UNIVERSIDAD DEL BÍO-BÍO
FACULTAD DE CIENCIAS EMPRESARIALES

Monitoreo de condición industrial usando Deep Learning

15 de mayo de 2026

Autor:
Héctor Ayala Peñailillo
hayala@ing.ucsc.cl
Profesor guía:
Mario Ramos Maldonado
Concepción

Resumen

El monitoreo de condiciones es una técnica indispensable para la industria, ya que permite la identificación de fallas en la maquinaria y la optimización de estrategias de mantenimiento. En consecuencia, ayuda a evitar accidentes producidos por el mal funcionamiento de algún sistema y reduce los tiempos de paro en las cadenas de producción. El monitoreo de condiciones conducido por datos requiere la recolección constante de registros producidos por sensores puestos en las máquinas. Como los registros tienen un orden temporal, estos corresponden a series de tiempo. Los algoritmos de Deep Clustering han demostrado ser efectivos para series de tiempo. Sin embargo, pocos estudios están dedicados al trabajo con maquinaria industrial, y una fracción aún más pequeña utiliza datos reales. Este estudio utiliza Deep Clustering para realizar monitoreo de condiciones en chancadores mineros. Los datos fueron obtenidos del funcionamiento real de 5 máquinas. Bajo el modelo de Sequential Multistep Deep Clustering, se entrenó un autoencoder convolucional para aprender una representación diferente de los datos. Posteriormente, k-means y DBSCAN fueron usados para crear los grupos. La idea central del procedimiento es caracterizar los estados de funcionamiento, identificando aquellos grupos anómalos. Debido a que no se disponía de etiquetas para los datos, se utilizó la distancia de Wasserstein en conjunto a la distribución de alertas de sistema, para identificar los grupos anómalos. Los resultados mostraron que ambos algoritmos, utilizados en Deep Clustering, son efectivos. Además se constató que la cantidad de ciclos de entrenamiento del autoencoder tiene un alto impacto en el agrupamiento.

Abstract

Condition monitoring is an indispensable technique for industry, as it enables the identification of machinery failures and the optimization of maintenance strategies. Consequently, it helps prevent accidents caused by system malfunctions and reduces downtime on production lines. Data-driven condition monitoring requires the constant collection of data logs generated by sensors installed on machines. Since these logs are time-ordered, they constitute time series. Deep clustering algorithms have proven effective for time series. However, few studies focus on industrial machinery, and an even smaller fraction uses real-world data. This study uses deep clustering to monitor conditions in mining crushers. The data were obtained from the actual operation of five machines. Using the Sequential Multistep Deep Clustering model, a convolutional autoencoder was trained to learn a different representation of the data. Subsequently, k-means and DBSCAN were used to create the clusters. The central idea of the procedure is to characterize operating states by identifying anomalous clusters. Since no labels were available for the data, Wasserstein distance was used in conjunction with the distribution of system alerts to identify the anomalous clusters. The results showed that both algorithms, when used in Deep Clustering, are effective. Furthermore, it was found that the number of autoencoder training cycles has a significant impact on clustering.

Agradecimientos

Agradezco al profesor Mario Ramos por su orientación en todo el proceso y por su disposición para discutir ideas que, en ocasiones, se salían del alcance del proyecto. Agradezco a Yasmany Prieto quien ofreció parte de su tiempo para ayudarme a entender algunas conceptos y por su opinión profesional en algunos puntos. Finalmente, agradezco a mi familia por el apoyo que me han brindado a lo largo de todos mis años de estudio.

Índice

Índice de figuras

Índice de tablas

Lista de Acrónimos

AE	Autoencoder
CNN	(*Convolutional Neural Network*) Red Neuronal Convolucional
CAE	(*Convolutional Autoencoder*) Autoencoder Convolucional
DBSCAN	*Density-Based Spatial Clustering of Applications with Noise*
DL	(*Deep Learning*) Aprendizaje Profundo
MC	Monitoreo de Condiciones
ML	(*Machine Learning*) Aprendizaje Automático
PCA	(*Principal Component Analysis*) Análisis de Componentes Principales
RL	*Representation Learning*
RNN	(*Recurrent Neural Network*) Red Neuronal Recurrente
RNN-AE	(*Recurrent Autoencoder*) Autoencoder Recurrente

1. Introducción

La producción industrial se sustenta en el uso de diferentes maquinarias que son manipuladas y controladas en mayor y menor medida por seres humanos. Una cadena de producción puede estar conformada por un conjunto de máquinas que deben operar en buenas condiciones para garantizar un cierto nivel de desempeño, como podría ser alcanzar un número específico de unidades generadas por día. Debido a que diversos factores, muchos de ellos incontrolables para la industria, afectan el desempeño de la producción, es que se tiene la necesidad de realizar un monitoreo de condiciones en tiempo real de la maquinaria. Algunos de estos factores son fallas no anticipadas en la maquinaria, fallas humanas, condiciones medioambientales, condiciones socio-económicas, entre otros.

Para realizar monitoreo de condiciones conducido por datos se requiere la recolección de datos desde uno o varios sensores instalados en las máquinas. Esto genera un enorme volumen de datos que requieren ser procesado mediante el uso de modelos analíticos o de datos. La utilización de estas técnicas también abre las puertas para generar procesos de control automatizados. Los modelos de *deep learning* (en español, aprendizaje profundo), un subconjunto dentro de la familia de algoritmos de *machine learning* (en español, aprendizaje automático), han mostrado utilidad para el manejo de grandes cantidades de datos, ademas de poseer la particularidad de que por regla general no requieren un análisis previo sobre los datos para poder utilizarlos. La idea central de la utilización de deep learning radica en que estos modelos pueden descubrir patrones en los datos que posteriormente pueden ser utilizados para diversas tareas como predecir o identificar fallas.

Este trabajo propone la utilización de modelos de deep learning, concretamente deep learning no supervisado, para realizar monitoreo de condiciones en maquinarias industriales.

El documento está organizado de la siguiente manera: en la Sección 2 se presenta el fundamento teórico base; la Sección 3 contiene el estado del arte centrándose en Deep Clustering utilizado para series de tiempo; en la Sección 4 se describe el problema abordado, se presentan las preguntas de investigación asociadas, se plantea la hipótesis y los objetivos, y se definen los modelos que se utilizarán; en la Sección 5 se describe de la metodología; la Sección 6 expone los resultados junto a una interpretación y a las recomendaciones asociadas y, finalmente, la Sección 7 entrega las conclusiones finales y plantea los desafíos futuros.

2. Fundamento teórico

En este capítulo se aborda el fundamento teórico con el fin de proporcionar las definiciones básicas y las nociones generales utilizadas en todo el documento. El capítulo se organiza de la siguiente manera: en la Sección 2.1 se explica el proceso y necesidad del monitoreo de condiciones; en las Sección 2.2 se entrega la definición y clasificación de las series de tiempo, además de recalcar su estrecha relación con el monitoreo de condiciones; la Sección 2.3 aborda el machine learning, haciendo hincapíe en el aprendizaje no supervisado y sus medidas de desempeño; finalmente, la Sección 2.6 está dedicada al deep Learning y deep clustering, profundizando en algunas de las arquitecturas más usadas con series de tiempo.

2.1. Monitoreo de condiciones

Las averías inesperadas en la maquinaria de una planta causan daños económicos debido, entre otras razones, al tiempo que se requiere para que dichas máquinas sean intervenidas y vuelvan a estar operativas. Es por tanto de interés de toda compañía tener la capacidad de identificar y anticipar fallas de esta naturaleza. La realización de mantenimiento sobre los equipos es una medida que pretende evitar estas interrupciones. Existen tres estrategias convencionales para la realización de mantenimiento [42]:

1. **Mantenimiento de averías:** el mantenimiento (o remplazo) se realiza solo cuando surge un problema.
2. **Mantenimiento planeado:** se realiza un mantenimiento periódico a intervalos de tiempo regulares.
3. **Mantenimiento basado en condiciones:** se monitorea el estado de salud de la maquinaria con el fin de sugerir cuando realizar el mantenimiento.

El mantenimiento basado en condiciones, también llamado mantenimiento predictivo, es una técnica de diagnóstico avanzada que permite revelar fallas de operación de los equipos en fases previas a la ocurrencia de una avería [42]. Puede realizarse de manera directa, inspeccionando la maquinaria en el sitio donde está emplazada, o de manera indirecta mediante sensores que tomen y registren muestras del comportamiento del equipo de manera periódica. Esta última, preferible en la mayoría de los casos debido a que no supone una interrupción en el uso de los equipos, es conocida también como **Monitoreo de Condiciones** (MC).

Mobley en [28] describe al MC como una técnica de gestión que usa evaluaciones regulares de la condición operacional actual de equipos de planta, de sistemas de producción, y de la

gestión de funciones de planta, para de esta manera optimizar el total de las operaciones. La agencia internacional de energía atómica define al MC como pruebas, inspecciones, mediciones o seguimiento, ya sea continuo o periódico, del rendimiento o de las características físicas de estructuras, sistemas y componentes, para indicar el rendimiento actual o futuro, y el potencial de falla [18].

Existen dos estrategias para realizar MC: una basada en modelos y otra conducida por datos [20]. La primera se apoya en un modelo matemático/físico, por lo que puede resultar impracticable en la mayoría de los casos debido a que varios procesos físicos son demasiado complejos como para ser modelados con precisión. La segunda depende de las características extraídas de los datos y hace uso de técnicas estadísticas o de inteligencia artificial para llegar a un diagnóstico.

La teoría de confiabilidad es ampliamente utilizada para el MC. Confiabilidad es la habilidad de un producto o sistema para operar bajo las condiciones de operación designadas, por un periodo de tiempo determinado [29]. El propósito de su estudio es el desarrollo de métodos que permitan medirla. Esto puede ser llevado a cabo mediante la utilización de probabilidades.

El MC sigue el siguiente esquema secuencial [25]:

1. **Adquisición de datos:** toma de registros mediante sensores. Para cada momento t, un registro puede contener una o más variables como por ejemplo temperatura, aceleración, entre otras.

2. **Análisis de datos:** análisis sobre los datos utilizando, por ejemplo, técnicas de procesamiento de señales.

3. **Extracción de atributos:** identificación y cuantificación de aquellos aspectos de los datos que funcionan como buenos indicadores de falla

4. **Toma de decisión:** identificación de fallo utilizando técnicas de inteligencia artificial.

5. **Diagnóstico de condición**

Existe una responsabilidad legal y moral para minimizar los riesgos asociados a procesos industriales. Además de pérdidas económicas, las fallas en la maquinaria pueden ser desencadenantes de accidentes que pueden costar vidas humanas. El monitoreo de condiciones provee los datos necesarios para poder diagnosticar cuando un equipo requiere mantenimiento. También puede usarse para identificar los factores que limitan la efectividad y la eficiencia total de una planta [28].

2.2. Series de tiempo

Una serie de tiempo es un conjunto de observaciones x_t, cada una registrada en un tiempo t específico [10]. El término suele usarse para diferentes tipos de datos con un componente temporal.

Alqahtani et al. [1] clasifican los datos de series de tiempo en 4 categorías:

- **Series de tiempo univariadas:** es una secuencia que contiene solo un valor por punto de tiempo t. Un ejemplo es tomar la temperatura de un lugar cada hora durante una semana.
- **Series de tiempo multivariads:** es un conjunto de series de tiempo que comparten los mismos instantes de tiempo. Si una serie de tiempo univariada se define como un conjunto $X = [x_1, x_2, ..., x_T]$ con T instantes, entonces una serie de tiempo multivariada se puede definir como una secuencia $X = [X^1, X^2, ..., X^M]$, que se compone de M series de tiempo univariadas diferentes, con $X^i \in \mathbb{R}^T$ [19].
- **Tensor fields:** se pueden describir como una cantidad asociada a cada punto en el espacio-tiempo.
- **Multifield:** en [16] se describen los datos multifield como un conjunto de campos (*fileds*) interrelacionados de alguna manera. Un campo (*single field*) es una representación de una cantidad específica, sobre algún dominio como por ejemplo, el tiempo.

Las series de tiempo están compuesta de tres componentes principales [14, 10]:

1. **Tendencia** (*Trend*): representa la dirección en la cual los datos se mueven a través del tiempo, excluyendo la estacionalidad e irregularidades.
2. **Estacionalidad** (*Seasonality*): patrones que ocurren a intervalos regulares.
3. **Residuales** (*Residuals*): despues de separar los componentes de tendencia y de estacionalidad, lo que queda son los residuales.

Un ejemplo de serie de tiempo se encuentra en la Figura 1. Los datos[1] corresponden a la temperatura de la ciudad Tetouan, Marruecos, en un periodo de 2 meses. La serie de tiempo es univariada (el conjunto completo es multivariado, pero solo se ha tomado la temperatura) y discreta. Ha sido sometida a una descomposición aditiva ($X[t] = T[t] + E[t] + R[t]$) utilizando un periodo mensual para diferenciar sus componentes.

[1] https://www.kaggle.com/datasets/fedesoriano/electric-power-consumption. Fecha de acceso: 05-04-2026

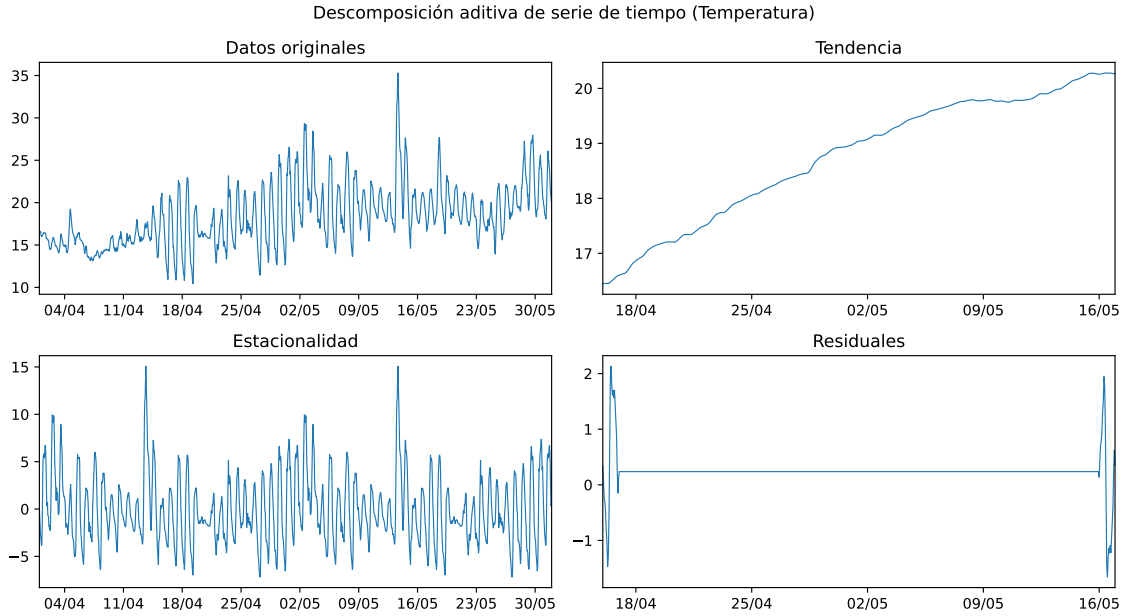

Figura 1: Gráfico de los componentes de series de tiempo

Los sistemas de producción generan datos con series de tiempo de manera inherente. En una industria se registra (o es deseable registrar) el comportamiento de los sistemas y las variaciones de funcionamiento de los equipos en todo momento. Una de las característica más significativas de las series de tiempo es que en ellas se encuentran los **patrones históricos** de funcionamiento del equipo o sistema que se esté midiendo. El MC conducido por datos es capaz de hacer uso de los datos para descubrir estos patrones y así predecir las potenciales fallas futuras, y optimizar las operaciones.

El trabajo con series de tiempo afronta varios desafíos. Algunos de estos son comunes con los datos sin secuencia temporal: datos nulos debido a errores en los sensores y registro de valores atípicos. Un problema adicional es que la secuencia de datos sea registrada a intervalos irregulares. Este fenómeno puede limitar los métodos disponibles para analizar. Si se requiere realizar agrupamiento con series de tiempo, la presencia de ruido, dimensiones grandes y una alta correlación entre los atributos, constituyen obstáculos para el diseño de algoritmos efectivos y eficientes. Esto, a comparación con registros sin componente temporal, resulta especialmente problemático en series de tiempo [1],

2.3. Machine Learning

Machine learning (ML) es un subcampo de la inteligencia artificial con una fuerte base teórica en las estadísticas. Los algoritmos de machine learning están diseñados para *aprender* o extraer conocimientos desde los datos mediante el aprendizaje con ejemplos. De manera formal, se dice que un programa computacional *aprende* de una experiencia

E con respecto a alguna clase de tarea T y medida de desempeño P, si su desempeño en la tarea T, medido mediante P, mejora con la experiencia E [27].

Los modelos de machine learning pueden ser precisos si una cantidad de datos adecuada es usada durante la etapa de aprendizaje [41], aunque dependiendo de las circunstancias, esto no siempre es posible.

La secuencia de trabajo general para machine learning es la siguiente. Primero **los datos deben ser capturados** mediante el uso de sensores u otros medios. Luego, los datos pueden ser **preprocesados**, ya sea para reducir sus dimensiones o para adaptarse a un modelo en concreto. Existen algoritmos que son sensibles a datos no normalizados. Dependiendo de la naturaleza del problema, los datos pueden ser divididos entre un **conjunto de entrenamiento** y un **conjunto de pruebas**. El primero es usado para el aprendizaje (o entrenamiento) del algoritmo y el segundo para la evaluación del desempeño del mismo. Esta división es necesaria ya que evaluar un modelo en base a los mismos ejemplos de entrenamiento puede llevar a problemas de *sobreajuste* que es, dicho de manera simple, cuando el algoritmo aprende los datos en vez de aprender los patrones implícitos en estos. Por último el modelo es creado, entrenado y evaluado. Dependiendo del desempeño mostrado en la etapa de evaluación y de los objetivos que se desean, pueden ser necesarios ajustes que lleven a repetir el proceso de manera completa o parcial.

Los algoritmos de machine learning pueden clasificarse de la siguiente manera:

- **Algoritmos de aprendizaje supervisado:** aprenden de datos que tienen una entrada y salida conocida. Se les llama supervisados ya que, al conocerse la salida esperada, existe una *supervición* que guía el proceso.
- **Algoritmos de aprendizaje no supervisado:** los datos solo contienen entrada y la salida es desconocida.

2.4. Aprendizaje supervisado

Los algoritmos de aprendizaje supervisado son utilizados para problemas de clasificación y regresión.

En un problema de **clasificación**, se pide especificar a qué categoría o clase pertenece alguna instancia de entrada x. Para llevar la tarea a cabo, el algoritmo de aprendizaje debe producir una función $f : \mathbb{R}^n \mapsto \{1, ..., k\}$, donde n es la dimensión del vector x. Cuando $y = f(x)$, el modelo asigna la clase y para la entrada x.

En un problema de **regresión**, la tarea es predecir un valor numérico dada una entrada x. De manera similar a los problemas de clasificación, el algoritmo debe producir una función $f : \mathbb{R}^n \mapsto \mathbb{R}$. A diferencia de la clasificación, cuya salida está contenida en un conjunto

discreto finito con una interpretación categórica, en los problemas de regresión la salida es numérica y puede pertenecer a un intervalo continuo o discreto, finito o infinito (un subconjunto de $\mathbb{R}$).

2.5. Aprendizaje no supervisado

Obtener datos junto a sus etiquetas, (una entrada x junto a la salida esperada y para esa entrada) no es posible en la mayoría de los casos, por lo que la utilización de modelos de aprendizaje supervisado podría no ser posible. El aprendizaje no supervisado se realiza sin etiquetas. Su utilización apunta a revelar la organización de patrones que permitan descubrir similitudes y diferencia entre los ejemplos.

Los modelos de aprendizaje no supervisado pueden clasificarse en algoritmos de transformación de datos y algoritmos de agrupamiento.

Los **algoritmos de transformación** crean representaciones nuevas para el conjunto de datos. El objetivo es que la nueva representación sea más fácil de comprender que la original, ya sea para humanos o para otros modelos de ML.

Los **algoritmos de agrupamiento** (en inglés, *clustering*) particionan el conjunto de datos en distintos grupos de similares características. En [43] se define el agrupamiento de la siguiente manera: sea X el conjunto de datos que contiene a todos los ejemplos x_i, es decir $X = \{x_1, x_2, ..., x_n\}$. Se define como *m-clustering* de X a la partición de X en m conjuntos $C_1, ..., C_m$, de tal forma que se cumplen las siguientes condiciones:

- Los conjuntos no pueden ser vacíos ($C_i \neq \emptyset, \quad i = 1, ..., m$)
- La unión de todos los m conjuntos es igual al conjunto X ($\cup_{i=1}^{m} C_i = X$)
- La intersección entre cualquier conjunto con otro es vacía ($C_i \cap C_j = \emptyset. \quad i \neq j; \quad i, j = 1, ..., m$)
- Los ejemplos contenidos en cada conjunto C deben ser *mas similares* entre ellos que con ejemplos de otros conjuntos. La manera en la que se calcula esta similitud depende del tipo de grupo

En [32] sumarizan diferentes criterios para categorizar a los algoritmos de agrupamiento. Uno de ellos es el tipo de estrategia que utiliza el algoritmo para crear los grupos.

- **Agrupamiento basado en particiones** (*Partition-based clustering*): se separan los objetos en k grupos, optimizando algún criterio como la distancia intra-cluster.

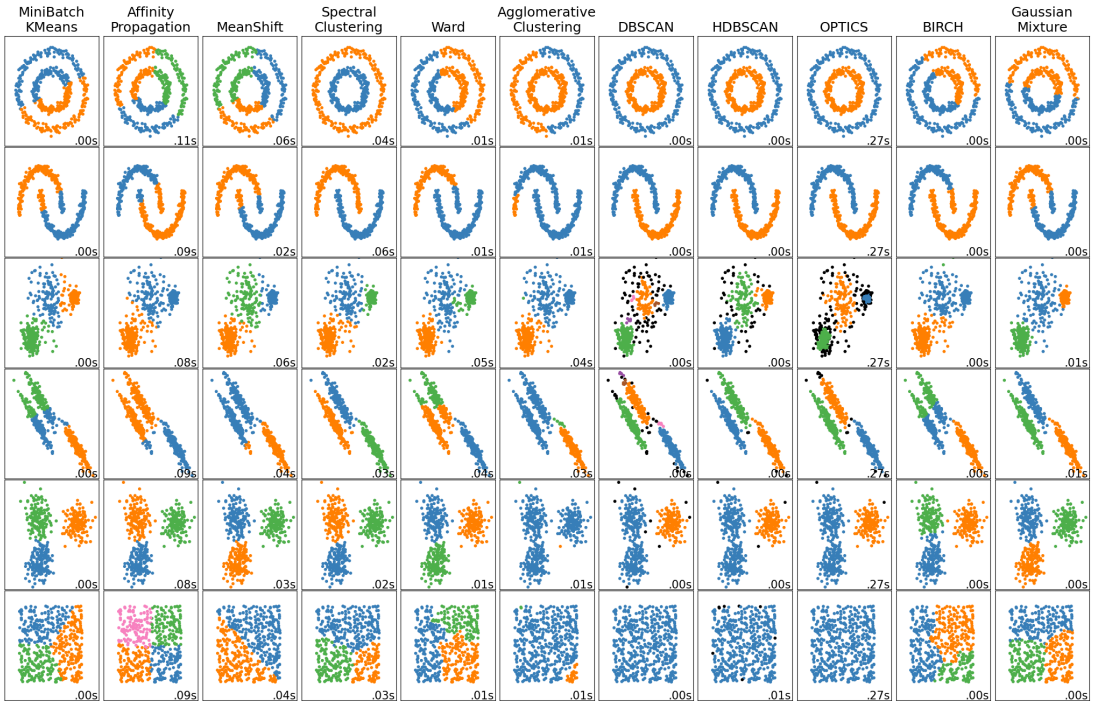

Figura 2: Diferentes algoritmos sobre un conjunto de datos 2D

- **Agrupamiento jerárquico** (*Hierarchical-based clustering*): en vez de generar un solo conjunto de grupos (como los basados en particiones), crean una jerarquía de agrupamientos.
- **Agrupamiento basado en cuadrícula** (*Grid-based clustering*): dividen el espacio en un número finito de celdas en una estructura cuadricular. El agrupamiento se realiza basado en las celdas y no en los objetos.
- **Agrupamiento basado en densidad** (*Density-based clustering*): se tiene una perspectiva de los grupos como regiones *densas* de datos en el espacio n-dimensional.
- **Agrupamiento basado en modelos** (*Model-based clustering*): asumen un modelo para los grupos e intentan *ajustar* los datos a dicho modelo.

Debido a que el aprendizaje no supervisado no utiliza datos con etiquetas, no se requiere separar los datos para entrenamiento y pruebas. En la Tabla 1 se listan algunos algoritmos de ML utilizados para diferentes tareas. La Figura 2 muestra de manera comparativa el comportamiento de diferentes algoritmos[2]. Todos los conjuntos de datos son de dos dimensiones, sin embargo la estructura varía.

[2]https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html. Fecha de acceso: 12-04-2026

Tarea	**Algoritmos**
Aprendizaje supervisado	K-Nearest Neighbors (K-NN), Random Forest (RF), Support Vector Machines (SVM), Árbol de decisión
Agrupamiento basado en particiones	K-means, arbitrarily ORiented projected CLUSter generation (ORCLUS), K-medoids, Clustering Large Applications Based upon Randomized Search (CLARANS)
Agrupamiento jerárquico	RObust Clustering using linKs (ROCK), Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH)
Agrupamiento basado en cuadrícula	CLustering In QUEst (CLIQUE), Statistical Information Grid (STING)
Agrupamiento basado en densidad	Density-based Spatial Clustering of Applications with Noise (DBSCAN), Ordering Points To Identify the Clustering Structure (OPTICS)
Agrupamiento basado en modelos	Gaussian Mixture (GM), Autoclass, Self Organising Map (SOM)
Transformación	Análisis de componentes principales (PCA), Non-negative Matrix Factorization (NMF)

Tabla 1: Tabla resumen con algunos algoritmos populares para tareas de machine learning.

2.5.1. k-means

K-means es un algoritmo de agrupamiento basado en particiones. Fue propuesto de manera independiente por diferentes investigadores de diferentes disciplinas entre las décadas de los 50 y los 60 [17].

A partir de un conjunto de datos X, se requiere crear K clusters o grupos distintos $C_1, ..., C_K$. Para cada uno es posible calcular el error cuadrático entre todos los puntos que lo conforman y un punto promedio o punto *representante* μ_k:

$$J(C_k) = \sum_{x_i \in C_k} ||x_i - \mu_k||^2 \tag{1}$$

La idea en K-means es minimizar la sumatoria de la Ecuación 1 para cada uno de los grupos, es decir minimizar J:

$$J = \sum_{k=1}^{K} \sum_{x_i \in C_k} ||x_i - \mu_k||^2 \tag{2}$$

El algoritmo opera como sigue: de manera aleatoria se escogen K centroides $\mu_1, ..., \mu_K$ desde el conjunto de datos X. Estos puntos representarán a cada uno de los grupos. Iterativamente, para cada punto $x \in X$, se calcula su distancia con respecto a los centroides y se le asigna el grupo al que representa el centroide más cercano. Luego de este último paso, el centroide del grupo al que x fue asignado es actualizado teniendo en cuenta el nuevo punto.

En la Figura 3 se muestra gráficamente el funcionamiento del algoritmo para un conjunto de dos dimensiones. Primero se seleccionan los centroides (marcados con una cruz) de manera aleatoria (a) para luego proceder a asignar a cada punto al centroide mas cercano (b). Posteriormente, los centroides son actualizados (c). El algoritmo continua iterando entre los pasos (b) y (c) hasta que se cumpla algun criterio de término.

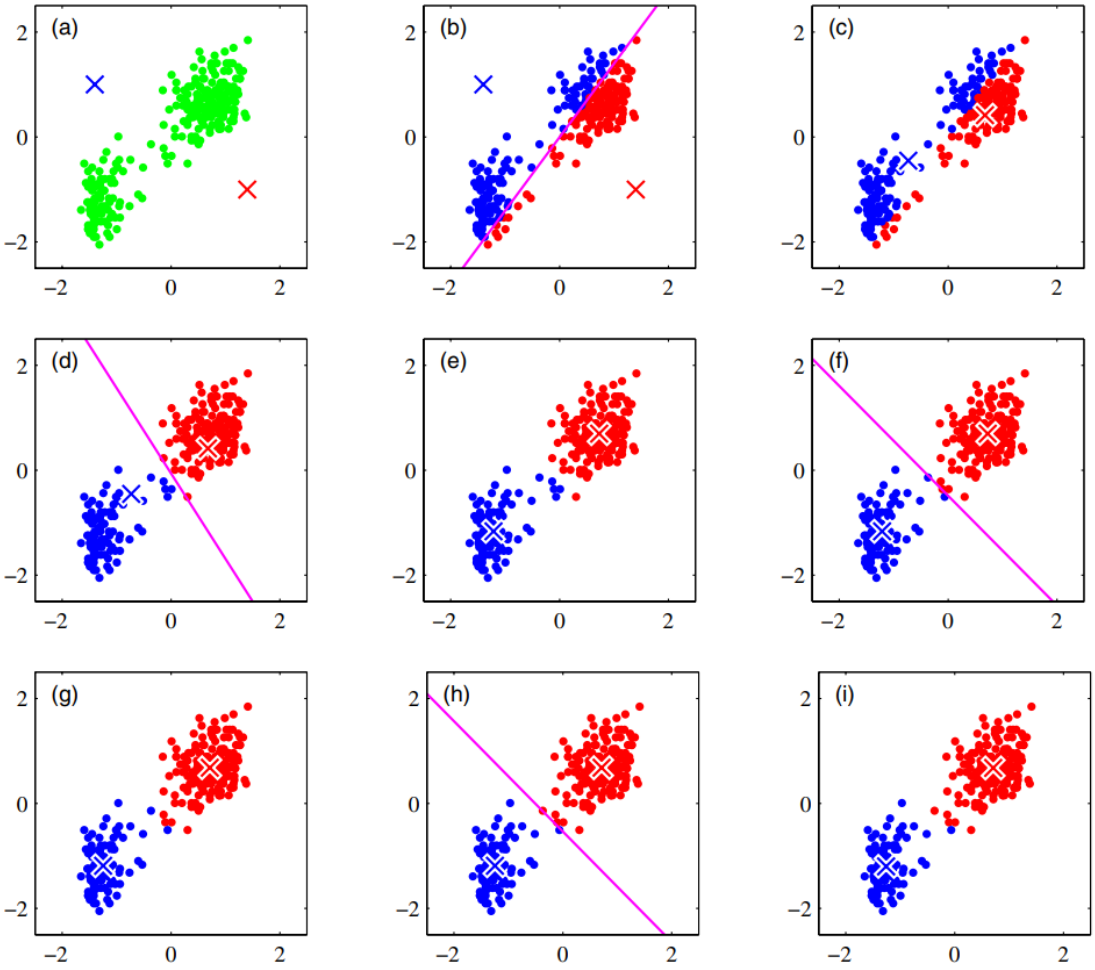

Figura 3: Funcionamiento secuencial de k-means para un conjunto de datos de 2 dimensiones [6].

Algunas características importantes de K-means se listan a continuación:

- El único hiperparámetro K corresponde a la cantidad de particiones a realizar. Una mala estimación de este valor puede provocar que el algoritmo no descubra la estructura subyacente del conjunto de datos.

- K-means es sensible a datos atípicos y ruido.
- El algoritmo a dado pie a diversas variantes. En [17] se realiza una revisión del algoritmo estándar y se exponen variantes.

2.5.2. DBSCAN

Density-based Spatial Clustering of Applications with Noise, más conocido como DBSCAN [13], es un algoritmo de agrupamiento basado en densidad. El algoritmo se sustenta en la idea de identificar regiones con alta densidad de puntos, separadas por regiones con una menor densidad.

Para explicar el funcionamiento de DBSCAN, se requiere definir algunos conceptos. Si se tiene el conjunto de datos X:

- La **densidad** alrededor de un punto $x \in X$ corresponde al número de puntos en el interior de una hiperesfera $V_\varepsilon(x)$ creada alrededor de x, con un radio ε. Si la cantidad supera un umbral q, se considera a x como un punto nuclear.
- Un punto y es **alcanzable por densidad** (*density reachable*) desde otro punto x si existe una secuencia $x_1, x_2, ..., x_p$, con $x_1 = x$ y $x_p = y$, de tal forma que x_{i+1} es **directamente alcanzable por densidad** (*directly density reachable*) desde x_i, es decir, si $x_{i+1} \in V_\varepsilon(x_i)$ y la cantidad de puntos en $V_\varepsilon(x_i)$ es mayor o igual a q.
- Un punto x está **conectado por densidad** (*density connected*) a otro punto x' si existe un punto $z \in X$ de tal forma que x y x' son density reachable desde z. En la Figura 4 se muestran ejemplos visuales para puntos alcanzables por densidad, directamente alcanzables por densidad y conectados por densidad.

Finalmente, un grupo o **cluster** $C \subset X$ es definido de la siguiente manera:

- Si $x \in C$ y $x' \in X$ es alcanzable por densidad a x, entonces $x' \in C$.
- Todos los pares en un cluster C están conectados por densidad.

Algunas características importantes de DBSCAN se listan a continuación.

- DBSCAN es capaz de identificar puntos atípicos o ruido. Se considera **ruido** al conjunto de puntos que no pertenecen a ningún cluster.
- Los dos hiperparámetros utilizados por DBSCAN son ε y q. En la Tabla 2 se presenta un resumen.

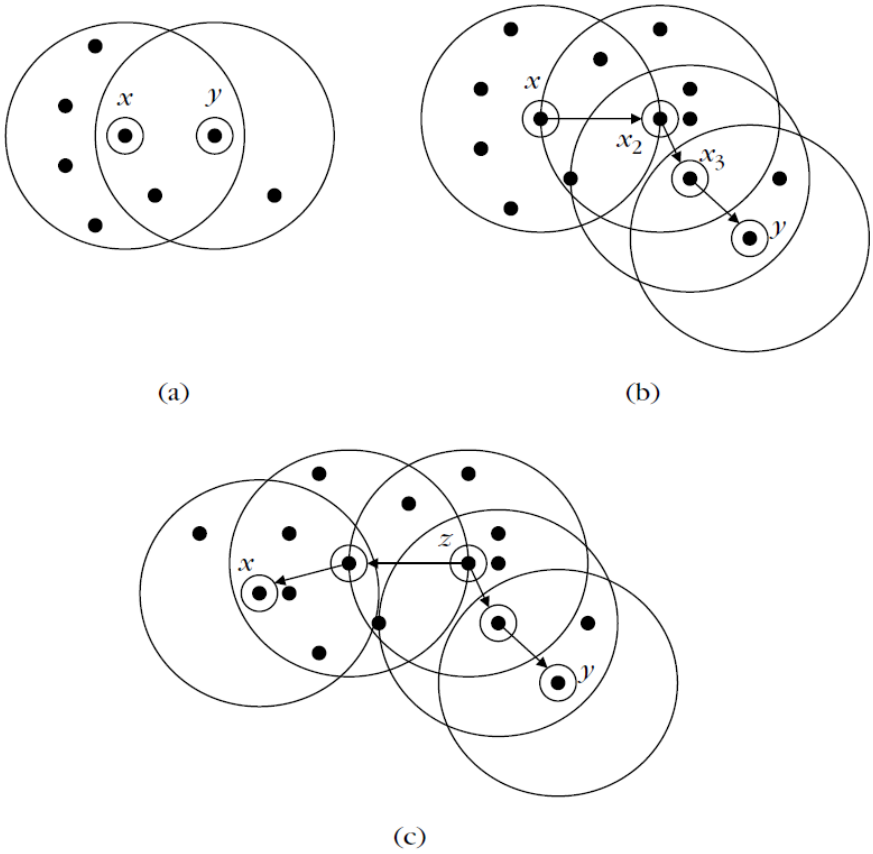

Figura 4: Ejemplos de relaciones entre puntos asumiendo $q = 5$: en (a) y es directamente alcanzable por densidad desde x, en (b) y es alcanzable por densidad desde x y en (c) x e y están conectados por densidad [44].

- La habilidad de DBSCAN para identificar regiones con alta densidad permite que la forma de sus grupos sea determinada por el conjunto de datos (X), a diferencia de otros métodos como K-means, que asume grupos de forma esférica [11].

Hiperparámetro	**Descripción**	**Configuración**
Épsilon (ε)	Radio de la hiperesfera centrada en un punto	Selección en base al método *K-distance* [13]. Se sugiere utilizar ($2 \cdot dimensin - 1$) vecinos más cercanos [40]
Cantidad mínima de puntos (q)	Medida de densidad alrededor de un punto: cantidad de puntos en vecindad	Comúnmente $q = 4$. Para datos de alta dimensión se sugiere $q \geq 2 \cdot dimensin$ [40]

Tabla 2: Hiperparámetros de DBSCAN

2.5.3. Medidas de evaluación para agrupamiento

La evaluación de un modelo de aprendizaje no supervisado se realiza utilizando **índices internos** e **índices externos**. Los primeros apuntan a verificar si la estructura de los grupos se acomoda a los datos, usando solo información contenida en ellos [45]. Los segundos se utilizan para asociar la estructura de los grupos a una clasificación predefinida de los objetos, con el objetivo de validar los resultados [32]. En la Tabla 3 se resumen algunas métricas de evaluación.

Medida	**Valores**	**Descripción**
Índice de Davies-bouldin	$[0, \infty[$	Similitud promedio entre cada grupo y su grupo mas similar. Valores bajos indican la presencia de grupos compactos y bien separados.
Coeficiente de silueta	$[-1, 1]$	Mide en base a la distancia, cuan cercano es cada elemento a su propio grupo, en relación a el grupo más cercano. Valores cercanos a 1 indican un mejor agrupamiento.
Accuracy	$[0, 1]$	Indica la proporción de ejemplos clasificados correctamente, contra el número total de instancias. Accuracy igual a 1 implica una asignación perfecta
Normalized Mutual Information	$[0, 1]$	Mide la consistencia entre los resultados y las etiquetas verdaderas mediante el cálculo la entroía normalizada y la información mútua ára cuantificar la similitud. Valores cercanos a 1 implican buenos resultados.
Adjusted Rand Index	$[-0{,}5, 1]$	El RI calcula la similitud entre dos grupos considerando todos los pares posibles. ARI es esta medida ajustada. Valores cercanos a 1 indican una buena solución, mientras que una asignación aleatoria debería dar un valor cercanoa cero.

Tabla 3: Índices internos y externos para evaluación de agrupamiento.

Detalles de los índices internos:

1. **Coeficiente de silueta (S):** Se tiene una solución que particiona un conjunto de datos X en m grupos. La distancia promedio entre un punto x_i, asignado al grupo C_{c_i}, y los demás puntos del mismo grupo es $a_i = d^{ps}_{avg}(x_i, C_{c_i} - \{x_i\})$. La distancia entre el mismo punto y el grupo más cercano (sin considerar el propio) es $b_i = min[d^{ps}_{avg}(x_i, C_k)]$, donde $k = 1, ..., m$ y $k \neq c_i$. El tamaño de silueta para x_i se define como sigue:

$$s_i = \frac{b_i - a_i}{max(b_i, a_i)} \tag{3}$$

El valor de silueta para un grupo C_j con n_j elementos, se define como:

$$S_j = \frac{1}{n_j} \sum_{i:x_i \in C_j} s_i \tag{4}$$

Finalmente, el coeficiente de silueta de global sería:

$$S_m = \frac{1}{m} \sum_{j=1}^{m} S_j \tag{5}$$

2. **Índice de Davies-bouldin (DB):** Sea s_i una medida de dispersión del grupo C_i y d_{ij} una medida simétrica de disparidad entre los grupos C_i y C_j. Se puede definir un índice de similitud como $R_{ij} = \frac{s_i + s_j}{d_{ij}}$. Sea $R_i = max(R_{ij})$, con $i, j = 1, ...m$ e $i \neq j$. Se define entonces el índice de Davies-Bouldin como:

$$DB_m = \frac{1}{m} \sum_{i=1}^{m} R_i \tag{6}$$

Se tienen N objetos con etiquetas $Y = \{y_1, ..., y_N\}$ conocidas. Si estos objetos son agrupados en m grupos, obteniéndose como resultado las etiquetas $\hat{Y} = \{\hat{y_1}, ..., \hat{y_N}\}$ desde los grupos, e pueden calcular los siguientes índices externos:

1. **Accuracy (ACC):** se define de la siguiente manera [48]:

$$ACC(Y, \hat{Y}) = \frac{1}{N} \sum_{i=1}^{N} f(y_i, \hat{y}_i), \quad f(a, b) = \begin{cases} 1 & si \quad a = b \\ 0 & si \quad a \neq b \end{cases} \tag{7}$$

2. **Normalized Mutual Information (NMI):** Sea $I(a, b)$ la información mutua entre a y b, y $H(Y)$ la entropía de Y. El NMI se calcula segun la formula [48]:

$$NMI(Y, \hat{Y}) = \frac{I(Y, \hat{Y})}{\frac{1}{2}[H(\hat{Y}) + H(Y)]} \tag{8}$$

3. **Adjusted Rand Index (ARI):** Sea TP (*true positives*) los elementos de una clase k clasificados correctamente y TN (*true negative*) los elementos que no pertenencen a k, que no fueron clasificados como de dicha clase. Se puede calcular el índice Rand (RI) de la siguiente manera:

$$RI = \frac{TP + TN}{C_N^2} \tag{9}$$

donde C_N^2 representa el número total de pares de elementos que pueden ser formados en el conjunto de datos. El cálculo del ARI es el siguiente [48]:

$$ARI = \frac{RI - E(RI)}{max(RI) - E(RI)} \tag{10}$$

siendo $E(RI)$ la esperanza (valor esperado) de RI.

2.6. Deep learning

El desempeño de los algoritmos de ML depende de la manera en la que se presentan los datos [15]. Por ejemplo, si se tiene un conjunto de imágenes, cada una de ellas puede ser representada como un vector $x \in \mathbb{R}^n$, donde cada atributo $x_1, ..., x_n$ corresponde a un pixel. Para varias tareas, la construcción de la representación no es un trabajo trivial, ya que no se conoce *a priori* que atributos son importantes y cuales se deben extraer para representar a los objetos. **Representatio learning** (RL) es un conjunto de métodos cuyo propósito es aprender una representación de los datos que permita una extracción más fácil de información útil, cuando se construyen clasificadores u otros predictores [5].

Deep Learning (DL) es una rama del ML que consigue flexibilidad y capacidad mediante la representación del mundo como una jerarquía de conceptos, con cada concepto definido en relación a conceptos más simples, y representaciones más abstractas calculadas en términos de representaciones menos abstractas [15]. DL está basado en redes neuronales, las cuales son modelos computacionales que se inspiraron originalmente por los mecanismos de aprendizaje y procesamiento del cerebro humano.

Una red neuronal clásica está compuesta de diferentes capas o *layers* (ver Figura 5) que a su vez están compuestas por neuronas artificiales. De esta manera, si se tiene un vector de entrada $x = x_1, ...x_D$, la primera capa con M neuronas, calculará sus M salidas z_j $(j = 1, ..., M)$ de la siguiente forma [9]:

$$z_j = h(\sum_{i=1}^{D} w_{ji} x_i + w_{j0}) \tag{11}$$

donde w_{ji} corresponde a los pesos, w_{j0} a los bíases y $h(.)$ a una función de activación. La salida es posteriormente utilizada por la siguiente capa como entrada para realizar el mismo proceso. De esta manera, de forma generalizada, para la capa l el cálculo está dado por [9]:

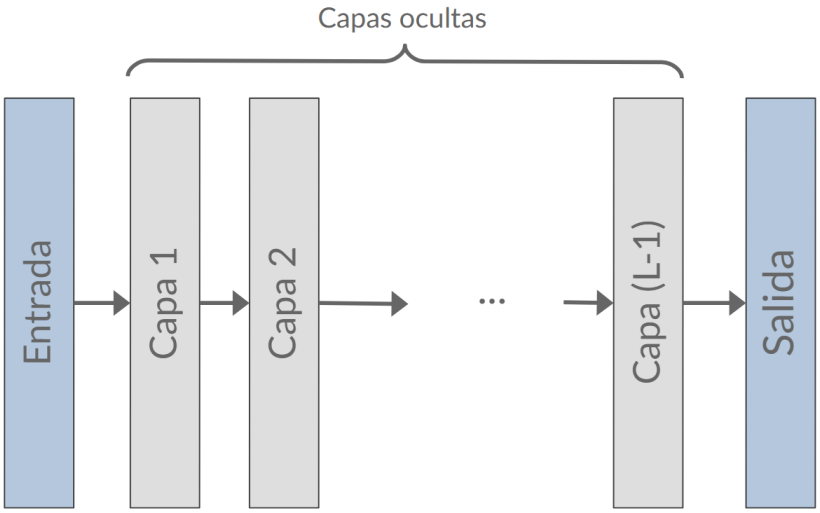

Figura 5: Esquema básico de una red neuronal profunda con L capas.

$$z^{(l)} = h^{(l)}(W^{(l)} z^{(l-1)}) \tag{12}$$

Cabe aclarar que en la Ecuación 12 el bias está incorporado en la matriz W y como una entrada adicional en $z^{(l-1)}$. Las capas intermedias a la entrada y la salida se conocen como capas ocultas o *hidden layers*.

Se puede ver a la sucesión de capas de una red neuronal como transformaciones realizadas sobre los datos, que facilitan la resolución de un problema dado. Esta habilidad para descubrir transformaciones no lineales de los datos es también RL. La Figura 6 muestra la relación entre DL, ML y RL.

2.6.1. Redes convolucionales

Las redes convolucionales (CNN, *Convolutional Neural Network*) son un tipo de red neuronal especializada en el proceso de datos de topología cuadricular, como las imágenes. Una característica particular de las CNNs es su capacidad de extraer dependencias espaciales o temporales a través de la aplicación de un conjunto de pequeños filtros espaciales (*convolutional kernels*) [23].

Una CNN usa convolución en lugar de multiplicación general de matrices en al menos una de sus capas [15]. La arquitectura básica de una CNN corresponde a una secuencia de 3 operaciones repetidas n veces: una capa convolucional, una capa de pooling y una función de activación.

La **convolucion** (en DL, que se diferencia de la convolución matemática) se calcula de la siguiente manera [8]:

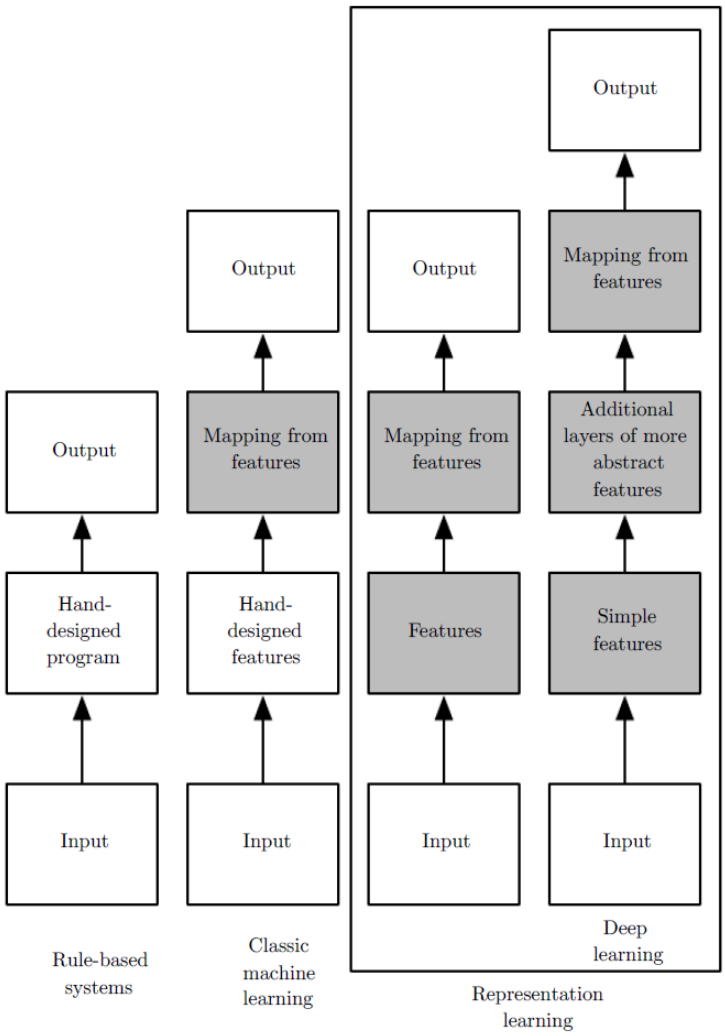

Figura 6: Diagrama de las partes de diferentes enfoques de inteligencia artificial. Los cuadros sombreados representan componentes con la capacidad de aprender [15].

$$C(j,k) = \sum_{l}\sum_{m} I(j+l, k+m)K(l,m), \tag{13}$$

donde I es una imagen con pixeles $i(j,k)$, y K es un filtro en dos dimensiones con valores $K(l,m)$. La Figura 7 muestra un ejemplo gráfico de la Ecuación 13. La convolución se aplica comúnmente en una dimensión (el tiempo en series de tiempo) y en dos dimensiones (largo y ancho en imágenes).

El **Pooling** es una función aplicada a la salida de una convolución, que reduce el tamaño de la convolución en una manera controlada. Las capas de Pool submuestrea el resultado de la capa previa. La Figura 8 muestra la aplicación de pooling (*MaxPooling*) sobre una entrada de 4×4, utilizando un filtro de 2×2.

Algunos conceptos comunes en los filtros de capas de Pooling y Convolución son:

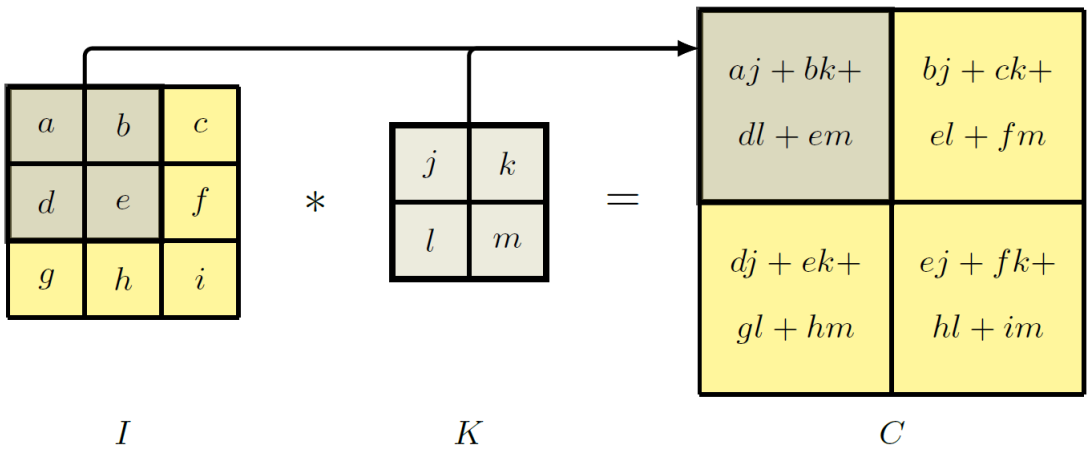

Figura 7: Imagen de 3×3 (I) convolucionada con un filtro 2×2 (K)[8].

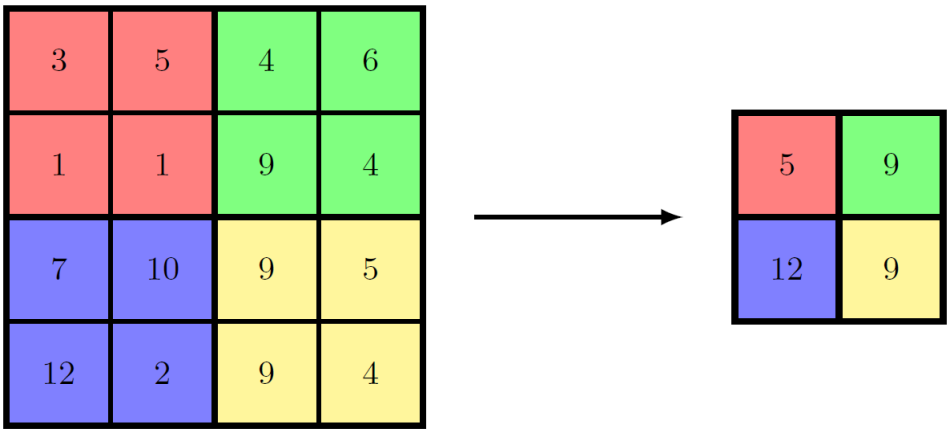

Figura 8: Aplicación de Pooling [8].

- **Padding:** Es una *extension* realizada en los bordes de los datos con el objetivo de controlar el tamaño de la salida.
- **Stride:** Es el tamaño del paso del movimiento de los filtros a través de las dimensiones.
- **Tamaño del filtro:** Es el tamaño o dimensiones de los filtros o *kernels* utilizado.

2.6.2. Autoencoder

Los autoencoders (AE) son un tipo específico de red neuronal diseñada para aprender una representación eficiente de datos no etiquetados, típicamente con el propósito de realizar reducción dimensional o para RL [26]. La red consta de dos partes: una función de codificación (*encoder*) $h = f(x)$ y un decodificador (*decoder*) que produce una reconstrucción $r = g(h)$ [15]. El codificador comprime la entrada y genera una representación diferente, denominada representación latente (*latent representation*), cuya característica es la de

contener las propiedades útiles de la entrada original. Esto se logra mediante la utilización de capas ocultas que funcionan como *cuello de botella* y fuerzan a red a priorizar aspectos prominentes en los datos e ignorar ruido y detalles poco importantes [26]. Durante el entrenamiento, la red mueve los datos a través del codificador y luego el decodificador. El objetivo es reducir las diferencias entre la entrada original y la salida. Es por esto que el proceso de entrenamiento de un AE involucra optimizar los parámetros del modelo con el fin de minimizar estas diferencias. La Figura 9 muestra la arquitectura usual de un AE. La entrada $x \in \mathbb{R}^D$ es transformada por el codificador F_1 para obtener la representación latente $z \in \mathbb{R}^M$, donde $D > M$. Posteriormente, el decodificador F_2 intenta rearmar la entrada original, constituyendo la salida $y \in \mathbb{R}^D$.

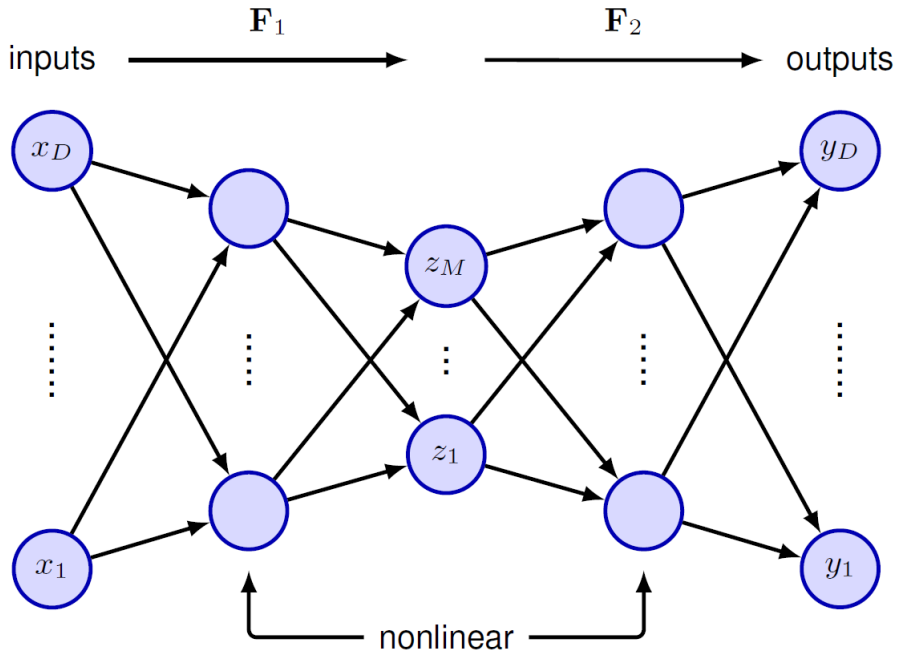

Figura 9: Diagrama de un AE [7].

2.6.3. Deep Clustering

El éxito del DL en tareas de aprendizaje supervisado ha inspirado el desarrollo de algoritmos de DL para agrupamiento que aprovechan la capacidad de estos para construir una nueva representación para los datos [30]. La mayoría de estos algoritmos tienen como objetivo principal generar una mejor representación y emplean el agrupamiento como un paso esencial para este propósito. Ren et al. en [38] afirman que estos modelos apuntan a extraer características más *amigables* para el proceso de agrupamiento. Todas estas técnicas son conocidas como **deep clustering**.

Entre las ventajas que ofrece el deep clustering sobre los métodos tradicionales están la incorporación de RL en el proceso, la captura de relaciones no lineales en los datos y la capacidad de reducción dimensional [48].

Nuttaki et al en [30] ofrecen una taxonomía para métodos de deep learning viéndolos como procesos:

- **Sequential Multistep Deep Clustering:** Este enfoque consiste en dos pasos bien definidos. El primero apunta a aprender una mejor representación de los datos de entrada (*latent representation* o representación latente), mientras que en el segundo paso se realiza el agrupamiento con técnicas de ML convencionales (Figura 10, arriba).
- **Joint Deep Clustering:** Esta familia de métodos realiza un solo paso para el RL y agrupamiento. Esto se consigue generalmente mediante la optimización de una función de pérdida que favorece una buena representación mientras toma en cuenta algún criterio de agrupamiento (Figura 10, centro).
- **Closed-loop Multistep Deep Clustering:** Esta familia de métodos, de manera similar a el Sequential Multistep Deep Clustering, consta de dos pasos. La diferencia radica en que ambos pasos se realizan en un ciclo y de manera alternada (Figura 10, abajo).

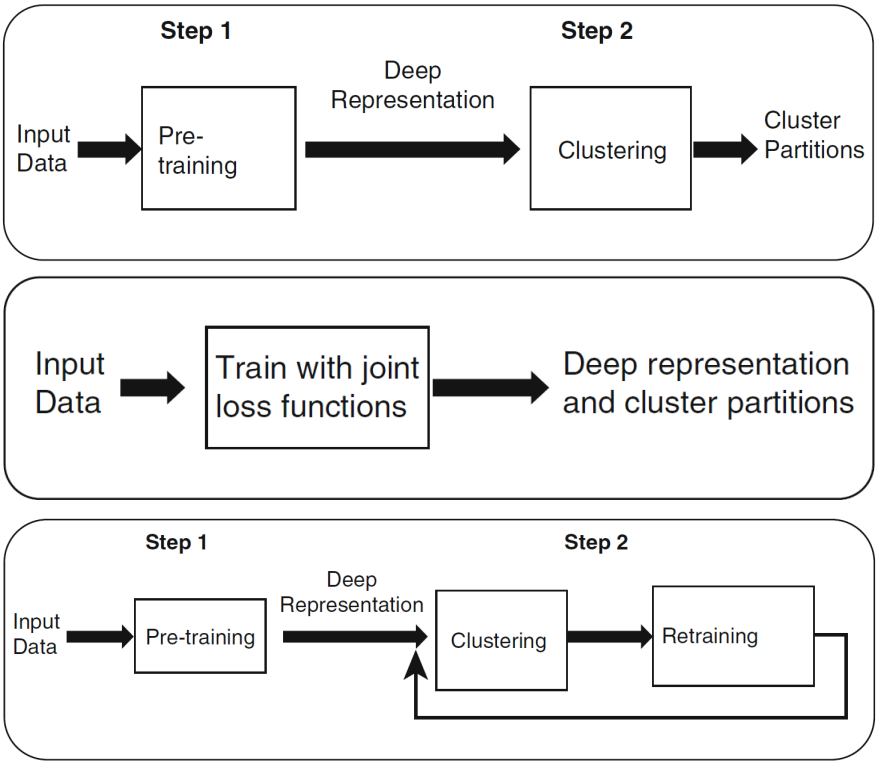

Figura 10: Sequential Multistep Deep Clusterin (arriba), Joint Deep Clustering (centro) y Closed-loop Multistep Deep Clustering (abajo) [30].

Ren et al. [38] dividen los métodos de deep clustering en 4 categorías principales de acuerdo a una perspectiva de las fuentes de datos y las condiciones iniciales. Si se tiene

un conjunto de datos X y un proceso de agrupamiento F cuyo resultado son las etiquetas Y, los métodos de deep clustering pueden ser:

- **Deep single-view clustering:** El agrupamiento se realiza con datos que tienen la misma estructura (*single-view data*). El proceso es $F(X) \rightarrow Y$.
- **Deep clustering based on semi-supervised learning:** Cuando los datos tienen restricciones asociadas, el agrupamiento puede utilizar esta información de manera semi-supervisada. El proceso es $F(X, A) \rightarrow Y$, donde A es una matriz de restricciones.
- **Deep clustering based on multi-view learning:** El agrupamiento se realiza sobre datos que pueden haber sido obtenidos de diferentes fuentes o tener diferente estructura (*multi-view data*). El propósito es utilizar información complementaria contenida en este tipo de datos. El proceso es $F(X^1, ..., X^n) \rightarrow Y$, donde X^i es la iésima *perspectiva* de X.
- **Deep clustering based on transfer learning:** Métodos basados en *Transfer Learning*, es decir, métodos que apuntan a mejorar el desempeño de la agrupación utilizando la información de problemas diferentes pero similares. El proceso es $F(X^s, Y^s, X) \rightarrow Y$, donde (X^s, Y^s) son los datos y etiquetas del problema o tarea del que se obtiene información.

3. Estado del arte

En este capítulo se presenta la revisión de literatura centrada en el agrupamiento de series de tiempo con deep learning. En la Sección 3.1 y la Sección 3.2 se muestran trabajos relizados mediante ML y DL respectivamente, mientras que en la Sección 3.4 se ofrece un resumen y se puntualizan las conclusiones obtenidas.

3.1. Agrupamiento de series de tiempo con machine learning

Los algoritmos de machine learning tradicionales de aprendizaje no supervisado han sido utilizados para el monitoreo de condiciones. Por nombrar algunos casos, en [50] se utilizó k-means para el diagnóstico automatizado de rodamientos, mientras que en [31] se implementó el mismo algoritmo, en conjunto con otros modelos de aprendizaje supervisado, para prevenir que columnas de destilación y extracción se desborden. El objetivo de k-means era crear grupos de acuerdo a la caida de presión registrada en la columna de destilación de banda giratoria. Altindal et al. en [2] usaron Análisis de Componentes Principales (PCA, del inglés *Principal Component Analysis*) e Isolation Forest (IF) contra un autoencoder recurrente (RNN-AE) para la detección de anomalías en el proceso de perforado de pozos. Si bien este último obtuvo los mejores resultados, PCA mostró un desempeño aceptable para casos en los cuales las anomalías eran escasas. En [24] los autores trabajan en la predicción de la vida útil de cojinetes. Uno de los desafíos en el problema es que las variaciones en las condiciones de trabajo hacen inapropiada la utilización de los datos históricos acumulados en el entrenamiento de modelos. Para abordar el problema, el estudio propone *transfer learning* (en español, aprendizaje por transferencia) en conjunto a un algoritmo de agrupamiento jerárquico aglomerativo (HAC, del inglés *Hierarchical Agglomerative Clustering*). Transfer learning se realiza cuando el aprendizaje para una tarea B es usado para resolver una tarea distinta A. Para esto se requiere que A y B compartan algo en común, de tal forma que las características de bajo nivel o la representación interna aprendida en B resulte ventajosa para para A.

La Tabla 4 sintetiza los estudios con ML.

3.2. Agrupamiento de series de tiempo con deep learning

Alqahtani et al. en [1] recopilan estudios que utilizan deep clustering para series de tiempo abarcando diferentes dominios. En el trabajo se recopilan 11 estudios que utilizaron Sequential Multistep Deep Clustering y 9 que usaron Joint Deep Clustering. Los autoencoder convolucionales (CAE, del inglés *Convolutional AutoEncoder*) y autoencoder recurrentes (RNN-AE) son los que se utilizaron más frecuentemente para obtener la repre-

Algoritmo	Dominio	Motivo	Estudio
k-means	diagnóstico de maquinaria	diagnóstico de rodamientos	[50]
k-means	diagnóstico de maquinaria	Identificar estados de flooding en columnas de destilación	[31]
PCA, IF	diagnóstico de procesos	identificación de fallas en proceso de excavación	[2]
HAC	diagnóstico de maquinaria	Crear grupos que identifiquen procesos de degradación similares	[24]
k-means	diagnóstico de maquinaria	Analizar ciclos de potencia en maquinaria	[41]

Tabla 4: Tabla comparativa de estudios que hacen uso de machine learning convencional no supervisado

sentación latente, mientras que el algoritmo k-means predomina la escena como método de agrupamiento, independiente del tipo de deep clustering.

El deep clustering aplicado sobre series de tiempo tiene gran utilidad en la industria de la salud, pudiendo dar asistencia a médicos durante la evaluación de exámenes fisiológicos. En [47] se analizan las señales de electrocardiogramas con el objetivo de encontrar patrones o estructuras en los datos mediante agrupamiento. En el estudio se utiliza *continuos wavelet transforms* sobre la señal del electrocardiograma para obtener los atributos, y luego se ocupa una red convolucional en conjunto con k-means (Joint deep clustering usando CAE y k-means) para obtener los resultados. En el mismo campo, en [46], se analizan electrocardiogramas señalando la utilidad de reconocer los exámenes anormales. Los autores utilizan un autoencoder para obtener una representación latente de los datos y posteriormente dividirlos en grupos mediante el uso de Permutation Distribution Clustering (PDC); un método jerárquico agromerativo (Sequential multistep clustering). La particularidad de este estudio es el uso de un Algoritmo Genético (AG) para encontrar una buena arquitectura para el AE. Las pruebas mostraron resultados favorables a esta aproximación.

Se ha utilizado deep clustering para series de tiempo en diversos dominios. En [39] se realiza agrupamiento para generar perfiles de consumo eléctrico estacional y diario para clientes, utilizando datos de Corea del Sur. En el estudio se utiliza un autoencoder convolucional (CAE) para obtener una nueva representación de los datos, reduciendo la dimensión de cerca de 8000 pixeles en la imagen original, a un vector de 100 elementos después de el proceso de codificación. Los grupos son creados utilizando k-means.

En [21], con el objetivo de monitorear las condiciones térmicas de un edificio, se realiza un *temporal clustering* utilizando las variaciones temporales de temperatura. Estas, junto

con las variaciones de humedad pueden causar daños en las superficies internas y externas de un edificio, lo que conlleva diversas implicancias económicas. Una red neuronal convolucional combinada con una red neuronal LSTM (Long Short-Term Memory) fue utilizada para obtener la representación latente, mientras que el algoritmo k-means creaba los grupos. Ambos procesos se llevaron a cabo simultáneamente (Joint deep clustering, Autoencoder híbrido junto a k-means).

En el contexto del **monitoreo de condiciones en maquinaria**, el agrupamiento se ha utilizado principalmente para la identificación de *estados no deseados*, frente a estados de funcionamiento normales. El objetivo común es anticipar situaciones que representen un riesgo para el funcionamiento mediante la identificación de los patrones propios de un estado de falla. En [51] se busca detectar fallas en maquinaria o estructuras mediante mediciones de vibración, como las que generan los motores de vehículos. Los autores utilizan una aproximación secuencial. Una red neuronal convolucional funciona como un autoencoder cuyo objetivo es obtener una representación. Posteriormente se utilizó k-means para obtener los centroides que servirán para clasificar nuevas señales en base a su distancia hacia estos (Sequential multistep clustering, CAE junto a k-means).

Algoritmo	**Dominio**	**Motivo**	**Estudio**
CAE, k-means	Medicina	Identificación de patrones en electrocardiogramas	[47]
CAE, k-means	Industria energética	Agrupar perfiles de clientes según su consumo eléctrico	[39]
AE, AG, PDC	Medicina	Agrupar electrocardiogramas anormales y normales	[46]
AE híbrido, k-means	Rendimiento térmico	Agrupar variaciones temporales de temperatura en edificios	[21]
CAE, k-means	diagnóstico de maquinaria	Agrupar de acuerdo a la condición de operación utilizando vibraciones	[51]
GCN-AE, Método jerárquico de agrupamiento	Industria energética	Identificar modos de funcionamiento en turbinas de viento para posteriormente hacer predicciones	[49]

Tabla 5: Tabla comparativa de estudios que utilizan *deep clustering*

Yang et.al. utilizaron un Graph Convolutional Autoencoder (GCN-AE) en conjunto a un método jerárquico de agrupamiento basado en Toeplitz Inverse covariance para detectar el modo de funcionamiento de turbinas de viento [49] (Joint deep clustering, GCN-AE junto a un algoritmo jerárquico). Una característica del problema es que las turbinas de viento están conformadas por subsistemas. Por este motivo los datos tienden a agruparse en

torno a grupos delimitados por los propios subsistemas. Esto genera que algunas variables tengan una fuerte correlación en el interior de un grupo, y una correlación entre grupos deducible a partir del conocimiento experto. Este escenario fue el que llevó a los autores a utilizar una estructura jerárquica para las variables.

La Tabla 5 resume estudios con deep learning para agrupamiento.

3.3. Otros métodos no supervisados con deep Learning

Otra idea utilizada para detectar estados anormales en procesos es evaluando la reconstrucción de los datos creada por un autoencoder. La red neuronal recurrente usada en [2], un LSTM-AE, detecta las fallas en datos sin etiquetar, identificando discrepancias en la reconstrucción de los valores originales tras la codificación y decodificación. Para este caso se tienen dos estados posibles: falla o no falla. Ambos estados están delimitados por un umbral que fue definido a través de pruebas.

3.4. Síntesis

Entre los estudios revisados, k-means se posiciona como el algoritmo de machine learning más popular para agrupar series de tiempo bajo el framework del deep clustering. En el ámbito de las arquitecturas de redes neuronales, las RNN y CNN son populares para su utilización en autoencoders. Para estos casos, su uso principal radica en la obtención de una representación latente a la que posteriormente se le aplica un algoritmo tradicional que divide los datos en grupos. Como las series de tiempo son datos secuenciales, las RNN se presentan como una solución indicada para su procesamiento. En el caso de las CNN, como estas sobresalen capturando características locales en los datos, han mostrado un buen desempeño que justifica su utilización. Particularmente, se suelen utilizar capas convolucionales de 1 dimensión. Estas capturan la información local utilizando filtros que se mueven a través de la dimensión temporal, en los n canales que componen a la serie de tiempo multivariada.

Además de lo ya explicado, es importante realizar los siguientes puntos:

- La búsqueda de literatura solo encontró un estudio que utiliza datos industriales reales con técnicas de deep learning no supervisado. En otros trabajos, como [51] y [24], se utilizaron conjuntos de datos de laboratorios para simular el comportamiento de ciertos sistemas. La falta de estudios con conjuntos de datos reales se constata también en [33], donde se señala como un obstáculo para el desarrollo de técnicas para la detección de anomalías.
- Todos los estudios utilizan conjuntos de datos que cuentan con etiquetas (*ground truth*) disponibles. Esto no es algo seguro para conjuntos de datos reales.

- Se observan pocos estudios de MC en maquinaria industrial.

4. El problema

En este capítulo se describe la problemática y se plantea una solución. En la Sección 4.1 se presenta el desafío del monitoreo de condiciones conducido por datos, mientras que en la Sección 4.2 y la Sección 4.3 se desarrollan las preguntas de investigación, la hipótesis y se plantean los objetivos. Finalmente, en la Sección 4.4 se define el modelo y los algoritmos que se utilizarán.

4.1. Planteamiento del problema

La industria depende de maquinaria que debe operar en óptimas condiciones para generar el producto final sin contratiempos. Fallas en la maquinaria traen consigo demoras que pueden significar un aumento en el tiempo que tarda un proceso en completarse, tiempos muertos de no producción y posibles accidentes. Por estos motivos es necesario mantener un control y un monitoreo del sistema que permita detectar a tiempo cualquier problema.

Las condiciones causantes de estos problemas son muchas veces inciertas y pueden ser ajenas al control de la planta. Esto es debido a que no se conocen las variables exactas que hacen que un sistema falle. Este tipo de situación puede ser tratada mediante el uso de Inteligencia Artificial (IA). Con sensores instalados en las máquinas industriales se puede generar un gran volumen de datos que indique el estado de la maquinaria y de manera constante. Tener tal cantidad de datos resulta idóneo para la realización de **monitoreo de condiciones conducido por datos** (ver Sección 2.1). Sin embargo, analizar tantos registros es un proceso costoso que se beneficia de la automatización: obtención automática de datos y ejecución modelos de IA. En la industria de la manufactura, la automatización afecta positivamente en la calidad y los plazos de producción [34]. Pese a esto, la utilización de modelos de IA automatizados para el monitoreo de condiciones no es algo que se aplique en todas las instancias industriales en las que se podría usar.

Con el objetivo de conocer cuando un sistema o maquinaria está fallando, se requiere tener la capacidad de reconocer el estado de operación del equipo. Para esto es necesario catalogar cuales son los diferentes estados por los cuales puede pasar el sistema y cuales de ellos solo aparecen cuando hay una situación de fallo. La caracterización de los estados de funcionamiento es un paso previo a su identificación. Esta tarea requiere el uso de datos históricos, ya que el estado para un tiempo t es producto de lo que ocurre en el mismo instante t y de lo que ocurrió en el sistema a partir de n unidades de tiempo atrás. De esta manera, los datos que se requieren para evaluar el funcionamiento de un equipo o un sistema corresponden a series de tiempo (ver Sección 2.2).

La dependencia temporal presente en las series de tiempo implica que los algoritmos para su procesamiento deben ser adecuados para el trabajo con secuencias temporales. Además, cuando no se cuenta con un conocimiento previo sobre el número de estados posibles o la estructura de cada uno de estos estados, la caracterización se debe llevar a cabo de manera no supervisada. Los algoritmos de agrupamiento son modelos no supervisados que permiten, mediante el descubrimiento de las estructuras subyacentes en los datos, identificar grupos que pueden representar los estados de funcionamiento.

4.2. Propuesta

El monitoreo de condiciones es deseable en cualquier industria. La tecnología actualmente permite la generación y almacenamiento de grandes cantidades de datos, por lo que el monitoreo de condiciones conducido por datos es una estrategia realista, adicional a otros modelos. De lo anterior, y teniendo en cuenta lo discutido en la Sección 4.1, se plantean las siguientes preguntas que darán lugar a la hipótesis: ¿Serán efectivos las técnicas de deep clustering si se aplican en datos industriales reales?¿Cuáles algoritmos de agrupamiento serán efectivos dentro del marco de trabajo del deep clustering?

Hipótesis
Es posible realizar monitoreo de condiciones para maquinaria industrial usando datos reales, mediante la utilización de técnicas de deep learning no supervisado sobre series de tiempo.

4.3. Objetivos

Objetivo general:
Evaluar un modelo de Deep Learning no supervisado que sea capaz de separar series de tiempo generadas por maquinaria industrial real, de tal forma de que sea posible identificar diferentes estados de funcionamiento, con especial énfasis en estados anómalos.

Objetivos específicos:

1. Identificar modelos de deep clustering adecuados para el agrupamiento de series de tiempo.
2. Implementar modelos de Deep clustering sobre series de tiempo generadas por un proceso industrial real.
3. Evaluar los modelos de aprendizaje no supervisado, analizando su desempeño de manera comparativa, en conjunto con el conocimiento experto.
4. Identificar los grupos anómalos en los datos.

4.4. Selección de modelo

Para el proceso de agrupamiento se decidió utilizar **Sequential Multistep Deep Clustering** (ver Sección 2.6.3). Este tipo de aproximación, al estar compuesto por 2 pasos fácilmente diferenciables, resultan sencilla de implementar. Adicionalmente, el algoritmo de agrupamiento utilizado en el segundo paso puede ser remplazado sin requerir un re entrenamiento del AE. La Figura 11 esquematiza el procedimiento llevado a cabo para un conjunto de datos dado. El paso 1 corresponde al entrenamiento de un modelo de autoencoder convolucional y el paso 2 es la utilización de algoritmos de ML sobre la representación latente para obtener los grupos finales.

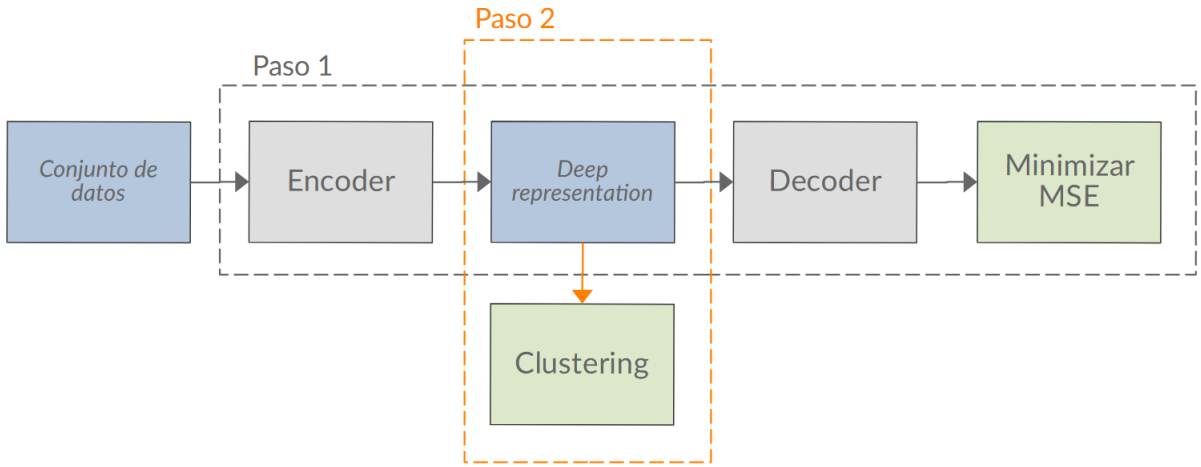

Figura 11: Proceso de deep clustering dividido en dos pasos.

Los CAE, junto con los RNN-AE, son ampliamente utilizados para el trabajo con series de tiempo debido a que pueden tomar en consideración el componente temporal de los datos en su funcionamiento. Ambas han mostrado resultados favorables para el manejo de series de tiempo. Esto se constata tanto en la revisión bibliográfica (Sección 3.2) como en múltiples reviews [33, 1, 3]. Teniendo presente que ambas opciones son efectivas, en este estudio se escogió un CAE por sobre un RNN-AE. Esto es debido a que las arquitecturas convolucionales han mostrado resultados a la par con las RNN para algunas tareas de predicción con datos secuenciales [22], tienen un tiempo más corto de entrenamiento y usualmente son menos costosas computacionalmente [3].

Para la realización del agrupamiento sobre la representación latente se escogieron 2 algoritmos diferentes: **k-means y DBSCAN**. El primero, como se constata en la Sección 3.2, es el más utilizado para deep clustering en series de tiempo y ha demostrado producir buenos resultados. La elección de DBSCAN se sustenta en la posibilidad de que la estructura de los datos no favorezca algoritmos que tiendan a formar grupos circulares, como k-means. Ambos algoritmos agrupan datos bajo diferentes ideas (basado en particiones y basado en densidad). DBSCAN tiene la capacidad de reconocer grupos con formas más

irregulares. Si los datos utilizados son de esta naturaleza, DBSCAN es idóneo para el trabajo.

5. Metodología

En este capítulo se describen los detalles de la metodología utilizada para cumplir con los objetivos señalados en la Sección 4. La Figura 12 esquematiza la metodología general.

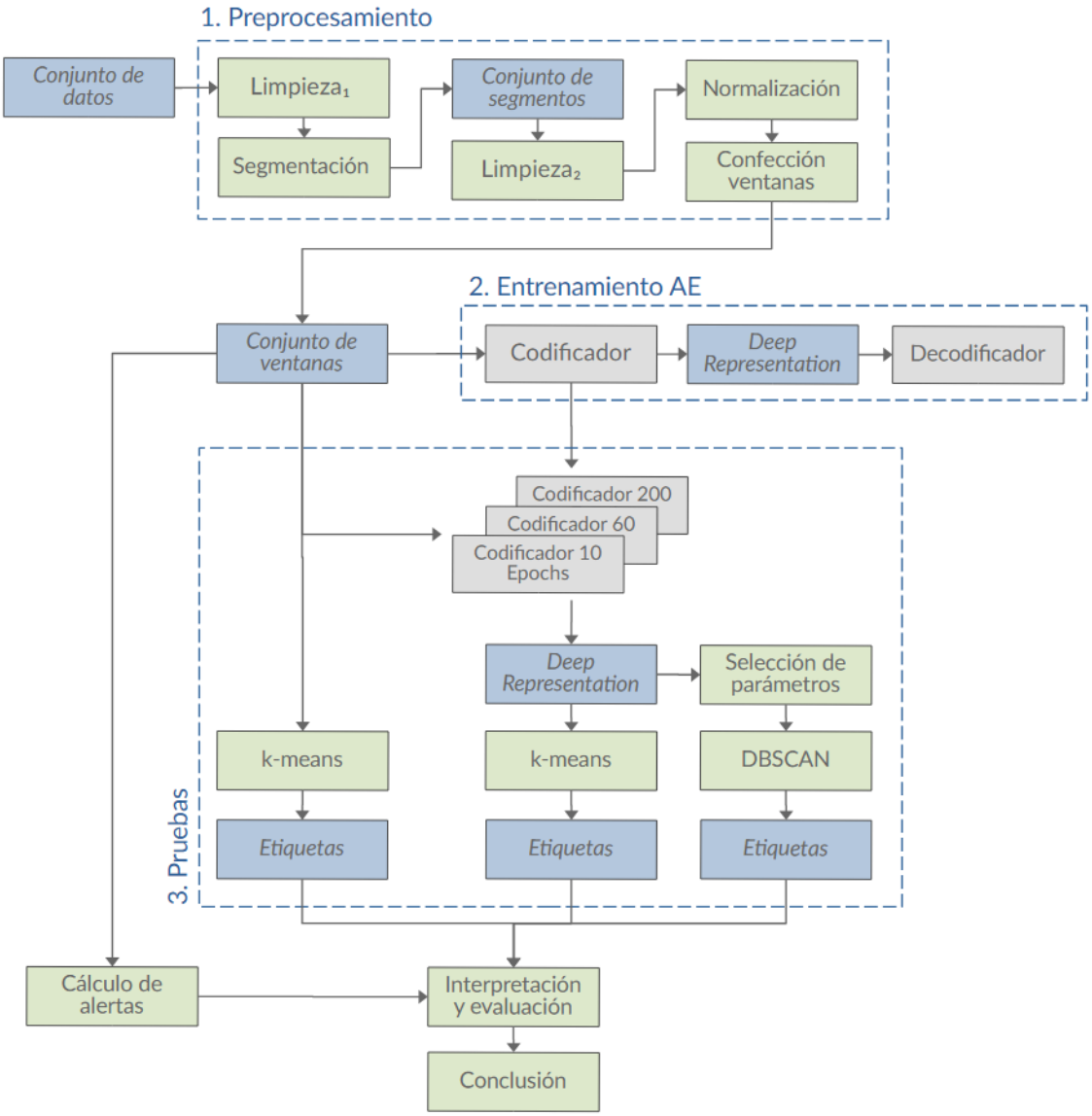

Figura 12: Metodología.

A partir de una serie de tiempo generada por un proceso industrial real, se configuran las siguientes etapas:

1. **Preprocesamiento:** Esta etapa, que está constituida por varias subetapas, tiene como fin extraer los datos útiles y moldearlos para que puedan ser aprovechados por el proceso de deep clustering.

2. **Entrenamiento AE:** El AE es entrenado con los datos ya preprocesados. Como resultado se generan diferentes codificadores que serán utilizados para RL en el proceso de deep clustering.

3. **Pruebas:** En esta etapa se realiza la selección de hiperparámetros para los algoritmos encargados de agrupar. Ya configurados, los algoritmos son utilizados en conjunto a los diferentes codificadores para generar los grupos finales.

Las secciones están organizadas de acuerdo al esquema: en la Sección 5.1 se presentan los detalles del conjunto de datos; en la Sección 5.2 se describe el preprocesamiento; la Sección 5.3 está dedicada a presentar detalles de la arquitectura y del entrenamiento del AE; la Sección 5.4 expone la selección de parámetros para las pruebas de los algoritmos de ML que formarán parte del deep clustering y finalmente, la Sección 5.5 explica la manera en la que se llevará a cabo la evaluación de resultados.

La implementación de los modelos y el manejo de los datos se realizó con el lenguaje de programación Python. Los detalles se pueden ver en la Tabla 6. Los experimentos fueron ejecutados en una máquina con procesador Intel(R) Core(TM) i9-13900KF, 64 Gb RAM y una GPU NVIDIA GeForce RTX 4080.

Tareas	**Bibliotecas Python**	**Otros**
Exploración de datos	Pandas (2.3.3), Matplotlib (3.10.8), Seaborn (0.13.2)	PostgreSQL, hojas de cálculo
Preprocesamiento de datos	Pandas (2.3.3), scikit-learn (1.8.0) [36]	
Implementación del autoencoder	PyTorch (2.9.1+cu126)	
Implementación de los algoritmos de agrupamiento	scikit-learn (1.8.0)	
Análisis de resultados	ScyPy (1.16.3), Matplotlib (3.10.8), Seaborn 0.13.2, scikit-learn (1.8.0)	Hojas de cálculo

Tabla 6: Herramientas utilizadas para los experimentos.

5.1. Conjunto de datos

Como caso de estudio se ha utilizado un conjunto de datos de origen industrial con el funcionamiento de chancadores mineros. Estas máquinas tienen el propósito de triturar materiales como rocas para así reducirlas a un tamaño adecuado para algún proceso. Los datos son reales y han sido obtenidos desde sensores ubicados en las máquinas. PI System

es utilizado para recoger y guardar los datos en una base de datos de donde, posteriormente, se extraen utilizando el formato CSV. En la Figura 13 se muestran 2 modelos de chancadores de cono diferentes: un chancador de cono compuesto HAMAC[3] (izquierda) y un chancador CH870i de Sandvik[4] (derecha). Las imágenes son solo referenciales.

Figura 13: Chancadores de cono.

5.1.1. Descripción

El conjunto de datos se generó a partir del funcionamiento de cinco chancadores durante el periodo de dos años (2023 y 2024). Todos los chancadores pertenecen al mismo modelo, son de la misma marca y del mismo año. Los 5 operan de manera continua y en igualdad de condiciones. La Tabla 7 muestra la cantidad de registros disponibles por cada chancador y por año. Las máquinas están identificadas con números del 1 al 5.

Chancador	2023	2024	Total
1	522.367	523.9993	1.046.366
2	522.367	523.9993	1.046.366
3	522.367	523.9993	1.046.366
4	522.367	523.9993	1.046.366
5	522.367	523.9993	1.046.366

Tabla 7: Cantidad de registros por chancador y por año

[3]https://www.hamacinc.com/products/crushing-screen/compound-cone-crusher.htmls. Fecha de acceso: 06-04-2026

[4]https://www.rockprocessing.sandvik/en/products/stationary-crushers/stationary-cone-crushers/ch870i-cone-crusher. Fecha de acceso: 06-04-2026

Para cada uno de los chancadores, se registran 17 diferentes variables a intervalos constantes de un minuto. Por consiguiente, el conjunto de datos se compone por las series de tiempo multivariadas y discretas generadas por las diferentes máquinas. Todas las variables son numéricas, no categóricas y de valores continuos. Adicionalmente, para cada registro, se cuenta con un índice único de tipo timestamp que corresponde al instante del registro. A continuación se describen las 17 variables:

1. **Corriente motor chancador (CM):** Mide la corriente eléctrica consumida por el motor principal del chancador. Es un indicador directo de la carga de material y posibles atascos o sobre fuerzas. Su valor está expresado en porcentaje en razón de la potencia máxima de la electrónica en el aparato (750 Kw).

2. **Plano inclinado alimentación (PI):** Sensor que indica el porcentaje de llenado.

3. **Presión entrada aceite lubricación chancador (PEL):** Mide la presión (kPa) de alimentación del aceite al sistema. Una baja presión puede provocar fallas catastróficasăpor falta de lubricación.

4. **Presión diferencial filtro sistema lubricación (PDF):** Calcula la diferencia de presión (kPa) antes y después del filtro de aceite. Aumentos indican obstrucción del filtro o necesidad de mantenimiento.

5. **Velocidad cero alimentador (RA):** Sensor de detección de rotación en el alimentador. Detecta si el equipo está detenido (0 rpm) y se usa paraăinterlocks y seguridad.

6. **Velocidad cero chancador (RM):** Sensor de detección de rotación que monitorea el motor del chancador. Se usa para disparar alarmas o detener la alimentación.

7. **Temperatura aceite buje excéntrica (T7):** Monitorea la temperatura ($°C$) del aceite que lubrica el buje excéntrico, una pieza crítica que permite el movimiento giratorio del eje principal. Un aumento puede indicar falla de lubricación o fricción excesiva

8. **Temperatura socket liner (T8):** Controla la temperatura ($°C$) en la zona del socket liner (casquillo de soporte del eje), útil para detectar ădesgaste, sobrecarga o falta de lubricación.

9. **Temperatura descanso contraeje (T9):** Indica la temperatura ($°C$) en el cojinete del contraeje, parte que transmite el giro al excéntrico. Una temperatura alta puede sugerir alineación deficiente o daño mecánico.

10. **Temperatura aceite retorno sistema de lubricación (T1):** Mide la temperatura ($°C$) del aceite que retorna desde el chancador, útil para analizar el comportamiento térmico general del sistema. Aumento indica calentamiento interno.

11. **Temperatura aceite entrada chancador (T2):** Mide la temperatura ($°C$) del aceite antes de ingresar al chancador. Compararla con la de retorno permite detectar problemas de refrigeración o eficiencia del sistema.

12. **Temperatura aceite tanque de lubricación (T5):** Supervisa la temperatura ($°C$) dentro del tanque principal de lubricación, donde se almacena y enfría el aceite.

13. **Vibración anillo de rebote A (V1):** Sensor de vibración radial o axial (mm/s) ubicado en el anillo de rebote. Cambios pueden indicar desalineación, sobrecarga o material no chancable.

14. **Vibración anillo de rebote B (V2):** Sensor de vibración radial o axial (mm/s) ubicado en el anillo de rebote, en una sección diferente a VITA.

15. **Vibración anillo de rebote C (V3):** Sensor de vibración radial o axial (mm/s) que permite generar un patrón de vibración en múltiples posiciones para diagnóstico completo.

16. **Vibración anillo de rebote D (V4):** Último sensor del conjunto de 4 vibraciones del anillo. Un análisis conjunto permite detección de eventos anómalos o desequilibrios.

17. **Configuración chancador (Conf):** Valor del ajuste operativo de la abertura del chancador (CSS o OSS). Determina el tamaño del producto y afecta el consumo energético y desgaste. El valor se expresa en mm.

La Figura 14 muestra la distribución de las variables para el chancador 3 durante dos años de funcionamiento. Los datos incluyen periodos de no funcionamiento, por lo que algunas variables como las vibraciones y la corriente tienen una proporción notoria de valores cero. La Figura 15 muestra la correlación lineal entre las variables del mismo chancador en el mismo periodo. Se observan correlaciones altas (sobre 0,9) entre todas las vibraciones y entre la corriente y diferentes variables de temperatura, vibración y velocidad de rotación.

5.1.2. Alertas

Los chancadores son operados por expertos que hacen uso de 32 reglas que constatan si alguno de los componentes del chancador se encuentra funcionando dentro de ciertos límites conocidos. Cuando una de estas reglas se cumple, se gatilla una alerta que es informada a los operarios. Las reglas tienen diferentes categorías y pueden ser:

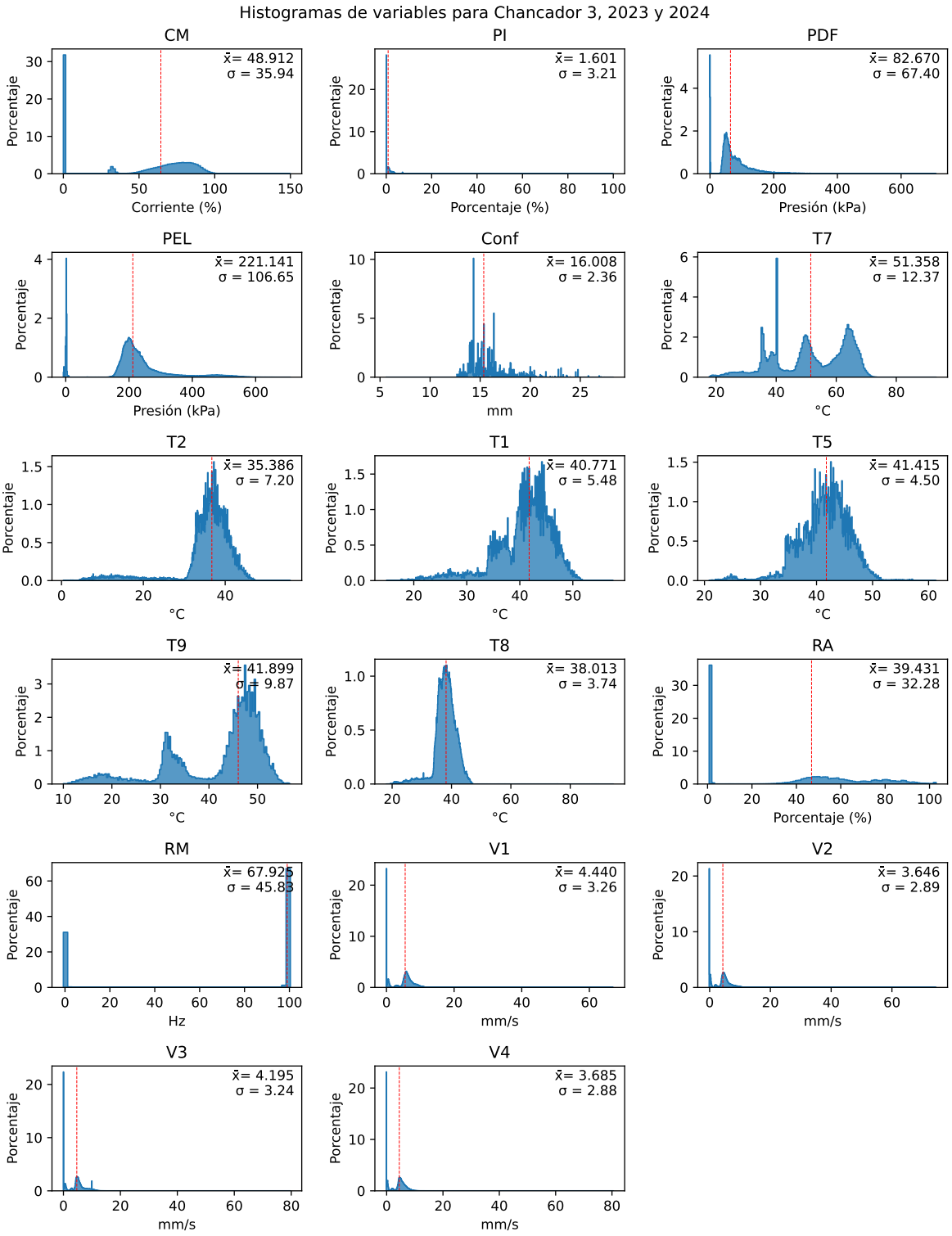

Figura 14: Histogramas de las variables del chancador 3 durante el 2023 y el 2024. La línea vertical roja marca la mediana.

- Límites inferiores o superiores para el valor de una variable
- Valores específicos que se prolonga en el tiempo

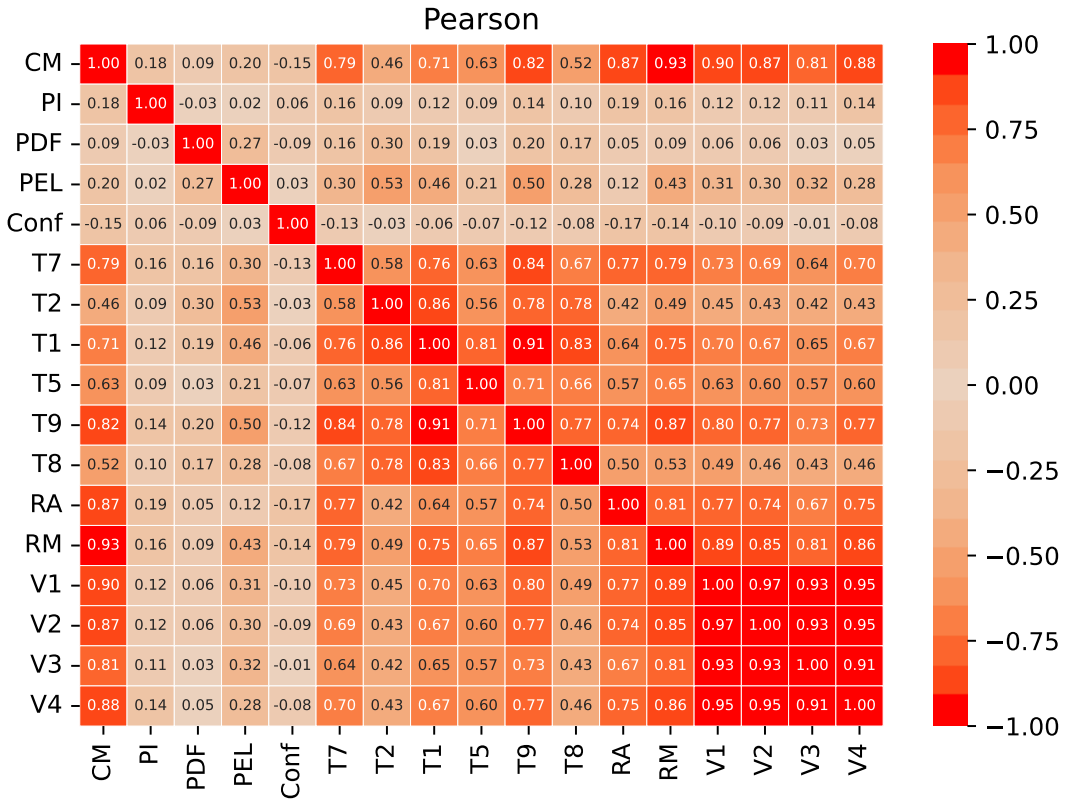

Figura 15: Matriz de correlación de variables para el chancador 3 durante 2023 y 2024.

- Límites inferiores o superiores para la diferencia de dos o más variables.

Cabe mencionar que cada una de estas reglas por sí mismas no implican mal funcionamiento.

5.2. Preprocesamiento

El preprocesamiento de los datos, tal como se señala en la Sección 2.3, es uno de los pasos en el framework general en la utilización de ML. Los datos de cada chancador fueron procesados siguiendo los pasos presentes en la Figura 16. En cada conjunto de datos, se aplicó una primera etapa de limpieza seguido de un proceso de segmentación que concluye con la generación de un conjunto de segmentos de datos continuos. Cada uno de estos segmentos es sometido a una segunda etapa de limpieza, a una normalización y a un proceso de confección de ventanas. Este último paso da como producto final un conjunto de ventanas, cada una de ellas con la información del funcionamiento de un chancador durante el periodo de una hora. Es este conjunto el que se utiliza con los modelos de ML y DL.

En esta sección se describen en detalle cada una de las etapas planteadas.

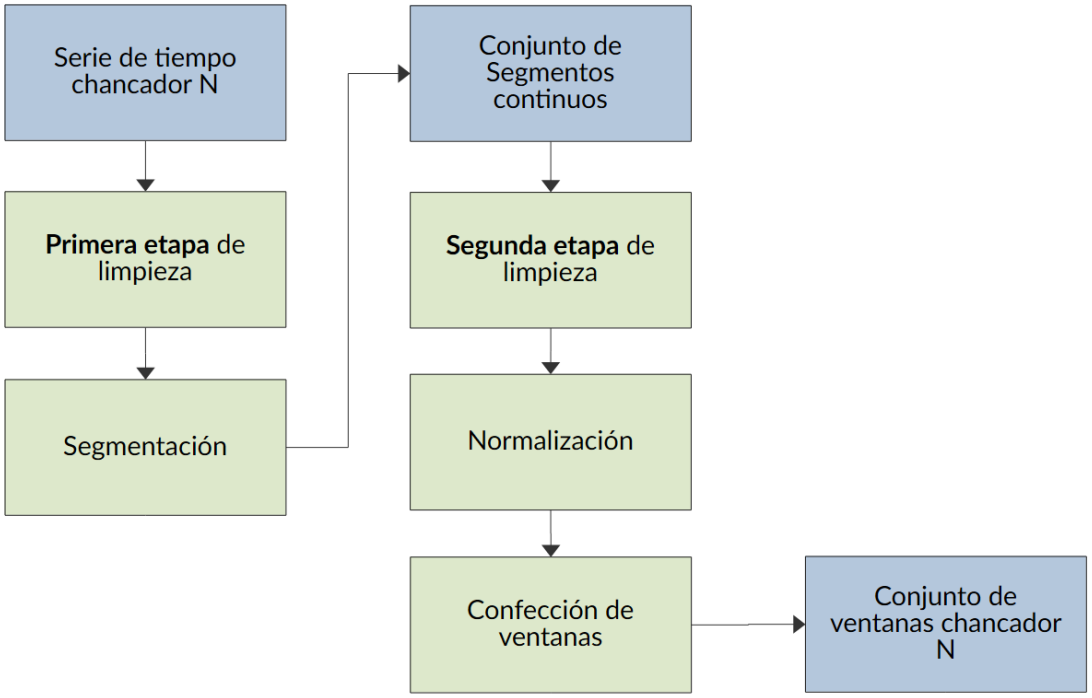

Figura 16: Diagrama con los pasos del preprocesamiento de datos para cada uno de los chancadores.

5.2.1. Limpieza de datos

El propósito de limpiar los datos es eliminar valores atípicos, completar valores faltantes y separar los datos útiles (chancador en funcionamiento) de los no útiles (chancador detenido o funcionando en vacío). La limpieza de datos se realiza en dos etapas:

- **Etapa 1:**

  1. **Eliminar las filas con todas las variables nulas:** Filas sin valores no tienen información.
  2. **Completar valores nulos de la variable Conf con el último valor registrado:** El valor de Conf se registra solo cuando hay un cambio.
  3. **Eliminar filas cuyo valor de CM sea inferior o igual a 25:** $CM = 0$ implica que el chancador está apagado. $0 < CM < 25$ implica que el chancador está funcionando en vacío. Como se quiere realizar MC sobre el chancador, estos dos estados no son de interés.
  4. **Eliminar filas cuyo valor de CM sea nulo:** Se ha decidido usar solamente registros en donde exista seguridad de que el chancador está en funcionamiento.

- **Etapa 2:**

    1. **Remplazar con nulo valores negativos en las variables PI, RA, T9 y T5:** Durante la exploración de los datos se constató que los valores negativos corresponden a errores del sensor.
    2. **Remplazar valores negativos de las variables VA, VB, VC y VD con 0:** Durante la exploración de los datos se constató que los valores negativos ocurrian esporádicamente en intervalos de tiempo donde no había vibración.
    3. **Realizar Completar valores nulos realizando interpolación lineal en todas las variables de temperatura, PDF, PEL, RA y RM:** La interpolación lineal se ajusta al comportamiento de estas variables en periodos de tiempo cortos.
    4. **Eliminar registros que tengan cualquiera de las 17 variables nula:** Para asegurar la fidelidad de los datos, cualquier fila que contenga valores nulos luego del proceso de limpieza, será eliminada. Esto puede ocurrir con registros que no tengan valores precedentes cuando se realiza el paso 2 de la primera etapa o el paso 3 de la segunda.

5.2.2. Normalización

Para asegurar que cada atributo contribuye de igual manera en cualquier proceso de ML, se requiere normalizar o escalar los datos. La normalización se realizó para cada chancador según la Ecuación 14. Para cada elemento x_i en el atributo x ($x_i \in x$) se calcula un nuevo valor $x'_i \in [0, 1]$. De esta manera, el mayor valor de x será 1 y el menor será 0.

$$x'_i = \frac{x_i - min(x)}{max(x) - min(x)} \tag{14}$$

5.2.3. Segmentación y confección de ventanas

El método de ventanas deslizantes (*Sliding Windows Approach*) se utiliza para segmentar series de tiempo continuas en un conjunto de segmentos más pequeños [1]. La serie de tiempo de cada chancador fue procesada mediante este método para generar ventanas de tiempo de 1 hora. De esta manera se obtiene un conjunto de objetos independientes entre si (ventanas) que conservan el componente temporal. El proceso se realiza como sigue:

1. Segmentación de los datos en intervalos continuos
2. Creación de ventanas a partir de cada segmento.

La **segmentación previa a la creación de ventanas** se realiza para conservar la continuidad temporal en los datos, es decir, para mantener la regularidad en la toma de registros. Debido a que el proceso de limpieza descrito en la Sección 5.2.1 puede eliminar filas completas, se corre el riesgos de tener interrupciones temporales en los datos útiles. Para evitar que las ventanas contengan dichas interrupciones, es necesario utilizar solo segmentos que tenga datos continuos para dividirlos en ventanas de manera independiente. La Figura 17 expone el gráficamente el fenómeno.

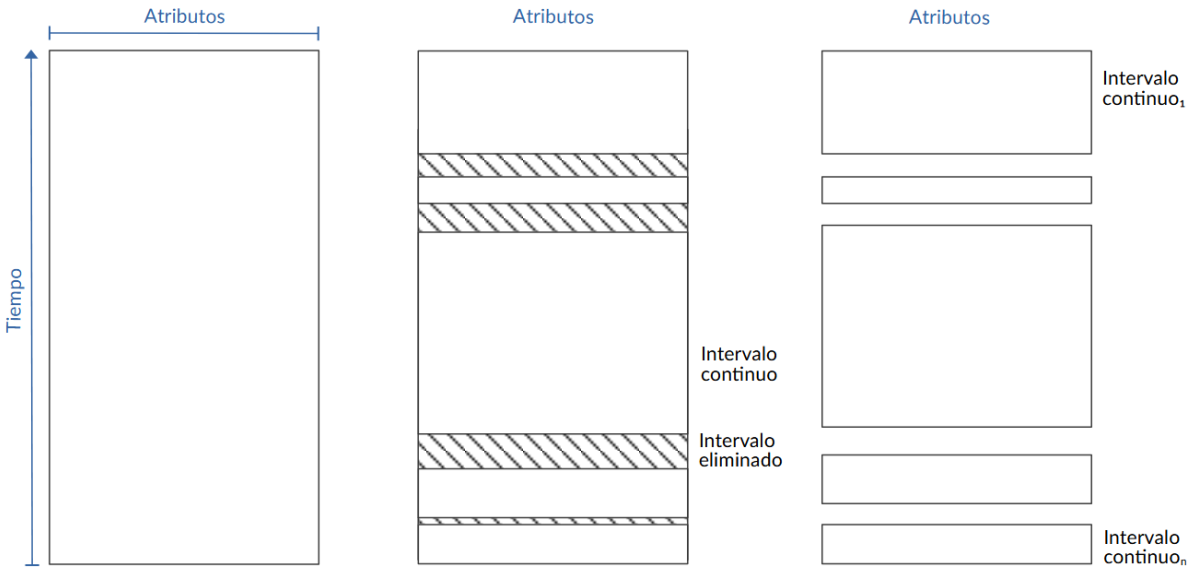

Figura 17: Segmentación del conjunto de datos en intervalos temporales. A la izquierda la serie de tiempo de un producida por un chancador. En el centro, algunos son extraídos en el proceso de limpieza. A la derecha, n segmentos continuos son los que finalmente se usarán para confeccionar ventanas.

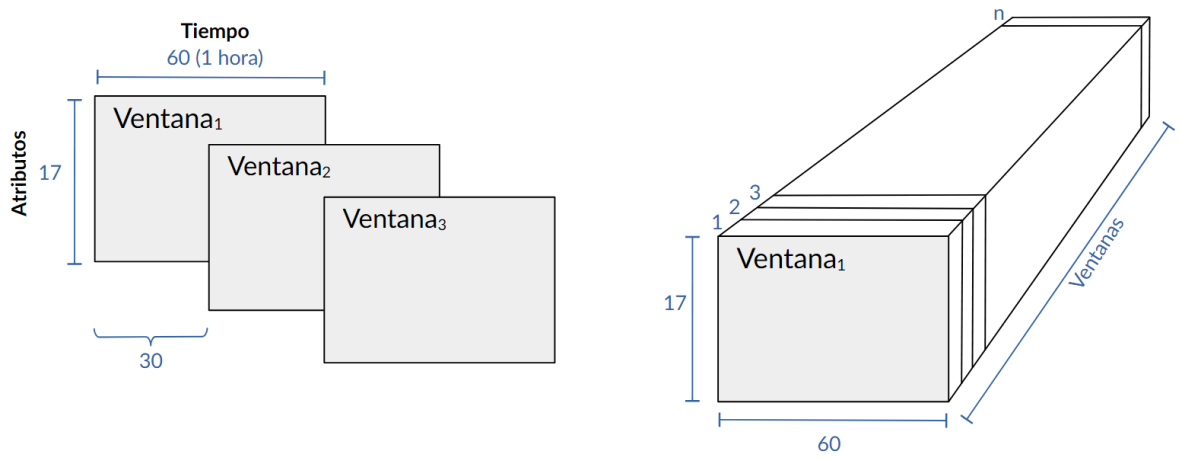

Figura 18: Creación de ventanas. Cada ventana comparte 30 registros con la ventana previa y con la siguiente (izquierda). El resultado es un conjunto de n ventanas de 17×60 (derecha).

La **creación de las ventanas** se realiza sobre los segmentos de 60 o más registros, ya que este es tamaño mínimo para contener 1 hora de datos continuos (registros cada 1 minuto). Las ventanas se crean a partir del primer registro en el segmento, tomando 60 registros y con un solapamiento de 30 (30 minutos) con la ventana previa. Esto se realiza para conservar posibles fenómenos que ocurren entre ventanas. El proceso se ilustra en la Figura 18. La cantidad de ventanas extraídas de los segmentos de datos se detalla en la Tabla 8.

Chancador	2023	2024	Total
1	10.395	10.440	20.835
2	10.332	10.468	20.800
3	11.079	10.591	21.670
4	10.714	10.741	21.455
5	10.993	10.072	21.065

Tabla 8: Cantidad de ventanas generadas a partir de los conjuntos de datos

5.3. Entrenamiento del Autoencoder

Desde la perspectiva de los datos, como todos provienen de la misma fuente de datos, comparten la misma estructura y no se ha utilizado transfer learning, el proceso corresponde a **Deep single-view clustering** (ver Sección 2.6.3), concretamente uno basado en Deep Autoencoders.

El CAE se diseñó basándose en los utilizados en [51] y [1]. La arquitectura se muestra en la Figuta 19. La red neuronal fue entrenado utilizando **descenso de gradiente estocástico** (SGD, del inglés *Stochastic Gradient Descent*) y **MSE Loss**, con *learning rate* (lr) de 6×10^{-3}, *weight decay* (wd) de 9×10^{-6} y *momentum* de 0,9. Los tres valores fueron ajustados a partir de los presentados en [1], con cambios realizados para mejorar el error de reconstrucción con mayor velocidad. Los detalles de la implementación del AE están en la Tabla 9. Las capas están ordenadas desde la entrada hacia la salida. Los parámetros no especificados tienen su valor por defecto.

El único objetivo del autoencoder es la obtención de la representación latente. Por esta razón, una vez entrenado un modelo, solo se extrae y conserva el codificador. Este será utilizado para la generación de la representación las ventanas de funcionamiento, como paso previo al proceso de agrupamiento. El AE se entrenó individualmente para cada uno de los chancadores. Todos los modelos utilizaron la misma arquitectura e hiperparámetros. Para el entrenamiento se utilizó el conjunto de ventanas de ambos años, mezclados aleatoriamente.

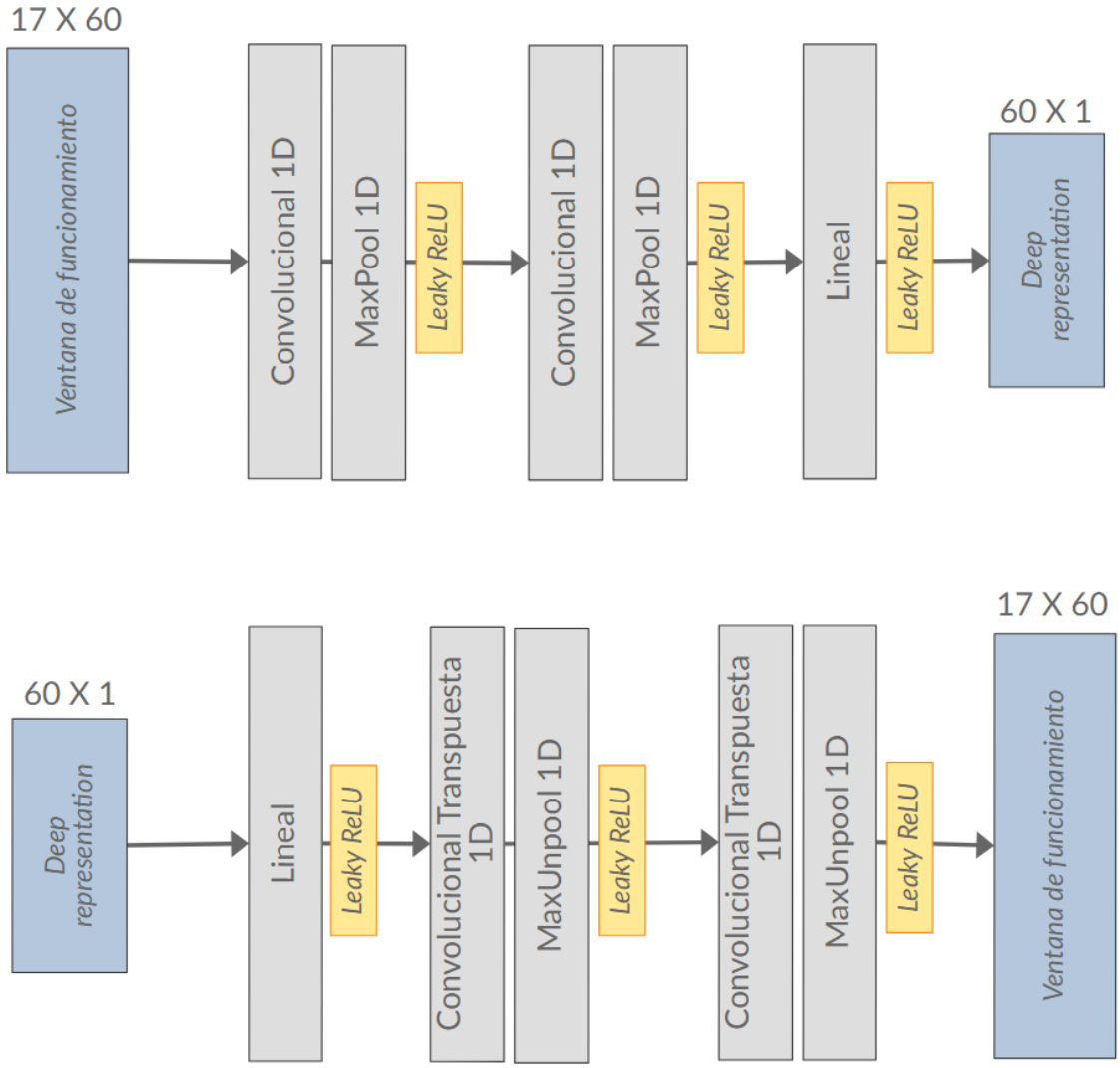

Figura 19: Arquitectura del autoencoder convolucional utilizado. La imagen está dividida separando al codificador (arriba) del decodificador (abajo).

Los modelos fueron entrenados con 10, 60 y 200 epochs, obteniendo un total de 3 codificadores por chancador. Un epoch es un ciclo de entrenamiento con el conjunto de datos completos. Para un AE, una mayor cantidad de epochs implica usualmente un menor error de reconstrucción. Una cantidad diferente de epochs de entrenamiento en un mismo AE puede entregar diferentes resultados en el proceso de deep clustering. Esta razón sostiene la incorporación de esta variación en las pruebas.

5.4. Pruebas

En deep clustering secuencial, el algoritmo de agrupamiento es un componente separado del AE. La **selección de parámetros para DBSCAN** se realizó de acuerdo a lo expuesto en la Sección 2.5.2, específicamente en la Tabla 4. Se fijó $q = (dimensin \cdot 2) = 120$ y se utilizaron gráficos K-distance, con $k = (dimensin \cdot 2 - 1) = 119$, para realizar el

Capa	**PyTorch**	**Configuración**
Convolucional 1D	Conv1d	$in\_channels = 17$, $out\_channels = 34$, $kernel\_size = 6$
MaxPool 1D	MaxPool1d	$kernel\_size = 2$, $stride = 2$
Convolucional 1D	Conv1d	$in\_channels = 34$, $out\_channels = 68$, $kernel\_size = 12$
MaxPool 1D	MaxPool1d	$kernel\_size = 2$, $stride = 2$
Lineal	Linear	$in\_features = 544$, $out\_features = 60$
Lineal	Linear	$in\_features = 60$, $out\_features = 544$
MaxUnpool 1D	MaxUnpool1d	$kernel\_size = 2$, $stride = 2$
Convolucional transpuesta 1D	ConvTranspose1d	$in\_channels = 68$, $out\_channels = 34$, $kernel\_size = 12$
MaxUnpool 1D	MaxUnpool1d	$kernel\_size = 2$, $stride = 2$
Convolucional transpuesta 1D	ConvTranspose1d	$in\_channels = 34$, $out\_channels = 17$, $kernel\_size = 6$

Tabla 9: Construcción del CAE con PyTorch

análisis visual y encontrar el valor de ε. Como el codo en los gráficos no es evidente para todos los conjuntos de datos o para todas las representaciones, resulta complicado escoger un ε. Para afrontar este problema se decidió seleccionar un conjunto de valores para su posterior evaluación. El procedimiento es como sigue:

- Tomar una cota inferior d_{min} y una superior d_{max} para delimitar el inicio y fin del codo, obteniéndose un intervalo $[d_{min}, d_{max}]$.
- Seleccionar 5 valores pertenecientes a $[d_{min}, d_{max}]$, incluyendo los valores extremos, de tal forma que la distancia entre dos valores consecutivos sea la misma para todos.
- Realizar pruebas con todas las configuraciones seleccionadas.

En la Figura 20 se muestra los gráficos K-distance para el chancador 1. Cada fila corresponde a la representación latente creada por un AE diferente (10, 60, 200 Epochs). La columna izquierda es el gráfico completo y la columna derecha es el mismo gráfico ampliado a una zona de interés. Las líneas rojas marcan los límites d_{min} y d_{max}. La Tabla 10 presenta todas las configuraciones para los experimentos.

Para k-means se realizaron pruebas variando el valor de K de 2 a 31 ($K = 2, 3, ..., 30, 31$) para cada uno de los AE. En todas las pruebas se utilizó distancia euclidiana.

Codificador	**Conjunto**	q	ε
CAE 10 Epochs	Chancador 1	120	{0,1; 0,11; 0,12; 0,13; 0,14}
	Chancador 2	120	{0,12; 0,1275; 0,135; 0,1425; 0,15}
	Chancador 3	120	{0,1; 0,125; 0,15; 0,175; 0,2}
	Chancador 4	120	{0,09; 0,1125; 0,135; 0,1575; 0,18}
	Chancador 5	120	{0,09; 0,1125; 0,135; 0,1575; 0,18}
CAE 60 Epochs	Chancador 1	120	{0,6; 0,75; 0,9; 1,05; 1,2}
	Chancador 2	120	{0,8; 0,95; 1,1; 1,25; 1,4}
	Chancador 3	120	{0,9; 1,15; 1,3; 1,45; 1,5}
	Chancador 4	120	{0,7; 0,9; 1,1; 1,3; 1,5}
	Chancador 5	120	{0,8; 1; 1,2; 1,4; 1,6}
CAE 200 Epochs	Chancador 1	120	{2; 2,25; 2,5; 2,75; 3}
	Chancador 2	120	{2; 2,25; 2,5; 2,75; 3}
	Chancador 3	120	{1,7; 2,15; 2,6; 3,05; 3,5}
	Chancador 4	120	{1,5; 2; 2,5; 3; 3,5}
	Chancador 5	120	{1,5; 2; 2,5; 3; 3,5}

Tabla 10: Configuración para experimentos con CAE DBSCAN con distancia euclidiana

5.5. Evaluación

Como no se cuenta con datos que directamente asocien fallas a ventanas de tiempo determinadas, se decidió utilizar las alertas como indicador. Las alertas funcionan como un síntoma de falla, lo que quiere decir que una alta cantidad de alertas durante un periodo de tiempo determinado implica una alta posibilidad de que algún componente en el chancador esté fallando. Para cada ventana del conjunto de datos, se calculó el número de alertas como la suma de todas las alertas gatilladas durante la hora que dura la ventana.

Los índices externos comúnmente utilizados para deep clustering son ARI, NMI y Accuracy [48]. Tal como se describe en la Sección 2.5.3, todos necesitan etiquetas conocidas para ser calculados. Debido a esta razón, ninguno de estos índices resulta apropiado para el problema con los chancadores; las alertas son síntomas pero no se conoce un umbral que pueda determinar un fallo de forma precisa. Para subsanar el problema, se utilizó la **distancia de Wasserstein** (W_{emd}) para cuantificar las diferencias en las distribuciones de las alertas de los grupos. Esta distancia toma en cuenta el costo de mover *masa* de una distribución a otra. En [37] se da una definición (Ecuación 15) y se describen algunas de sus propiedades:

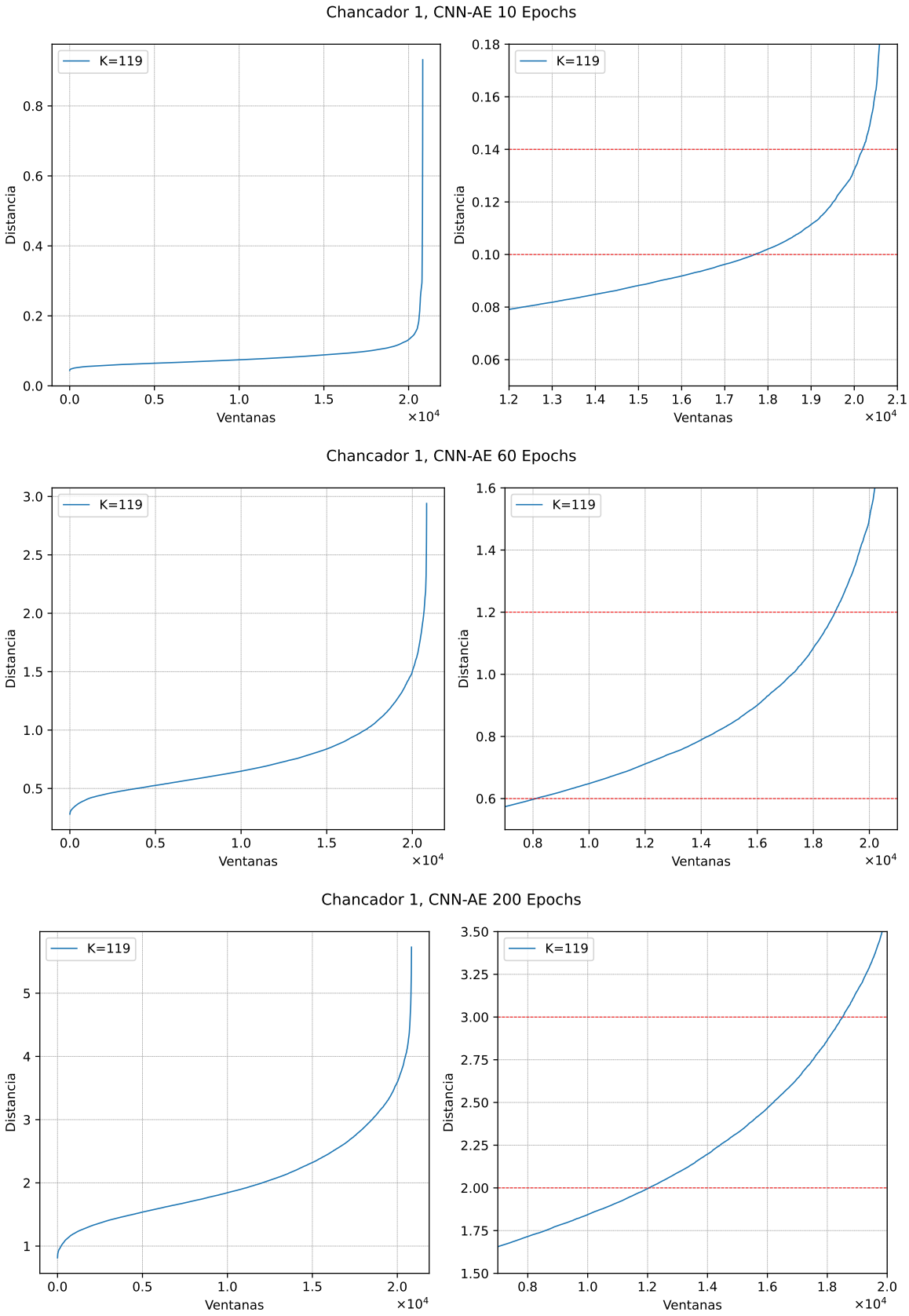

Figura 20: Gráfico K-Distance de la representación latente de las ventanas del chancador 1.

- Es una función continua, por lo que cambios pequeños en las distribuciones de entrada dan como resultado diferencias pequeñas en la distancia.
- Satisface las propiedades de una métrica: simetría, desigualdad triangular y no negatividad.
- Tiene una interpretación natural: el costo mínimo de transformar una distribución en otra.

Se puede definir la distancia de Wasserstein de la siguiente manera:

$$W_{emd}(\mu, \upsilon) = \min_{\gamma \in P} \int ||x - y|| \gamma(x, y) dx dy \tag{15}$$

donde P es el conjunto de todas las distribuciones conjuntas $\Pi(\mu, \upsilon)$ cuyas distribuciones marginales son μ y υ, $||x - y||$ es el costo de transportar una unidad de masa desde el punto x al punto y, y la función $\gamma(x, y)$ retorna la cantidad de masa a mover desde x a y.

Finalmente, para la evaluación de resultados se utilizaron las siguientes medidas:

- El **promedio de alertas** de cada grupo generado C se utilizó para distinguir a los grupos de ventanas con funcionamiento anómalo y grupos de ventanas de funcionamiento no anómalo. Si el promedio de alertas de un grupo ($\bar{x}_C$) es mayor al promedio del conjunto original ($\bar{x}_C > \bar{x}$), el grupo tiene un funcionamiento anómalo. En caso contrario, el grupo tienen un funcionamiento no anómalo.
- La **distancia de Wasserstein** entre las distribuciones de alertas del conjunto de datos original y de un grupo C se utilizó para cuantificar el nivel de anomalía de ese grupo (*cuan malo* es). Si C es un grupo anómalo, mientras mayor sea la distancia, más anómalo se considera (es *peor*). Si C es un grupo no anómalo, mientras mayor sea la distancia, mayor es el nivel de estabilidad de funcionamiento del grupo. Como el objetivo del MC es distinguir el mal funcionamiento, las soluciones que distingan peores grupos tienen prioridad.
- **Índices internos** como el coeficiente de silueta o el índice de Davies-Bouldin se utilizaron para cuantificar la calidad de la solución de agrupamiento desde un punto de vista estructural (ver Sección 2.5.3). Se busca un valor de silueta lo más cercano a 1 y un valor de Davies-Bouldin lo más bajo posible.

Finalmente, los criterios para determinar comparativamente cual es la mejor solución se listan a continuación por orden de importancia:

1. Identificación de un grupo anómalo con la máxima distancia de Wasserstein posible.

2. Solución de agrupamiento con un alto coeficiente de silueta y un bajo índice de Davies-bouldin.

3. Identificación de grupos no anómalos con la máxima distancia de Wasserstein posible.

6. Resultados e interpretación

En este capítulo se exponen los resultados obtenidos a partir de la metodología planteada en la Sección 5. La Sección 6.1 y la Sección 6.2 muestra el resultado de los experimentos con k-means y DBSCAN respectivamente, mientras que en la Sección 6.3 se da una interpretación de lo obtenido.

6.1. Deep clustering con k-means

Los resultados de los experimentos con k-means (puro, CAE 10 epochs, CAE 60 epochs y CAE 200 epochs) están resumidos en la Figura 21, la Figura 22 y la Figura 23. La primera muestra el valor de los índices internos (coeficiente de silueta e índice de Davies-Bouldin) obtenidos por cada una de las soluciones, a medida que aumenta el valor de k. La segunda y la tercera exponen la distancia W_{emd} máxima entre un grupo anómalo/no anómalo y el conjunto original, respectivamente ($\text{máx}_{C \in A} W_{emd}(C, X)$, siendo A todos los grupos identificados como anómalos (no anómalos para la tercera figura) y X es el conjunto completo de ventanas). En el Apéndice A se proveen más detalles.

6.2. Deep clustering con DBSCAN

El agrupamiento con DBSCAN en muy pocos casos creó más de un grupo. Sin embargo, el ruido identificado durante el proceso de agrupamiento está formado por ventanas de alta cantidad de alertas. De esta manera, se consideró como grupo no anómalo al grupo principal y como grupo anómalo al conformado por ruido. La Figura 24 muestra para cada chancador y para cada valor de ε, la distancia W_{emd} máxima entre un grupo anómalo y el conjunto original.

Los índices internos de los experimentos con DBSCAN se presentan en la Figura 25. En la imagen, cada fila representa a un chancador diferente ordenado de arriba a abajo, desde el chancador 1 hasta el 5. Es importante recalcar que tanto el coeficiente de silueta como el índice de Davies-Bouldin pueden ser imprecisos cuando son usados para evaluar grupos con forma cóncava [12], por lo que si los resultados generados por DBSCAN presentan esta estructura, ambos valores pueden no ser los adecuados para una evaluación. Además, ambos índices se han calculado incorporando al ruido como un solo grupo. Esto, por la manera en la que está diseñado DBSCAN, no es ideal ya que el ruido corresponde a elementos atípicos cuya similitud interna no está asegurada. Por ejemplo, los puntos de ruido podrían estar dispersos alrededor de los grupos. Agrupar dichos puntos en un solo grupo empeoraría índices internos como el silueta debido a que muchos puntos serian, en promedio, más cercanos matemáticamente a otros grupos que, en promedio, al grupo conformado por el ruido. Los detalles de los experimentos se entregan en el Apéndice A.

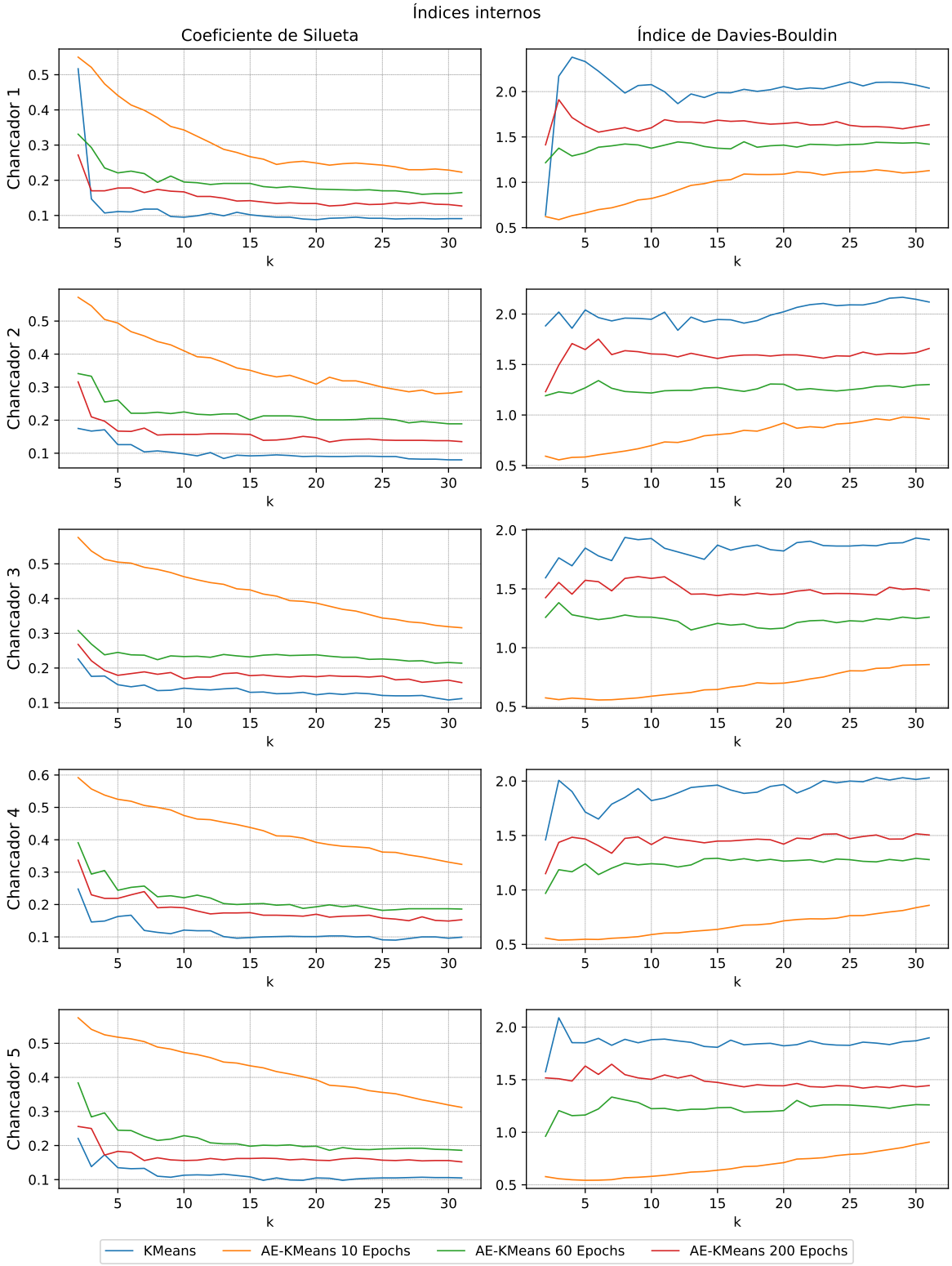

Figura 21: Comparación del valor del índice de silueta y el índice de Davies-Bouldin entre diferentes soluciones de k-means, a diferentes valores de k.

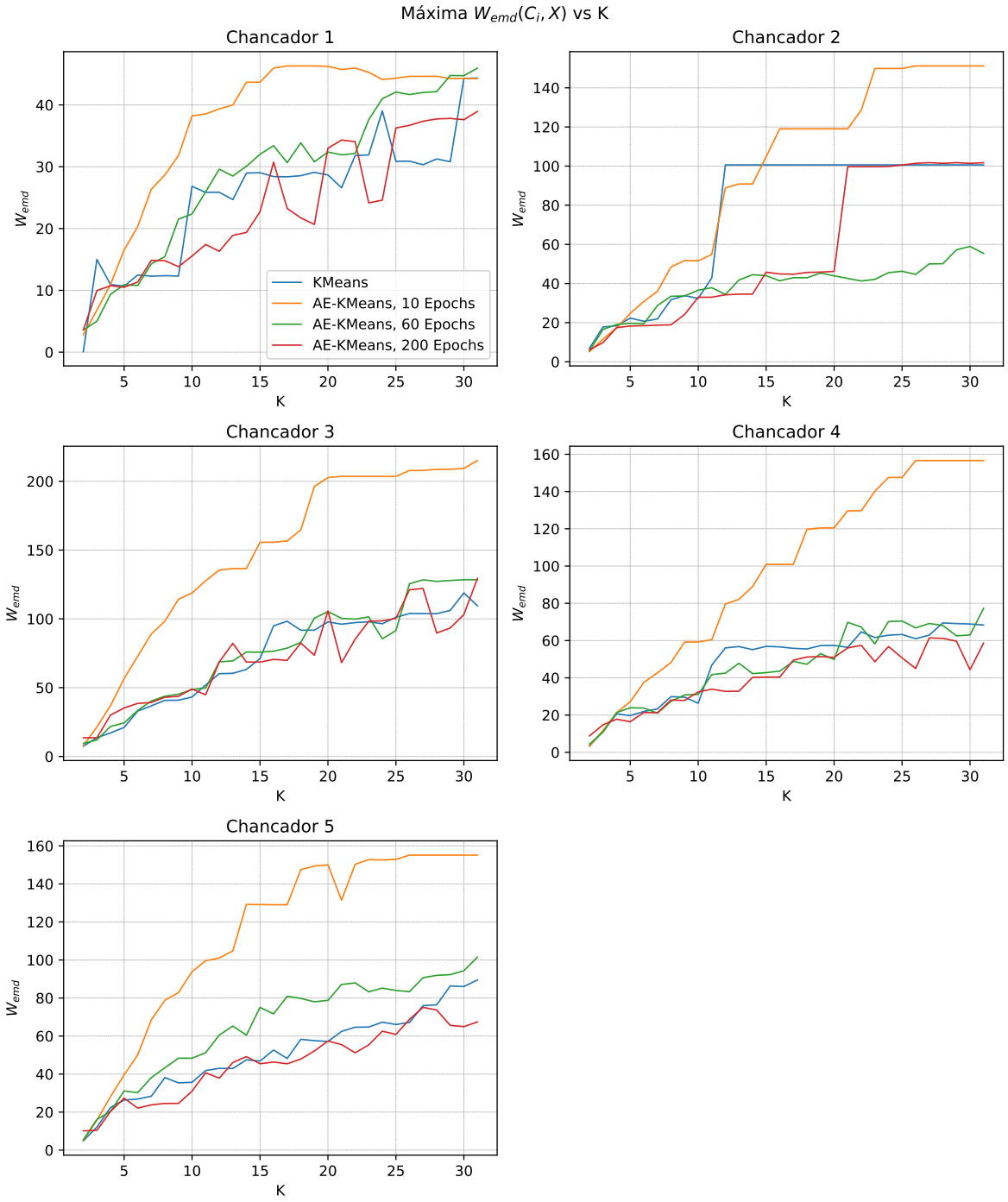

Figura 22: Valor de W_{emd} máximo entre un grupo anómalo y el conjunto completo, vs valor de K.

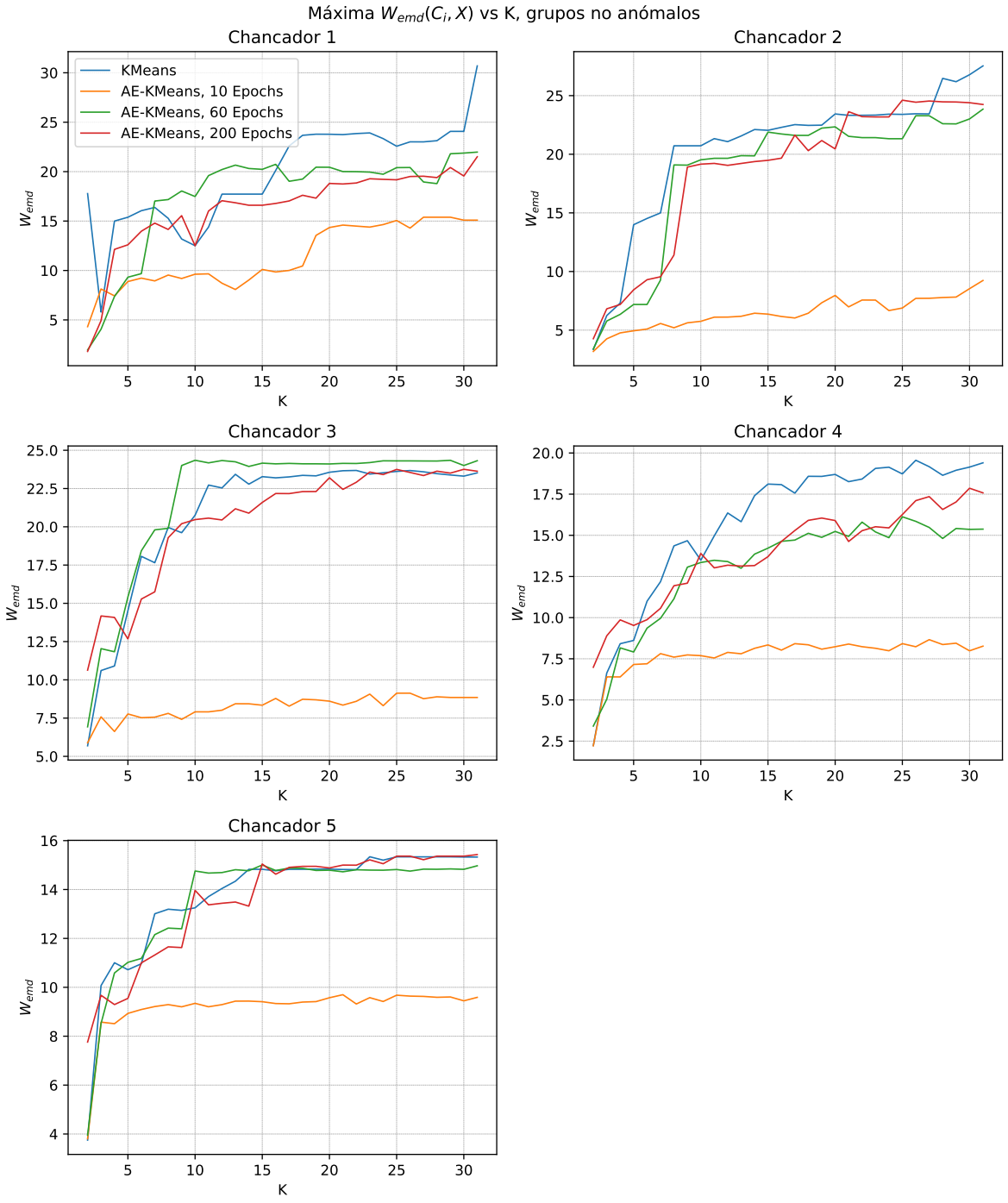

Figura 23: Valor de W_{emd} máximo entre un grupo no anómalo y el conjunto completo, vs valor de K.

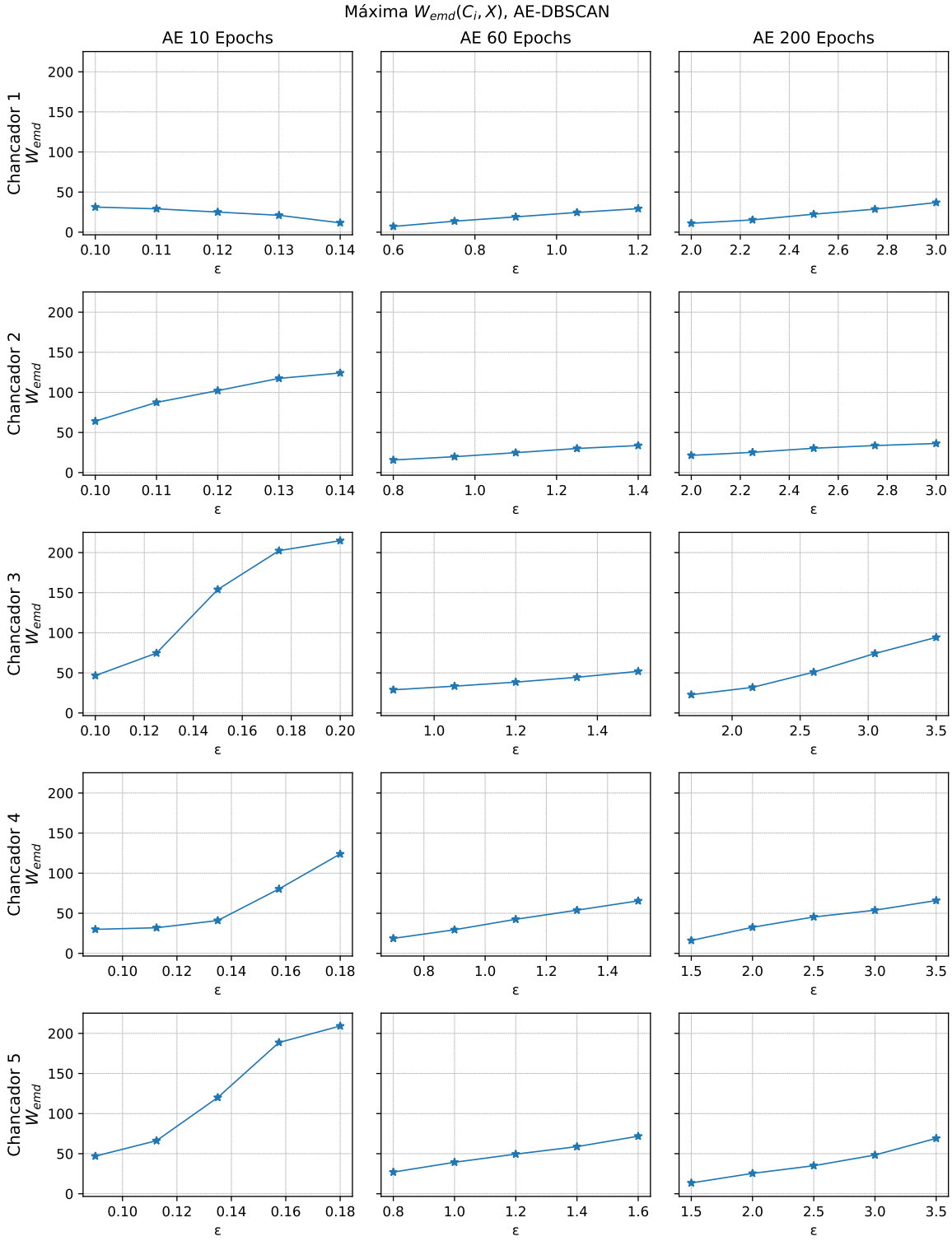

Figura 24: Valor de W_{emd} máximo entre un grupo anómalo y el conjunto completo, vs valor de ε.

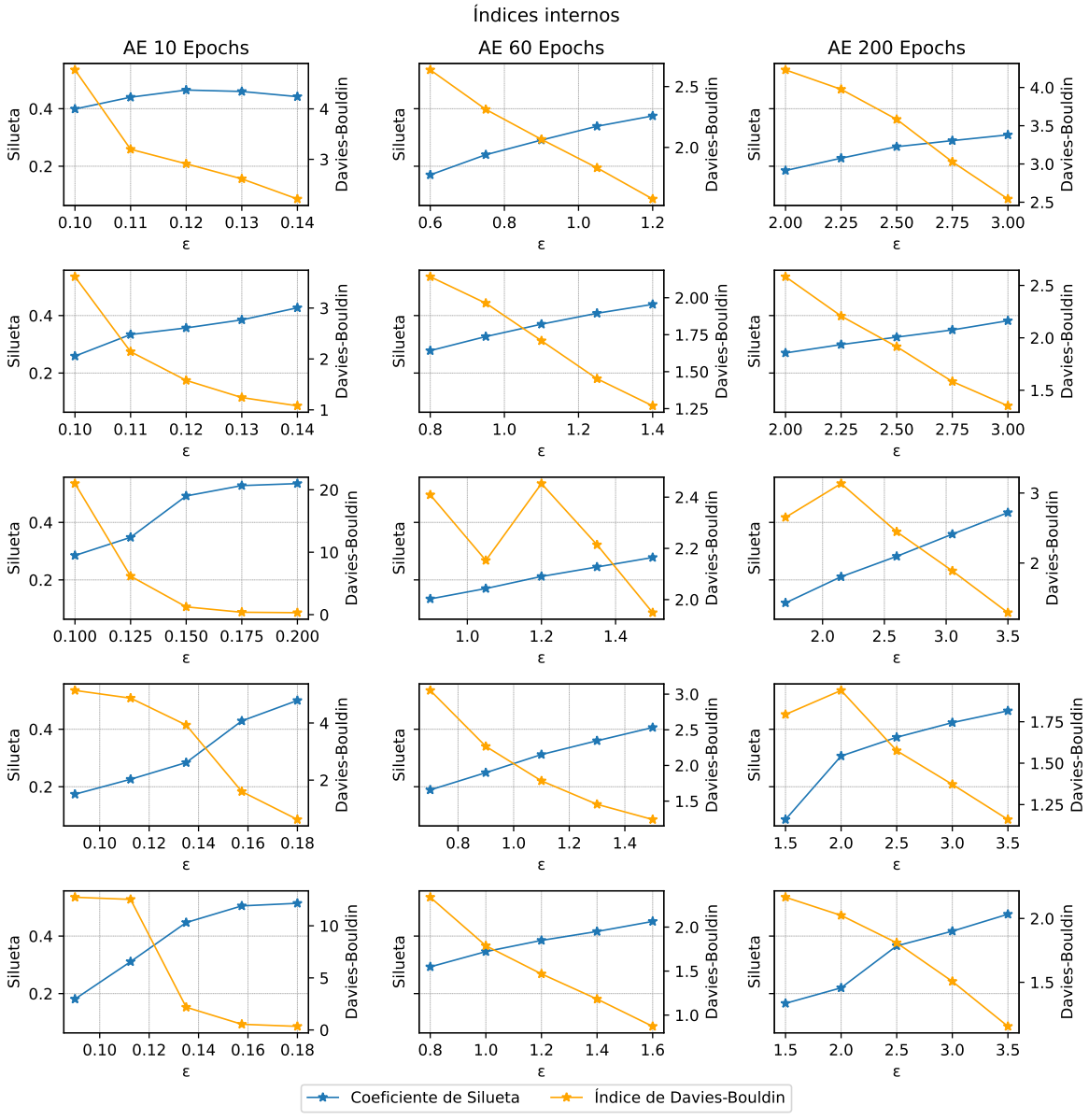

Figura 25: Índices internos para AE con DBSCAN.

6.3. Discusión e interpretación

6.3.1. K-means

De los resultados obtenidos con k-means, sin y con AE, se pueden evidenciar los siguientes puntos:

1. Los resultados donde se utiliza deep clustering (CAE y k-means) tienen mejor coeficiente de silueta e índice de Davies-Bouldin que aquellos en los que se utiliza ML convencional (k-means).
2. Una mayor cantidad de Epochs en el entrenamiento empeora los valores del coeficiente de silueta y el índice de Davies-Bouldin.
3. El AE con 10 epochs es el mejor diferenciando grupos anómalos.
4. El AE con 10 epochs tiene problemas para diferenciar grupos de buen comportamiento.

Con respecto a los índices internos, el primer punto se puede explicar debido a la reducción dimensional realizada por el AE. En [4] se explica la **maldición de la dimensionalidad** (*curse of dimensionality*) señalando que si la cantidad de datos que se disponen para entrenar un modelo es fija, un aumento de la dimensionalidad puede conducir a sobreajuste. Esto puede ser subsanado incrementando los datos de manera exponencial, en función de las dimensiones adicionales. También es sabido que a medida que incrementa el número de dimensiones en los datos, las medidas de distancia para el cálculo de similitud se vuelven menos eficientes [32]. La reducción conducida por los AEs transforma datos de 17×60 dimensiones a 60, evitando los problemas asociados.

Los AE se entrenan para reconstruir los datos con el menor error posible. Esto quiere decir que mientras más ciclos de entrenamiento se realicen, menor será el error de reconstrucción y por consiguiente, es más probable que el AE aprenda los detalles (útiles o inútiles) y el ruido presente en los datos originales. Como todos los AE generan una representación latente de 60 dimensiones, los desafíos del primer punto no son decisivos en el segundo punto. En este caso, el AE de 10 epochs aprendió a armar una representación con la suficiente información para crear grupos bien diferenciados. Que esta capacidad, según los índices internos, se vaya perdiendo mientras más ciclos de entrenamiento se utilizan, sugiere la existencia de ciertos *detalles* en los datos que tardan en ser aprendidos, y que homogeneizan la representación latente.

El tercer y cuarto punto profundizan lo constatado en el segundo. Si bien los índices internos indican la presencia de ciertos componentes acentuados en las reconstrucciones con más ciclos, no entregan un juicio sobre la utilidad de estos. El tercer punto, observado en

todos los chancadores, sugiere que este componente dificulta el reconocimiento de grupos anómalos. Esta es la razón por la que el grupo con más alto W_{emd} con respecto a los datos originales fue creado por el AE de 10 epochs. Por el contrario, los detalles no capturados por el AE de 10 epcohs facilitan la identificación de grupos de buen comportamiento. Esto se señala en el cuarto punto y concuerda con el hecho de que k-means sin AE (datos sin pérdida de información) es el que generó grupos de comportamiento no anómalo con mayor W_{emd} con respecto al conjunto original (grupos *buenos*).

Las soluciones tienden a separar mejor los grupos anómalos mientras mayor sea el valor de k. Esto no ocurre en todos los casos; por ejemplo, al utilizar k-means puro en el chancador 2, la distancia se estanca en $k = 12$ con el valor $W_{emd} = 100{,}624$. De ahí en adelante, no se crea ningún grupo con mayor distancia al conjunto completo. Pese a esto, no es conveniente elegir un valor alto para k, ya que los índices internos demuestran que las soluciones se van degenerando mientras más particiones se crean. Una de las razones que podrían explicar esta divergencia entre índices internos y externos es que, a medida que k aumenta, se dividen algunos grupos relativamente cohesivos. Los dos grupos resultantes quedan geométricamente cerca y hacen que los indices internos empeoren. Por este motivo, se sugiere fijar un valor para k de tal forma que el coeficiente de silueta no baje de 0,3. Este límite concuerda con resultados como los presentados en [35], donde se utiliza k-means para agrupar conjuntos de datos reales. En dicho trabajo se incluyendo series de tiempo y conjuntos de datos con dimensiones similares. Además, el valor de k es ideal, dado por la naturaleza de los datos, es conocido.

6.3.2. DBSCAN

De los resultados de las pruebas con DBSCAN se puede observar que:

1. Para todas las pruebas, a excepción del chancador 1 con el AE de 10 epochs, a medida que ε aumenta, mejoran tanto índices internos como la separación entre el grupo anómalo y el conjunto completo.
2. A medida que ε aumenta, el grupo anómalo se hace más pequeño.
3. El AE con 10 epochs es el mejor diferenciando grupos anómalos.
4. En todos los experimentos, el grupo no anómalo con más ventanas presenta una distribución de alertas muy similar al conjunto completo (W_{emd} bajo).

El primer punto sugiere la utilización de valores de ε altos para obtener mejores resultados. Sin embargo, el segundo indica que esto trae como consecuencia la identificación de grupos anómalos cada vez más pequeños. En la figura 26 se contrastan ambos fenómenos. Que un grupo anómalo sea pequeño, por si mismo, no es un problema; resulta natural que

los estados de falla sean minoría durante las operaciones de maquinaria industrial. Sin embargo, al elegir un valor de ε demasiado alto, existe la posibilidad de que ventanas con un número de alertas importante, que no forman parte de la fracción con más alertas, queden clasificadas como parte del grupo no anómalo. Esto sucede debido a que, según las pruebas realizadas, el valor de ε funciona como un parámetro ajustable que determina un *umbral de aceptación* para que el algoritmo considere una ventana como no anómala: mientras mayor sea su valor, más estricto es el criterio.

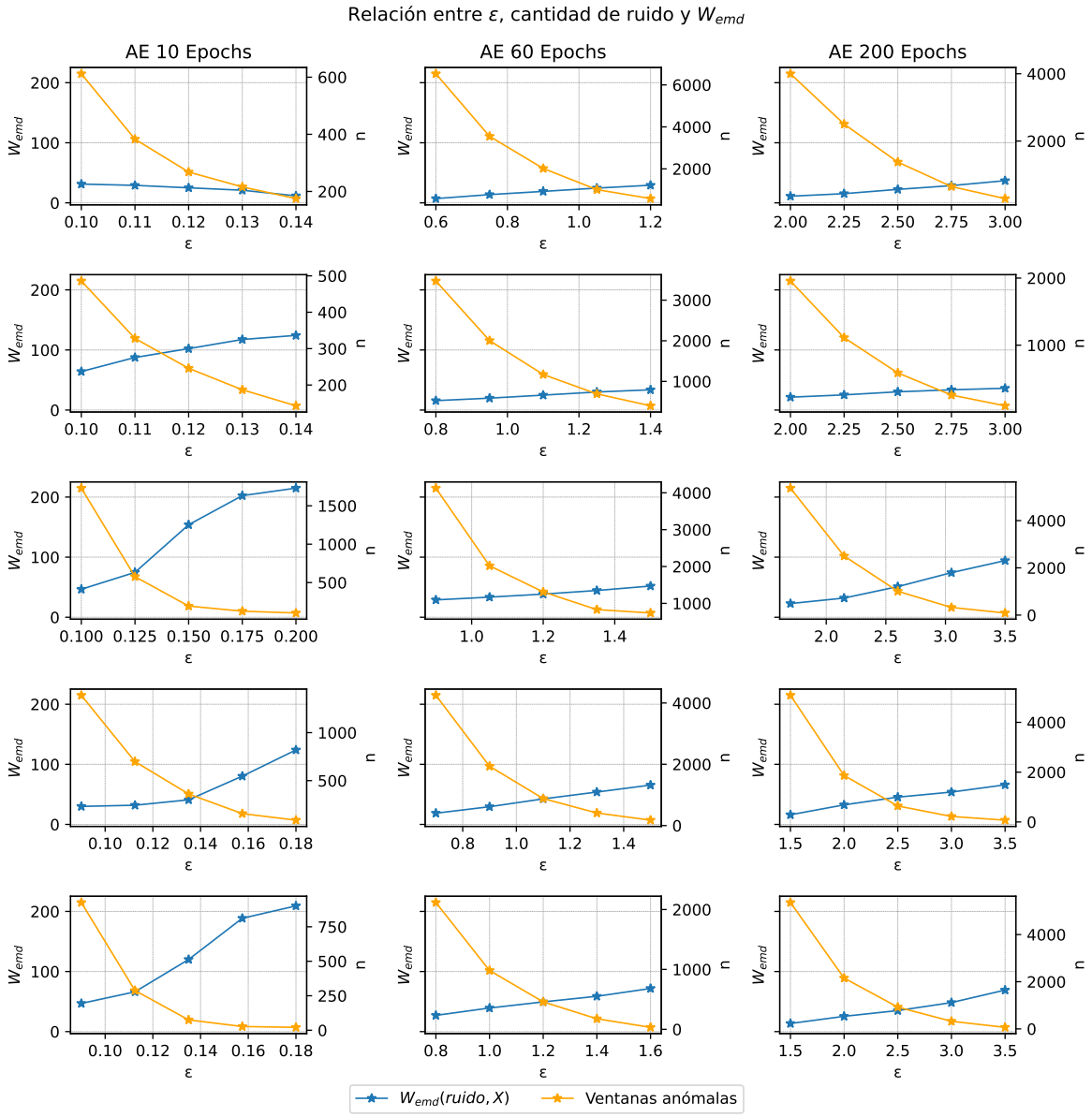

Figura 26: W_{emd} vs cantidad de ruido, para cada chancador (dilas 1 a 5, chancador 1 a 5 respectivamente), AE y ε.

Este comportamiento no se observa en los experimentos con k-means debido a que estos producen muchos grupos, pudiendo distinguirse diferentes estados de falla y, en algunos casos, diferentes niveles de funcionamiento normal. En conclusión, el valor ε debe ser ajus-

tado de acuerdo a las necesidades del problema. No obstante, se sugiere una configuración que represente un compromiso entre el valor W_{emd} y la cantidad de ventanas anómalas. En la Tabla 11 se calculan valores de ε eligiendo un punto medio entre los intervalos utilizados en los gráficos k-distance: $\varepsilon_m = \frac{d_{max}+d_{min}}{2}$ (ver Sección 5.4). Se calcula también el promedio (columna $\bar{x}$) como valor sugerido en independencia del chancador.

Modelo	**Chancadores** (ε_m)					$\bar{x}$
	1	**2**	**3**	**4**	**5**	
AE 10 Epochs	0,1	0,13125	0,1375	0,14625	0,12375	0,12775
AE 60 Epochs	0,825	1,025	1,225	1	1,15	1,045
AE 200 Epochs	2,375	2,375	2,375	2,25	2,25	2,325

Tabla 11: Valores sugeridos para modelos de AE con DBSCAN.

El tercer punto reafirma lo constatado en los experimentos con k-means: las representación latente generada por el AE de 10 epochs es codificada de tal forma de que el algoritmo de agrupamiento (DBSCAN o k-means) es capaz de diferencial fácilmente los estados anómalos, superando en ello a los AE de 60 y 200 epochs.

Una diferencia importante entre los resultados de DBSCAN y k-means está marcada por el cuarto punto. Mientras k-means particiona el conjunto de datos en grupos con diferentes distribuciones de alertas, DBSCAN *extrae* ventanas anómalas desde el conjunto de datos (el ruido). Esta es una de las razones por la que el conjunto no anómalo tiene una baja distancia W_{emd} con el conjunto original.

En síntesis, Sequential Multistep Deep Clustering con k-means divide los datos en múltiples grupos, pudiendo interpretarse algunos de ellos como diferentes estados de funcionamiento anómalo, siendo unos más graves que otros. Tambien se obtienen algunos grupos de estados no anómalos. La misma estrategia de Deep Clustering con DBSCAN obtiene desde el conjunto de datos completo, un grupo con una alta cantidad de alertas, es decir, un único grupo que se puede interpretar como un estado de funcionamiento anómalo grave.

7. Conclusión

En este trabajo se utilizó deep clustering para realizar monitoreo de condiciones de maquinaria industrial real. El caso de estudio fue el funcionamiento de chancadores mineros durante un periodo de 2 años. Se adoptó la estrategia de Sequential Multistep deep clustering para el agrupamiento de series de tiempo. Un Autoencoder Convolucional fue construido para la obtención de la representación latente, mientras que para el agrupamiento se utilizó DBSCAN y k-means. La arquitectura de la red neuronal fue diseñada en base a existentes en la literatura y diferentes cantidades de ciclos de entrenamiento fueron probados. Los resultados mostraron que un autoencoder convolucional junto a k-means o DBSCAN tiene la capacidad de identificar con éxito estados anómalos utilizando un enfoque de Sequential Multistep Deep Clustering. Los dos algoritmos de agrupamiento mostraron resultados favorables, siendo el autoencoder entrenado con 10 ciclos el que generó la representación con mejor desempeño.

Las principales contribuciones de este trabajo son: deep clustering con series de tiempo al servicio de datos industriales reales para realizar monitoreo de condiciones; utilización de la distancia de Wasserstein como métrica de evaluación de grupos ante un problema sin etiquetas categóricas conocidas y un análisis empírico del efecto de la variación de ciclos de entrenamiento para AE sobre la representación latente, en procesos de deep clustering.

7.1. Trabajo futuro

- **Algoritmos de agrupamiento diferentes:** Si bien k-means es el más utilizado para series de tiempo debido a su efectividad, otro tipos de algoritmos no han sido masivamente explorados para deep clusering. Esto incluye algoritmos derivados de DBSCAN, como HDBSCAN, OPTICS y otras variaciones más modernas.

- **Diferentes arquitecuras para AE:** Autoencoders Recurrentes, autoencoder variacionales y redes generativas antagónicas se presentan como alternativas viables para su uso en deep clustering.

- **Disminución de dimensión en la representación latente:** Cambios de parámetros y configuraciones de capas diferentes en el AE pueden conducir a una reducción dimensional de la representación latente sin aumentos importantes en el error de reconstrucción. Una reducción dimensional seria beneficioso para los algoritmos de agrupamiento, especialmente aquellos no adecuados para dimensiones altas, como es el caso de DBSCAN [44].

- **Predicción de comportamientos futuros:** A través de deep clusterin es posible caracterizar los estados de funcionamiento de la maquinaria. La predicción de esta-

dos futuros es ir un paso más adelante y permitiría realizar correcciones antes de que ocurra un problema.

Referencias

[1] Ali Alqahtani et al. «Deep time-series clustering: A review». En: *Electronics (Switzerland)* 10.23 (2021). ISSN: 20799292. DOI: 10.3390/electronics10233001. URL: https://www.mdpi.com/2079-9292/10/23/3001.

[2] Mehmet Cagri Altindal et al. «Anomaly detection in multivariate time series of drilling data». En: *Geoenergy Science and Engineering* 237 (2024), pág. 212778. ISSN: 2949-8910. DOI: https://doi.org/10.1016/j.geoen.2024.212778. URL: https://www.sciencedirect.com/science/article/pii/S2949891024001489.

[3] Xiqin Ao, Yujie Gong y Aixiang He. «A Review of Time Series Prediction Models Based on Deep Learning». En: *IEEE Access* 13 (2025), págs. 153696-153712. URL: https://api.semanticscholar.org/CorpusID:280930559.

[4] Shaeela Ayesha, Muhammad Kashif Hanif y Ramzan Talib. «Overview and comparative study of dimensionality reduction techniques for high dimensional data». En: *Information Fusion* 59 (2020), págs. 44-58. ISSN: 15662535. DOI: 10.1016/j.inffus.2020.01.005. URL: https://www.sciencedirect.com/science/article/pii/S156625351930377X.

[5] Yoshua Bengio, Aaron Courville y Pascal Vincent. «Representation Learning: A Review and New Perspectives». En: *IEEE Trans. Pattern Anal. Mach. Intell.* 35.8 (2013), págs. 1798-1828. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2013.50. URL: https://doi.org/10.1109/TPAMI.2013.50.

[6] Christopher Bishop. *Pattern Recognition and Machine Learning*. Springer New York, 2006. ISBN: 978-1-4939-3843-8. DOI: 10.1007/978-0-387-45528-0. URL: https://www.microsoft.com/en-us/research/publication/pattern-recognition-machine-learning/.

[7] Christopher M. Bishop y Hugh Bishop. «Autoencoders». En: *Deep Learning*. Cham: Springer International Publishing, 2024. Cap. 19, págs. 563-579. ISBN: 978-3-031-45468-4. DOI: 10.1007/978-3-031-45468-4_19. URL: https://doi.org/10.1007/978-3-031-45468-4%7B%5C_%7D19.

[8] Christopher M. Bishop y Hugh Bishop. «Convolutional Networks». En: *Deep Learning*. Cham: Springer International Publishing, 2024. Cap. 10, págs. 287-324. ISBN: 978-3-031-45468-4. DOI: 10.1007/978-3-031-45468-4_10. URL: https://doi.org/10.1007/978-3-031-45468-4%7B%5C_%7D10.

[9] Christopher M. Bishop y Hugh Bishop. «Deep Neural Networks». En: *Deep Learning*. Cham: Springer International Publishing, 2024. Cap. 6, págs. 171-207. ISBN: 978-3-031-45468-4. DOI: 10.1007/978-3-031-45468-4_6. URL: https://doi.org/10.1007/978-3-031-45468-4%7B%5C_%7D6.

[10] Peter J Brockwell y Richard A Davis. «Introduction». En: *Introduction to Time Series and Forecasting*. Cham: Springer International Publishing, 2016, págs. 1-37. ISBN: 978-3-319-29854-2. DOI: 10.1007/978-3-319-29854-2_1. URL: https://doi.org/10.1007/978-3-319-29854-2%7B%5C_%7D1.

[11] Adil Abdu Bushra y Gangman Yi. «Comparative Analysis Review of Pioneering DBSCAN and Successive Density-Based Clustering Algorithms». En: *IEEE Access* 9 (2021), págs. 87918-87935. ISSN: 21693536. DOI: 10.1109/ACCESS.2021.3089036.

[12] Davide Chicco et al. «The DBCV index is more informative than DCSI, CDbw, and VIASCKDE indices for unsupervised clustering internal assessment of concave-shaped and density-based clusters». En: *PeerJ Computer Science* 11 (2025), e3095. ISSN: 23765992. DOI: 10.7717/peerj-cs.3095. URL: https://doi.org/10.7717/peerj-cs.3095.

[13] Martin Ester et al. «A density-based algorithm for discovering clusters in large spatial databases with noise». En: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. KDD'96. Portland, Oregon: AAAI Press, 1996, págs. 226-231.

[14] Syeda Sitara Wishal Fatima y Afshin Rahimi. «A Review of Time-Series Forecasting Algorithms for Industrial Manufacturing Systems». En: *Machines* 12.6 (2024). ISSN: 20751702. DOI: 10.3390/machines12060380. URL: https://www.mdpi.com/2075-1702/12/6/380.

[15] Ian Goodfellow, Yoshua Bengio y Aaron Courville. *Deep Learning*. MIT Press, 2016.

[16] Ingrid Hotz y Ronald Peikert. «Definition of a Multifield». En: *Scientific Visualization: Uncertainty, Multifield, Biomedical, and Scalable Visualization*. Ed. por Charles D Hansen et al. London: Springer London, 2014, págs. 105-109. ISBN: 978-1-4471-6497-5. DOI: 10.1007/978-1-4471-6497-5_10. URL: https://doi.org/10.1007/978-1-4471-6497-5%7B%5C_%7D10.

[17] Abiodun M. Ikotun et al. «K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data». En: *Information Sciences* 622 (2023), págs. 178-210. ISSN: 00200255. DOI: 10.1016/j.ins.2022.11.139. URL: https://www.sciencedirect.com/science/article/pii/S0020025522014633.

[18] International Atomic Energy Agency. *Terminology Used in Nuclear Safety and Radiation Protection, 2016 Revision*. 2016. URL: https://www-ns.iaea.org/downloads/standards/glossary/iaea-safety-glossary-rev2016.pdf (visitado 05-05-2026).

[19] Hassan Ismail Fawaz et al. «Deep learning for time series classification: a review». En: *Data Mining and Knowledge Discovery* 33.4 (2019), págs. 917-963. ISSN: 1573-756X. DOI: 10.1007/s10618-019-00619-1. URL: https://doi.org/10.1007/s10618-019-00619-1.

[20] Alaa Abdulhady Jaber. «Literature Review». En: *Design of an Intelligent Embedded System for Condition Monitoring of an Industrial Robot*. Cham: Springer International Publishing, 2017, págs. 11-51. ISBN: 978-3-319-44932-6. DOI: 10.1007/978-3-319-44932-6_2. URL: https://doi.org/10.1007/978-3-319-44932-6%7B%5C_%7D2.

[21] Naima Khan, Masud Ahmed y Nirmalya Roy. «Temporal Clustering Based Thermal Condition Monitoring in Building». En: *Sustainable Computing: Informatics and Systems* 29 (2021), pág. 100441. ISSN: 22105379. DOI: 10.1016/j.suscom.2020.100441. URL: https://www.sciencedirect.com/science/article/pii/S2210537920301682.

[22] Xiangjie Kong et al. «Deep learning for time series forecasting: a survey». En: *International Journal of Machine Learning and Cybernetics* 16.7-8 (2025), págs. 5079-5112. ISSN: 1868808X. DOI: 10.1007/s13042-025-02560-w. arXiv: 2503.10198. URL: https://doi.org/10.1007/s13042-025-02560-w.

[23] Manel Martinez-Ramon; Meenu Ajith; Aswathy Rajendra Kurup. «Convolutional Neural Networks». En: *Deep Learning*. John Wiley & Sons, Ltd, 2024. Cap. 4, págs. 153-185. ISBN: 9781119861898. DOI: https://doi.org/10.1002/9781119861898.ch4. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119861898.ch4.

[24] Wentao Mao et al. «Prediction of Bearings Remaining Useful Life Across Working Conditions Based on Transfer Learning and Time Series Clustering». En: *IEEE Access* 9 (2021), págs. 135285-135303. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2021.3117002. URL: https://ieeexplore.ieee.org/document/9555648/.

[25] Tshilidzi Marwala. «Introduction to Condition Monitoring in Mechanical and Electrical Systems». En: *Condition Monitoring Using Computational Intelligence Methods: Applications in Mechanical and Electrical Systems*. London: Springer London, 2012, págs. 1-25. ISBN: 978-1-4471-2380-4. DOI: 10.1007/978-1-4471-2380-4_1. URL: https://doi.org/10.1007/978-1-4471-2380-4%7B%5C_%7D1.

[26] Ibomoiye Domor Mienye y Theo G Swart. «Deep Autoencoder Neural Networks: A Comprehensive Review and New Perspectives». En: *Archives of Computational Methods in Engineering* 32.7 (2025), págs. 3981-4000. ISSN: 1886-1784. DOI: 10.1007/s11831-025-10260-5. URL: https://doi.org/10.1007/s11831-025-10260-5.

[27] Thomas M Mitchell. *Machine Learning*. 1.ª ed. USA: McGraw-Hill, Inc., 1997. ISBN: 0070428077.

[28] R Keith Mobley. «Condition based maintenance». En: *Handbook of Condition Monitoring: Techniques and Methodology*. Ed. por A Davies. Dordrecht: Springer Netherlands, 1998, págs. 35-53. ISBN: 978-94-011-4924-2. DOI: 10.1007/978-94-011-4924-2_2. URL: https://doi.org/10.1007/978-94-011-4924-2%7B%5C_%7D2.

[29] Mohammad Modarres, Mark P. Kaminskiy y Vasiliy Krivtsov. *Reliability engineering and risk analysis a practical gide*. 3.ª ed. CRC Press, 2016, págs. 1-504. ISBN: 9781315382425. DOI: 10.1201/9781315382425.

[30] Gopi Chand Nutakki et al. «An Introduction to Deep Clustering». En: 2019, págs. 73-89. ISBN: 978-3-319-97863-5. DOI: 10.1007/978-3-319-97864-2_4.

[31] Jonas Oeing et al. «Flooding Prevention in Distillation and Extraction Columns with Aid of Machine Learning Approaches». En: *Chemie-Ingenieur-Technik* 93.12 (2021), págs. 1917-1929. ISSN: 15222640. DOI: 10.1002/cite.202100051. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cite.202100051.

[32] Gbeminiyi John Oyewole y George Alex Thopil. «Data clustering: application and trends». En: *Artificial Intelligence Review* 56.7 (2023), págs. 6439-6475. ISSN: 15737462. DOI: 10.1007/s10462-022-10325-y. URL: https://doi.org/10.1007/s10462-022-10325-y.

[33] Guansong Pang et al. «Deep Learning for Anomaly Detection: A Review». En: *ACM Comput. Surv.* 54.2 (2021). ISSN: 0360-0300. DOI: 10.1145/3439950. URL: https://doi.org/10.1145/3439950.

[34] Zuzana Papulová, Andrea Gaová y ubomír ufliarský. «Implementation of Automation Technologies of Industry 4.0 in Automotive Manufacturing Companies». En: *Procedia Computer Science* 200 (2022), págs. 1488-1497. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2022.01.350. URL: https://www.sciencedirect.com/science/article/pii/S1877050922003593.

[35] John Pavlopoulos, Georgios Vardakas y Aristidis Likas. «Revisiting Silhouette Aggregation». En: *Discovery Science*. Ed. por Dino Pedreschi et al. Cham: Springer Nature Switzerland, 2025, págs. 354-368. ISBN: 978-3-031-78977-9.

[36] F. Pedregosa et al. «Scikit-learn: Machine Learning in Python». En: *Journal of Machine Learning Research* 12 (2011), págs. 2825-2830.

[37] Xiaotong Qian et al. «Incremental clustering based on Wasserstein distance between histogram models». En: *Pattern Recognition* 162 (2025), pág. 111414. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2025.111414. URL: https://www.sciencedirect.com/science/article/pii/S0031320325000743.

[38] Yazhou Ren et al. «Deep Clustering: A Comprehensive Survey». En: *IEEE Transactions on Neural Networks and Learning Systems* 36.4 (2025), págs. 5858-5878. DOI: 10.1109/TNNLS.2024.3403155.

[39] Seunghyoung Ryu et al. «Residential Load Profile Clustering via Deep Convolutional Autoencoder». En: *2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids, SmartGridComm 2018*. 2018, págs. 1-6. ISBN: 9781538679548. DOI: 10.1109/SmartGridComm.2018.8587454.

[40] Erich Schubert et al. «DBSCAN revisited, revisited: Why and how you should (still) use DBSCAN». En: *ACM Transactions on Database Systems* 42.3 (2017). ISSN: 15574644. DOI: 10.1145/3068335. URL: https://doi.org/10.1145/3068335.

[41] Jan-Peter Seevers et al. «Automatic Detection of Manufacturing Equipment Cycles Using Time Series». En: *Journal of Computing and Information Science in Engineering* 20.3 (2020), pág. 31005. ISSN: 1530-9827. DOI: 10.1115/1.4046208. URL: https://doi.org/10.1115/1.4046208.

[42] Onur Surucu, Stephen Andrew Gadsden y John Yawney. «Condition Monitoring using Machine Learning: A Review of Theory, Applications, and Recent Advances». En: *Expert Systems with Applications* 221 (2023), pág. 119738. ISSN: 09574174. DOI: 10.1016/j.eswa.2023.119738. URL: https://www.sciencedirect.com/science/article/pii/S0957417423002397.

[43] Sergios Theodoridis y Konstantinos Koutroumbas. «Chapter 11 - Clustering: Basic Concepts». En: *Pattern Recognition (Fourth Edition)*. Ed. por Sergios Theodoridis y Konstantinos Koutroumbas. Fourth Edi. Boston: Academic Press, 2009, págs. 595-625. ISBN: 978-1-59749-272-0. DOI: https://doi.org/10.1016/B978-1-59749-272-0.50013-X. URL: https://www.sciencedirect.com/science/article/pii/B978159749272050013X.

[44] Sergios Theodoridis y Konstantinos Koutroumbas. «Chapter 15 - Clustering Algorithms IV». En: *Pattern Recognition (Fourth Edition)*. Ed. por Sergios Theodoridis y Konstantinos Koutroumbas. Fourth Edi. Boston: Academic Press, 2009, págs. 765-862. ISBN: 978-1-59749-272-0. DOI: https://doi.org/10.1016/B978-1-59749-272-0.50017-7. URL: https://www.sciencedirect.com/science/article/pii/B9781597492720500177.

[45] Sergios Theodoridis y Konstantinos Koutroumbas. «Chapter 16 - Cluster Validity». En: ed. por Sergios Theodoridis y Konstantinos B T - Pattern Recognition (Fourth Edition) Koutroumbas. Boston: Academic Press, 2009, págs. 863-913. ISBN: 978-1-59749-272-0. DOI: https://doi.org/10.1016/B978-1-59749-272-0.50018-9. URL: https://www.sciencedirect.com/science/article/pii/B9781597492720500189.

[46] Tippaya Thinsungnoen, Kittisak Kerdprasop y Nittaya Kerdprasop. «Deep Autoencoder Networks optimized with genetic algorithms for efficient ECG clustering». En: *International Journal of Machine Learning and Computing* 8.2 (2018), págs. 112-116. ISSN: 20103700. DOI: 10.18178/ijmlc.2018.8.2.672.

[47] Mark P. Wachowiak, Jason J. Moggridge y Renata Wachowiak-Smolikova. «Deep Embedded Clustering for Data-Driven ECG Exploration Using Continuous Wavelet Transforms». En: *Proceedings of the International Conference on Information and Digital Technologies 2019, IDT 2019*. 2019, págs. 551-556. ISBN: 9781728114019. DOI: 10.1109/DT.2019.8813501.

[48] Xiuxi Wei et al. «An overview on deep clustering». En: *Neurocomputing* 590 (2024), pág. 127761. ISSN: 18728286. DOI: 10.1016/j.neucom.2024.127761. URL: https://www.sciencedirect.com/science/article/pii/S09252312%2024005320.

[49] Luo Yang, Kaibo Wang y Jie Zhou. «Multimode high-dimensional time series clustering and monitoring for wind turbine SCADA data». En: *Quality and Reliability Engineering International* 40.8 (2024), págs. 4285-4306. ISSN: 10991638. DOI: 10.1002/qre.3626. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/qre.3626.

[50] C T Yiakopoulos, K C Gryllias e I A Antoniadis. «Rolling element bearing fault detection in industrial environments based on a K-means clustering approach». En: *Expert Systems with Applications* 38.3 (2011), págs. 2888-2911. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2010.08.083. URL: https://www.sciencedirect.com/science/article/pii/S0957417410008791.

[51] Geng Zhang, Adam R. Singer y Nickolas Vlahopoulos. *Temporal clustering network for self-diagnosing faults from vibration measurements*. 2020. DOI: 10.48550/arXiv.2006.09505.

A. Apéndice

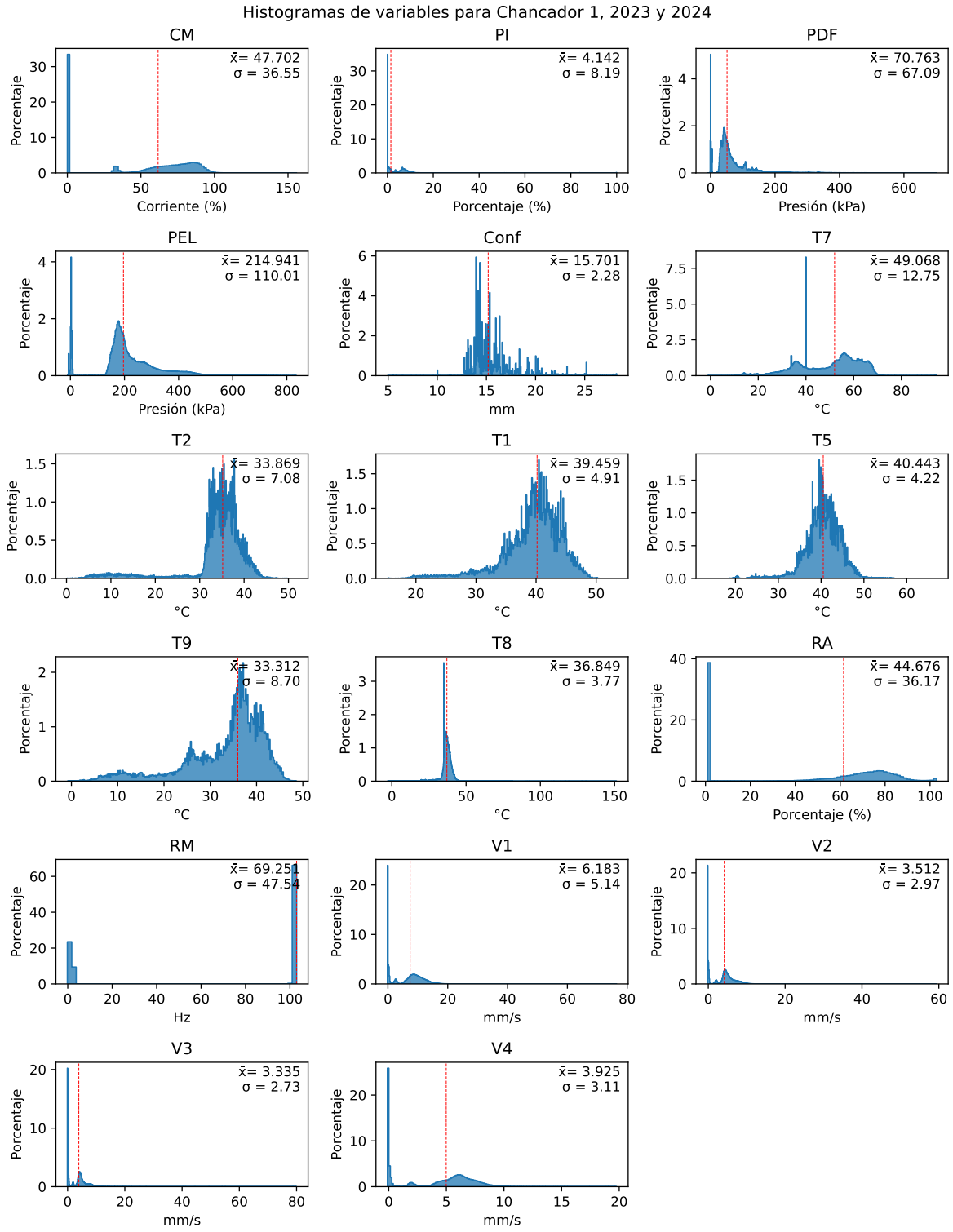

Figura 27: Histogramas de las variables del chancador 1 (2023 y 2024).

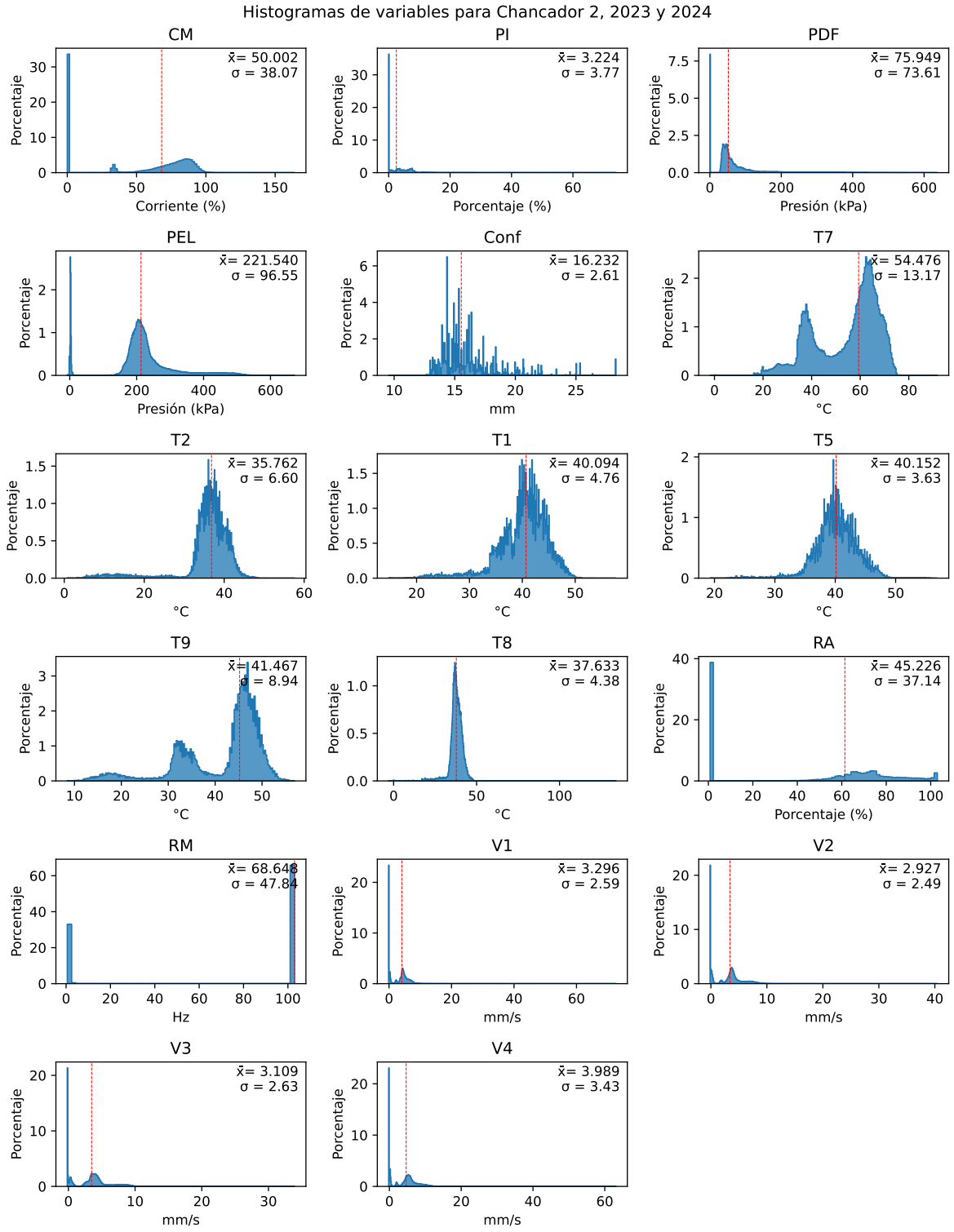

Figura 28: Histogramas de las variables del chancador 2 (2023 y 2024).

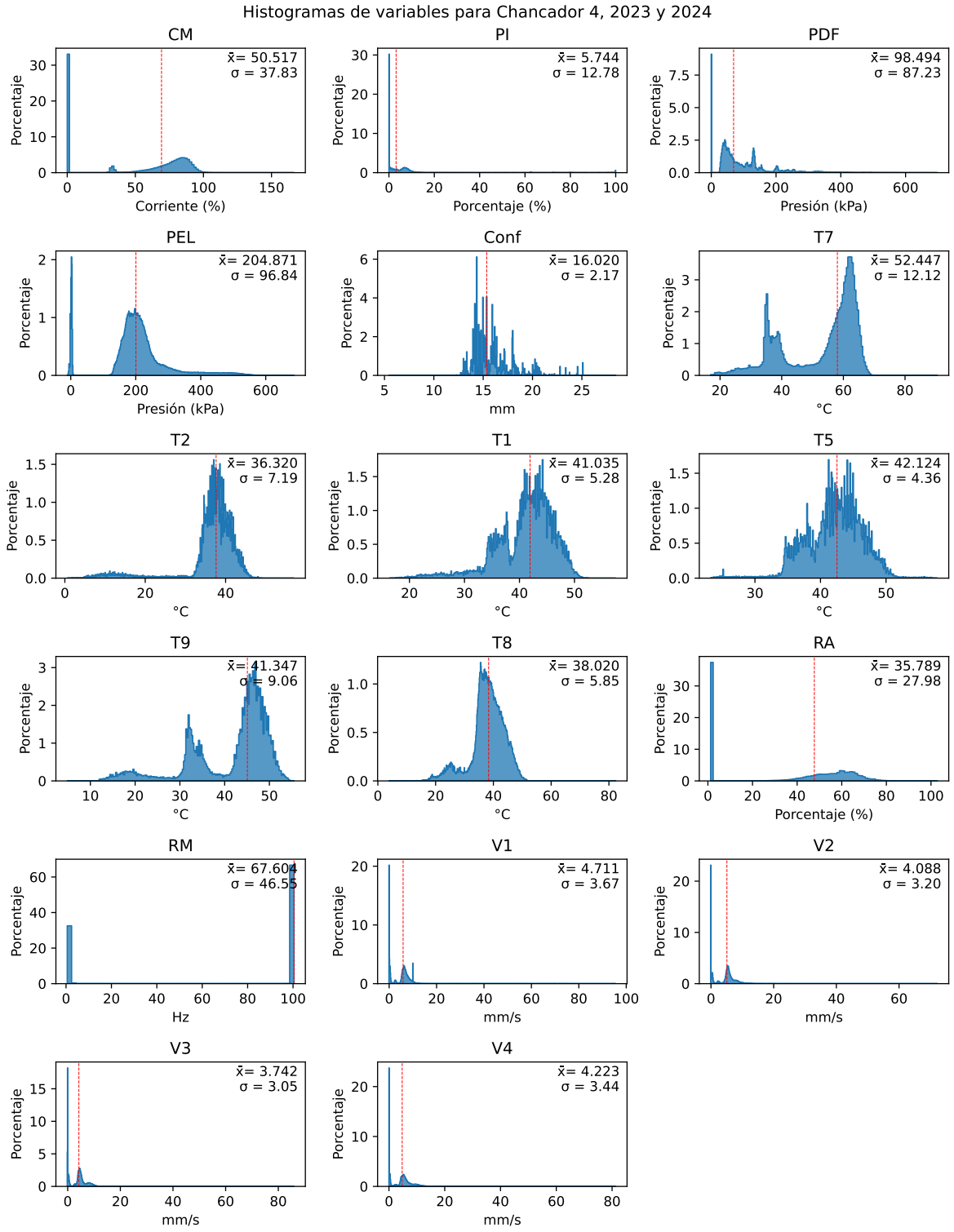

Figura 29: Histogramas de las variables del chancador 4 (2023 y 2024).

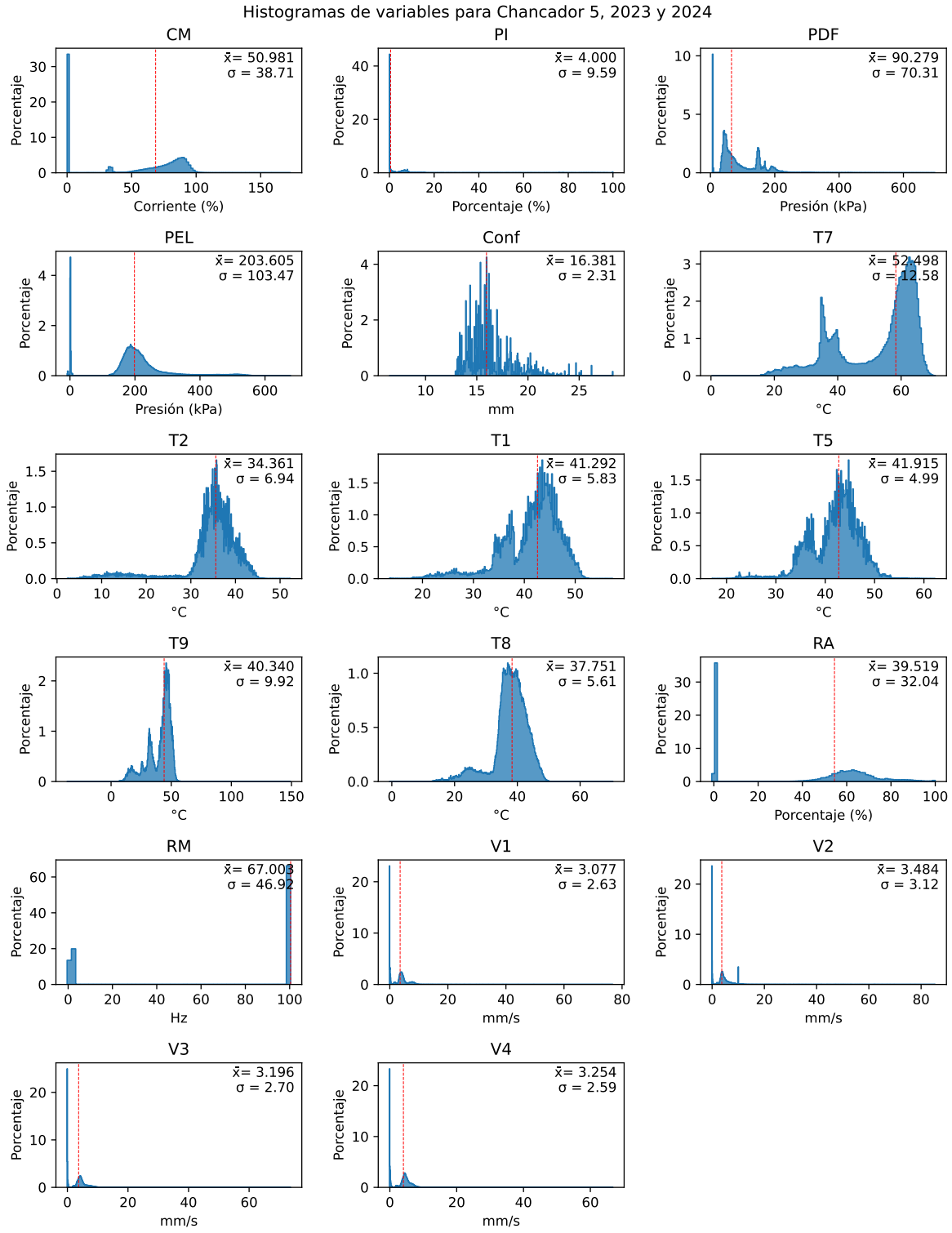

Figura 30: Histogramas de las variables del chancador 5 (2023 y 2024).

k	AE 10 E.		AE 60 E.		AE 200 E.		k-means	
	S	DB	S	DB	S	DB	S	DB
2	0,543	0,622	0,331	1,216	0,272	1,413	0,517	0,639
3	0,513	0,588	0,293	1,377	0,170	1,911	0,147	2,170
4	0,466	0,632	0,235	1,290	0,170	1,714	0,107	2,380
5	0,434	0,661	0,221	1,325	0,178	1,622	0,111	2,331
6	0,408	0,699	0,226	1,387	0,178	1,553	0,110	2,225
7	0,393	0,719	0,219	1,402	0,165	1,579	0,118	2,106
8	0,374	0,757	0,194	1,422	0,174	1,603	0,118	1,984
9	0,350	0,805	0,212	1,412	0,169	1,564	0,097	2,067
10	0,341	0,821	0,195	1,376	0,167	1,601	0,095	2,077
11	0,325	0,861	0,193	1,410	0,154	1,690	0,099	1,998
12	0,308	0,914	0,188	1,445	0,154	1,665	0,106	1,867
13	0,290	0,966	0,191	1,432	0,149	1,665	0,099	1,974
14	0,282	0,985	0,191	1,395	0,141	1,654	0,109	1,935
15	0,270	1,019	0,191	1,376	0,142	1,685	0,102	1,989
16	0,264	1,029	0,182	1,369	0,138	1,672	0,098	1,987
17	0,249	1,091	0,179	1,447	0,134	1,678	0,095	2,025
18	0,253	1,086	0,182	1,387	0,136	1,656	0,095	2,003
19	0,252	1,086	0,179	1,403	0,134	1,641	0,090	2,021
20	0,248	1,090	0,175	1,410	0,134	1,648	0,088	2,055
21	0,240	1,116	0,174	1,388	0,127	1,660	0,092	2,025
22	0,245	1,107	0,173	1,419	0,129	1,632	0,093	2,041
23	0,247	1,081	0,172	1,415	0,135	1,636	0,095	2,031
24	0,243	1,103	0,173	1,409	0,131	1,669	0,092	2,067
25	0,239	1,114	0,170	1,416	0,132	1,628	0,092	2,106
26	0,235	1,118	0,170	1,420	0,136	1,613	0,090	2,063
27	0,228	1,138	0,166	1,441	0,133	1,614	0,091	2,102
28	0,227	1,123	0,160	1,437	0,137	1,607	0,091	2,104
29	0,229	1,103	0,162	1,432	0,132	1,590	0,090	2,098
30	0,226	1,112	0,162	1,437	0,131	1,614	0,091	2,073
31	0,220	1,129	0,165	1,420	0,127	1,636	0,091	2,038

Tabla 12: Índices internos de las pruebas de k-means y AE-KMeans sobre el chancador 1

k	AE 10 E.		AE 60 E.		AE 200 E.		k-means	
	S	DB	S	DB	S	DB	S	DB
2	0,572	0,592	0,341	1,191	0,316	1,231	0,175	1,883
3	0,546	0,556	0,333	1,229	0,210	1,494	0,167	2,020
4	0,505	0,580	0,255	1,214	0,197	1,708	0,171	1,859
5	0,494	0,584	0,261	1,269	0,167	1,648	0,126	2,041
6	0,468	0,606	0,221	1,341	0,166	1,752	0,126	1,965
7	0,455	0,624	0,221	1,265	0,176	1,598	0,104	1,932
8	0,438	0,643	0,224	1,233	0,155	1,637	0,107	1,960
9	0,428	0,666	0,220	1,225	0,157	1,627	0,103	1,957
10	0,410	0,697	0,225	1,218	0,157	1,605	0,098	1,948
11	0,392	0,733	0,218	1,240	0,157	1,601	0,092	2,019
12	0,389	0,728	0,216	1,244	0,159	1,576	0,102	1,839
13	0,375	0,754	0,219	1,244	0,159	1,611	0,084	1,970
14	0,358	0,794	0,219	1,267	0,158	1,585	0,094	1,920
15	0,351	0,806	0,201	1,273	0,157	1,560	0,092	1,946
16	0,339	0,817	0,213	1,251	0,139	1,583	0,093	1,943
17	0,331	0,848	0,213	1,234	0,140	1,594	0,095	1,910
18	0,336	0,840	0,213	1,259	0,144	1,595	0,093	1,935
19	0,323	0,877	0,210	1,307	0,151	1,584	0,090	1,989
20	0,309	0,921	0,201	1,305	0,147	1,596	0,091	2,022
21	0,330	0,869	0,201	1,249	0,134	1,596	0,090	2,066
22	0,319	0,884	0,201	1,261	0,140	1,582	0,090	2,093
23	0,319	0,875	0,202	1,248	0,142	1,563	0,091	2,105
24	0,310	0,910	0,205	1,238	0,143	1,586	0,091	2,083
25	0,300	0,919	0,205	1,250	0,140	1,583	0,090	2,091
26	0,293	0,939	0,201	1,263	0,139	1,623	0,090	2,090
27	0,286	0,962	0,192	1,285	0,139	1,597	0,083	2,114
28	0,291	0,949	0,196	1,290	0,139	1,609	0,082	2,157
29	0,280	0,980	0,193	1,274	0,138	1,607	0,082	2,166
30	0,282	0,973	0,189	1,296	0,138	1,617	0,080	2,146
31	0,286	0,959	0,189	1,302	0,135	1,659	0,080	2,119

Tabla 13: Índices internos de las pruebas de k-means y AE-KMeans sobre el chancador 2

k	AE 10 E.		AE 60 E.		AE 200 E.		k-means	
	S	DB	S	DB	S	DB	S	DB
2	0,576	0,574	0,308	1,258	0,268	1,425	0,226	1,594
3	0,537	0,559	0,269	1,383	0,221	1,555	0,176	1,764
4	0,513	0,572	0,238	1,280	0,193	1,455	0,177	1,696
5	0,505	0,565	0,245	1,259	0,179	1,573	0,152	1,847
6	0,502	0,556	0,238	1,239	0,184	1,561	0,146	1,780
7	0,490	0,558	0,237	1,253	0,189	1,483	0,151	1,740
8	0,484	0,566	0,224	1,278	0,182	1,589	0,135	1,937
9	0,475	0,574	0,235	1,261	0,187	1,604	0,136	1,918
10	0,463	0,588	0,233	1,260	0,169	1,589	0,142	1,928
11	0,454	0,600	0,234	1,246	0,174	1,603	0,139	1,845
12	0,446	0,610	0,231	1,224	0,174	1,532	0,137	1,814
13	0,441	0,620	0,239	1,151	0,184	1,455	0,140	1,783
14	0,428	0,642	0,235	1,179	0,186	1,457	0,142	1,751
15	0,425	0,645	0,232	1,207	0,178	1,443	0,130	1,872
16	0,413	0,665	0,237	1,192	0,180	1,456	0,131	1,829
17	0,407	0,677	0,239	1,201	0,176	1,449	0,126	1,856
18	0,394	0,702	0,236	1,169	0,174	1,464	0,127	1,872
19	0,392	0,696	0,237	1,160	0,177	1,452	0,130	1,833
20	0,387	0,699	0,238	1,167	0,175	1,458	0,123	1,822
21	0,378	0,714	0,234	1,214	0,178	1,481	0,127	1,893
22	0,369	0,735	0,231	1,229	0,176	1,492	0,124	1,905
23	0,364	0,751	0,231	1,233	0,176	1,458	0,128	1,868
24	0,354	0,780	0,225	1,213	0,174	1,461	0,126	1,864
25	0,344	0,804	0,226	1,229	0,177	1,460	0,121	1,864
26	0,340	0,803	0,224	1,224	0,166	1,455	0,120	1,871
27	0,333	0,825	0,220	1,247	0,168	1,448	0,120	1,866
28	0,330	0,828	0,221	1,238	0,159	1,514	0,121	1,888
29	0,323	0,851	0,214	1,260	0,162	1,496	0,114	1,892
30	0,319	0,854	0,216	1,248	0,165	1,503	0,108	1,932
31	0,316	0,857	0,214	1,260	0,158	1,487	0,112	1,918

Tabla 14: Índices internos de las pruebas de k-means y AE-KMeans sobre el chancador 3

k	AE 10 E.		AE 60 E.		AE 200 E.		k-means	
	S	DB	S	DB	S	DB	S	DB
2	0,592	0,558	0,391	0,969	0,337	1,150	0,248	1,460
3	0,557	0,539	0,294	1,186	0,230	1,437	0,146	2,007
4	0,538	0,542	0,305	1,167	0,219	1,484	0,149	1,904
5	0,525	0,547	0,244	1,240	0,219	1,468	0,163	1,716
6	0,519	0,545	0,253	1,141	0,230	1,407	0,167	1,651
7	0,506	0,556	0,257	1,200	0,240	1,337	0,120	1,788
8	0,500	0,562	0,224	1,247	0,190	1,474	0,114	1,850
9	0,492	0,571	0,227	1,231	0,192	1,487	0,110	1,931
10	0,475	0,590	0,221	1,241	0,190	1,417	0,121	1,821
11	0,464	0,604	0,229	1,234	0,180	1,486	0,119	1,845
12	0,462	0,606	0,220	1,211	0,171	1,466	0,119	1,891
13	0,454	0,619	0,203	1,230	0,174	1,451	0,101	1,941
14	0,447	0,628	0,200	1,286	0,174	1,433	0,096	1,953
15	0,438	0,638	0,202	1,291	0,175	1,449	0,098	1,963
16	0,428	0,657	0,203	1,271	0,167	1,450	0,100	1,918
17	0,412	0,677	0,198	1,287	0,167	1,459	0,101	1,887
18	0,411	0,680	0,200	1,268	0,166	1,467	0,102	1,898
19	0,405	0,690	0,188	1,281	0,164	1,461	0,101	1,951
20	0,392	0,716	0,193	1,265	0,170	1,421	0,101	1,968
21	0,385	0,728	0,199	1,270	0,161	1,476	0,103	1,890
22	0,380	0,735	0,193	1,277	0,164	1,468	0,103	1,938
23	0,378	0,734	0,197	1,255	0,165	1,512	0,100	2,004
24	0,375	0,741	0,189	1,284	0,167	1,515	0,101	1,984
25	0,362	0,764	0,182	1,278	0,158	1,471	0,091	2,000
26	0,361	0,765	0,184	1,263	0,155	1,491	0,090	1,994
27	0,353	0,782	0,187	1,258	0,150	1,505	0,095	2,032
28	0,347	0,798	0,187	1,280	0,162	1,467	0,100	2,010
29	0,339	0,811	0,187	1,268	0,151	1,468	0,100	2,031
30	0,331	0,837	0,187	1,290	0,149	1,516	0,096	2,015
31	0,324	0,859	0,186	1,279	0,153	1,505	0,099	2,030

Tabla 15: Índices internos de las pruebas de k-means y AE-KMeans sobre el chancador 4

k	AE 10 E.		AE 60 E.		AE 200 E.		k-means	
	S	DB	S	DB	S	DB	S	DB
2	0,575	0,578	0,384	0,962	0,256	1,517	0,221	1,576
3	0,541	0,558	0,284	1,206	0,250	1,509	0,138	2,089
4	0,525	0,548	0,296	1,158	0,172	1,487	0,173	1,852
5	0,518	0,543	0,245	1,164	0,183	1,630	0,135	1,851
6	0,513	0,544	0,244	1,222	0,180	1,550	0,132	1,893
7	0,505	0,549	0,227	1,335	0,156	1,647	0,133	1,827
8	0,489	0,567	0,215	1,308	0,164	1,548	0,110	1,884
9	0,483	0,572	0,219	1,283	0,158	1,517	0,107	1,851
10	0,473	0,580	0,229	1,224	0,156	1,503	0,113	1,880
11	0,467	0,591	0,223	1,227	0,157	1,545	0,114	1,886
12	0,458	0,605	0,208	1,206	0,162	1,516	0,113	1,869
13	0,445	0,621	0,205	1,219	0,158	1,542	0,116	1,855
14	0,442	0,626	0,205	1,219	0,162	1,486	0,112	1,816
15	0,434	0,639	0,198	1,233	0,162	1,474	0,108	1,808
16	0,428	0,652	0,201	1,236	0,163	1,452	0,098	1,876
17	0,417	0,673	0,200	1,191	0,162	1,432	0,105	1,832
18	0,410	0,678	0,202	1,195	0,158	1,452	0,099	1,841
19	0,402	0,695	0,197	1,198	0,160	1,443	0,098	1,846
20	0,393	0,711	0,198	1,206	0,157	1,442	0,105	1,822
21	0,377	0,745	0,186	1,303	0,156	1,464	0,104	1,833
22	0,374	0,750	0,194	1,244	0,161	1,434	0,098	1,869
23	0,370	0,758	0,189	1,260	0,163	1,429	0,102	1,839
24	0,361	0,778	0,188	1,261	0,161	1,444	0,104	1,829
25	0,356	0,790	0,190	1,259	0,157	1,440	0,105	1,827
26	0,352	0,796	0,191	1,251	0,156	1,420	0,105	1,857
27	0,343	0,817	0,192	1,242	0,158	1,434	0,106	1,848
28	0,334	0,836	0,192	1,228	0,155	1,424	0,107	1,834
29	0,327	0,855	0,189	1,249	0,156	1,446	0,106	1,861
30	0,319	0,884	0,188	1,263	0,156	1,432	0,106	1,870
31	0,312	0,906	0,186	1,260	0,152	1,444	0,105	1,898

Tabla 16: Índices internos de las pruebas de k-means y AE-KMeans sobre el chancador 5

k	AE 10 E.		AE 60 E.		AE 200 E.		k-means	
	A	NA	A	NA	A	NA	A	NA
2	2,808	4,314	3,593	1,956	3,611	1,810	0,082	17,776
3	6,810	8,129	5,003	4,058	9,958	4,921	8,332	5,795
4	10,940	7,433	9,358	7,365	10,746	12,135	10,945	14,998
5	16,539	8,880	10,911	9,320	10,495	12,598	10,671	15,387
6	20,328	9,226	10,791	9,680	11,351	13,971	12,476	16,048
7	26,348	8,945	14,279	17,026	14,844	14,791	12,296	16,379
8	28,673	9,537	15,456	17,174	14,799	14,148	12,371	15,273
9	31,813	9,188	21,517	18,043	13,837	15,545	12,308	13,185
10	38,215	9,622	22,358	17,481	15,545	12,501	26,831	12,509
11	38,540	9,659	25,940	19,597	17,424	16,016	25,844	14,391
12	39,352	8,713	29,602	20,211	16,318	17,059	25,871	17,725
13	39,980	8,075	28,493	20,659	18,896	16,850	24,676	17,725
14	43,672	9,033	30,071	20,317	19,371	16,600	28,975	17,725
15	43,672	10,104	32,013	20,231	22,704	16,600	29,030	17,725
16	45,963	9,849	33,406	20,740	30,732	16,782	28,407	20,123
17	46,305	10,005	30,651	19,022	23,245	17,033	28,370	22,596
18	46,305	10,453	33,862	19,245	21,730	17,610	28,553	23,674
19	46,305	13,550	30,795	20,449	20,656	17,316	29,091	23,788
20	46,244	14,346	32,371	20,444	33,000	18,807	28,665	23,782
21	45,728	14,592	31,927	20,007	34,308	18,750	26,587	23,745
22	45,962	14,489	32,136	19,996	34,061	18,850	31,801	23,851
23	45,268	14,389	37,614	19,952	24,159	19,279	31,910	23,916
24	44,110	14,643	41,002	19,736	24,592	19,219	39,036	23,339
25	44,321	15,906	42,075	20,405	36,269	19,173	30,864	22,568
26	44,629	14,285	41,677	20,417	36,693	19,507	30,899	23,014
27	44,629	15,388	42,014	18,954	37,362	19,536	30,321	23,011
28	44,629	15,388	42,143	18,786	37,730	19,397	31,260	23,130
29	44,271	15,388	44,761	21,808	37,815	20,419	30,808	24,072
30	44,271	15,091	44,729	21,877	37,607	19,557	44,294	24,072
31	44,271	15,091	45,941	21,975	38,945	21,503	44,356	30,696

Tabla 17: Máxima distancia W_{emd} entre los datos del chancador 1 y un grupo anómalo (A) y uno no anómalo (NA)

k	AE 10 E.		AE 60 E.		AE 200 E.		k-means	
	A	NA	A	NA	A	NA	A	NA
2	5,190	3,183	5,558	3,404	6,184	4,268	6,927	3,343
3	11,635	4,260	16,572	5,776	9,795	6,817	17,829	6,241
4	17,364	4,764	19,060	6,346	17,486	7,187	18,388	7,309
5	24,737	4,943	19,667	7,188	18,258	8,438	22,329	13,991
6	30,883	5,093	19,404	7,188	18,445	9,299	20,703	14,524
7	35,999	5,566	28,747	9,265	18,723	9,550	21,898	14,996
8	48,531	5,203	33,488	19,089	18,887	11,400	31,744	20,722
9	51,721	5,623	33,627	19,071	24,177	18,904	33,803	20,722
10	51,600	5,753	36,592	19,527	32,989	19,155	32,397	20,722
11	54,838	6,106	37,870	19,655	33,001	19,220	42,862	21,332
12	88,861	6,112	34,393	19,655	34,224	19,053	100,624	21,076
13	90,856	6,180	41,790	19,879	34,553	19,215	100,624	21,557
14	90,856	6,443	44,500	19,862	34,615	19,375	100,624	22,108
15	104,793	6,365	43,996	21,887	45,779	19,482	100,624	22,042
16	119,042	6,164	41,376	21,729	44,883	19,665	100,624	22,294
17	119,042	6,036	42,990	21,609	44,738	21,637	100,624	22,527
18	119,042	6,444	42,937	21,609	45,660	20,304	100,624	22,462
19	119,042	7,335	45,345	22,228	45,834	21,170	100,624	22,481
20	119,042	7,963	43,925	22,336	46,184	20,455	100,624	23,437
21	119,042	6,981	42,612	21,529	99,685	23,631	100,624	23,321
22	128,645	7,566	41,288	21,418	99,685	23,207	100,624	23,323
23	149,866	7,562	42,123	21,418	99,685	23,183	100,624	23,338
24	149,866	6,669	45,561	21,324	99,775	23,183	100,624	23,419
25	149,866	6,886	46,257	21,324	100,549	24,612	100,624	23,401
26	151,173	7,709	44,661	23,284	101,410	24,437	100,624	23,453
27	151,173	7,715	50,019	23,284	101,762	24,540	100,624	23,440
28	151,173	7,781	50,130	22,603	101,410	24,469	100,624	26,471
29	151,173	7,820	57,221	22,589	101,762	24,456	100,624	26,187
30	151,173	8,535	58,956	23,011	101,410	24,396	100,624	26,782
31	151,173	9,235	55,334	23,846	101,762	24,245	100,624	27,535

Tabla 18: Máxima distancia W_{emd} entre los datos del chancador 2 y un grupo anómalo (A) y uno no anómalo (NA)

k	AE 10 E.		AE 60 E.		AE 200 E.		k-means	
	A	NA	A	NA	A	NA	A	NA
2	7,789	5,886	9,448	6,928	13,590	10,632	7,433	5,691
3	21,478	7,581	11,341	12,039	13,590	14,174	13,669	10,607
4	36,909	6,624	21,833	11,834	30,063	14,078	17,109	10,905
5	56,573	7,771	24,372	15,413	35,241	12,674	21,104	14,518
6	72,797	7,527	33,426	18,432	38,551	15,278	32,950	18,070
7	88,956	7,562	40,319	19,800	39,237	15,750	36,786	17,652
8	98,390	7,809	43,791	19,899	43,025	19,288	40,776	19,959
9	114,270	7,417	45,227	24,005	43,719	20,202	40,873	19,613
10	118,801	7,909	48,644	24,348	49,015	20,470	43,226	20,755
11	127,714	7,908	49,740	24,177	44,847	20,566	51,951	22,728
12	135,426	8,016	68,711	24,329	68,671	20,447	60,167	22,539
13	136,583	8,438	69,487	24,246	82,248	21,178	60,430	23,429
14	136,583	8,435	75,895	23,943	68,614	20,891	63,281	22,785
15	155,688	8,342	75,794	24,159	68,614	21,592	71,156	23,276
16	155,688	8,786	76,520	24,113	70,554	22,175	94,933	23,198
17	156,657	8,280	78,749	24,142	69,878	22,169	98,275	23,260
18	164,612	8,732	82,863	24,116	82,425	22,299	91,783	23,361
19	196,173	8,694	100,513	24,115	73,568	22,303	91,783	23,328
20	202,687	8,607	105,275	24,106	105,791	23,204	97,799	23,564
21	203,595	8,348	100,377	24,144	68,156	22,444	96,135	23,663
22	203,595	8,597	99,776	24,136	85,211	22,910	97,281	23,682
23	203,595	9,067	101,492	24,195	98,265	23,577	97,929	23,447
24	203,595	8,312	85,505	24,311	98,542	23,413	96,416	23,529
25	203,595	9,129	91,455	24,306	100,323	23,750	101,214	23,617
26	207,838	9,129	125,538	24,306	121,175	23,551	103,892	23,676
27	207,838	8,768	128,404	24,300	122,075	23,348	103,821	23,593
28	208,734	8,892	127,249	24,297	89,747	23,635	103,762	23,471
29	208,734	8,848	127,914	24,344	93,383	23,508	106,143	23,386
30	209,335	8,848	128,451	24,004	103,091	23,757	118,892	23,310
31	215,091	8,848	128,433	24,317	129,403	23,630	109,518	23,524

Tabla 19: Máxima distancia W_{emd} entre los datos del chancador 3 y un grupo anómalo (A) y uno no anómalo (NA)

k	AE 10 E.		AE 60 E.		AE 200 E.		k-means	
	A	NA	A	NA	A	NA	A	NA
2	3, 325	2,216	4,406	3,413	8,827	6,985	3,275	2,233
3	11, 719	6,402	10,859	5,042	14,823	8,896	11,478	6,633
4	21, 317	6,398	21,345	8,164	17,781	9,864	20,652	8,416
5	27, 19	7,155	23,907	7,912	16,431	9,526	19,720	8,614
6	37, 537	7,197	23,781	9,362	21,384	9,876	21,932	10,997
7	42, 692	7,813	21,028	9,959	21,251	10,567	23,304	12,181
8	48, 261	7,600	27,431	11,142	28,109	11,937	29,903	14,358
9	59, 232	7,735	30,839	13,065	27,841	12,092	29,590	14,672
10	59, 232	7,690	31,122	13,349	32,411	13,898	26,350	13,488
11	60, 469	7,549	41,733	13,483	33,916	13,022	46,846	14,964
12	79, 597	7,889	42,513	13,406	32,724	13,185	56,066	16,358
13	82, 131	7,805	47,824	12,997	32,833	13,131	56,799	15,824
14	88, 973	8,136	42,260	13,856	40,236	13,162	55,123	17,407
15	100, 9	8,343	42,787	14,220	40,380	13,703	56,901	18,112
16	100, 9	8,026	43,603	14,635	40,397	14,627	56,619	18,078
17	100, 9	8,419	48,810	14,711	49,469	15,292	55,830	17,561
18	119, 694	8,349	47,280	15,123	51,188	15,909	55,493	18,584
19	120, 483	8,084	53,005	14,876	51,379	16,048	57,298	18,578
20	120, 483	8,233	49,810	15,237	50,968	15,898	57,392	18,699
21	129, 688	8,395	69,800	14,932	56,065	14,624	56,359	18,259
22	129, 688	8,235	67,248	15,802	57,433	15,280	64,678	18,417
23	140, 162	8,139	58,249	15,205	48,622	15,521	61,640	19,064
24	147, 564	7,987	70,212	14,855	56,801	15,451	62,903	19,136
25	147, 564	8,421	70,491	16,125	50,756	16,260	63,284	18,728
26	156, 655	8,232	66,821	15,848	44,994	17,107	60,995	19,559
27	156, 655	8,662	69,166	15,477	61,405	17,351	62,927	19,167
28	156, 655	8,365	68,129	14,807	61,180	16,568	69,460	18,645
29	156, 655	8,447	62,529	15,416	59,687	17,025	69,134	18,951
30	156, 655	7,985	63,010	15,360	44,326	17,860	68,898	19,143
31	156, 655	8,268	77,305	15,375	58,636	17,579	68,346	19,402

Tabla 20: Máxima distancia W_{emd} entre los datos del chancador 4 y un grupo anómalo (A) y uno no anómalo (NA)

k	AE 10 E.		AE 60 E.		AE 200 E.		k-means	
	A	NA	A	NA	A	NA	A	NA
2	5,080	3,805	5,470	3,954	10,154	7,760	4,895	3,749
3	15,745	8,576	16,145	8,518	10,392	9,674	12,121	10,070
4	28,127	8,512	20,309	10,587	20,354	9,294	22,403	11,005
5	39,553	8,933	31,121	11,021	27,333	9,547	26,339	10,718
6	50,088	9,092	30,234	11,180	22,110	10,999	26,873	10,963
7	68,452	9,213	38,125	12,155	23,801	11,321	28,339	13,008
8	78,797	9,292	43,316	12,422	24,558	11,655	38,159	13,194
9	82,822	9,202	48,334	12,394	24,558	11,625	35,363	13,147
10	93,867	9,344	48,335	14,757	31,016	13,965	35,674	13,251
11	99,607	9,204	51,240	14,672	40,693	13,373	41,853	13,711
12	101,044	9,291	60,448	14,693	37,824	13,438	42,981	14,041
13	104,755	9,434	65,259	14,810	46,055	13,488	42,925	14,339
14	129,210	9,436	60,468	14,769	49,162	13,319	47,464	14,826
15	129,107	9,411	75,071	15,003	45,429	15,043	46,831	14,826
16	129,016	9,336	71,634	14,778	46,305	14,626	52,607	14,767
17	129,016	9,326	80,907	14,875	45,401	14,906	48,201	14,826
18	147,483	9,396	79,803	14,869	47,880	14,947	58,251	14,826
19	149,400	9,415	77,982	14,780	52,097	14,948	57,572	14,834
20	150,022	9,573	78,778	14,790	57,376	14,882	57,129	14,833
21	131,448	9,698	87,098	14,725	55,511	14,999	62,422	14,819
22	150,288	9,317	87,953	14,808	51,185	14,995	64,636	14,812
23	152,777	9,577	83,282	14,794	55,321	15,214	64,730	15,340
24	152,555	9,420	85,173	14,790	62,557	15,054	67,270	15,201
25	152,890	9,677	83,958	14,817	60,848	15,364	66,050	15,336
26	155,134	9,640	83,364	14,753	68,762	15,364	67,157	15,336
27	155,134	9,628	90,676	14,833	75,087	15,219	75,973	15,336
28	155,134	9,591	91,905	14,828	73,695	15,364	76,407	15,336
29	155,134	9,605	92,276	14,843	65,613	15,364	86,353	15,336
30	155,134	9,447	94,324	14,825	64,920	15,364	86,004	15,328
31	155,134	9,589	101,505	14,971	67,440	15,428	89,492	15,328

Tabla 21: Máxima distancia W_{emd} entre los datos del chancador 5 y un grupo anómalo (A) y uno no anómalo (NA)

Chancador	ε	**N**	**S**	**DB**	W_{emd}
Chancador 1	0,1	1	0,399	4,772	31,253
	0,11	1	0,440	3,199	29,133
	0,12	1	0,465	2,912	25,047
	0,13	1	0,460	2,613	21,035
	0,14	1	0,442	2,214	11,650
Chancador 2	0,12	1	0,259	3,615	64,088
	0,128	1	0,334	2,142	87,477
	0,135	1	0,357	1,579	102,217
	0,143	1	0,385	1,242	117,431
	0,15	1	0,427	1,079	123,370
Chancador 3	0,1	1	0,285	20,995	46,549
	0,125	1	0,348	6,161	74,664
	0,15	1	0,492	1,244	153,953
	0,175	1	0,528	0,384	202,373
	0,2	1	0,535	0,314	214,862
Chancador 4	0,09	1	0,174	5,140	29,968
	0,1125	1	0,226	4,870	31,931
	0,135	1	0,284	3,928	40,965
	0,1575	1	0,429	1,599	80,324
	0,18	1	0,500	0,621	123,943
Chancador 5	0,09	1	0,181	12,739	46,919
	0,1125	1	0,311	12,535	66,201
	0,135	1	0,447	2,182	120,040
	0,1575	1	0,505	0,547	188,643
	0,18	1	0,514	0,345	209,204

Tabla 22: Resultados con Autoencoder de 10 Epochs y DBSCAN

Chancador	ε	**N**	**S**	**DB**	W_{emd}
Chancador 1	0,6	1	0,170	2,637	7,097
	0,75	1	0,240	2,312	13,713
	0,9	1	0,291	2,066	19,068
	1,05	1	0,339	1,832	24,573
	1,2	1	0,375	1,576	29,311
Chancador 2	0,8	1	0,278	2,142	15,668
	0,95	1	0,327	1,963	19,781
	1,1	1	0,370	1,709	24,811
	1,25	1	0,408	1,452	30,021
	1,4	1	0,439	1,269	33,698
Chancador 3	0,9	2	0,134	2,409	28,871
	1,05	2	0,170	2,153	33,428
	1,2	1	0,212	2,453	38,475
	1,35	1	0,245	2,214	44,506
	1,5	1	0,278	1,949	51,909
Chancador 4	0,7	1	0,189	3,052	18,675
	0,9	1	0,249	2,267	29,448
	1,1	1	0,312	1,784	42,499
	1,3	1	0,360	1,455	53,862
	1,5	1	0,406	1,242	65,484
Chancador 5	0,8	1	0,293	2,339	27,006
	1	1	0,346	1,788	39,281
	1,2	1	0,385	1,468	49,440
	1,4	1	0,416	1,182	58,776
	1,6	1	0,451	0,872	71,784

Tabla 23: Resultados con Autoencoder de 60 Epochs y DBSCAN

Chancador	ε	**N**	**S**	**DB**	W_{emd}
Chancador 1	2	1	0,185	4,230	11,063
	2,25	1	0,228	3,976	15,300
	2,5	1	0,268	3,583	22,420
	2,75	1	0,289	3,028	28,675
	3	1	0,309	2,542	36,923
Chancador 2	2	1	0,270	2,583	21,430
	2,25	1	0,299	2,208	25,236
	2,5	1	0,325	1,915	30,333
	2,75	1	0,350	1,583	33,679
	3	1	0,382	1,351	36,255
Chancador 3	1,7	2	0,120	2,652	22,764
	2,15	1	0,211	3,135	31,920
	2,6	1	0,282	2,446	50,931
	3,05	1	0,359	1,888	74,176
	3,5	1	0,434	1,291	94,253
Chancador 4	1,5	3	0,086	1,794	16,075
	2	1	0,307	1,939	32,546
	2,5	1	0,372	1,575	45,391
	3	1	0,423	1,372	53,819
	3,5	1	0,464	1,160	65,897
Chancador 5	1,5	2	0,166	2,166	13,520
	2	2	0,220	2,024	25,426
	2,5	1	0,366	1,809	34,996
	3	1	0,417	1,507	48,312
	3,5	1	0,476	1,155	69,130

Tabla 24: Resultados con Autoencoder de 200 Epochs y DBSCAN