

**UNIVERSIDAD DE COSTA RICA**  
**Facultad de Ingeniería**  
**Escuela de Ingeniería Eléctrica**

**IE0499 – Proyecto Eléctrico**

**Simulación de Aplicaciones de la Inteligencia Artificial  
(IA) en la Ingeniería Eléctrica: Métodos de agrupamiento**

por

**Brandon José Esquivel Molina**

**Ciudad Universitaria Rodrigo Facio**

Agosto de 2021



# **Simulación de Aplicaciones de la Inteligencia Artificial (IA) en la Ingeniería Eléctrica: Métodos de agrupamiento**

por

**Brandon José Esquivel Molina**

B52571

**IE0499 – Proyecto Eléctrico**

Aprobado por

---

Dr. Marvin Coto Jimenez

*Profesor guía*

---

Dr. Jorge Arturo Romero Chacón

*Profesor lector*

---

Dr. Federico Ruiz Ugalde

*Profesor lector*

Agosto de 2021



## Resumen

# Simulación de Aplicaciones de la Inteligencia Artificial (IA) en la Ingeniería Eléctrica: Métodos de agrupamiento

por

**Brandon José Esquivel Molina**

Universidad de Costa Rica

Escuela de Ingeniería Eléctrica

Profesor guía: Dr. Marvin Coto Jimenez

Agosto de 2021

El amplio campo que trata problemas que pueden ser formulados y resueltos por maquinas se conoce como inteligencia artificial (IA), problemas que en un principio parecían imposibles, intratables y difíciles de formular utilizando ordenadores. La Inteligencia Artificial (IA) es un área de gran importancia en la actualidad, gracias a la gran cantidad de aplicaciones exitosas y a su gran difusión e inversión a nivel mundial. Hoy, el campo de la IA engloba varias subareas tales como los sistemas expertos, la demostración automática de teoremas, el juego automático, el reconocimiento de la voz y de patrones, el procesamiento del lenguaje natural, la visión artificial, la robótica, las redes neuronales, procesamiento y ordenamiento de datos o métodos de agrupamiento, entre otros.

El presente proyecto tiene un enfoque teórico/práctico/didáctico que relaciona estos conceptos para comprender las técnicas específicas de inteligencia artificial seleccionadas y su relación con la ingeniería eléctrica, se busca generar un conjunto de datos de prueba que recree alguna de las experiencias reportadas en las publicaciones académicas. Para esto, se busca crear datos de manera artificial, que tengan magnitudes y unidades según la aplicación, correspondiente a las secciones de simulaciones y pruebas. Finalmente se genera un cuaderno en Jupyter, y una actividad didáctica donde se aplican los conceptos de Inteligencia Artificial para que eventualmente un grupo de estudiantes pueda explorarlo y llegar a conclusiones sobre el mismo.

**Palabras claves:** Inteligencia Artificial, IA, métodos de agrupamiento, Jupyter Notebook, simulación, Datos.

---

### Acerca de IE0499 – Proyecto Eléctrico

El Proyecto Eléctrico es un curso semestral bajo la modalidad de trabajo individual supervisado, con el propósito de aplicar estrategias de diseño y análisis a un problema de temática abierta de la ingeniería eléctrica. Es un requisito de graduación para el grado de Bachiller en Ingeniería Eléctrica de la Universidad de Costa Rica.



*Dedicado a mi hermano Dylan, por su completa confianza en mi y todo su apoyo.*

## **Agradecimientos**

A mi padre y madre por haber dado lo mejor de si para enseñarme el camino al estudio, a mi amigo Yu, por sus consejos y apoyo como hermano mayor de carrera, a mis amigos cercanos de la facultad, que siempre me hicieron dar lo mejor de mi, a las personas que estuvieron cerca, día a día, dándome energía e impulso para seguir hacia la meta, y finalmente, gracias a mi, por no haberme dado por vencido a pesar de las adversidades y los retos, por haber soñado más allá del horizonte que me rodeaba y destruir cualquier idea conformista que pudiera llegar.



# Índice general

<b>Índice general</b>	<b>ix</b>
<b>Índice de figuras</b>	<b>x</b>
<b>1 Introducción</b>	<b>1</b>
1.1. Justificación . . . . .	1
1.2. Objetivos . . . . .	2
1.3. Alcance . . . . .	3
1.4. Cronograma . . . . .	4
<b>2 Marco teórico</b>	<b>5</b>
2.1. Inteligencia Artificial (IA) y Teorías de aprendizaje . . . . .	5
2.1.1. Definiciones . . . . .	5
2.1.2. Breve reseña Histórica . . . . .	6
2.1.3. Actualidad . . . . .	7
2.2. Métodos de agrupamiento (clustering) . . . . .	7
2.2.1. Tipos y Definiciones . . . . .	8
2.2.2. Métodos y Algoritmos comunes . . . . .	9
2.3. Sistemas de evaluación . . . . .	13
2.3.1. Validación Interna . . . . .	13
2.3.2. Validación Externa . . . . .	14
2.4. Herramientas de software: Python y Jupyter . . . . .	15
2.4.1. NumPy . . . . .	15
2.4.2. Matplotlib . . . . .	15
2.4.3. Pandas . . . . .	15
2.4.4. scikit-learn . . . . .	15
<b>3 Desarrollo de aplicaciones</b>	<b>17</b>
3.1. Aplicaciones implementadas . . . . .	17
3.1.1. Clustering en Generación de Energía por Región . . . . .	17
3.1.2. Clustering con K-means - Catado de cafe . . . . .	18

3.1.3.	Comparación de Métodos de Agrupamiento - EMG y Redes neuronales . . . . .	18
3.1.4.	DSP. Analizando sonidos de manatíes - Actividad práctica . . . . .	19
<b>4</b>	<b>Resultados Obtenidos</b>	<b>21</b>
4.1.	Introducción . . . . .	21
4.2.	Conjuntos de datos generados . . . . .	21
4.3.	Gráficos y resultados obtenidos . . . . .	23
4.3.1.	Aplicando KMeans a la industria cafetera . . . . .	23
4.3.2.	Clustering en Generación de Energía . . . . .	25
4.3.3.	EMG - Comparando Algoritmos de Clustering . . . . .	28
4.4.	Actividades específicas por implementar . . . . .	30
4.5.	Cuadernos de trabajo Jupyter Notebook . . . . .	30
<b>5</b>	<b>Conclusiones y Recomendaciones</b>	<b>31</b>
5.1.	Conclusiones . . . . .	31
5.2.	Recomendaciones . . . . .	32
5.3.	Anexos . . . . .	32
5.3.1.	Código . . . . .	32
5.3.2.	Conjuntos de datos utilizados . . . . .	32
5.3.3.	Cuadernos Jupyter Notebook creados . . . . .	32
<b>Bibliografía</b>		<b>47</b>

# Índice de figuras

1.1.	Construcción del objetivo general. . . . .	3
2.1.	Distinción entre IA como ciencia e IA como ingeniería. Adaptado de: [9] . . . . .	6
2.2.	Árbol de tipos de métodos de clasificación. Adaptado de: [Jain y otros, 1999] y [Lance y Williams, 1967] . . . . .	8
2.3.	Ejemplo visual usando K-means. Adaptado de Internet . . . . .	9
2.4.	Ejemplo visual de la agrupación por densidad usando DBSCAN. Tomado de: [5] . . . . .	10
2.5.	Ejemplo comparativo sobre la agrupación por forma de DBSCAN vs K-means. Tomado de Github . . . . .	11
2.6.	Gráfica intermedia de generación de regiones con el método HDBSCAN. Tomado de su documentación en Github . . . . .	12

4.1.	Conjunto de datos EMG, mostrado en el cuaderno Jupyter . . . . .	22
4.2.	Conjunto de datos DSP, mostrado en el cuaderno Jupyter . . . . .	22
4.3.	Conjunto de datos Generación de Energía, mostrado en el cuaderno Jupyter . . . . .	23
4.4.	Gráfico codo de Jambú, actividad Kmeans . . . . .	24
4.5.	Gráfico obtenido, actividad Kmeans . . . . .	24
4.6.	Gráfico obtenido, actividad Kmeans PCA . . . . .	25
4.7.	Gráfico obtenido, actividad clustering jerárquico HDBSCAN . . . . .	26
4.8.	Gráfico remarcado, actividad HDBSCAN . . . . .	26
4.9.	Gráfico obtenido, actividad hdbscan, post-procesado con PCA . . . . .	27
4.10.	Gráfico obtenido del clustering con MS, actividad comparando algoritmos . . . . .	28
4.11.	Gráfico obtenido del clustering con AP, actividad comparando algoritmos . . . . .	28
4.12.	Gráfico obtenido del clustering con Kmeans, actividad comparando algoritmos . . . . .	29
4.13.	Gráfico obtenido del clustering con SC, actividad comparando algoritmos . . . . .	29



## Capítulo 1

# INTRODUCCIÓN

La IA o inteligencia artificial es un área de estudio con una definición comúnmente aceptada que la relaciona con el diseño y el análisis de sistemas con naturaleza artificial y que resulten autónomos, capaces de exhibir un comportamiento inteligente. Se puede contemplar de dos enfoques: como ciencia o como ingeniería. En el presente documento se hablará de forma resumida sobre éstos dos enfoques, los conceptos que involucra la IA, su historia y su contexto actual, se dará enfoque a los métodos de agrupamiento o clustering, teorías de aprendizaje en clustering, sus algoritmos y sus métodos de evaluación, específicamente en los métodos no supervisados, además, se orientará a su relación con la ingeniería eléctrica y una serie de aplicaciones seleccionadas a replicar de forma didáctica/práctica con el fin de representar una base metodológica de aprendizaje a los estudiantes de la Escuela de Ingeniería eléctrica interesados en la materia.

### 1.1. Justificación

La Inteligencia Artificial es la parte de la Ciencia que se encarga del diseño y estudio de sistemas computacionales inteligentes, es decir, sistemas que muestran características que relacionamos con la inteligencia humana, que se refiere, por ejemplo, a comprensión del lenguaje, el razonamiento lógico, el aprendizaje, la resolución de problemas entre otros. Este enorme campo de estudio da pie a una gran cantidad de aplicaciones (IA como ingeniería) en áreas de estudio como la Medicina, biología, estudios sociológicos, análisis de datos, etc, razón por la cual su importancia ha ido en crecimiento importante en los últimos años y se predice que siga creciendo al punto de ser una de las ramas dominantes de la tecnología en los próximos años.

Una aplicación de la IA son los métodos de agrupamiento o clustering, que son las técnicas o algoritmos que permiten de forma automática y eficiente agrupar, categorizar y etiquetar bases de datos o información no organizada en diferentes grupos o clusters. En un mundo digital donde la cantidad de datos es cada vez mayor, alcanzando cifras inimaginables, la necesidad de ordenamiento y agrupación se vuelve un factor urgente y primordial para lograr analizar ésta información de forma efectiva, coherente y sobre todo automática e inteligente. Son cada vez más las aplicaciones que requieren de este tipo y magnitud de trabajo sistémico y siempre será mayor, en proporción del crecimiento de los horizontes de datos e información obtenible. Por lo anteriormente expuesto, resulta necesario introducir a los

estudiantes de ingeniería en ésta área de estudio, sus aplicaciones actuales y las posibles aplicaciones abarcables y para el alcance de este proyecto, a los estudiantes de Ingeniería Eléctrica específicamente.

Algunas razones específicas de justificación del estudio de la IA y los métodos de agrupamiento son, por ejemplo, que el mantenimiento y el coste marginal de su uso repetido es relativamente bajo, las ganancias en términos monetarios, de tiempo y eficiencia resultantes del uso de los sistemas IA son muy altas, se puede responder a preguntas y resolver problemas mucho más rápidamente que un experto humano, suministrando respuestas fiables en situaciones en las que los humanos expertos no pueden. Los sistemas de análisis de datos con métodos de agrupamiento eficientes pueden ser utilizados para realizar operaciones monótonas, extenuantes, intratables o inconfortables para los humanos, en materia de ordenamiento y clasificación o para detectar nuevos patrones no siempre visibles manualmente. [1]

## **1.2. Objetivos**

El presente proyecto tiene como objetivo general OG: Desarrollar una propuesta de recursos para la simulación y aprendizaje de aplicaciones relevantes de la inteligencia artificial aplicables en la ingeniería eléctrica, específicamente de métodos de agrupamiento. La construcción de este objetivo primario se llevará a cabo por medio de los objetivos específicos que buscan OE1: comprender los algoritmos principales de agrupamiento y sus métricas de evaluación, y el papel que juegan dentro de la inteligencia artificial, OE2: seleccionar aplicaciones relevantes de la ingeniería eléctrica donde se apliquen los métodos de agrupamiento, OE3: generar o adaptar un conjunto de datos donde los métodos de agrupamiento se apliquen en problemas relevantes de la ingeniería eléctrica, OE4: generar actividades didácticas donde se puedan ilustrar, simular y estudiar los métodos de agrupamiento y plantear un proyecto destinado a estudiantes de ingeniería eléctrica en una lección sobre métodos de agrupamiento. En el siguiente diagrama se muestra la construcción de estos objetivos, brindando un panorama metodológico de actividades por objetivo.

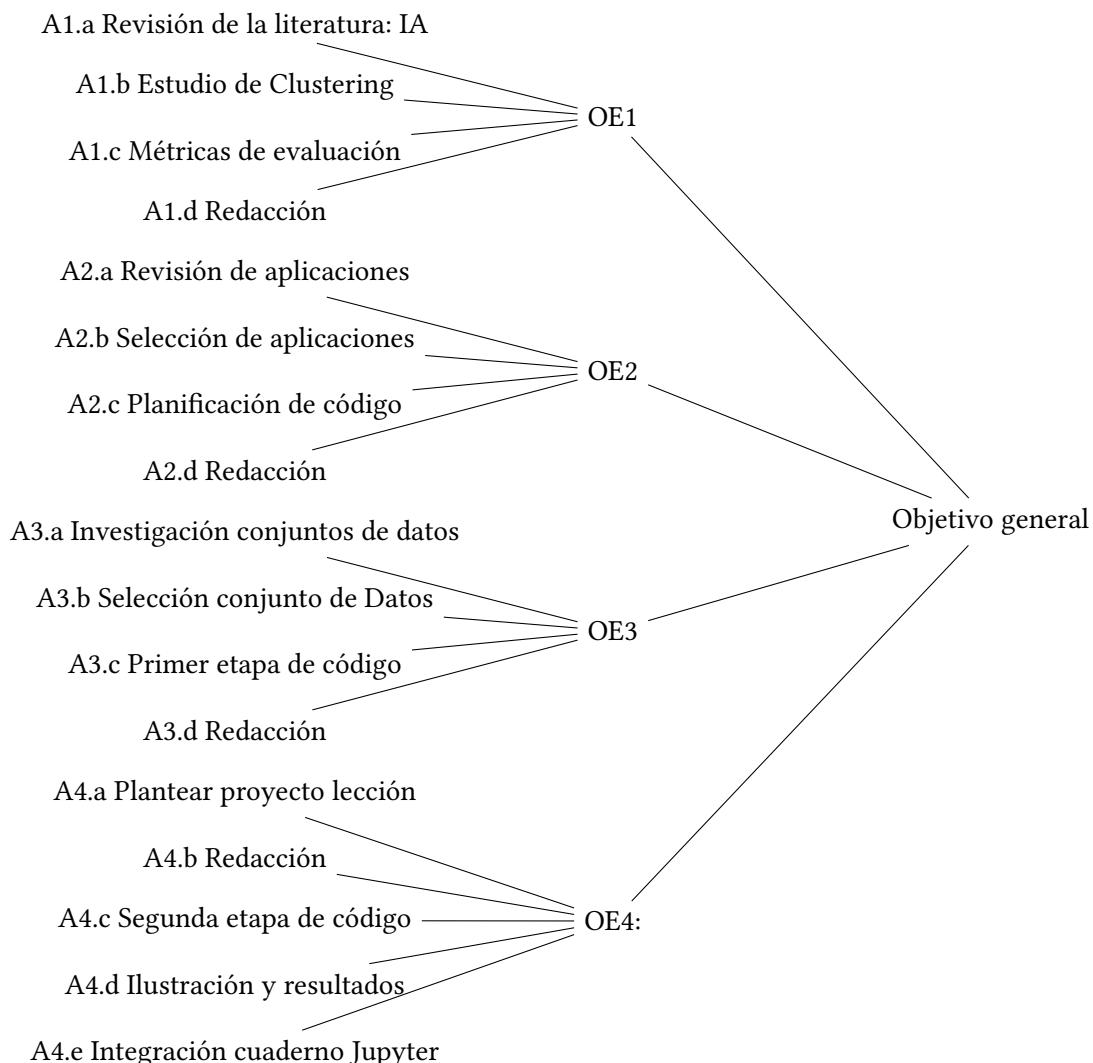


Figura 1.1: Construcción del objetivo general.

### 1.3. Alcance

Este trabajo se limita a exponer diferentes técnicas y algoritmos de métodos de agrupamiento seleccionados, no supervisados, su eficiencia y sus principales medidas de evaluación, se seleccionarán tres aplicaciones documentadas en relación específica con la ingeniería Eléctrica para su adaptación práctica, simulación y su estudio por parte de los estudiantes de la Escuela de Ingeniería Eléctrica de la Universidad de Costa Rica, siendo así un recurso didáctico importante para un posible curso o una evaluación en materia de Inteligencia Artificial: métodos de agrupamiento. Este trabajo no busca expandir ampliamente el estudio de la Inteligencia Artificial, ya que representaría objetivos mucho más

amplios, sino más bien brindar una breve introducción a sus conceptos, sus algoritmos y su historia, dando énfasis a los métodos de agrupamiento, sus aplicaciones y su simulación.

## **1.4. Cronograma**

Desde el siguiente link se puede acceder a un archivo amplio de cronograma que detalla las actividades, tareas a realizar y sus fechas tentativas:

<https://drive.google.com/file/d/1kXw9nj6ToDHrIw6POuGC8Lekw9qQKOWg/view?usp=sharing>

## Capítulo 2

# MARCO TEÓRICO

**E**N EL PRESENTE CAPÍTULO se expone una breve reseña teórica sobre la inteligencia artificial, sus conceptos y su historia, las teorías de aprendizaje y su evolución. También se explican algunos métodos de agrupamiento, sus fórmulas, tipos y definiciones, además de sus métricas de evaluación.

### 2.1. Inteligencia Artificial (IA) y Teorías de aprendizaje

#### 2.1.1. Definiciones

Existen muchas definiciones de la Inteligencia Artificial en la literatura, a lo largo de la historia se encontraron muchas variables en su definición debido a su carácter pragmático en la modernidad o sus bases teóricas. Palma y Marín(2008, [9]) afirman que una definición aceptada de forma común enlaza la disciplina de la Inteligencia Artificial(IA) con el estudio, análisis y el diseño de sistemas autónomos de carácter artificial capaces de mostrar un comportamiento considerado inteligente. Por su parte Arrúa y Meza (2003, [1]) relacionan la IA con la ciencia y computación afirmando que la IA es la parte de la Ciencia que se encarga del diseño de sistemas de computación inteligentes, es decir, sistemas que presentan las características que relacionamos con la inteligencia en el comportamiento humano, entiéndase elementos como la comprensión del lenguaje, el aprendizaje, el razonamiento y la resolución de problemas. Desde su aplicación práctica, el propósito general de la IA es desarrollar Modelos conceptuales, procedimientos de reescritura formal de esos modelos y estrategias de programación y construcción de máquinas físicas para reproducir de la forma más eficiente y completa posible las tareas de carácter cognitivo y científico-técnico, e igualmente genuinas y naturales que otros sistemas o entes biológicos que consideramos inteligentes. Esta idea trae consigo la delimitación pragmática de la IA, pues su desarrollo está necesariamente limitado por los avances en las técnicas de modelado, formalización, programación, por la evolución de los materiales y las arquitecturas de los computadores ( dispositivos electromecánicos) o microelectrónicos (hardware) en los que se instala el cálculo autónomo. ( [9])

Se asume que, para que un agente actúe inteligentemente, debe poder percibir su entorno, elegir y planificar sus objetivos, actuar hacia la consecución de estos objetivos aplicando algún principio de racionalidad e interactuar con otros agentes inteligentes, sean estos artificiales o humanos.

### 2.1.2. Breve reseña Histórica

Muchas autores afirman que la historia de la Inteligencia Artificial es muy antigua en relación a sus ideas básicas, remontándose a la historia primitiva de la humanidad. Desde el comienzo de la evolución del humano como "ser racional", con su constante intento de extender las facultades físicas, pero especialmente las facultades intelectuales. Munera(1990) expone que la historia de la IA tiene en la antigua Grecia uno de sus momentos interesantes, con la aparición de algunos geniales matemáticos y de los primeros automatismos o inventos. Sin embargo, la formalización de la IA como ciencia puede marcarse históricamente en tres puntos claves iniciales en la época conocida como era pionera:

La creación del modelo neuronal del cerebro realizado por el doctor McCulloch de la Universidad de Illinois y el matemático Pitts, se unieron para hacer un modelo del cerebro (1942-1943). Hicieron un estudio desde el punto de vista booleano de las neuronas y construyeron un primer modelo formal del procesamiento de información a nivel del cerebro.

El segundo es en 1950 cuando Alan Turing publicó su famoso artículo "Computing Machinery and Intelligence", donde describe un método para que los humanos podamos testear programas de IA.

El otro momento clave es la reunión o conferencia de Dartmouth College, en el verano de 1956, donde se reunieron un grupo de investigadores para dar origen a lo que oficialmente ya se conoce con el nombre de Inteligencia Artificial. Muchos de ellos venían del grupo de Norbert Wiener, del MIT que inició el estudio de la cibernetica. [8]

A partir de este momento se inicia la etapa de desarrollo e investigación en la materia. De este grupo final, se crearon subgrupos de investigación en dos diferentes corrientes que posteriormente se conocerían con los nombres de conexionismo e ingeniería del conocimiento. Otros puntos importantes son: la creación del MYCIN, primer sistema experto formal; el desarrollo de PROLOG, un lenguaje de programación lógico e interpretado; en 1958 se desarrolla el lenguaje LISP, lenguaje con el que se desarrollan la mayoría de sistemas expertos; el desarrollo del programa General Problem Solver (GPS), entre muchos otros siguientes que relacionan la computación, informática, matemática y la IA. Finalmente, en 1997 Gari Kaspárov, campeón mundial de ajedrez, pierde ante la computadora autónoma Deep-Blue.

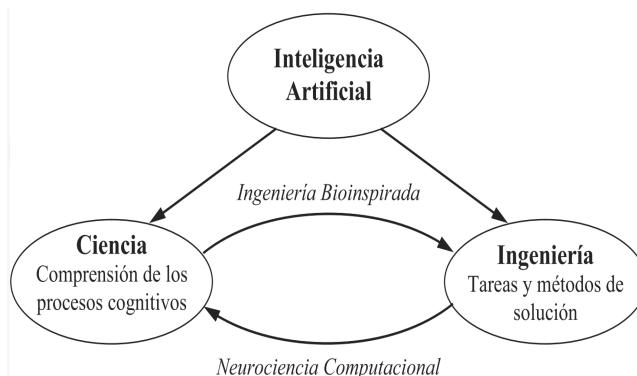


Figura 2.1: Distinción entre IA como ciencia e IA como ingeniería. Adaptado de: [9]

Gracias a estos avances en la materia, se fueron desarrollando algoritmos que permitiesen ordenar la información de las bases de datos de los mecanismos y sistemas a implementar o de su propia información recabada, resultado en los métodos de agrupamiento o algoritmos de agrupamientos (*clustering*) como k-means (algoritmo de Lloyd como base), Fuzzy-C, k-medoid, ordenamiento jerárquico entre otros que se analizarán más adelante. Nuevamente es evidente la estrecha relación mencionada anteriormente entre el desarrollo de la IA y sus factores: matemática, computación ingeniería etc. Con el crecimiento del mundo digital, cada vez es más necesario que los sistemas autónomos procesen grandes cantidades de datos por lo que los métodos de agrupamiento se convirtieron en una extremaidad operativa fundamental para estos sistemas, cuando antes se apreciaban como una herramienta opcional.

### 2.1.3. Actualidad

En la etapa moderna de la IA encontramos sistemas muy complejos y muy amplios, donde se puede mencionar la programación en lenguajes abstractos (interpretes), la recopilación de datos a gran escala (big data), el análisis automático, la robótica autónoma, el análisis de emociones, entre muchos otros, que han sufrido muchas diferencias en su forma conceptual, debido a los cambios contemporáneos del humano, la sociedad y claro, la ciencia. Desde el punto de vista teórico e incluso filosófico se ha cuestionado mucho la forma en que se planteó y formalizó esta ciencia, uno de los argumentos es la propia naturaleza general de la palabra inteligencia que manifiesta una mensura global de la calidad de los procesos cognitivos de un ser vivo y de su capacidad de adaptación al medio en que se desarrolla y coexiste. [9]

Este término inteligencia se categoriza como un concepto precientífico que en la actualidad apenas es discutible en utilidad, obsoleto, ya que son muchas las nuevas ideas que buscan interpretar mejor este concepto, como las inteligencias múltiples desde la psicología o la inteligencia colectiva desde la sociología. Con la formalización del conocimiento, la IA se logró ordenar de diferentes formas, una de ellas es por la escuela de pensamiento: se encuentran la inteligencia artificial convencional y la inteligencia computacional. También se pueden encontrar otras ramas o corrientes de la IA moderna como los dominios formales, donde se pretende solucionar problemas mediante modelos de búsquedas en un espacio de estados, ya sean modelos de tipo algorítmico o heurístico, y dominios técnicos, donde se utiliza conocimiento científico-técnico muy elaborado, obtenido de un experto (sistemas expertos). [7]

El paso clave de la IA en la actualidad es sin duda, su auge en la industria o paso a la industrialización, esto conllevó un crecimiento exponencial en la investigación e implementación de las aplicaciones que utilizan IA y métodos de agrupamiento, hasta nuestros días, aproximadamente 65 años después y que irá creciendo aún más. Es posible que si se lee este documento en el futuro, se tendrán más contradicciones, críticas y aplicaciones emergentes, de la misma forma en que nosotros observamos a la literatura de la era clásica y pionera de la IA.

## 2.2. Métodos de agrupamiento (*clustering*)

El agrupamiento de información ha sido analizado con mucho empeño, como se mencionó anteriormente, desde hace ya aproximadamente 60 años. A partir del final de la década de los 60, se han estudiado

muchos métodos de agrupamiento en el marco de la Taxonomía Numérica y del Análisis de Datos. Como se comentó, luego las técnicas de agrupamiento se incluyeron firmemente dentro del campo de la IA, en el área del Aprendizaje no Supervisado. a posteriori, también se fue incorporando en la Minería de Datos, retomando muchos métodos previamente desarrollados, extendiéndolos y adaptándolos al mundo digital de datos masivos. [12]

Los métodos de agrupamiento son el proceso de agrupar datos de manera que todos los objetos del mismo grupo son similares y los objetos de otro grupo son diferentes, es decir, un procedimiento de agrupación de una serie de unidades de información de acuerdo con un criterio o regla establecida. Basándose en la definición clásica, el clustering se pueden definir como:

Una técnica de clasificación no supervisada de elementos (observaciones, datos o vectores de características) en grupos (clusters). [9]

El análisis y desarrollo de estos es uno de los campos de investigación más desafiantes referentes a IA. El análisis de clusters se denomina segmentación de datos y en la literatura, existen muchas categorías de algoritmos. A continuación se definen y explican algunos, así como su categorización.

### 2.2.1. Tipos y Definiciones

La clasificación o agrupamiento se puede dividir en dos grandes áreas: los métodos supervisado y los no supervisados. Para los Supervisados se posee cierta información preestablecida como la clase a la que pertenece cada elemento, y se busca encontrar cuáles parámetros intervienen en sus valores, por su parte, los no supervisados buscan generar ésta información que será procesada por el analista luego del proceso, es decir, no se tiene ninguna información acerca del orden de los elementos en agrupaciones o clases, por ende, la meta focal es determinar dicha organización en base a la proximidad entre los elementos. Se ampliarán tres de estos métodos no supervisados, como se mencionó en el enfoque del trabajo.

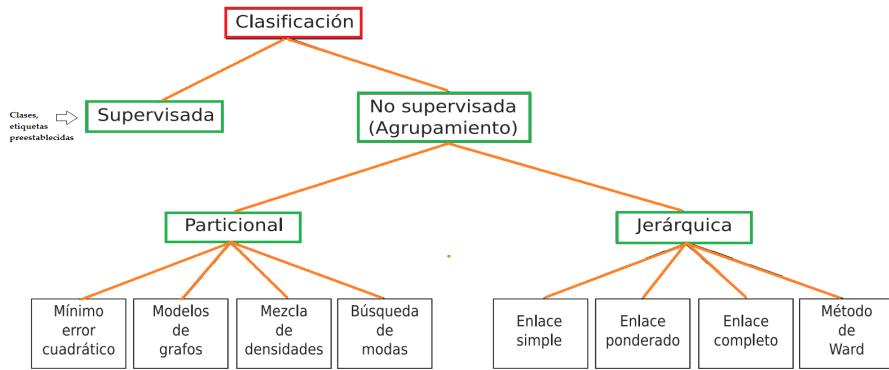


Figura 2.2: Árbol de tipos de métodos de clasificación. Adaptado de: [Jain y otros, 1999] y [Lance y Williams, 1967]

Generalmente, los métodos existentes se pueden dividir aproximadamente en tres categorías: métodos basados en la distancia, basados en la densidad y basados en la conectividad. Los métodos basados

en la distancia, (como K-means y el agrupamiento aglomerativo) buscan la relación entre datos basándose en métricas de distancia, como la Euclidiana. Los métodos basados en densidad agrupan los datos de acuerdo con una función de densidad predefinida, como por ejemplo, la agrupación espacial basada en densidad de aplicaciones y datos con ruido. Los métodos basados en conectividad agrupan datos en conjunto si están muy conectados, por lo que definen parámetros de enlace o relación. Un ejemplo es el agrupamiento espectral. El enfoque de partición es adecuado para datos de baja dimensión y bien separados. Sin embargo, en grandes cantidades pueden presentar métricas menos eficientes que los jerárquicos. En esencia, estos algoritmos actúan como vectores n-dimensionales para datos y aplican funciones de distancia o correlación para determinar la cantidad de similitud entre dos series. La distancia euclidiana, la distancia de Manhattan y el coeficiente de correlación de Pearson son funciones muy utilizadas.

### 2.2.2. Métodos y Algoritmos comunes

Las ideas antes mencionadas forman la base de varios métodos de los que pueden mencionarse el agrupamiento por conjuntos, el agrupamiento basado en la factorización matricial no negativa (NMF), métodos difusos(Fuzzy), métodos de distancias ponderadas, métodos jerárquicos, entre otros más complejos que podría ser una combinación de los métodos base para obtener mejor rendimiento.

#### K-means

El algoritmo K-means o k-medias es un algoritmo típico de agrupación en clústeres no supervisado basado en particiones, de modo que los datos se dividen en varios conjuntos predefinidos optimizando ciertos criterios definidos. La ventaja más importante es su sencillez y rapidez. Por lo tanto, se puede aplicar a grandes conjuntos de datos. Sin embargo, es posible que el algoritmo no produzca el mismo resultado en cada ejecución y/o que no pueda manejar correctamente datos atípicos. [13]

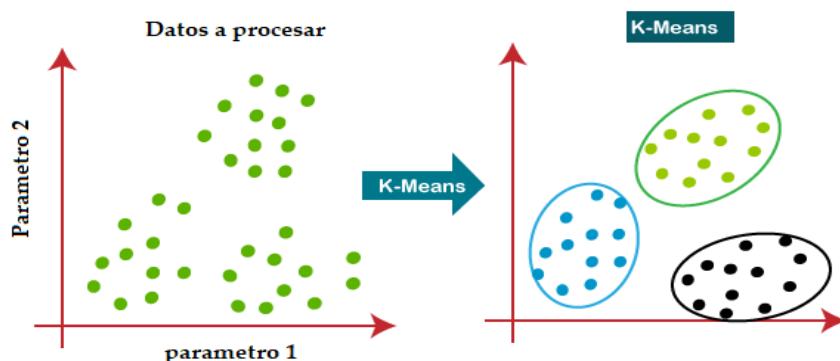


Figura 2.3: Ejemplo visual usando K-means. Adaptado de Internet

La idea principal detrás de este método es la minimización de una función objetivo, que normalmente se elige como la distancia total entre todos los patrones de sus respectivos clústeres. Su solución

se basa en un esquema iterativo, que comienza con centros de clústeres iniciales elegidos arbitrariamente. La distribución de objetos entre clústeres y la actualización de los centros son los dos pasos principales del algoritmo k-means. Para mejorar su eficiencia, se suele combinar con algún método de preprocessado de los datos, para obtener los centros iniciales de forma más precisa. [4] Los pasos del algoritmo, a grandes rasgos son:

1.  $k$  centroides iniciales son generados aleatoriamente.
2. Se generan  $k$  grupos, asociándole el punto con la media mas próxima.
3. Se itera desde el paso 1, recalculando el centroide de cada grupo generado hasta que converjan.

## DBSCAN

Uno de los algoritmos más interesantes y utilizados es la agrupación espacial de aplicaciones con ruido basado en densidad, DBSCAN (del inglés Density-based spatial clustering of applications with noise), propuesto en 1996, que por su simplicidad base permite observar y estudiar muchas características del agrupamiento no supervisado. DBSCAN utiliza la densidad basada en centroides estadísticos, lo que permite estimar la densidad de cada punto del arreglo de datos en concreto, contando el número de puntos que caen dentro de un área centrado en él de radio epsilon. [5]

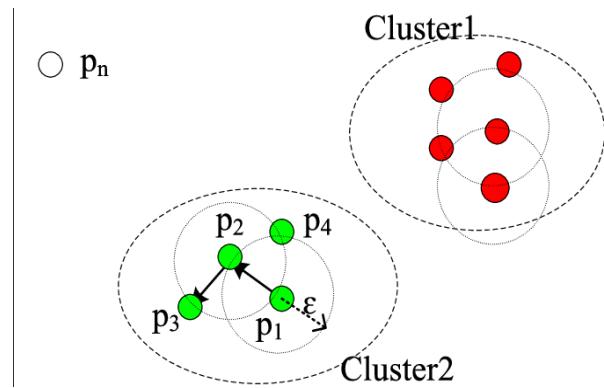


Figura 2.4: Ejemplo visual de la agrupación por densidad usando DBSCAN. Tomado de: [5]

Como se puede ir intuyendo, este algoritmo, a diferencia de media K explicado anteriormente, no necesita especificar el número de agrupaciones ya que tiene la capacidad de detectar automáticamente el número de clústeres en función de sus datos y parámetros de entrada, pero la característica más importante, es que DBSCAN puede encontrar agrupaciones por formas que k-medias no pueden. Por ejemplo, un grupo rodeado por un grupo diferente.

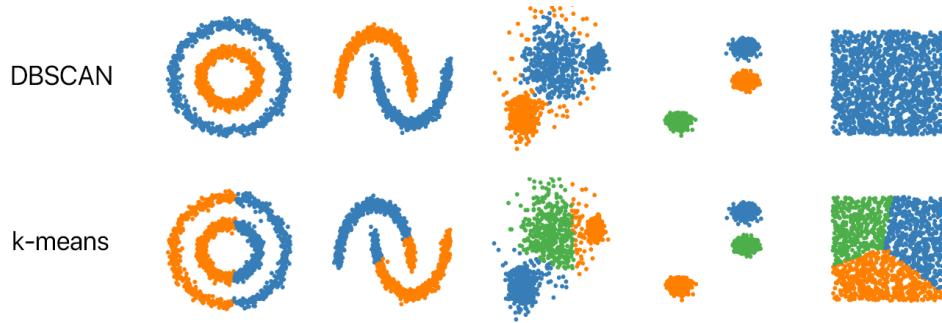


Figura 2.5: Ejemplo comparativo sobre la agrupación por forma de DBSCAN vs K-means. Tomado de Github

Para entender el paso a paso del algoritmo, es necesario definir los elementos que se estarán utilizando:

- Eps: Parámetro radio de contorno definido, máximo radio del vecindario de puntos.
- MinPTS: Parámetro cantidad de puntos mínimo en un contorno para definirse como un grupo valido. Umbral definido.
- Punto Núcleo: es el centro de un contorno de radio eps que tiene más del umbral de puntos mínimos *MinPts* definido.
- Punto Frontera: Punto que se encuentra en un entorno de radio eps que tiene como centro un punto núcleo. Puede ocurrir que un punto frontera pertenezca a varios contornos.
- Punto Ruido. Un punto que no es núcleo, ni frontera. Se supone que va a estar en regiones muy poco densas y que no va a formar parte de ningún grupo.

## HDBSCAN

El agrupamiento espacial jerárquico de aplicaciones con ruido basado en densidad (de Hierarchical Density-Based Spatial Clustering of Applications with Noise, por sus siglas en inglés) extiende el método DBSCAN y lo convierte en uno de tipo jerárquico, aprovechando sus excelentes cualidades, se basa en la estabilidad de los agrupamientos para generar nuevos grupos más eficientes. [10]

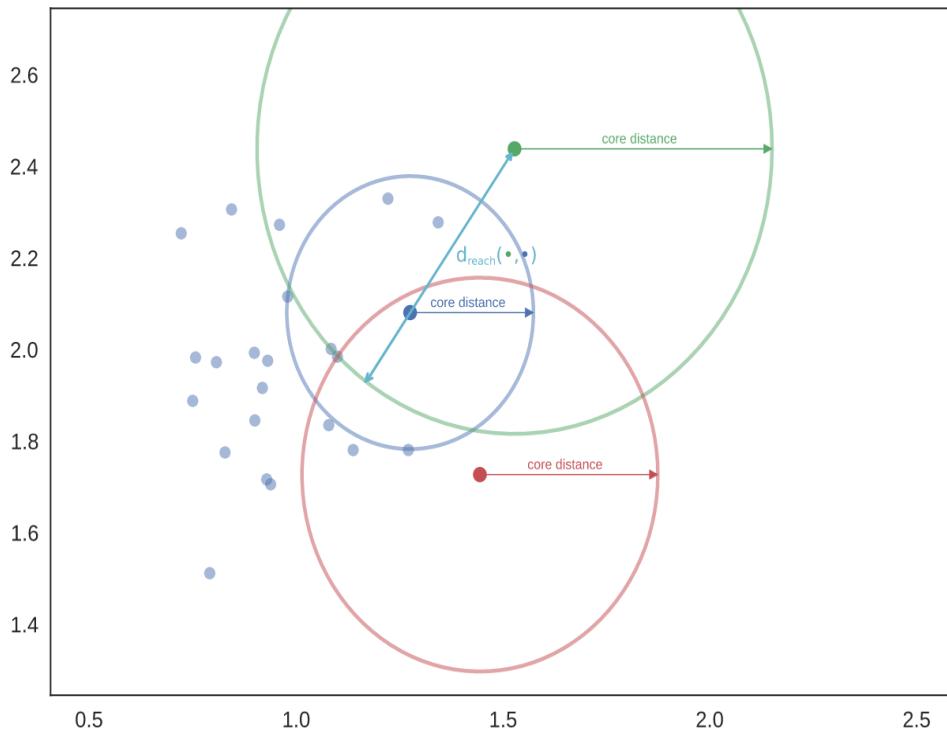


Figura 2.6: Gráfica intermedia de generación de regiones con el método HDBSCAN. Tomado de su documentación en Github

HDBSCAN realiza DBSCAN sobre valores de épsilon y luego integra esos resultados para encontrar un agrupamiento que brinde la mejor estabilidad sobre un cierto radio. Esto permite que HDBSCAN encuentre grupos de densidades variables (a diferencia de DBSCAN) y sea más robusto para la selección de parámetros. Por ello este método devuelve un mejor agrupamiento con poco o ningún ajuste de parámetros, además, aprovechando que los parámetros restantes a seleccionar manualmente son intuitivos. [6]

## 2.3. Sistemas de evaluación

Como se ha venido exponiendo, el Clustering es un proceso no supervisado ampliamente utilizado y que es especialmente sensible a los parámetros de entrada, por ello, es de suma importancia evaluar el resultado de los algoritmos de clustering y, consecuentemente, definir cuando nuestro resultado de agrupamiento es aceptable, u óptimo para cada aplicación. Por esta razón existen técnicas e índices para la validación de un agrupamiento realizado. [3]

La validación externa y la validación interna son las dos categorías más importantes para la validación de clustering. La principal diferencia es si se usa o no información externa para la validación, es decir, información que no es producto de la técnica de agrupación utilizada.

A diferencia de técnicas de validación externas, las de validación interna miden el clustering únicamente basadas en información de los datos. Evalúan que tan buena es la estructura del clustering sin necesidad de información ajena al propio algoritmo y su resultado.

### 2.3.1. Validación Interna

#### Cohesión y separación

**Cohesión:** Los elementos de cada agrupación debe estar lo más cerca posible de los otros elementos de ese mismo clúster.

**Separación:** Los grupos deben estar lo más separado posible entre ellos.

Hay distintas formas de calcular la distancia entre los elementos de los grupos, entre ellas podemos mencionar la distancia entre el elemento más cercano, distancia de los elementos más distanciados o la distancia entre los centroides.

#### Sum of Squared Within (SSW)

Medida interna que permite evaluar la cohesión (es decir, que tan juntos están los elementos de un grupo) de las agrupaciones generadas por el algoritmo. Se calcula como la suma de la distancia cuadrada de cada punto del clúster, respecto a su centroide, así para cada clúster. Es una simple suma acumulativa promedio.

$$SSW = \sum_{i=1}^k \sum_{x \in C_i} dist^2(m_i, x) \quad (2.1)$$

Donde  $k$  es la cantidad de clústeres,  $x$  un elemento o punto del clúster,  $C_i$  es el clúster en cuestión y  $m_i$  el centroide del clúster.

#### Sum of Squared Between (SSB)

Es una medida interna utilizada para evaluar la separación entre los grupos generados. Utiliza una sumatoria de la distancia cuadrada entre el centroide de cada clúster y la media del data set, además, agrega el peso del número de puntos o elementos de cada grupo generado.

$$SSW = \sum_{i=1}^k n_i \cdot dist^2(C_i - \bar{X}) \quad (2.2)$$

Donde  $k$  es la cantidad de clústeres,  $n$  es la cantidad de elementos del clúster  $i$ ,  $\bar{X}$  es la media del data set y  $C_i$  es el centroide del clúster.

Existen muchos otros índices de validación interna que se pueden encontrar en la literatura.

### 2.3.2. Validación Externa

Cuando si se tiene información externa tal como la clase de cada dato u otras etiquetas, es muy utilizado el análisis de comparación de Resultado(hipótesis) vs los resultados conocidos de antemano.

Esto es, que una vez realizado el agrupamiento mediante algún algoritmo para ese propósito como los mencionados no supervisados anteriormente, k-means o DBSCAN, el algoritmo puede sugerir un nuevo agrupamiento de los datos, diferente a los datos conocidos de partida. al tener estos datos, se realiza una comparación y se utilizan algunas métricas numéricas y lógico-tabulares. Se pueden mencionar las métricas precisión, recall y medida F.

#### Precisión y Recall

estas medidas intentan estimar si la predicción de un par de datos es correcta, como si estuviera en el mismo grupo, con respecto a las categorías verdaderas subyacentes en los datos. La precisión se calcula como la fracción de pares colocados correctamente en el mismo grupo, el recall es la fracción de pares reales y correctos que se identificaron, es decir, ambos son un porcentaje de acierto del resultado.

#### Medida F

La medida F es la media armónica de precisión y Recall. En particular la medida F maneja un parámetro  $\alpha$  que es justamente esa media harmónica.

## 2.4. Herramientas de software: Python y Jupyter

Python es una potente herramienta en materia de manejo de datos, mantiene librerías o módulos de gran capacidad para estadística, operaciones matemáticas, probabilidad y métodos de agrupamiento. Por ello se decide utilizarlo para la implementación de ejemplos y aplicaciones simuladas en esta materia de estudio.

Jupyter Notebook por su parte, es una aplicación web de código abierto que permite crear y compartir documentos que contienen código en vivo, ecuaciones, visualizaciones y texto narrativo, muy adecuado para procesos didácticos e investigativos.

### 2.4.1. NumPy

Es una biblioteca de Python que se utiliza para trabajar con arreglos de datos o vectores. También tiene funciones para trabajar en el dominio de álgebra lineal, transformada de Fourier y matrices.

### 2.4.2. Matplotlib

Es una biblioteca completa para crear visualizaciones estáticas, animadas e interactivas en Python. Permitirá la visualización de los datos agrupados, tendencias y evaluaciones.

### 2.4.3. Pandas

Es un paquete de Python de código abierto que se usa más ampliamente para la ciencia de datos o análisis de datos, y tareas de aprendizaje automático. Está construido sobre Numpy, que proporciona soporte para matrices multidimensionales.

### 2.4.4. scikit-learn

Biblioteca que contiene herramientas para el análisis predictivo de datos, accesible y es construido sobre NumPy, SciPy y matplotlib. Permite métodos de clasificación, de agrupamiento, regresiones, preprocesado entre otras importantes características.



## Capítulo 3

# DESARROLLO DE APLICACIONES

En la literatura se encuentran múltiples aplicaciones relacionadas a la ingeniería eléctrica y clustering [2], desde análisis de consumo eléctrico [11], reconocimiento de patrones [4], hasta procesamiento de señales de audio. En este capítulo, se explican las aplicaciones implementadas, para el uso de los métodos de agrupamiento, sus principales conceptos de ingeniería eléctrica y su desarrollo didáctico.

### 3.1. Aplicaciones implementadas

Las tres aplicaciones son muy similares, se varía el método de agrupamiento utilizado, la forma de pre-procesamiento de los datos y los datos utilizados. Cada una se implementa de forma que se obtengan las mejores métricas posibles, con un enfoque completamente didáctico.

En el código, cada línea está comentada y explicada desde la teoría vista en los capítulos anteriores. El estudiante podrá explotar estas aplicaciones de forma libre.

#### 3.1.1. Clustering en Generación de Energía por Región

En la ingeniería eléctrica, específicamente el énfasis de sistemas de potencia, la generación de energía es una de las ramas principales de aplicación. En muchos artículos académicos e investigaciones se analiza la generación y consumo de energía en diferentes regiones y con múltiples variables con el fin de optimizar estepreciado recurso.

En este caso se tiene una base de datos con los resultados de diferentes mediciones de generación de energía por tipo y por región, a su vez que la cantidad de energía a producir estimada en unidades de Medición fasorial.

Las unidades de Medición fasorial como su nombre lo indica realizan una medida de los fasores de corriente y tensión de la red eléctrica, garantizando sincronización en las medidas y una alta tasa de muestreo que permite una visualización en tiempo real del sistema, siendo un factor clave para determinar flujos de potencia y en general el estado del sistema. El conjunto de dos o más PMU's instaladas en el sistema y los software de análisis de datos provenientes de estas es llamado un WAMS de PMU's. Todos los valores tomados, que se muestran en la base de datos están en MU.

Se utilizará el método HDBSCAN para ordenar los datos en clusters relacionados, con el fin de analizar características geográficas y técnicas de cada región. Este algoritmo realiza DBSCAN sobre diferentes valores de épsilon e integra el resultado para encontrar un agrupamiento que brinde la mejor estabilidad sobre épsilon. Esto permite que HDBSCAN encuentre grupos de diferentes densidades (a diferencia de DBSCAN) y sea más robusto para la selección de parámetros.

### **3.1.2. Clustering con K-means - Catado de café**

Costa Rica fue un país que se caracterizó por su época cafetalera, el auge de este producto conocido como el oro líquido mantuvo a flote la economía del país. Hoy, esta industria es enorme y mantiene estándares de calidad mundial, muchos profesionales de procesamiento, cultivo generación y particularmente, de catación.

Catación es la descripción o medición de las características organolépticas y físicas del café. Este método permite evaluar atributos, cualidades y defectos del producto, por lo que se transforma en una muy importante herramienta de control de calidad. En esta implementación, se tiene una base de datos con los resultados de diferentes catados de múltiples muestras de café.

Se desea realizar un agrupamiento de éstas muestras según sus métricas estadísticas. Entre ellas se encuentra la calificación promedio del catador certificado y niveles de sabor: Vainilla, floral, cereral, cocoa, alcohol, fermentado, tostado, oscuro, amargo, entre otros.

En este caso se hace uso del algoritmo KMeans. El método de KMeans presenta gran efectividad y velocidad para datos no tan amplios, sin embargo, su principal debilidad es la selección de parámetros de entrada, entiéndase, número de clusters a realizar. Como este valor no se conoce, se debe usar algún método para optimizar su implementación, en este caso se utilizará el método del codo de Jambú para encontrar un número de clusters óptimo y se analizarán los resultados.

### **3.1.3. Comparación de Métodos de Agrupamiento - EMG y Redes neuronales**

En muchas aplicaciones de Clustering y artículos académicos, es común apreciar comparaciones entre métodos o algoritmos de agrupamiento, esto se da ya que los resultados son muy variados y dependen del dataset, los parámetros utilizados y las métricas analizadas. Encontrar el mejor fit es una tarea interesante y que ayuda a profundizar sobre los diferentes métodos de agrupamiento.

Otra característica a tomar en cuenta de los datos a analizar en diferentes aplicaciones, es que los datasets vienen en formatos específicos y muchas veces no visibles de forma fácil sin un procesamiento adecuado; un conjunto de datos „ciegas”.

En este caso se analizará un dataset en formato de salida (output) codificado de forma que no es visible en cualquier aplicación, para ello se procesan los datos con los paquetes de Python para este fin.

Los datos representan la salida de una aplicación de Red Neuronal, capturas de señales de los músculos responsables del movimiento de los dedos de la mano en una prótesis EMG. La electromiografía (EMG) consta en colocar electrodos superficiales sobre la extremidad seleccionada por medio de electrodos conectados a un sensor mioeléctrico cuya salida va a una tarjeta de adquisición de datos.

El entrenamiento de los datos en el sistema se realiza por medio de un sistema RNA. Los RNA se definen como un sistema de mapeo no lineales. Constan de un número de procesadores simples ligados por conexiones con pesos.

Las unidades de procesamiento se denominan neuronas. Cada unidad recibe entradas de otros nodos y generan una salida simple escalar que depende de la información local disponible, y guardada internamente o que llega a través de las conexiones con pesos. Estos son datos los obtenidos y se desean analizar por medio de clustering para determinar las principales relaciones entre sí, definiendo así grupos de acción neuronal.

En esta actividad, se utilizan, aplican y comparan diferentes algoritmos de agrupamiento de la librería sklearn.

### 3.1.4. DSP. Analizando sonidos de manatíes - Actividad práctica

Una de las ramas mas importantes de la electrónica y sistemas de control es el procesamiento Digital de señales. Son muchísimas las posibilidades de aplicación al procesar señales de distintas fuentes: Sonido, transductores, señales eléctricas o electromagnéticas puras entre muchas otras.

Los manatíes no se pueden ver en ríos de aguas turbias, por lo que se identifican por sus sonidos. Al igual que las personas, los manatíes emiten sus sonidos con ligeras variaciones, aunque cada uno tiene una "voz"única. Por esto, contar cuántos manatíes hay en un río se puede realizar usando técnicas de clustering y DSP, procesando los sonidos que se captan, pensando en que cada clúster corresponde a un manatí en particular.

En este caso se cuenta con una base de datos de muestras de sonidos de manatíes captados en un río, que se procesan para obtener un número aproximado de individuos en la zona, entre otras características conocidas, tras su análisis. Se espera que el estudiante pueda implementar un procedimiento completo basándose en la información y ejemplos presentados, su propia investigación y su análisis.



## Capítulo 4

# RESULTADOS OBTENIDOS

### 4.1. Introducción

**E**N este capítulo se exponen los principales resultados obtenidos, tanto de las simulaciones en gráficos, como de las actividades sugeridas, además se exponen los diferentes enlaces a los recursos y archivos generados. La mayoría de procesos didácticos son ejemplificados y explicados en los cuadernos de Jupyter NB creados. En ellos se puede obtener información detallada de como se obtienen los resultados acá mostrados y su respectivo análisis.

### 4.2. Conjuntos de datos generados

Se adaptaron 3 conjuntos de datos de aplicaciones de ingeniería eléctrica: Generación de Energía, Procesamiento digital de Señales (DSP) y datos de Redes neuronales (EMG). Además se incluye un dataset sobre resultados de catado de café, por sabores, para implementar un análisis un poco diferente a lo acostumbrado.

## Conjunto de datos para EMG - Redes neuronales

### Se cargan los datos de la aplicación EMG

```
# Se utiliza la función Load de numpy para cargar Los datos
data_EMG = np.load('../datasets/EMG Red neuronal Protesis.npy')

# Visualizamos de forma rápida Los datos leido
plt.scatter(data_EMG.T[0], data_EMG.T[1], c='b', **plot_kwds)
frame = plt.gca()
frame.axes.set_title("EMG Impulsos Eléctricos Neuronales")
frame.set_xlabel("Pesos sensado Mioeléctrico")
frame.set_ylabel("Potencial Eléctrico (mV)")
frame.axes.get_xaxis().set_visible(True)
frame.axes.get_yaxis().set_visible(True)
```

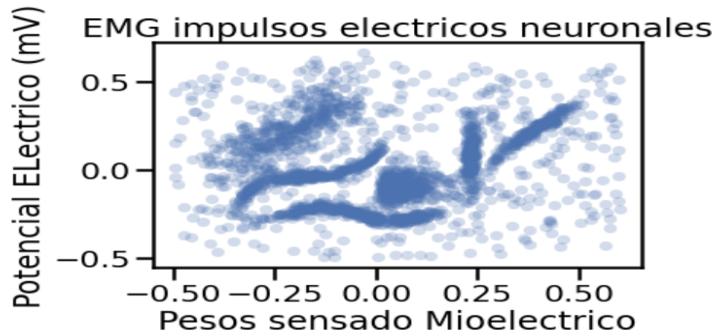


Figura 4.1: Conjunto de datos EMG, mostrado en el cuaderno Jupyter

## Conjunto de datos para Procesamiento de señales(DSP) - Manatíes

### Datos a procesar

Se lee el archivo csv con el dataset

```
manaties = pd.read_csv('../datasets/dataset_manaties.csv', engine='python')
manaties.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 4 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Frecuencia Fundamental (Khz)    150 non-null   float64 
 1   Longitud de sonido (s)         150 non-null   float64 
 2   Intensidad promedio (W/m2)     150 non-null   float64 
 3   Frecuencia promedio por hora(veces) 150 non-null   float64 
dtypes: float64(4)
memory usage: 4.8 KB
```

Figura 4.2: Conjunto de datos DSP, mostrado en el cuaderno Jupyter

## Conjunto de datos para Análisis de Producción de Energía

### Importando Dataset y visualizando sus características

```
data = pd.read_csv('../datasets/generación energía por tipo region.csv', engine='python')
data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4945 entries, 0 to 4944
Data columns (total 9 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   indice          4945 non-null    int64  
 1   fecha            4945 non-null    object  
 2   Region           4945 non-null    object  
 3   Generacion Termica actual (MU) 4945 non-null    float64 
 4   Generacion Termica Estimada (MU) 4945 non-null    float64 
 5   Generacion Eolica Actual (MU) 4945 non-null    float64 
 6   Generacion Eolica Estimada (MU) 4945 non-null    float64 
 7   Hidro Generacion Actual (MU) 4945 non-null    float64 
 8   Hidro Generacion Estimada (MU) 4945 non-null    float64 
dtypes: float64(6), int64(1), object(2)
memory usage: 347.8+ KB
```

Figura 4.3: Conjunto de datos Generación de Energía, mostrado en el cuaderno Jupyter

## Conjunto de datos Industria Cafetalera - catado de café

Estos datasets se encuentran en el repositorio del proyecto. Se pueden accesar en el siguiente link: [https://github.com/brandonequivel/IA\\_Clustering\\_Apps\\_Simulation/tree/main/datasets](https://github.com/brandonequivel/IA_Clustering_Apps_Simulation/tree/main/datasets)

### 4.3. Gráficos y resultados obtenidos

Los resultados de cada implementación se muestran a continuación:

#### 4.3.1. Aplicando KMeans a la industria cafetera

Se inició realizando un pre-procesamiento de los datos mostrados en 4.2, se analizó el número de clúster óptimo con el método del codo de Jambú y se realizó el agrupamiento con KMeans:

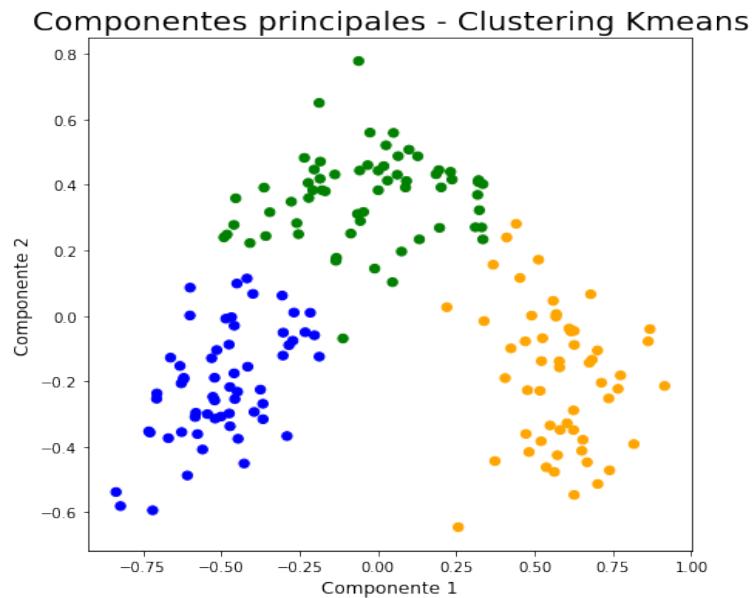


Figura 4.4: Gráfico codo de Jambú, actividad Kmeans

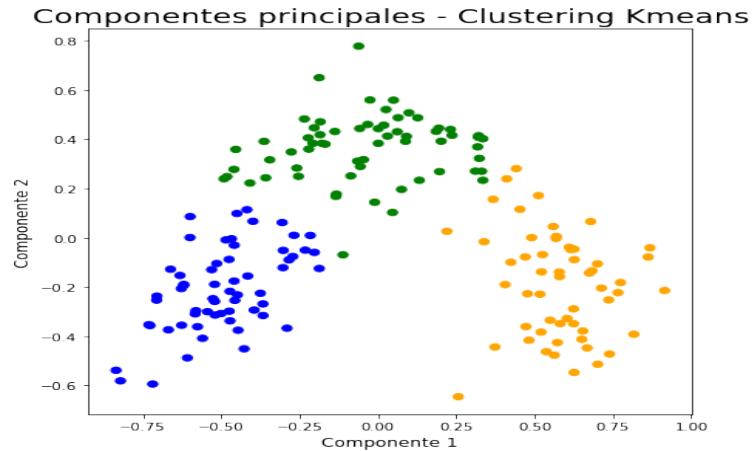


Figura 4.5: Gráfico obtenido, actividad Kmeans

Para efectos didácticos, los datos se mostraron en dos dimensiones ¿Cuales? se seleccionaron las variables que mejor caracterizan a todos los datos, para ello se usó el Análisis de Componentes Principales (PCA) para reducir el número de variables a analizar, en este caso a visualizar.

Se hace uso del paquete decomposition de sklearn y se crea un dataframe a partir de estos componentes para graficarlo.

3 Componentes principales - Clustering Kmeans

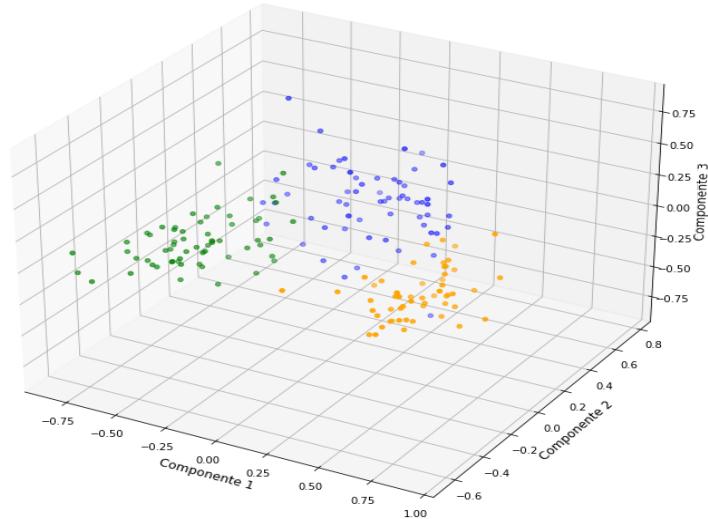


Figura 4.6: Gráfico obtenido, actividad Kmeans PCA

Los datos agrupados y etiquetados (bajo una cierta configuración base) se pueden encontrar en el link:

[https://github.com/brandonequivel/IA\\_Clustering\\_Apps\\_Simulation/blob/main/Results/cafe-kmeans.csv](https://github.com/brandonequivel/IA_Clustering_Apps_Simulation/blob/main/Results/cafe-kmeans.csv)

#### 4.3.2. Clustering en Generación de Energía

En esta aplicación se tienen datos con variables no numéricas, por lo que inicialmente se pre-procesaron para obtener un dataset adecuado. Se utilizó el método HDBSCAN para ordenar los datos en clusters relacionados, con el fin de analizar características geográficas y técnicas de cada región. Estos datos son muy útiles para diferentes análisis de producción y estimación. Otra buena forma de tratar los datos es normalizándolos.

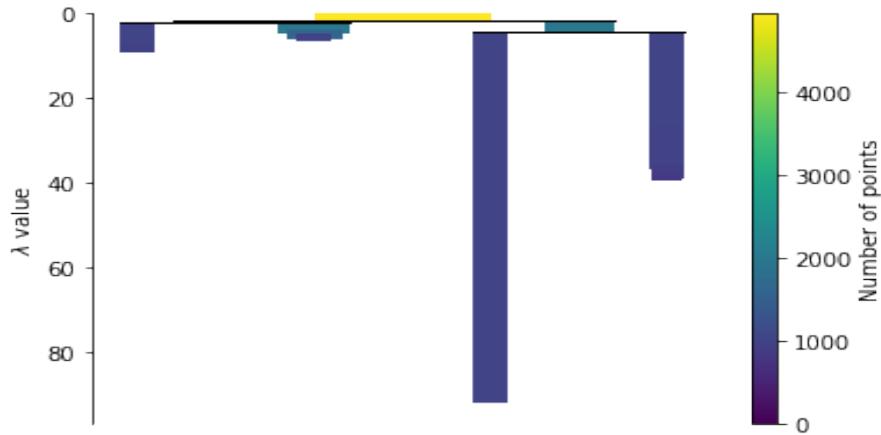


Figura 4.7: Gráfico obtenido, actividad clustering jerárquico HDBSCAN

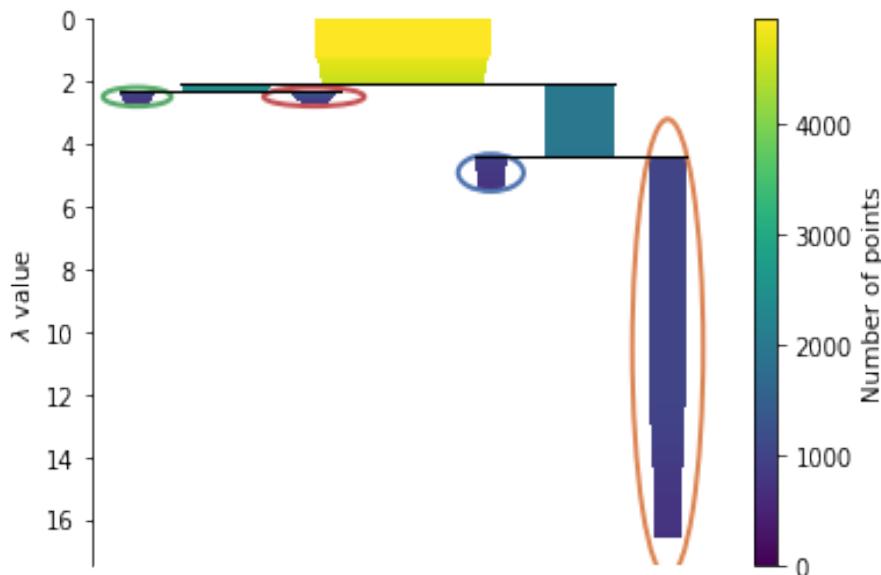


Figura 4.8: Gráfico remarcado, actividad HDBSCAN

El otro enfoque mencionado, es a partir de los datos estadísticos que mejor representan las relaciones del dataset, para ello se hizo uso del Análisis de Componentes Principales (PCA):

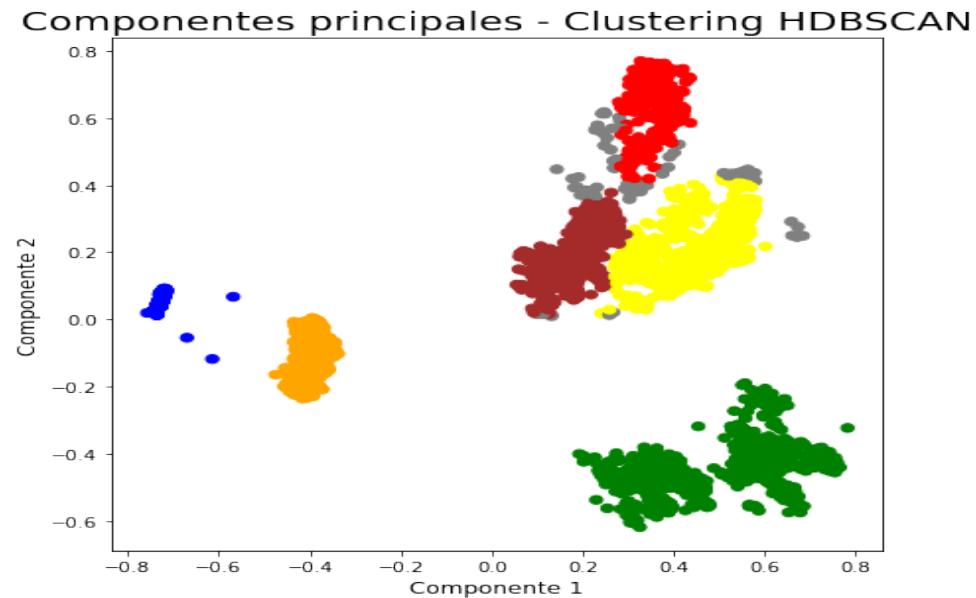


Figura 4.9: Gráfico obtenido, actividad hdbSCAN, post-procesado con PCA

Los datos agrupados y etiquetados se pueden encontrar en el link: <https://n9.cl/energy-hdbSCAN>

### 4.3.3. EMG - Comparando Algoritmos de Clustering

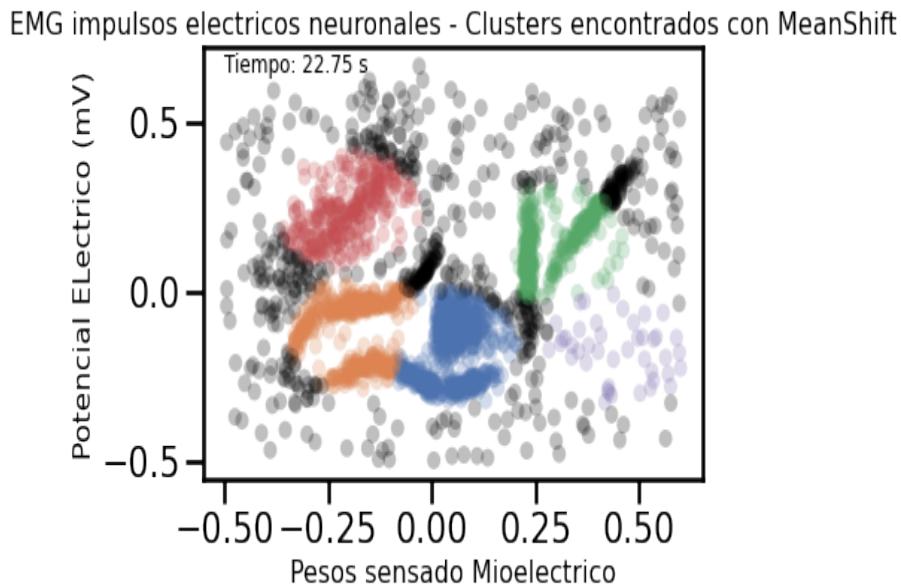


Figura 4.10: Gráfico obtenido del clustering con MS, actividad comparando algoritmos

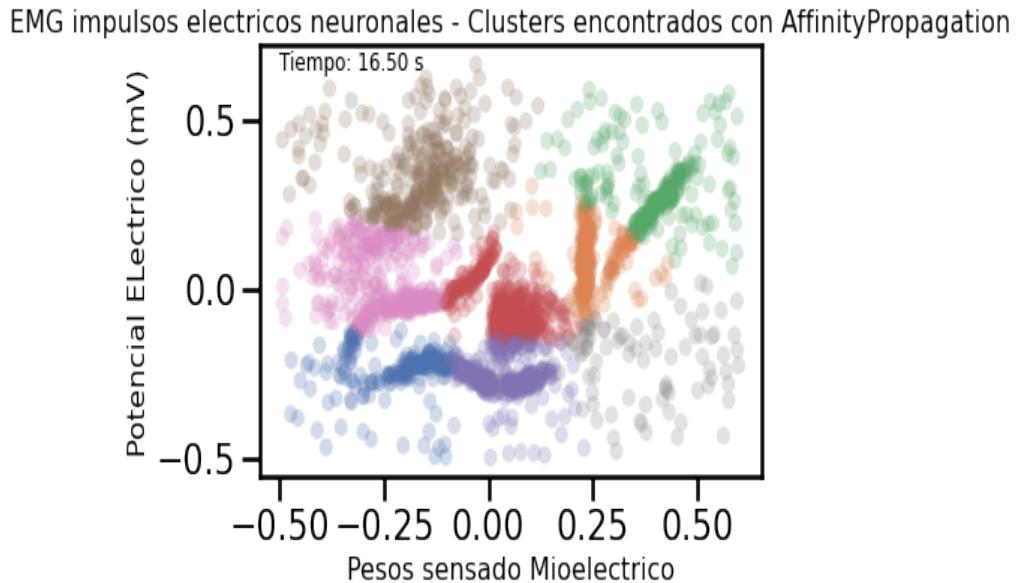


Figura 4.11: Gráfico obtenido del clustering con AP, actividad comparando algoritmos

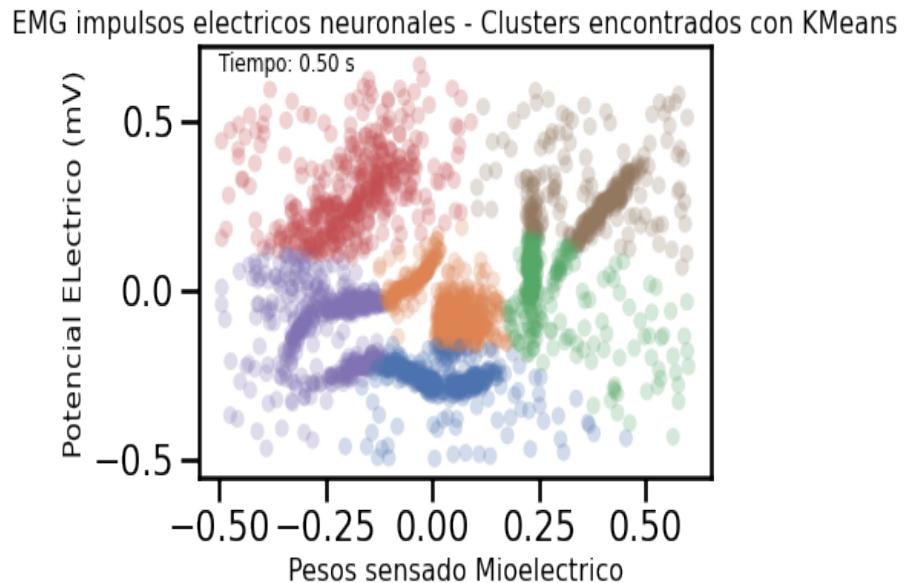


Figura 4.12: Gráfico obtenido del clustering con Kmeans, actividad comparando algoritmos

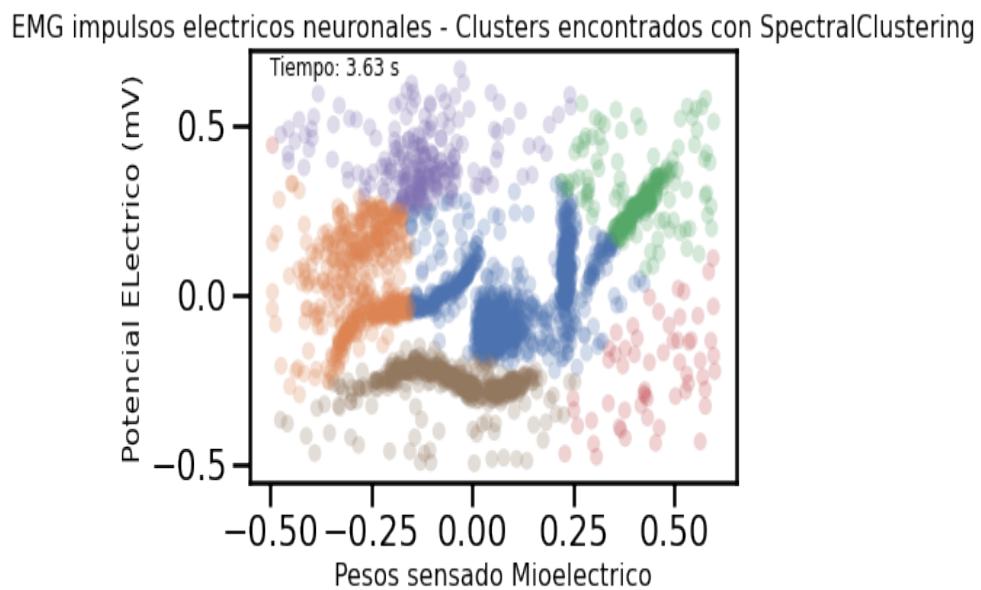


Figura 4.13: Gráfico obtenido del clustering con SC, actividad comparando algoritmos

Estos resultados gráficos pueden ser visualizados por medio del siguiente link:

[https://github.com/brandoneEsquivel/IA\\_Clustering\\_Apps\\_Simulation/tree/main/docs/img](https://github.com/brandoneEsquivel/IA_Clustering_Apps_Simulation/tree/main/docs/img)

#### 4.4. Actividades específicas por implementar

Se implementan actividades didácticas donde el estudiante deberá:

1. Resolver ejercicios o preguntas propuestas relacionadas a la teoría del tema expuesto.
2. Investigar distintos elementos propuestos de forma estratégica para llevar al estudiante a reconocer y fortalecer conceptos seleccionados.
3. Implementar ejercicios en código, similar a los ejemplos proporcionados, que le permita mejorar su comprensión de la materia.
4. Variar parámetros específicos de los ejemplos proporcionados para contestar preguntas y llegar a conclusiones personales sobre cada método de agrupamiento.

#### 4.5. Cuadernos de trabajo Jupyter Notebook

Se crearon 4 cuaderno de Jupyter Notebook con ejemplos y ejercicios implementados utilizando los conjuntos de datos adaptados.

En cada uno de estos cuadernos, el estudiante profundizará temas relacionados a los métodos de agrupación, sus parámetros y su implementación.

Estos cuatro cuadernos implementados son:

1. Aplicando KMeans a la industria cafetera.ipynb  
[https://github.com/brandonequivel/IA\\_Clustering\\_Apps\\_Simulation/blob/main/IA\\_Clustering\\_Apps\\_Simulation/Aplicando%20KMeans%20a%20la%20industria%20cafetera.ipynb](https://github.com/brandonequivel/IA_Clustering_Apps_Simulation/blob/main/IA_Clustering_Apps_Simulation/Aplicando%20KMeans%20a%20la%20industria%20cafetera.ipynb)
2. Clustering en Generacion de Energia.ipynb  
[https://github.com/brandonequivel/IA\\_Clustering\\_Apps\\_Simulation/blob/main/IA\\_Clustering\\_Apps\\_Simulation/Clustering%20en%20Generacion%20de%20Energia.ipynb](https://github.com/brandonequivel/IA_Clustering_Apps_Simulation/blob/main/IA_Clustering_Apps_Simulation/Clustering%20en%20Generacion%20de%20Energia.ipynb)
3. DSP estudio de manaties.ipynb  
[https://github.com/brandonequivel/IA\\_Clustering\\_Apps\\_Simulation/blob/main/IA\\_Clustering\\_Apps\\_Simulation/DSP%20estudio%20de%20manaties.ipynb](https://github.com/brandonequivel/IA_Clustering_Apps_Simulation/blob/main/IA_Clustering_Apps_Simulation/DSP%20estudio%20de%20manaties.ipynb)
4. EMG - Comparando Algoritmos de Clustering.ipynb  
[https://github.com/brandonequivel/IA\\_Clustering\\_Apps\\_Simulation/blob/main/IA\\_Clustering\\_Apps\\_Simulation/EMG%20-%20Comparando%20Algoritmos%20de%20Clustering.ipynb](https://github.com/brandonequivel/IA_Clustering_Apps_Simulation/blob/main/IA_Clustering_Apps_Simulation/EMG%20-%20Comparando%20Algoritmos%20de%20Clustering.ipynb)

## Capítulo 5

# CONCLUSIONES Y RECOMENDACIONES

### 5.1. Conclusiones

- La IA y sus ramas es un área de estudio en crecimiento exponencial y con mucha demanda laboral actualmente. El conocimiento básico de los algoritmos y métodos más utilizados en el área es fundamental en los estudiantes de Ingeniería eléctrica para que su visión profesional se amplíe y las opciones de especialización sean mayores. Al poder interactuar con estos principios y conceptos, los estudiantes obtienen un primer vistazo estructurado del mundo de la IA, lo que podría encender su deseo por ésta rama y profundizar la materia.
- Es posible concluir que cada método de agrupamiento mantiene carencias y fortalezas. En la teoría, se puede concluir que para obtener un método ideal se deben cumplir ciertos requisitos: Debe tener gran escalabilidad, la posibilidad de manejar muchos y distintos tipos de data sets, completa independencia del orden inicial y del orden de representación, poder identificar clusters con formas arbitrarias, mantener un número mínimo de parámetros configurables manualmente, capacidad de manejar múltiples espacios de dimensiones, capacidad para que el usuario agregue restricciones de forma simple y rápida y por supuesto, la máxima tolerancia al ruido y datos atípicos. Cada usuario especializado tendrá las características particulares que desea para su modelo ideal.
- Sobre las métricas de evaluación de algoritmos de agrupación, es posible concluir que la calidad o eficiencia de los agrupamientos generados es comúnmente medida sobre su exactitud, se puede mencionar por ejemplo, la proporción de objetos agrupados correctamente según sus características, etiquetas o clases. Sin embargo, en los métodos de agrupamiento no supervisados, en la práctica, la estructura de las agrupaciones suele ser desconocida y no se tiene un grupo de datos con información entrenada o una verdad base de la cual partir para las comparaciones.
- Los algoritmos de agrupamiento son generalmente evaluados manualmente bajo la supervisión de profesionales expertos, por lo que se puede concluir que esto conlleva algunos problemas, ya que éste recurso humano experto no estará disponible siempre, es muy costosos y no siempre se tendrá la calidad de los resultados deseados, por ello se destaca la importancia de la automatización de los procesos de evaluación de los algoritmos y agrupaciones generados.

- En este trabajo se mostró que la automatización de una evaluación básica puede realizarse de forma relativamente sencilla, tomando en cuenta variables de tiempo, efectividad, manejo de ruido y claridad. Se comprueba que cada método mantiene un desempeño diferente según el tipo de dataset y su escala de tamaño.
- Se concluye que el tamaño de los datos utilizados es relativamente pequeño, después de la investigación de diferentes fuentes de dataset donde se pudo apreciar escalas que rondan el Big Data. Optimizar, estudiar y comprender mejor los métodos de agrupamiento permitirá extender su dominio y aplicabilidad a estos horizontes de procesamiento.

## 5.2. Recomendaciones

Se recomienda realizar diagramas, dibujos y cualquier infograma que ayude a comprender la organización y los conceptos de los métodos de agrupamiento, ya que su materia puede ser bastante abstracta, a su vez, respecto a los algoritmos implementados en las librerías utilizadas, se recomienda estudiar cada método y su documentación a detalle, para comprender el uso y efecto de cada parámetro de entrada.

## 5.3. Anexos

### 5.3.1. Código

El código se puede descargar del siguiente repositorio:

[https://github.com/brandoneEsquivel/IA\\_Clustering\\_Apps\\_Simulation](https://github.com/brandoneEsquivel/IA_Clustering_Apps_Simulation)

### 5.3.2. Conjuntos de datos utilizados

Es posible acceder estos archivos por medio del siguiente link:

[https://github.com/brandoneEsquivel/IA\\_Clustering\\_Apps\\_Simulation/tree/main/datasets](https://github.com/brandoneEsquivel/IA_Clustering_Apps_Simulation/tree/main/datasets)

### 5.3.3. Cuadernos Jupyter Notebook creados

Se muestran los cuadernos de Jupyter Notebook en formato PDF. Estos pueden ser accedidos desde el repositorio.

## Clustering con K-means - Catado de café



Se tiene un conjunto de datos con los resultados de diferentes catados de múltiples muestras de café.

Se desea realizar un agrupamiento de éstas muestras según sus métricas estadísticas. Entre ellas se encuentra la calificación promedio del catador certificado y niveles de sabor: Vainilla, floral, cereral, cocoa, alcohol, fermentado, tostado, oscuro, amargo, entre otros.

En este caso se hace uso del algoritmo KMeans, que se explica más adelante.

In [1]:

```
# Imports
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
import pandas as pd

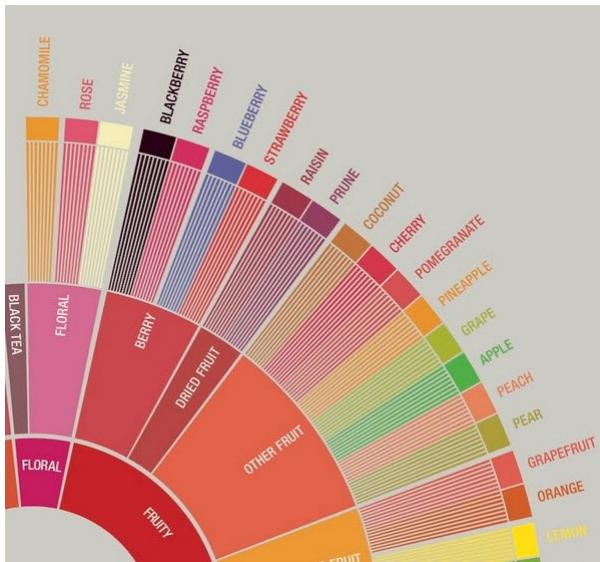
%matplotlib inline
```

### Importando Dataset y visualizando sus Características

In [2]:

```
cafes = pd.read_csv('../datasets/catacafe.csv', engine='python')
cafes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 178 entries, 0 to 177
Data columns (total 14 columns):
 #   Column           Non-Null Count  Dtype  
 ---  -- 
 0    Cafe             178 non-null    int64  
 1    Dulce            178 non-null    float64 
 2    Floral           178 non-null    float64 
 3    Especias         178 non-null    float64 
 4    Tostado          178 non-null    float64 
 5    Frutal           178 non-null    int64  
 6    Fermentado       178 non-null    float64 
 7    Vegetal          178 non-null    float64 
 8    Otro              178 non-null    float64 
 9    Cocoa             178 non-null    float64 
 10   Cereal            178 non-null    float64 
 11   Vainilla          178 non-null    float64 
 12   Picante           178 non-null    float64 
 13   Calificacion promedio 178 non-null    int64  
dtypes: float64(11), int64(3)
memory usage: 19.6 KB
```



Mostrar las primeras filas para una previsualización del orden de los datos

In [3]:

```
cafes.head()
```

Out[3]:

	Cafe	Dulce	Floral	Especias	Tostado	Frutal	Fermentado	Vegetal	Otro	Cocoa	Cereal	Vainilla	Picante	Calificación promedio
0	1	14.23	1.71	2.43	15.6	127	2.80	3.06	0.28	2.29	5.64	1.04	3.92	1065
1	2	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	1.28	4.38	1.05	3.40	1050
2	3	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30	2.81	5.68	1.03	3.17	1185
3	4	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24	2.18	7.80	0.86	3.45	1480
4	5	13.24	2.59	2.87	21.0	118	2.80	2.69	0.39	1.82	4.32	1.04	2.93	735

Se puede ver que la columna "Cafe" es el índice numérico de la muestra, pero que ya se ha enumerado con el método de lectura, por ello, se procede a eliminarla.

In [4]:

```
cafes = cafes.drop(['Cafe'],axis=1)
```

```
cafes.head()
```

Out[4]:

	Dulce	Floral	Especias	Tostado	Frutal	Fermentado	Vegetal	Otro	Cocoa	Cereal	Vainilla	Picante	Calificación promedio
0	14.23	1.71	2.43	15.6	127	2.80	3.06	0.28	2.29	5.64	1.04	3.92	1065
1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	1.28	4.38	1.05	3.40	1050
2	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30	2.81	5.68	1.03	3.17	1185
3	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24	2.18	7.80	0.86	3.45	1480
4	13.24	2.59	2.87	21.0	118	2.80	2.69	0.39	1.82	4.32	1.04	2.93	735

Se obtienen las variables estadísticas de los datos por columna.

In [5]:

```
cafes.describe()
```

Out[5]:

	Dulce	Floral	Especias	Tostado	Frutal	Fermentado	Vegetal	Otro	Cocoa	Cereal	Vainilla	Picante	Calificación promedio
count	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000
mean	13.000618	2.336348	2.366517	19.494944	99.741573	2.295112	2.029270	0.361854	1.590899	5.058090	0.957449	2.611685	746.893258
std	0.811827	1.171746	0.274344	3.339564	14.282484	0.625851	0.998859	0.124453	0.572359	2.318286	0.228572	0.709990	314.907474
min	11.030000	0.740000	1.360000	10.600000	70.000000	0.980000	0.340000	0.130000	0.410000	1.280000	0.480000	1.270000	278.000000
25%	12.362500	1.602500	2.210000	17.200000	88.000000	1.742500	1.205000	0.270000	1.250000	3.220000	0.782500	1.937500	500.500000
50%	13.050000	1.865000	2.360000	19.500000	98.000000	2.355000	2.135000	0.340000	1.555000	4.690000	0.965000	2.780000	673.500000
75%	13.677500	3.082500	2.557500	21.500000	107.000000	2.980000	2.875000	0.437500	1.950000	6.200000	1.120000	3.170000	985.000000
max	14.830000	5.800000	3.230000	30.000000	162.000000	3.880000	5.080000	0.660000	3.580000	13.000000	1.710000	4.000000	1680.000000

Se normalizan los datos en un rango adecuado y se vuelven a obtener sus métricas

In [6]:

```
cafes_normalizado = (cafes - cafes.min())/(cafes.max()-cafes.min())
```

```
cafes_normalizado.describe()
```

Out[6]:

	Dulce	Floral	Especias	Tostado	Frutal	Fermentado	Vegetal	Otro	Cocoa	Cereal	Vainilla	Picante	Calificación promedio
count	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000
mean	0.518584	0.315484	0.538244	0.458502	0.323278	0.453487	0.356386	0.437460	0.372523	0.322363	0.388170	0.491460	0.334446
std	0.213639	0.220780	0.146708	0.172142	0.155244	0.215811	0.210730	0.234818	0.180555	0.197806	0.185831	0.260070	0.224613
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.350658	0.170455	0.454545	0.340206	0.195652	0.262931	0.182489	0.264151	0.264984	0.165529	0.245935	0.244505	0.158702
50%	0.531579	0.222332	0.534759	0.458763	0.304348	0.474138	0.378692	0.396226	0.361199	0.290956	0.394309	0.553114	0.282097
75%	0.696711	0.462945	0.640374	0.561856	0.402174	0.627586	0.534810	0.580189	0.485804	0.419795	0.520325	0.695971	0.504280
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

Se puede observar que ahora los datos tienen valores entre cero y uno, max y min.

## Ahora con este preprocesamiento se tienen datos ordenados, numéricos y normalizados, listos para un agrupamiento óptimo.

El método de KMeans presenta gran efectividad y velocidad para datos no tan amplios, sin embargo, su principal debilidad es la selección de parámetros de entrada, entiéndase número de clusters a realizar. Como este valor no se conoce, se debe usar algún método para optimizar su implementación, en este caso se utilizará el método del codo de Jambú para encontrar un número de clusters óptimo:

### Obtención del gráfico del Codo de Jambú

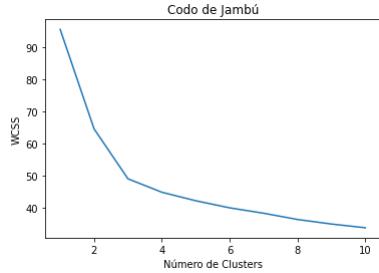
Se desean que los clusteres sean lo más separados entre sí y que sus elementos sean lo más cercanos entre sí, para ello se utiliza la medida WCSS: suma de los cuadrados de las distancias de cada punto de datos, en todos los grupos a sus respectivos centroides, es decir, es una medida de similitud. La idea es minimizar esta suma. Para ello se obtiene la inercia de cada clustering realizado con KMeans para un cierto número de grupos, desde 1 hasta uno deseado ( $n+1$  en este caso), estos valores obtenidos en cada iteración se almacena en WCSS, donde luego se imprimen en una gráfica para su análisis.

In [7]:

```
num_clusters = 10
wcss = []
for i in range(1,num_clusters+1):
    kmeans_model = KMeans(n_clusters=i,max_iter=300)
    kmeans_model.fit(cafes_normalizado)
    wcss.append(kmeans_model.inertia_)

## Ahora se grafican los resultados:

plt.plot(range(1,num_clusters+1),wcss)
plt.title("Codo de Jambú")
plt.xlabel("Número de Clusters")
plt.ylabel("WCSS")
plt.show()
```



Se observa que el número de clusteres óptimo es 3, para este método y este dataset. Ahora se procede a utilizar el método Kmeans con este parametro. Igual que anteriormente, se crea el modelo de clustering y luego se aplica con .fit

In [8]:

```
agrupamiento = KMeans(n_clusters=3, max_iter=300)
agrupamiento.fit(cafes_normalizado)
```

Out[8]:

```
KMeans(n_clusters=3)
```

Este método crea un atributo `label_` dentro del modelo clustering generado. Se agrega esta calificación al archivo original del Dataset. Finalmente los datos procesados obtenidos se muestran:

In [9]:

```
cafes['KMeans_clusters'] = agrupamiento.labels_
cafes.head()
```

Out[9]:

	Dulce	Floral	Especias	Tostado	Frutal	Fermentado	Vegetal	Otro	Cocoa	Cereal	Vainilla	Picante	Calificación promedio	KMeans_clusters	
0	14.23	1.71	2.43	15.6	127		2.80	3.06	0.28	2.29	5.64	1.04	3.92	1065	2
1	13.20	1.78	2.14	11.2	100		2.65	2.76	0.26	1.28	4.38	1.05	3.40	1050	2
2	13.16	2.36	2.67	18.6	101		2.80	3.24	0.30	2.81	5.68	1.03	3.17	1185	2
3	14.37	1.95	2.50	16.8	113		3.85	3.49	0.24	2.18	7.80	0.86	3.45	1480	2
4	13.24	2.59	2.87	21.0	118		2.80	2.69	0.39	1.82	4.32	1.04	2.93	735	2

## Visualización de los clusters Generados

Los datos tienen múltiples variables que los caracterizan, en este caso se desea visualizar un gráfico lo mayor resumido posible, y en la naturaleza humana se alcanzan a visualizar hasta tres dimensiones.

Para efectos didácticos, se mostraran en dos dimensiones ¿Cuales? se seleccionan las variables que mejor caracterizan a todos los datos, para ello se hace uso del Análisis de Componentes Principales (PCA) para reducir el número de variables a analizar, en este caso a visualizar. Se hace uso del paquete decomposition de sklearn y se crea un dataframe a partir de estos componentes para graficarlo.

In [10]:

```
from sklearn.decomposition import PCA
pca = PCA(n_components=2)           # Dos componentes principales
pca_cafes = pca.fit_transform(cafes_normalizado)
pca_cafes_df = pd.DataFrame(data=pca_cafes, columns=['Componente_1', 'Componente_2'])
pca_names_cafes = pd.concat([pca_cafes_df, cafes[['KMeans_clusters']]], axis=1)

# veamos el resultado de los datos procesados:
pca_names_cafes
```

Out[10]:

	Componente_1	Componente_2	KMeans_clusters
0	-0.706336	-0.253193	2
1	-0.484977	-0.008823	2
2	-0.521172	-0.189187	2
3	-0.821644	-0.580906	2
4	-0.202546	-0.059467	2
...	...	...	...
173	0.739510	-0.471901	1
174	0.581781	-0.348366	1
175	0.626313	-0.546857	1
176	0.572991	-0.425516	1

	Componente_1	Componente_2	KMeans_clusters
177	0.701764	-0.513505	1

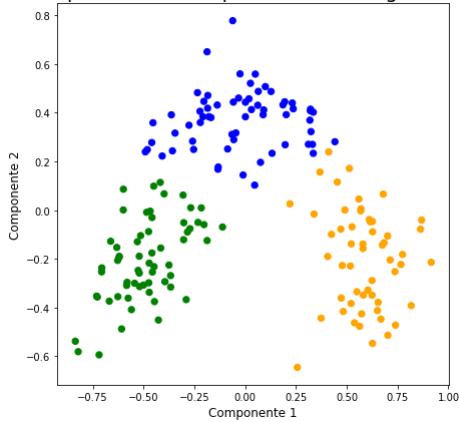
178 rows × 3 columns

## Graficar el dataframe procesado

Ahora se configura la figura plot a mostrar con estos datos obtenidos

```
In [11]: # Configurando La figura plot
fig = plt.figure(figsize=(7,7))
grafico = fig.add_subplot(1,1,1)
grafico.set_xlabel('Componente 1',fontsize = 12 )
grafico.set_ylabel('Componente 2',fontsize = 12 )
grafico.set_title('Componentes Principales - Clustering Kmeans',fontsize = 20 )
Colores = np.array(['blue", "orange", "green"])
grafico.scatter(x=pca_names_cafes.Componente_1, y=pca_names_cafes.Componente_2, c=Colores[pca_names_cafes.KMeans_clusters], s=40) #llamado al método de figura de puntos de dispersión.
plt.show()
```

Componentes Principales - Clustering Kmeans



## Guardar los datos generados

Se procede a guardar el dataframe en formato csv:

```
In [12]: # Se crea un archivo csv en la carpeta Results
cafes.to_csv('../Results/cafe-kmeans.csv')
```

## Ejercicios/ Experimentos propuestos

1. Elija un valor aleatorio para el número de clusters a implementar, suponiendo que no conoce el resultado del método Codo de Jambú. ¿Cómo cambia el resultado? ¿Qué+e se nota?
2. ¿Qué sucede al aumentar o disminuir el número de clusters a implementar en la llamada a KMeans? ¿Por qué?
3. ¿Qué sucede con los clusters al aumentar o disminuir significativamente el número de iteraciones máxima (seteado en 300) al usar el metodo del Codo de Jambú y en la llamada a KMeans? ¿Por qué?
4. Explique las ventajas y desventajas que tiene el algoritmo KMeans. Puede investigar diferentes fuentes.
5. Aumente el número de PCA a 3 componentes y grafíquelo en 3 Dimensiones. ¿Qué es lo que cambió y qué se está añadiendo?

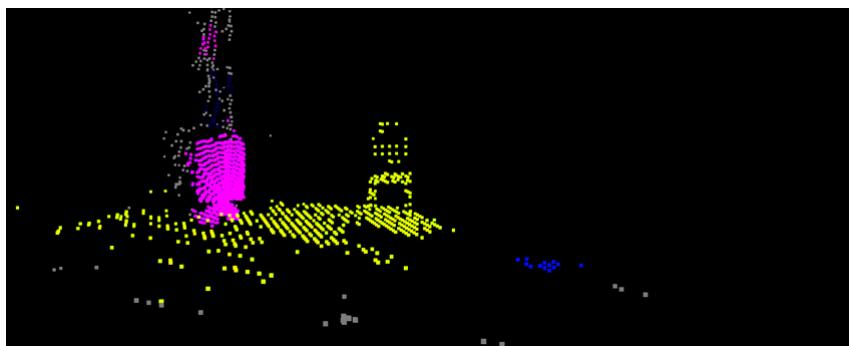
# Clustering con HDBSCAN - Generación de energía por Región

En la ingeniería eléctrica, específicamente el énfasis de sistemas de potencia, la generación de energía es una de las ramas principales de aplicación. En muchos artículos académicos e investigaciones se analiza la generación y consumo de energía en diferentes regiones y con múltiples variables con el fin de optimizar estepreciado recurso.



En este caso se tiene una base de datos con los resultados de diferentes mediciones de generación de energía por tipo y por región, a su vez que la cantidad de energía a producir estimada en unidades de Medición fensorial.

Las unidades de Medición fensorial como su nombre lo indica realizan una medida de los fasores de corriente y tensión de la red eléctrica, garantizando sincronización en las medidas y una alta tasa de muestreo que permite una visualización en tiempo real del sistema, siendo un factor clave para determinar flujos de potencia y en general el estado del sistema. El conjunto de dos o más PMU's instaladas en el sistema y los software de análisis de datos provenientes de estas es llamado un WAMS de PMU's. Todos los valores tomados, que se muestran en la base de datos están en MU.



Se utilizará el método HDBSCAN para ordenar los datos en clusteres relacionados, con el fin de analizar

características geográficas y técnicas de cada región. Este algoritmo realiza DBSCAN sobre diferentes valores de  $\epsilon$  y integra el resultado para encontrar un agrupamiento que brinde la mejor estabilidad sobre  $\epsilon$ . Esto permite que HDBSCAN encuentre grupos de diferentes densidades (a diferencia de DBSCAN) y sea más robusto para la selección de parámetros.

In [217]

```
# Imports

import numpy as np
import matplotlib.pyplot as plt
import hdbscan
import seaborn as sns
import pandas as pd
import sklearn.cluster as cluster

%matplotlib inline
```

## Importando Dataset y visualizando sus características

In [218]

```
data = pd.read_csv('../datasets/generación energía por tipo region.csv', engine='python')
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4945 entries, 0 to 4944
Data columns (total 9 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   indice          4945 non-null   int64  
 1   fecha            4945 non-null   object  
 2   Region           4945 non-null   object  
 3   Generacion Termica actual (MU) 4945 non-null   float64 
 4   Generacion Termica Estimada (MU) 4945 non-null   float64 
 5   Generacion Eolica Actual (MU) 4945 non-null   float64 
 6   Generacion Eolica Estimada (MU) 4945 non-null   float64 
 7   Hidro Generacion Actual (MU) 4945 non-null   float64 
 8   Hidro Generacion Estimada (MU) 4945 non-null   float64 
dtypes: float64(6), int64(1), object(2)
memory usage: 347.8+ KB
```

## Mostrar las primeras filas para una previsualización del orden de los datos

In [219]

```
data.head()
```

Out[219]

indice	fecha	Region	Generacion Termica actual (MU)	Generacion Termica Estimada (MU)	Generacion Eolica Actual (MU)	Generacion Eolica Estimada (MU)	Hidro Generacion Actual (MU)	Hidro Generacion Estimada (MU)	
0	0	9/1/2017	Central y Norte	624.23	484.21	30.36	35.57	273.27	320.81
1	1	9/1/2017	Pacifico	1106.89	1024.33	25.17	3.81	72.00	21.53
2	2	9/1/2017	Zona Sur	576.66	578.55	62.73	49.80	111.57	64.78
3	3	9/1/2017	Caribe	441.02	429.39	38.94	36.45	85.94	69.36
4	4	9/1/2017	Huetar Norte	29.11	15.91	0.00	0.00	24.64	21.21

Se puede ver que la columna "indice" es el índice numérico de la muestra, pero que ya se ha enumerado con el método de lectura, por ello, se procede a eliminarla

In [220]

```
data = data.drop(['indice'], axis=1)
data.head()
```

Out[220]

	fecha	Region	Generacion Termica actual (MU)	Generacion Termica Estimada (MU)	Generacion Eolica Actual (MU)	Generacion Eolica Estimada (MU)	Hidro Generacion Actual (MU)	Hidro Generacion Estimada (MU)
0	9/1/2017	Central y Norte	624.23	484.21	30.36	35.57	273.27	320.81
1	9/1/2017	Pacifico	1106.89	1024.33	25.17	3.81	72.00	21.53
2	9/1/2017	Zona Sur	576.66	578.55	62.73	49.80	111.57	64.78
3	9/1/2017	Caribe	441.02	429.39	38.94	36.45	85.94	69.36
4	9/1/2017	Huetar Norte	29.11	15.91	0.00	0.00	24.64	21.21

## Se obtienen las variables estadísticas de los datos por columna, para los valores numéricos.

In [221]

```
data_std = data.describe()
data_std
# En este punto se puede optar por un análisis de estos elementos, que caracterizan Los datos procesados, teniendo hasta N componentes
# a Los cuales aplicar métodos de agrupamiento y visualizarlos.
# En este caso, se hará un enfoque directo con los datos y no sus variables estadísticas.
```

Out[221]

	Generacion Termica actual (MU)	Generacion Termica Estimada (MU)	Generacion Eolica Actual (MU)	Generacion Eolica Estimada (MU)	Hidro Generacion Actual (MU)	Hidro Generacion Estimada (MU)
count	4945.000000	4945.000000	4945.000000	4945.000000	4945.000000	4945.000000
mean	603.978358	575.395116	22.353199	22.200097	73.305921	76.842965
std	383.534208	383.387299	22.005852	20.188407	74.482145	82.043952
min	12.340000	12.380000	0.000000	0.000000	0.000000	0.000000
25%	470.050000	427.460000	0.000000	0.000000	26.910000	23.310000
50%	615.280000	535.980000	25.130000	28.540000	52.960000	50.270000
75%	689.530000	672.740000	34.020000	36.600000	85.940000	95.800000
max	1395.970000	1442.380000	68.740000	76.640000	348.720000	397.380000

Estos datos son muy útiles para diferentes análisis de producción y estimación. Otra buena forma de tratar los datos es normalizándolos.

Nótese que este dataset mantiene datos numéricos y no-numéricos(texto) por lo que su procesamiento no se puede realizar de forma directa. En este caso existen varias formas de procesamiento y ordenamiento: Por fecha, por región o por valores numéricos. Una posibilidad es tratar estos datos texto como números o etiquetas numéricas, o simplemente omitiéndolos del procesamiento numérico.

Para esto, es posible crear un dataset o un dataframe que contenga los valores a procesar, mientras se guardan las etiquetas de texto en el vector original de datos o en otro lugar conocido para su procesamiento posterior.

In [222]

```
# Se quitan Las columnas fecha y región, que se dejarán como etiquetas posteriores.
data = data.drop(['fecha'],axis=1)
data = data.drop(['Region'],axis=1)
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4945 entries, 0 to 4944
Data columns (total 6 columns):
 #   Column           Non-Null Count  Dtype  
 --- 
 0   Generacion Termica actual (MU)  4945 non-null   float64
 1   Generacion Termica Estimada (MU) 4945 non-null   float64
 2   Generacion Eolica Actual (MU)    4945 non-null   float64
 3   Generacion Eolica Estimada (MU)  4945 non-null   float64
 4   Hidro Generacion Actual (MU)    4945 non-null   float64
 5   Hidro Generacion Estimada (MU)  4945 non-null   float64
dtypes: float64(6)
memory usage: 231.9 KB
```

In [223]

```
# Ahora que se tienen todos Los valores numéricos, se normalizan Los datos en un rango adecuado y se vuelven a obtener sus métricas:
data_norm = (data - data.min())/(data.max() - data.min())
data_norm.describe()
```

Out[223]

	Generacion Termica actual (MU)	Generacion Termica Estimada (MU)	Generacion Eolica Actual (MU)	Generacion Eolica Estimada (MU)	Hidro Generacion Actual (MU)	Hidro Generacion Estimada (MU)
count	4945.000000	4945.000000	4945.000000	4945.000000	4945.000000	4945.000000
mean	0.427599	0.393717	0.325185	0.289667	0.210214	0.193374
std	0.277194	0.268103	0.320132	0.263419	0.213587	0.206462
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.330804	0.290266	0.000000	0.000000	0.077168	0.058659
50%	0.435767	0.366154	0.365580	0.372390	0.151870	0.126504
75%	0.489430	0.461790	0.494908	0.477557	0.246444	0.241079
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

Se puede observar que ahora los datos tienen valores entre cero y uno, max y min.

Ahora con este preprocesamiento se tienen datos ordenados, numéricos y normalizados, listos para un agrupamiento óptimo.

In [224]

```
# Configuración del modelo HDBSCAN (una forma es aplicar fit_predict para evitar setear algunos parámetros)
# un ejemplo de configuración manual es: hdbscan_model = hdbscan.HDBSCAN(algorithm='best', alpha=1.0,approx_min_span_tree=True,gen_min_span_tree=True,Leaf_size=40,metric='euclidean',min_cluster_size=650, min_samp
```

Out[224]

col_0	Cantidad Elementos
row_0	
-1	671
0	916
1	991
2	957
3	1410

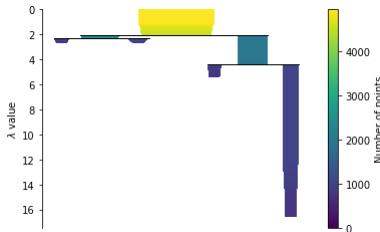
Se puede observar que con esta configuración de parámetros, HDBSCAN encuentra varios clusters con diferentes cantidades de elementos. HDBSCAN no tiene dificultades reconociendo los datos atípicos o outliers del dataset, como se puede ver en la agrupación -1. Esta es una de las principales fortalezas del método.

In [225]

```
## Ahora se grafican estos resultados:
hdbscan_auto.condensed_tree_.plot()
```

Out[225]

```
<AxesSubplot: xlabel='$\lambda$ value'>
```



Este método crea un atributo `label_` dentro del modelo clustering generado. Se agrega esta calificación al archivo original del Dataset:

```
In [226]: data['HDBSCAN clusters'] = cluster_labels
data.head()
```

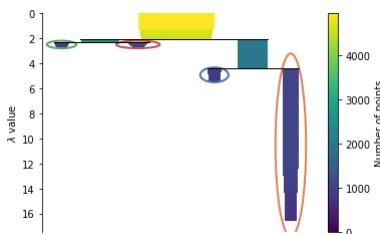
	Generacion Termica actual (MU)	Generacion Termica Estimada (MU)	Generacion Eolica Actual (MU)	Generacion Eolica Estimada (MU)	Hidro Generacion Actual (MU)	Hidro Generacion Estimada (MU)	HDBSCAN clusters
0	624.23	484.21	30.36	35.57	273.27	320.81	-1
1	1106.89	1024.33	25.17	3.81	72.00	21.53	-1
2	576.66	578.55	62.73	49.80	111.57	64.78	3
3	441.02	429.39	38.94	36.45	85.94	69.36	3
4	29.11	15.91	0.00	0.00	24.64	21.21	1

## Visualización de los clusters Generados

Ahora se pueden colorear los clusters generados en la imagen.

```
In [227]: hdbscan_auto.condensed_tree_.plot(select_clusters=True, selection_palette=sns.color_palette('deep',10))
```

```
Out[227]: <AxesSubplot:ylabel='$\lambda$ value'>
```



```
In [228]: # El otro enfoque mencionado, es a partir de los datos estadisticos que mejor representen las relaciones del dataset, para ello se puede hacer uso del Análisis de Componentes Principales (PCA)
from sklearn.decomposition import PCA

# Dos componentes principales
pca = PCA(n_components=2)

# Aplicar transformacion
pca_data = pca.fit_transform(data_norm)

# Obtener DataFrame ordenado
pca_df = pd.DataFrame(data=pca_data, columns=['Componente_1', 'Componente_2'])

# Aplicar el modelo clustering HDBSCAN y obtener etiquetas ordenadas
hdbscan_auto_PCA = hdbscan.HDBSCAN(min_cluster_size=60)

labels = pd.DataFrame(data=hdbscan_auto_PCA.fit_predict(pca_df), columns=['HDBSCANS_Labels'])

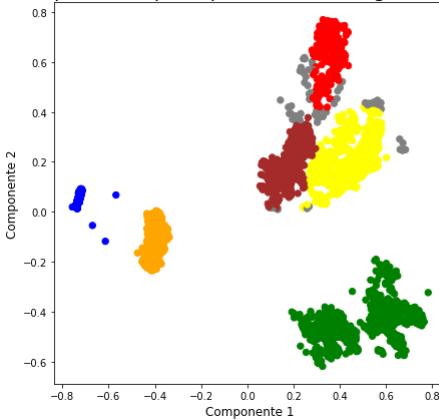
# Agregar etiquetas de agrupaciones y dataframe en un solo arreglo
pca_df = pd.concat([pca_df, labels], axis=1)
```

## Graficar el dataframe procesado

Ahora se configura la figura plot a mostrar con estos datos obtenidos

```
In [229]: fig = plt.figure(figsize=(7,7))
grafico = fig.add_subplot(1,1,1)
grafico.set_xlabel('Componente 1', fontsize = 12 )
grafico.set_ylabel('Componente 2', fontsize = 12 )
grafico.set_title('Componentes principales - Clustering HDBSCAN', fontsize = 20 )
Colores = np.array(["blue", "orange", "green", "yellow", "red", "brown", "purple", "pink", "grey"])
grafico.scatter(x=pca_df.Componente_1, y=pca_df.Componente_2, c=Colores[pca_df.HDBSCANS_Labels], s=40)
plt.show()
```

Componentes principales - Clustering HDBSCAN



## Guardar los datos generados

Se procede a guardar el dataframe en formato csv. Como se modificó el archivo inicial, ahora se carga el original, se agregan las etiquetas generadas y se vuelve a guardar como csv ya con el clustering realizado.

```
In [230]: csv = pd.read_csv('../datasets/generación energía por tipo region.csv',engine='python')  
csv['HDBSCAN_clusters'] = hdbscan_auto.labels_  
csv.to_csv('../Results/energy-HDBSCAN.csv') # Se crea un archivo csv en la carpeta Results
```

## Ejercicios/ Experimentos propuestos

1. Elija diferentes valores aleatorios para el número mínimo de clusters a implementar en el modelo HDBSCAN ¿Cómo cambia el resultado? ¿Qué se nota?
2. Investigue cuales son los parámetros configurables del método HDBSCAN (los mostrados en el comentario del código) y qué determinan cada uno de ellos.
3. Implemente el modelo HDBSCAN con estos parámetros seteados a su parecer. ¿Qué cambia?
4. Explique las ventajas y desventajas que tiene el algoritmo HDBSCAN. Puede investigar diferentes fuentes.
5. Aumente el número de PCA a 3 componentes y grafiquelo en 3 Dimensiones. ¿Qué es lo que cambio y que se esta añadiendo?

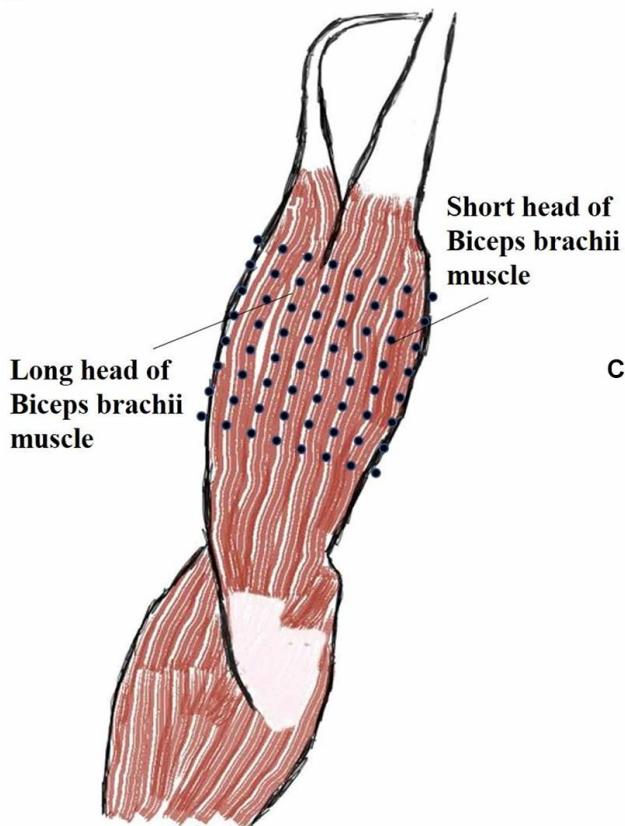
# Comparando Métodos de Agrupamiento

En muchas aplicaciones de Clustering y artículos académicos, es común apreciar comparaciones entre métodos o algoritmos de agrupamiento, esto se da ya que los resultados son muy variados y dependen del dataset, los parámetros utilizados y las métricas analizadas. Encontrar el mejor fit es una tarea interesante y que ayuda a profundizar sobre los diferentes métodos de agrupamiento.

Otra característica a tomar en cuenta de los datos a analizar en diferentes aplicaciones, es que los datasets vienen en formatos específicos y muchas veces no visibles de forma fácil sin un procesamiento adecuado; un conjunto de datos "a ciegas".

En este caso se analizará un dataset en formato de salida (output) codificado de forma que no es visible en cualquier aplicación, para ello se procesan los datos con los paquetes de Python para este fin.

A



B

Rows							
Columns							
1	2	3	4	5	6	7	8
2							
3							
4							
5							
6							
7							
8							

● channel position

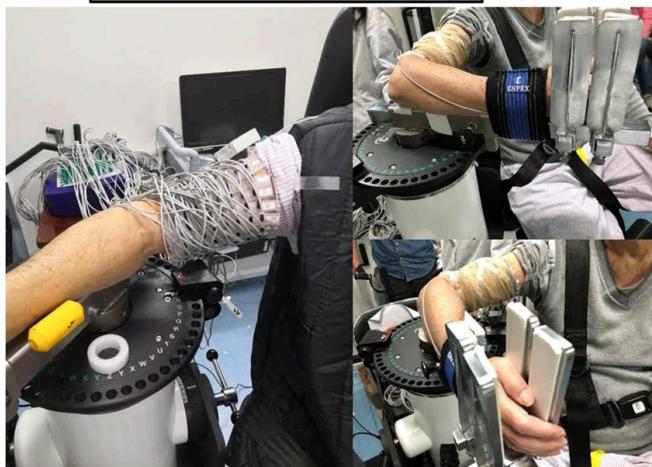
Proximal

Lateral

Medial

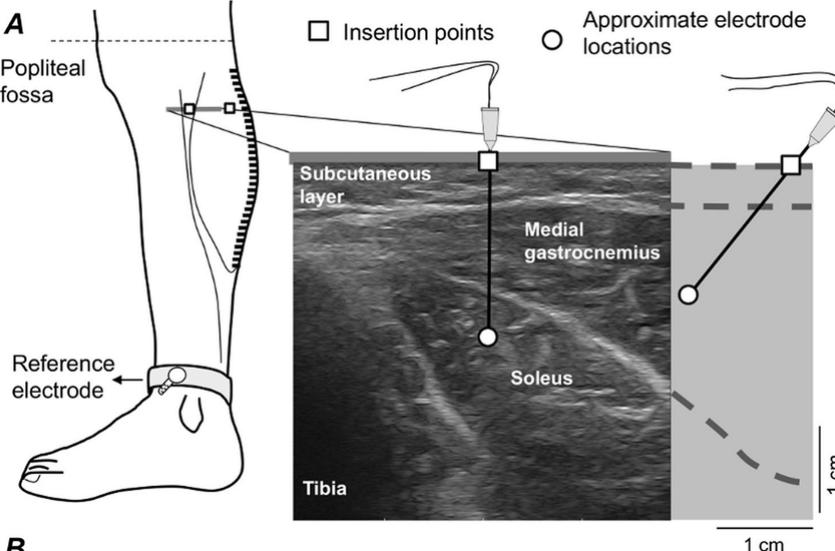
Distal

C

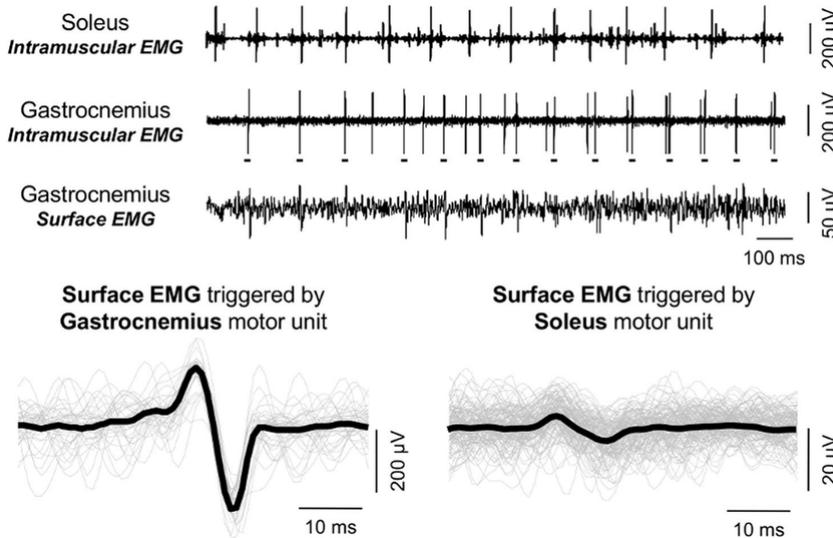


Los datos representan la salida de una aplicación de Red Neuronal, capturas de señales de los músculos responsables del movimiento de los dedos de la mano en una protesis EMG. La electromiografía (EMG) consta en colocar electrodos superficiales sobre la extremidad seleccionada por medio de electrodos conectados a un sensor mioeléctrico cuya salida va a una tarjeta de adquisición de datos.

El entrenamiento de los datos en el sistema se realiza por medio de un sistema RNA. Los RNA se definen como un sistema de mapeo no lineales. Constan de un número de procesadores simples ligados por conexiones con pesos.



**B**



Las unidades de procesamiento se denominan neuronas. Cada unidad recibe entradas de otros nodos y generan una salida simple escalar que depende de la información local disponible, y guardada internamente o que llega a través de las conexiones con pesos. Estos son datos los obtenidos y se desean analizar por medio de clustering para determinar las principales relaciones entre si, definiendo así grupos de acción neuronal.

In [25]:

```
# Imports y Configuracion de plot y seaborn

import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import sklearn.cluster as cluster
import time

%matplotlib inline

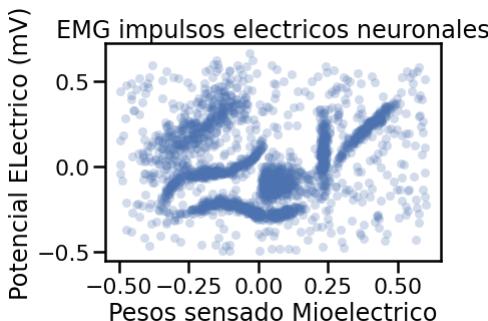
sns.set_context('poster')
sns.set_color_codes()
plot_kwds = {'alpha' : 0.25, 's' : 80, 'linewidths':0}
```

## Se cargan los datos de la aplicación EMG

In [26]:

```
# Se utiliza la funcion Load de numpy para cargar Los datos
data_EMG = np.load('../datasets/EMG Red neuronal Protesis.npy')

# Visualizamos de forma rapida Los datos Leido
plt.scatter(data_EMG.T[0], data_EMG.T[1], c='b', **plot_kwds)
frame = plt.gca()
frame.axes.set_title("EMG Impulsos Eléctricos Neuronales")
frame.axes.set_xlabel("Pesos sensado Mioeléctrico")
frame.axes.set_ylabel("Potencial Eléctrico (mV)")
frame.axes.get_xaxis().set_visible(True)
frame.axes.get_yaxis().set_visible(True)
```



Funcion para plotear graficos

Para comparar diferentes métodos de agrupamiento, se requiere de un proceso de prueba homogéneo para cada uno, por ello, se emplea un método automático para aplicar el algoritmo seleccionado bajo mismas condiciones y graficar los resultados. El código ha sido adaptado de la página oficial de la librería hdbSCAN, github.

In [27]:

```
# Se define la función para graficar
def plot_clusters(data, algorithm, args, kwds):
    start_time = time.time()
    labels = algorithm(*args, **kwds).fit_predict(data)
    end_time = time.time()
    palette = sns.color_palette('deep', np.unique(labels).max() + 1)
    colors = [palette[x] if x >= 0 else (0.0, 0.0, 0.0) for x in labels]
    plt.scatter(data.T[0], data.T[1], c=colors, **plot_kwds)
    frame = plt.gca()
    # Se definen las características del gráfico, leyenda, ejes y tiempo empleado
    frame.set_xlabel("Pesos sensado Mioeléctrico", fontsize=16)
    frame.set_ylabel("Potencial Eléctrico (mV)", fontsize=16)
    frame.axes.get_xaxis().set_visible(True)
    frame.axes.get_yaxis().set_visible(True)
    # Se definen las características del gráfico, leyenda, ejes y tiempo empleado
    plt.title('EMG Impulsos Eléctricos Neuronales - Clusters encontrados con {}'.format(str(algorithm.__name__)), fontsize=16)
    plt.text(-0.5, 0.65, 'Tiempo: {:.2f} s'.format(end_time - start_time), fontsize=12)
```

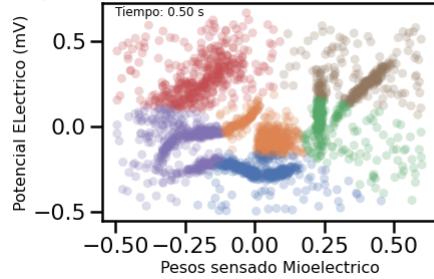
## Comparativa de métodos

### Usando KMEANS

In [28]:

```
# Se llama a la función definida, usando el dataset procesado, el método kmeans y un número de cluster de 6.
plot_clusters(data_EMG, cluster.KMeans, (), {'n_clusters':6})
```

EMG impulsos eléctricos neuronales - Clusters encontrados con KMeans



Se pueden apreciar los 6 clusters indicados, y el hecho de poder distinguirlos a simple vista indica que el agrupamiento es aceptable. Se recomienda utilizar el método del codo de Jambú para encontrar el número de Clusters óptimo. (ver aplicación de KMeans)

### Usando Propagación por Afinidad (Affinity Propagation)

La propagación de afinidad es un algoritmo de agrupación más nuevo que utiliza un enfoque basado en gráficos para permitir que los puntos "voten" por su "ejemplo" preferido. El resultado final es un conjunto de "ejemplos" de agrupaciones de las que derivamos agrupaciones básicamente haciendo lo que hace K-Means y asignando cada punto al grupo de su ejemplar más cercano.

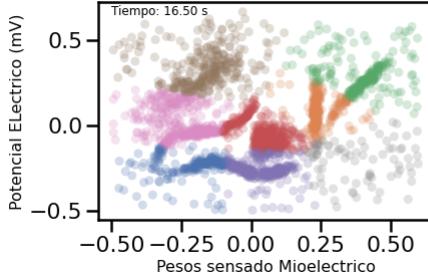
<https://www.scielo.br/j/bcg/a/s3rpjYbQHs5PBRVnsW4CkKS/?lang=en>

In [29]:

```
# Se llama a la función definida utilizando el dataset procesado, el algoritmo de afinidad y dos parámetros de configuración:
# Preference indica la probabilidad de que se seleccione el i-ésimo punto como ejemplo.
# La preferencia se pueden establecer en un valor global o para puntos de datos particulares.
# Damping factor: El factor de amortiguación (entre 0.5 y 1) es la medida en que el valor actual se mantiene en relación con los valores entrantes. Esto con el fin de evitar oscilaciones numéricas al actualizar el punto.
plot_clusters(data_EMG, cluster.AffinityPropagation, (), {'preference':-5.0, 'damping':0.95})
```

/home/bran/.local/lib/python3.8/site-packages/sklearn/cluster/\_affinity\_propagation.py:148: FutureWarning: 'random\_state' has been introduced in 0.23. It will be set to None starting from 1.0 (renaming of 0.25) which means that results will differ at every function call. Set 'random\_state' to None to silence this warning, or to 0 to keep the behavior of versions <0.23.  
warnings.warn(

EMG impulsos eléctricos neuronales - Clusters encontrados con AffinityPropagation



Se puede apreciar que este método generó 8 clusters diferentes, es decir categorizó más detalladamente el dataset, sin embargo esto no garantiza que sea la mejor representación. Además se puede ver que tardó mucho tiempo más que KMeans. Variando los parámetros de entrada se pueden obtener diferentes resultados.

### Usando Cambio promedio (Mean Shift)

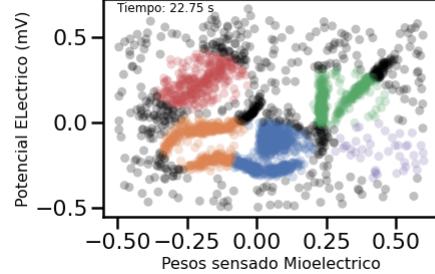
El cambio promedio es otra opción si no se desea especificar el número de grupos. Está basado en centroides (K-centroids), como K-Means y propagación por afinidad, pero puede devolver clústeres en lugar de una partición. La idea subyacente del algoritmo Mean Shift es que existe alguna función de densidad de probabilidad de la que se extraen los datos y trata de colocar centroides de grupos en los máximos de esa función de densidad.

In [34]:

```
# Se llama a la función definida y se selecciona el parámetro cluster_all como False:
# Si fuese verdadero, entonces todos los puntos están agrupados, incluso aquellos huérfanos que no están dentro de ningún grupo. Los huérfanos (outliers) se asignan al núcleo más cercano. Si es falso, los datos se agrupan en los clústeres que se definen.
# Por otro lado, se puede pasar el argumento bandwidth (float) que a su vez se le pasaría al método MeanShift, indicando el ancho de banda del proceso, que configura estados de escalabilidad (debilidad de este método).
plot_clusters(data_EMG, cluster.MeanShift, (0.175,), {'cluster_all':False})
```

/home/bran/.local/lib/python3.8/site-packages/sklearn/utils/validation.py:70: FutureWarning: Pass bandwidth=0.175 as keyword args. From version 1.0 (renaming of 0.25) passing these as positional arguments will result in an error  
warnings.warn(f"Pass {args\_msg} as keyword args. From version "

## EMG impulsos electricos neuronales - Clusters encontrados con MeanShift

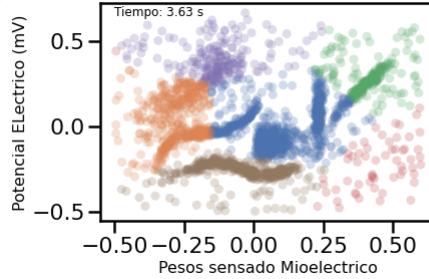


## Usando Clustering Espectral

La agrupación espectral se puede considerar mejor como una agrupación de gráficos. Para datos espaciales, se puede pensar en generar un gráfico basado en las distancias entre puntos como un gráfico k-NN, o de densidades. A partir de ahí, la agrupación espectral examinará los vectores propios del Laplaciano del gráfico para intentar encontrar una buena conjunción del gráfico en el espacio euclídeo. En este caso podemos pensar en términos de distancias, espacio y densidades.

```
In [35]: # Se llama a la función definida, con el clustering espectral un número de clusters de 6, al igual de Kmeans  
plot_clusters(data_EMG, cluster.SpectralClustering, (), {'n_clusters':6})
```

## EMG impulsos electricos neuronales - Clusters encontrados con SpectralClustering



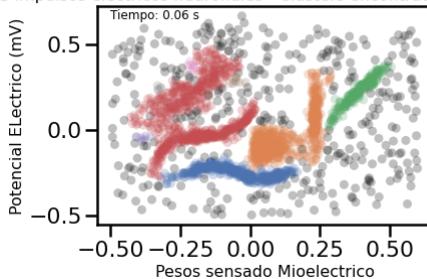
Muy similar a Kmeans, este método resulta útil para datos muy ordenados o delgados, nuevamente el ruido afecta en gran medida su desempeño.

## Usando DBSCAN

DBSCAN es un algoritmo basado en densidad: asume clústeres para regiones densas. No requiere que cada punto se asigne a un clúster y, por lo tanto, no partitiona los datos, sino que extrae los clústeres 'densos' y deja un fondo disperso clasificado como 'ruido'. Es decir, tiene facilidad para distinguir los datos atípicos. Existe una variación mejorada de este método usando clustering jerárquico, llamado HDBSCAN. (ver "Generación de energía").

```
In [37]: # En este caso se tiene otros parámetros muy diferentes que se pueden seleccionar, pero en general se recomienda solo manipular y variar el parámetro epsilo:  
# Los enlaces simples del algoritmo generan un dendrograma espacial, que debe ser cortado, este nivel de corte se da en el valor designado por epsilo (que resulta muy gráfico y empírico)  
plot_clusters(data_EMG, cluster.DBSCAN, (), {'eps':0.025})
```

## EMG impulsos electricos neuronales - Clusters encontrados con DBSCAN



Como se puede ver, este método determinó menos clusters, en total se pueden distinguir 5, excluyendo los outliers. Como se comentó, este método determinó una muy grande cantidad de estos datos como ruido y sus clusters parecen estar muy bien definidos.

## Actividades / Ejercicios propuestos

- Realice una tabla comparativa resumen entre los diferentes métodos de agrupamiento. Incluya tiempo ejecución utilizando estos resultados (o bien, realizar sus propias pruebas y obtener un promedio), escalabilidad, estabilidad o rendimiento y facilidad de parámetros (entiéndase, qué tan intuitivos o fáciles de elegir). Explique sus decisiones y muestre sus conclusiones.
- Realice experimentos variando los parámetros de cada método ¿qué ocurre? ¿Hay casos de fallos? ¿Por qué?
- Investigue qué otros métodos incluye la librería sklearn.cluster. Incluya algunos parámetros.

## Procesamiento Digital de Señales (DSP)

Una de las ramas más importantes de la electrónica y sistemas de control es el procesamiento Digital de señales. Son muchísimas las posibilidades de aplicación al procesar señales de distintas fuentes: Sonido, transductores, señales eléctricas o electromagnéticas puras entre muchas otras.



Los manatíes no se pueden ver en ríos de aguas turbias, por lo que se identifican por sus sonidos. Al igual que las personas, los manatíes emiten sus sonidos con ligeras variaciones, aunque cada uno tiene una "voz" única. Por esto, contar cuántos manatíes hay en un río se puede realizar usando técnicas de clustering y DSP, procesando los sonidos que se captan, pensando en que cada cluster corresponde a un manatí en particular.

En este caso se cuenta con una base de datos de muestras desonidas de manatíes captados en un río.

```
In [2]: # Imports
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

%matplotlib inline
```

### Datos a procesar

Se lee el archivo csv con el dataset

```
In [3]: manaties = pd.read_csv('../datasets/dataset_manaties.csv', engine='python')
manaties.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 4 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Frecuencia Fundamental (Khz)    150 non-null   float64 
 1   Longitud de sonido (s)         150 non-null   float64 
 2   Intensidad promedio (W/m2)     150 non-null   float64 
 3   Frecuencia promedio por hora(veces) 150 non-null   float64 
dtypes: float64(4)
memory usage: 4.8 KB
```

```
In [4]: # Observar datos de forma rápida
manaties.head()
```

	Frecuencia Fundamental (Khz)	Longitud de sonido (s)	Intensidad promedio (W/m2)	Frecuencia promedio por hora(veces)
0	7.9	3.8	6.4	2.0
1	7.7	3.8	6.7	2.2
2	7.7	2.6	6.9	2.3
3	7.7	2.8	6.7	2.0

Frecuencia Fundamental (Khz)	Longitud de sonido (s)	Intensidad promedio (W/m2)	Frecuencia promedio por hora(veces)
4	7.7	3.0	6.1

## Preprocesar los datos

Ahora se tiene los datos en formato leído de csv, pero resulta más fácil trabajar con ellos ya normalizados, y en formato de dataset. Como se puede ver, se tienen 4 variables para cada muestra.

Se obtienen también las métricas estadísticas para cada columna y las más significativas.

In [5]:

```
manaties.describe()
```

Dout[5]:

	Frecuencia Fundamental (Khz)	Longitud de sonido (s)	Intensidad promedio (W/m2)	Frecuencia promedio por hora(veces)
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333
std	0.828066	0.435866	1.765298	0.762238
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

In [6]:

```
# Normalizar Los datos
manaties_normalizado = (manaties - manaties.min())/(manaties.max() - manaties.min())
manaties_normalizado.describe()
```

Dout[6]:

	Frecuencia Fundamental (Khz)	Longitud de sonido (s)	Intensidad promedio (W/m2)	Frecuencia promedio por hora(veces)
count	150.000000	150.000000	150.000000	150.000000
mean	0.428704	0.440556	0.467458	0.458056
std	0.230018	0.181611	0.299203	0.317599
min	0.000000	0.000000	0.000000	0.000000
25%	0.222222	0.333333	0.101695	0.083333
50%	0.416667	0.416667	0.567797	0.500000
75%	0.583333	0.541667	0.694915	0.708333
max	1.000000	1.000000	1.000000	1.000000

Ahora que se tienen los datos normalizados entre 0 y 1, se pueden obtener sus métricas estadísticas en dos o tres variables PCA

In [7]:

```
from sklearn.decomposition import PCA
pca = PCA(n_components=3) # tres componentes principales
pca_manaties = pca.fit_transform(manaties_normalizado)
DataFrame_manaties = pd.DataFrame(data=pca_manaties, columns=['Componente_1', 'Componente_2', 'Componente_3'])
```

## Actividad propuesta:

1. Ahora que se tienen los datos procesados ordenados y normalizados, elija un método de agrupamiento y realice el análisis correspondiente, puede utilizar PCA (2 o 3 elementos) o usar los datos normalizados directamente.
2. Obtenga los diferentes clusters y cree un archivo csv con los resultados en la carpeta Results.
3. Realice gráficas del procedimiento y ajuste lo mejor posible el método. Justifique su elección.
4. Indique sus principales Conclusiones.

# Bibliografía

- [1] Eduardo Meza Fernandez. *Inteligencia artificial: Sistemas expertos y redes neuronales*. Universidad Nacional del Nordeste, Corrientes, Argentina, primera edition, 2003.
- [2] Divam Gupta, Ramachandran Ramjee, Nipun Kwatra, and Muthian Sivathanu. Unsupervised clustering using pseudo-semi-supervised learning. In *International Conference on Learning Representations*, 2020.
- [3] IBM. Sum of squared error (sse) (cluster evaluation algorithms), 2015.
- [4] T. Warren Liao. Clustering of time series data—a survey. *Pattern Recognition Society, Elsevier S.A*, 2005.
- [5] Wei Lu, Yongliang Wang, Qiqing Fang, and Peng Shixin. Downlink compressive channel estimation with support diagnosis in fdd massive mimo. *EURASIP Journal on Wireless Communications and Networking*, 2018, 05 2018.
- [6] Leland McInnes and John Healy. Accelerated hierarchical density based clustering. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 33–42, 2017.
- [7] José Mira M. *Aspectos conceptuales de la Inteligencia Artificial y la Ingeniería del Conocimiento*. McGRAW-HILL/INTERAMERICANA DE ESPAÑA, Universidad de Murcia, Cap.1 de: Inteligencia Artificial: Métodos, técnicas y aplicaciones de Inteligencia, Universidad Nacional de Educación a Distancia, primera edition, 2008.
- [8] E. Munera. *Inteligencia Artificial y Sistemas Expertos*. CESI, Universidad Politécnica de Madrid, Madrid, España, 1990.
- [9] Roque Palma M, José.; Marín M. *Inteligencia Artificial: Métodos, técnicas y aplicaciones*. McGRAW-HILL/INTERAMERICANA DE ESPAÑA, Universidad de Murcia, Aravaca (Madrid), primera edición, 2008.
- [10] Teng Qiu, Chaoyi Li, and Yongjie Li. D-nnd: A hierarchical density clustering method via nearest neighbor descent. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 1414–1419, 2018.
- [11] Joshua D. Rhodes, Wesley J. Cole, Charles R. Upshaw, Thomas F. Edgar, and Michael E. Webber. Clustering analysis of residential electricity demand profiles. *Applied Energy*, 135:461–471, 2014.

- [12] Vila; Delgado C. Miguel Vila M. *Técnicas de Agrupamiento*. McGRAW-HILL/INTERAMERICANA DE ESPAÑA, Universidad de Murcia, Cap.16 de: Inteligencia Artificial: Métodos, técnicas y aplicaciones de Inteligencia, Universidad de Granada, primera edition, 2008.
- [13] Esma Ergüner Özkoç. Clustering of time-series data. *Başkent University, Ankara, Turkey*, 2018.