un modelo solo con las observaciones alimentadas por variadores AB. Seguidamente, se realiza una reducción del número de variables. Como podemos apreciar en la figura D.5, el mejor compromiso de tasa de acierto por número de variables explicativas incluidas en el modelo se encuentra con 6 variables.

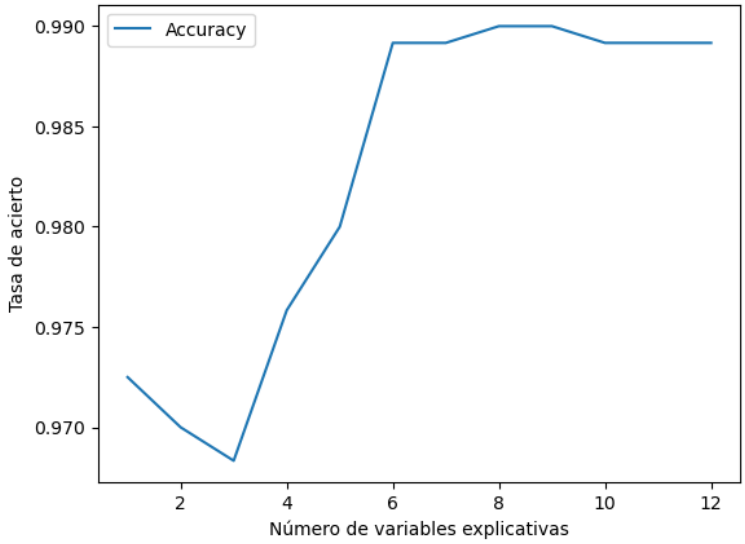

Figura D.5: Tasa de acierto vs. número de variables explicativas de vibraciones (eje X) en observaciones alimentadas por variador AB

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla D.10.

Modelo	**Repeticiones**	**Tasa de acierto promedio**	**Desviación típica**
c_2, c_4, m_2, m_4, x_r y sf	50	0,9933	0,0176

Tabla D.10: Resumen de la estabilidad del modelo obtenido con variables de vibraciones (eje X) en observaciones alimentadas por variador AB

Alimentación ABB

Ahora, se lleva a cabo un modelo solo con las observaciones alimentadas por variadores ABB. Seguidamente, se realiza una reducción del número de variables. De este modo, se obtiene que con tan solo la variable m_4 se alcanza el 100 % de tasa de acierto. Llevando a cabo un pequeño análisis descriptivo, se descubre que dicha variable es separadora lineal para los dos estados del motor tal y como se aprecia en la figura D.6. El segundo de los gráficos de la figura D.6 es una ampliación del primero en la que se añade la recta que separa los datos linealmente.

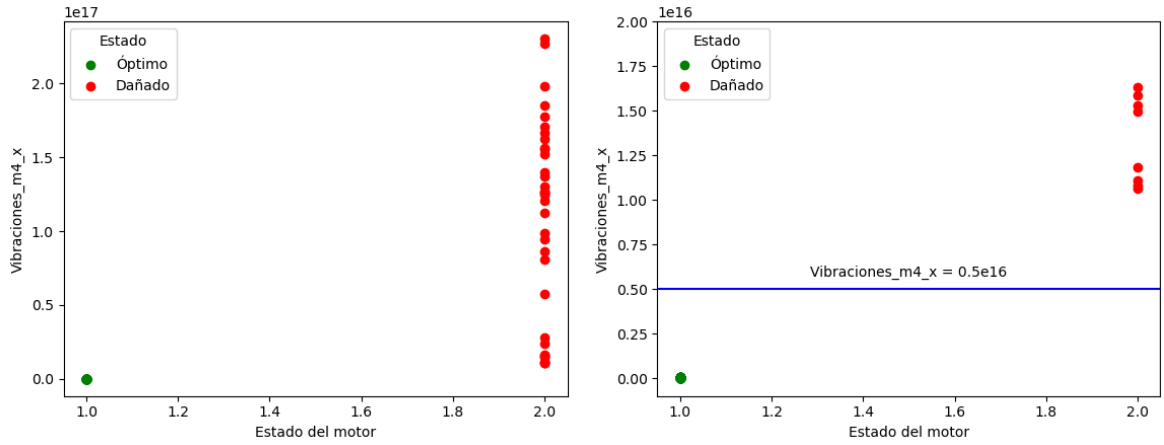

Figura D.6: Vibraciones_m4_x vs. estado del motor

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla D.11.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
m_4	50	1	0

Tabla D.11: Resumen de la estabilidad del modelo obtenido con variables de vibraciones (eje X) en observaciones alimentadas por variador ABB

Alimentación Red

Ahora, se lleva a cabo un modelo solo con las observaciones alimentadas por Red. Seguidamente, se lleva a cabo una reducción del número de variables. Como podemos apreciar en el gráfico de la figura D.7, con 3 variables obtenemos la máxima tasa de acierto posible con las variables: c_1, m_4 y x_r.

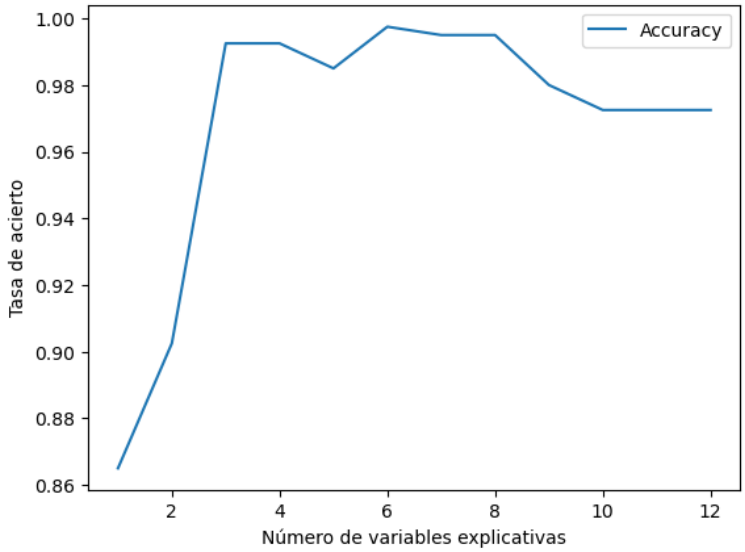

Figura D.7: Tasa de acierto vs. número de variables explicativas de vibraciones (eje X) en observaciones alimentadas por Red

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla D.12.

Modelo	**Repeticiones**	**Tasa de acierto promedio**	**Desviación típica**
c_1, m_4 y x_r	50	0,99	0,0343

Tabla D.12: Resumen de la estabilidad del modelo obtenido con variables de vibraciones (eje X) en observaciones alimentadas por Red

Alimentación WEG

Por último, para la alimentación por variador WEG, se lleva a cabo un modelo solo con las observaciones alimentadas por variadores WEG. Seguidamente, llevamos a cabo el mismo tipo de reducción de número de variables que venimos haciendo donde se descubre que con solo la variable m_2 se alcanza el 100 % de tasa de acierto por lo que realizamos un pequeño estudio sobre ella. El resultado se muestra en la figura D.8. En ella, se ve que esta variable separa linealmente los dos tipos de motores. El segundo de los gráficos de la figura D.8 es una ampliación del primero en la que se añade la recta que separa los datos linealmente.

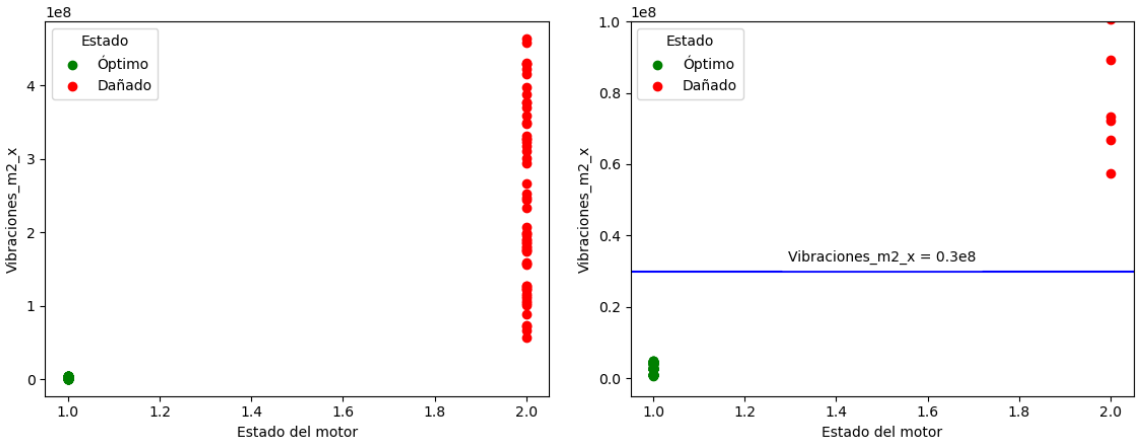

Figura D.8: Vibraciones_m2_x vs. estado del motor

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla D.13.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
m_2	50	1	0

Tabla D.13: Resumen de la estabilidad del modelo obtenido con variables de vibraciones (eje X) en observaciones alimentadas por variador WEG

D.3.2. Eje Y

Puesto que al llevar a cabo el análisis sin diferenciar por tipo de alimentación ya se encontró una variable que separaba perfectamente ambos conjuntos, los modelos que se aplicarán para cada tipo de alimentación sera con esa variable (c_1).

Alimentación AB

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla D.14.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
c_1	50	1	0

Tabla D.14: Resumen de la estabilidad del modelo obtenido con variables de vibraciones (eje Y) en observaciones alimentadas por variador AB

Alimentación ABB

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla D.15.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
c_1	50	1	0

Tabla D.15: Resumen de la estabilidad del modelo obtenido con variables de vibraciones (eje Y) en observaciones alimentadas por variador ABB

Alimentación Red

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla D.16.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
c_1	50	1	0

Tabla D.16: Resumen de la estabilidad del modelo obtenido con variables de vibraciones (eje Y) en observaciones alimentadas por Red

Alimentación WEG

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla D.17.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
c_1	50	1	0

Tabla D.17: Resumen de la estabilidad del modelo obtenido con variables de vibraciones (eje Y) en observaciones alimentadas por variador WEG

D.3.3. Eje Z

Alimentación AB

En primer lugar, se realiza una reducción del número de variables. Se obtiene que con tan solo la variable c_1 ya se alcanza un 100 % de tasa de acierto por lo que se decide

estudiar dicha variable. Como podemos apreciar en la figura D.9, esta variable separa linealmente los dos tipos de motores.

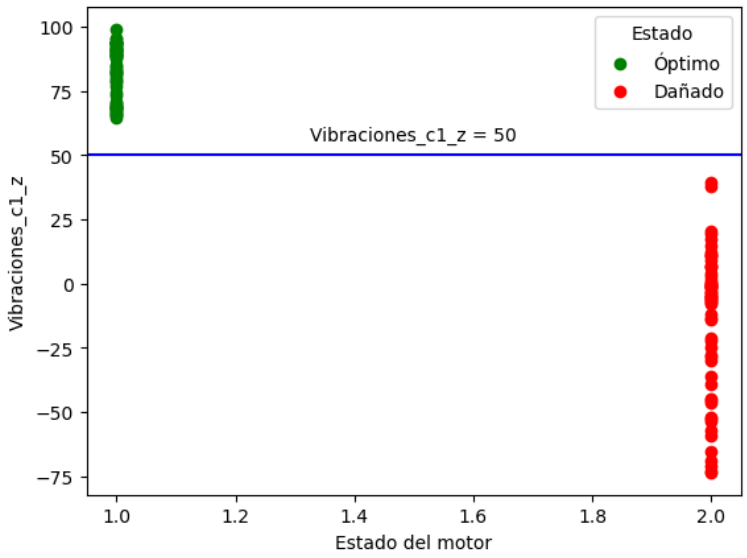

Figura D.9: Vibraciones_c1_z vs. estado del motor

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla D.18.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
c_1	50	1	0

Tabla D.18: Resumen de la estabilidad del modelo obtenido con variables de vibraciones (eje Z) en observaciones alimentadas por variador AB

Alimentación ABB

En este caso, solo con la variable m_4 se obtiene una tasa de acierto del 100 %. Si estudiamos dicha variable nos encontramos con que separa linealmente las dos categorías de motores tal y como se ve en la figura D.10. El segundo de los gráficos de la figura D.10 es una ampliación del primero en la que se añade la recta que separa los datos linealmente.

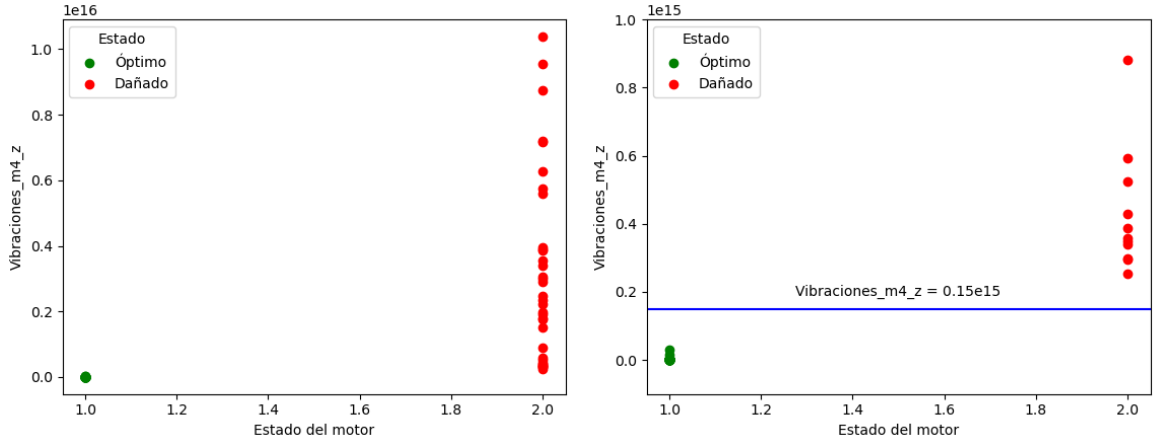

Figura D.10: Vibraciones_m4_z vs. estado del motor

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla D.19.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
m_4	50	1	0

Tabla D.19: Resumen de la estabilidad del modelo obtenido con variables de vibraciones (eje Z) en observaciones alimentadas por variador ABB

Alimentación Red

Llevando a cabo una reducción del número de variables, se obtiene que el mejor modelo es con sólo la variable *cf*. Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla D.20.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
cf	50	0,9825	0,0506

Tabla D.20: Resumen de la estabilidad del modelo obtenido con variables de vibraciones (eje Z) en observaciones alimentadas por Red

Alimentación WEG

Por último, solo con la variable m_2 se alcanza una tasa de acierto del 100 %. Estudiando esta variable, se aprecia en la figura D.11 que separa linealmente ambos tipos de motores. El segundo de los gráficos de la figura D.11 es una ampliación del primero en la que se añade la recta que separa los datos linealmente.

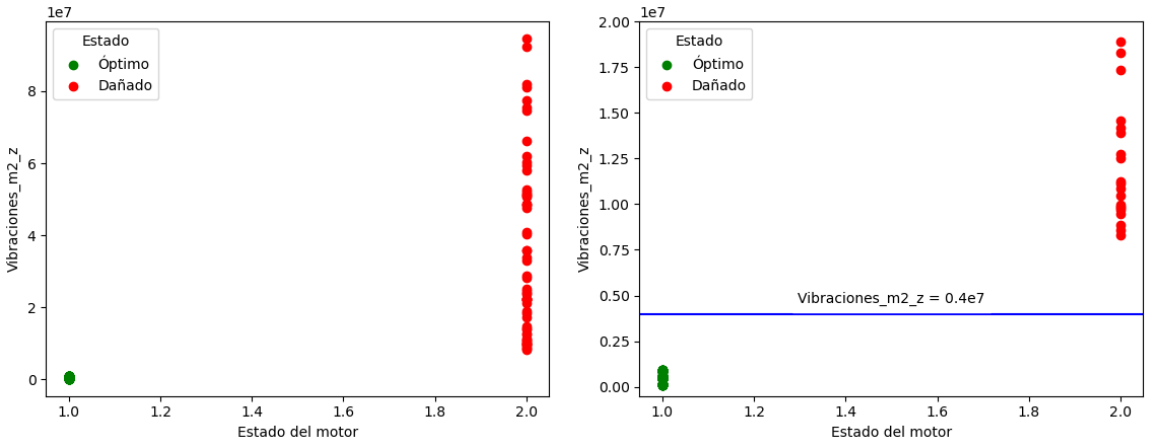

Figura D.11: Vibraciones_m2_z vs. estado del motor

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla D.21.

Modelo	**Repeticiones**	**Tasa de acierto promedio**	**Desviación típica**
m_2	50	1	0

Tabla D.21: Resumen de la estabilidad del modelo obtenido con variables de vibraciones (eje Z) en observaciones alimentadas por variador WEG

Apéndice E

Obtención de los modelos por tipo de carga

E.1. Análisis solo con los datos de corriente

E.1.1. Modelos separados para cada tipo de carga

Carga Baja

En primer lugar, se realiza un modelo solo con las observaciones con carga baja. Seguidamente, se lleva a cabo una reducción del número de variables cuyo resultado se encuentra en la figura E.1. Como podemos apreciar en la figura E.1, con 5 variables se obtiene ya una tasa de acierto óptima. Las variables que se incluyen en el modelo son: c_1, m_3, x_r, cf y sf.

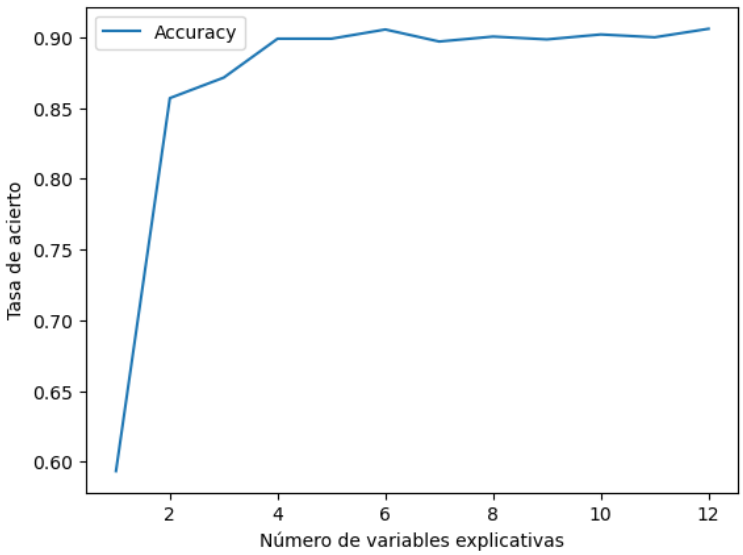

Figura E.1: Tasa de acierto vs. número de variables explicativas de corriente promediadas en observaciones con carga baja

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla E.1.

Modelo	**Repeticiones**	**Tasa de acierto promedio**	**Desviación típica**
c_1, m_3, x_r, cf y sf	50	0,947	0,0415

Tabla E.1: Resumen de la estabilidad del modelo obtenido con variables de corriente promediadas en observaciones con carga baja

Carga Alta

Ahora, se realiza un modelo solo con las observaciones con carga alta. Seguidamente, se lleva a cabo una reducción del número de variables cuyo resultado se encuentra en la figura E.2. En esta figura, se aprecia como el mejor modelo posible que mantiene la tasa de acierto con el menor número de variables posibles es: c_1, c_3, *skew*, *kurt*, x_r y *sf*.

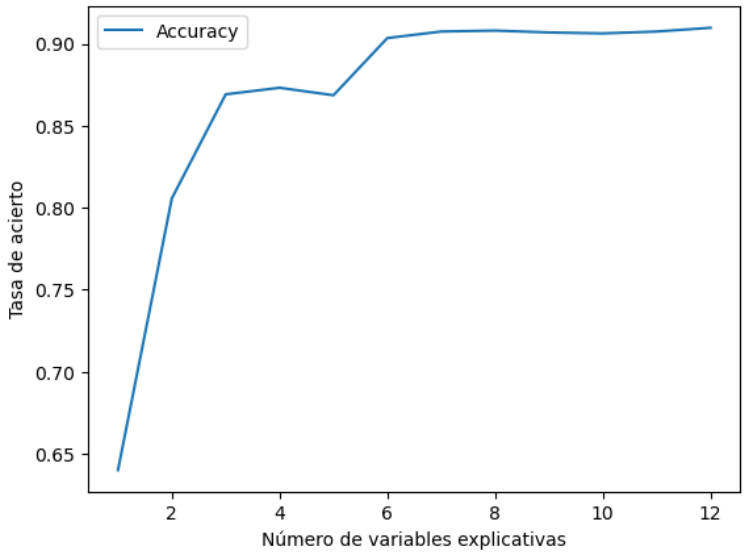

Figura E.2: Tasa de acierto vs. número de variables explicativas de corriente promediadas en observaciones con carga alta

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla E.2.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
c_1, c_3, skew, kurt, x_r y sf	50	0,9497	0,0430

Tabla E.2: Resumen de la estabilidad del modelo obtenido con variables de corriente promediadas en observaciones con carga alta

E.2. Análisis solo con los datos de sonido

Puesto que al llevar a cabo el análisis sin diferenciar por tipo de carga ya se encontró una variable que separaba perfectamente ambos conjuntos, los modelos que se aplicarán para cada tipo de carga son con esa variable (c_2).

E.2.1. Modelos separados para cada tipo de carga

Carga Baja

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla E.3.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
c_2	50	0,9995	0,0035

Tabla E.3: Resumen de la estabilidad del modelo obtenido con variables de sonido en observaciones con carga baja

Carga Alta

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla E.4.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
c_2	50	0,9977	0,0078

Tabla E.4: Resumen de la estabilidad del modelo obtenido con variables de sonido en observaciones con carga alta

E.3. Análisis solo con los datos de vibraciones

Para cada uno de los ejes (X, Y, Z) se lleva a cabo modelos separados para cada tipo de carga.

E.3.1. Eje X

Carga Baja

En primer lugar, se lleva a cabo un modelo solo con las observaciones con carga baja. Seguidamente, se realiza una reducción del número de variables. Como podemos apreciar en la figura E.3, la tasa de acierto aumenta muy poco al incrementar el número de variables por lo que nos quedamos con 3 variables por ser el más sencillo que funciona similar al modelo con mayor tasa de acierto. El modelo es: c_2, m_2 y x_r.

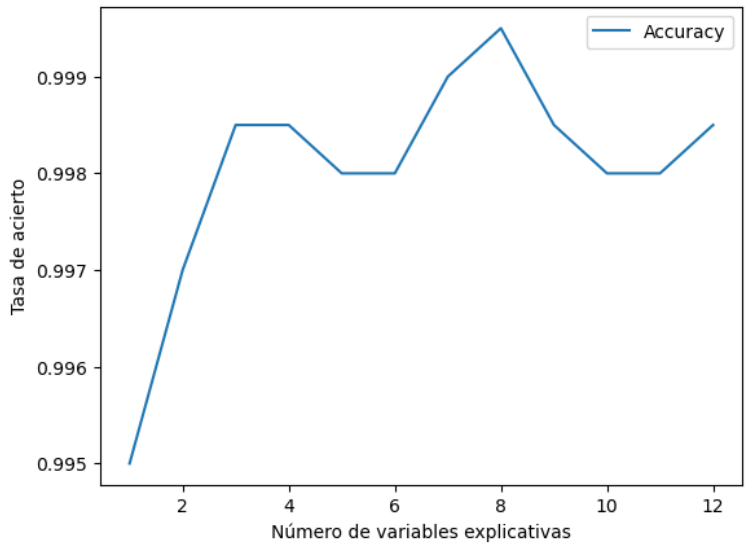

Figura E.3: Tasa de acierto vs. número de variables explicativas de vibraciones (eje X) en observaciones con carga baja

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla E.5.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
c_2, m_2 y x_r	50	0,9975	0,0076

Tabla E.5: Resumen de la estabilidad del modelo obtenido con variables de vibraciones (eje X) en observaciones con carga baja

Carga Alta

Ahora, se lleva a cabo un modelo solo con las observaciones con carga alta. Seguidamente, se realiza una reducción del número de variables. Como podemos apreciar en el gráfico de la figura E.4, con una sola variable se tiene la misma tasa de acierto que incluyendo más variables. El modelo sería: m_2.

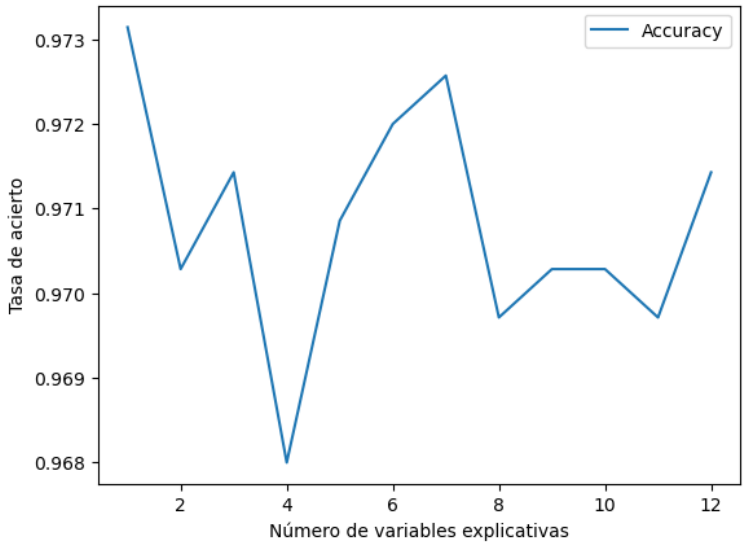

Figura E.4: Tasa de acierto vs. número de variables explicativas de vibraciones (eje X) en observaciones con carga alta

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla E.6.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
m_2	50	0,9874	0,0175

Tabla E.6: Resumen de la estabilidad del modelo obtenido con variables de vibraciones (eje X) en observaciones con carga alta

E.3.2. Eje Y

Puesto que al llevar a cabo el análisis sin diferenciar por tipo de carga ya se encontró una variable que separaba perfectamente ambos conjuntos, los modelos que se aplicarán para cada tipo de carga sera con esa variable (c_1).

Carga Baja

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla E.7.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
c_1	50	1	0

Tabla E.7: Resumen de la estabilidad del modelo obtenido con variables de vibraciones (eje Y) en observaciones con carga baja

Carga Alta

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla E.8.

Modelo	Repeticiones	Tasa de acierto promedio	Desviación típica
c_1	50	1	0

Tabla E.8: Resumen de la estabilidad del modelo obtenido con variables de vibraciones (eje Y) en observaciones con carga alta

E.3.3. Eje Z

Carga Baja

En primer lugar, se lleva a cabo un modelo solo con las observaciones con carga baja. Seguidamente, se realiza una reducción del número de variables. Como podemos apreciar en la figura E.5, el modelo con 3 variables es el más sencillo que funciona similar al modelo con mayor tasa de acierto. El modelo es: c_2, x_r y *cf.*

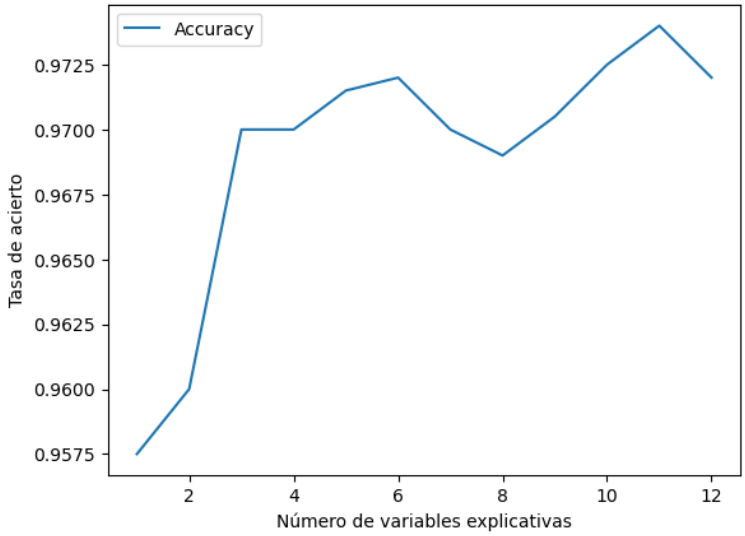

Figura E.5: Tasa de acierto vs. número de variables explicativas de vibraciones (eje Z) en observaciones con carga baja

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla E.9.

Modelo	**Repeticiones**	**Tasa de acierto promedio**	**Desviación típica**
c_2, x_r y cf	50	0,988	0,0184

Tabla E.9: Resumen de la estabilidad del modelo obtenido con variables de vibraciones (eje Z) en observaciones con carga baja

Carga Alta

Ahora, se lleva a cabo un modelo solo con las observaciones con carga alta. Seguidamente, se realiza una reducción del número de variables. Como podemos apreciar en el gráfico de la figura E.6, con 2 variables se tiene la misma tasa de acierto que incluyendo más. El modelo sería: c_1 y x_r.

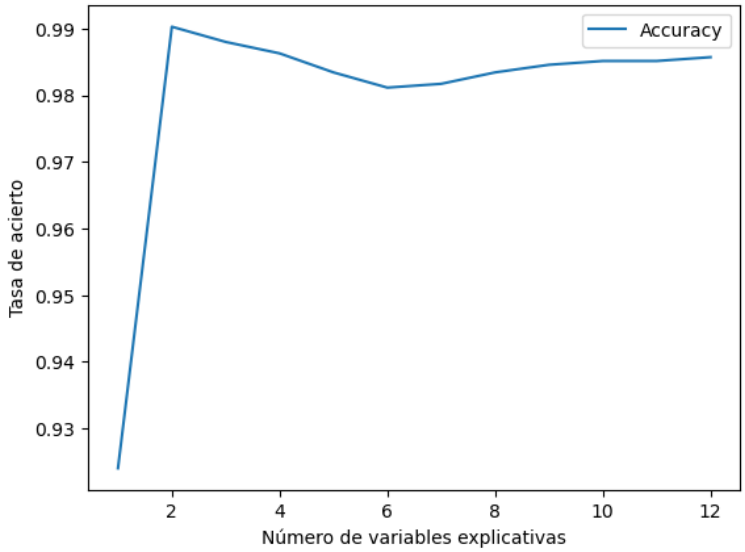

Figura E.6: Tasa de acierto vs. número de variables explicativas de vibraciones (eje Z) en observaciones con carga alta

Para poder comparar el modelo con otros, se lleva a cabo replicaciones *Bootstrap* del mismo. El resumen de las tasas de acierto obtenidas por las replicaciones *Bootstrap* se encuentran en la tabla E.10.

Modelo	**Repeticiones**	**Tasa de acierto promedio**	**Desviación típica**
c_1 y x_r	50	0,9903	0,0188

Tabla E.10: Resumen de la estabilidad del modelo obtenido con variables de vibraciones (eje Z) en observaciones con carga alta