PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

FACULTAD DE INGENIERÍA

# “Clasificación de los productos de una empresa de Quito considerando el recurso tiempo de mano de obra asignado a cada uno de sus procesos durante el año 2021-2022 utilizando algoritmos de aprendizaje no supervisado”

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE MÁSTER EN SISTEMAS DE INFORMACIÓN, MENCIÓN CIENCIA DE DATOS

NORKA GERMANIA GUAÑUNA VITERI

TUTOR: ING. RAFAEL MELGAREJO HEREDIA, PHD

QUITO, 2022

# TABLA DE CONTENIDO

Índice de gráficos

Índice de Figuras

Índice de Tablas

# CAPÍTULO 1

# 1. DESCRIPCIÓN DEL PROBLEMA

## 1.1 Planteamiento del problema

Se ha evidenciado que dentro de la organización no se cuenta con un proceso totalmente controlado al momento de asignar el recurso de mano de obra en cada uno de sus productos dentro de sus respectivos procesos, lo que generó una gran cantidad de horas extras que están afectando a la rentabilidad de la empresa. Por lo que conocer el recurso de tiempo de mano de obra asignado a cada uno de los procesos se vuelve un aspecto crítico para solventar. El desconocimiento de la cantidad de recurso tiempo de mano de obra asignados genera un descontrol en el mismo por lo que no se puede monitorear la variación y tomar medidas correctivas en el momento oportuno generando así mayor cantidad de horas extras siendo estas totalmente innecesarias. El conocer que productos tienen mayor variación en la asignación de recurso tiempo de mano de obra permitirá monitorear estos productos para plantear planes de mejora dentro de sus procesos, permitiendo generar una mejor distribución de este recurso.

## 1.2 OBJETIVOS

### 1.2.1 Objetivo General

- Clasificar los productos de una empresa de Quito considerando el recurso tiempo de mano de obra asignado a cada uno de sus procesos durante el año 2021-2022 utilizando algoritmos de aprendizaje no supervisado

### 1.2.2 Objetivos Específicos

- Analizar el porcentaje de variación del recurso de mano de obra asignado a cada uno de los procesos de manufactura dentro de una empresa de Quito.
- Aplicar dos algoritmos de aprendizaje no supervisado para la clasificación de los productos de una empresa de Quito.

- Conocer qué productos son los que están demandando mayor cantidad de recurso tiempo de mano de obra dentro de cada proceso.

## 1.3 JUSTIFICACIÓN

Dentro de una organización se generan grandes cantidades de datos las mismas que al no ser utilizadas se convierten en información poco útil, por eso incursionar con este tema ayuda a visibilizar datos y problemas que al ser tomados en consideración y clasificados ayudarán a que la empresa actúe de forma oportuna frente a ellos sin esperar que estos se vuelvan más grandes e incontrolables.

Al usar un exceso de mano de obra en un producto sube el precio del producto y también le resta el uso del recurso en otros procesos productivos que se encuentran planificados, si estos recursos se usan eficientemente la planificación programada se puede cumplir a cabalidad, cumpliendo con los requerimientos de los clientes a tiempo y aumentando así la confiabilidad de estos hacia la compañía ayudándole a que su posicionamiento en el mercado sea mucho mayor al que tiene actualmente.

# CAPÍTULO II

# 2. MARCO TEÓRICO

Dentro de este capítulo, se mencionan conceptos acerca de los costos por órdenes de producción y mano de obra directa. También se abordan temas como Machine Learning, aprendizaje no Supervisado y sus principales algoritmos dado a que el objetivo de este proyecto es clasificar datos es necesario conocer estos fundamentos.

## 2.1.1 Costos

Según el concepto de costos de (Pacheco Bautista, 2019) “son todos los valores monetarios empleados en un tiempo determinado para la elaboración de servicios y son recuperables”. El autor clasifica los costos en directos e indirectos, los costos directos son los que influyen en la fabricación de un producto, como por ejemplo los materiales y mano de obra directa y los costos indirectos se caracterizan por que no son asignados directamente al producto pero son esenciales para llevar a cabo la producción, por ejemplo, materiales o mano de obra indirecta y otros costos generales (luz, agua, depreciación, arrendamiento, entre otros) (Pacheco Bautista, 2019).

## 2.1.2 Costos por órdenes de producción

Según (Pacheco Bautista, 2019) Las empresas industriales transforma materia prima en un producto terminado, con el objetivo de ser comercializado, por lo que, este proceso se basa en la técnica de costeo total o absorbente, que se reflejan en los estados financieros de la empresa.

La técnica de costeo total también se la denomina costos por órdenes de fabricación o específicas de producción, lotes de trabajo, o pedidos de los clientes, el objetivo de este sistema es encontrar el precio total de materiales, mano de obra y costos indirectos manejados en el proceso de fabricación hasta obtener un producto terminado y luego ser entregados al sector comercial (Pacheco Bautista, 2019).

Dentro de la industria se emplea el sistema de costos por órdenes de producción cuando el tiempo de fabricación de una unidad de producto es relativamente largo, el precio de venta depende directamente del costo de producción y cuando la producción se programa por trabajos. Cuando la producción no tiene un ritmo constante bajo la modalidad de ordenes de trabajo es indispensable realizar una

planificación estratégica para lograr mayor eficiencia en uso del potencial humano y maquinaria. La planificación de la producción inicia con la solicitud del cliente, a partir de la cual se genera y emite una orden de producción (Rojas Medina , 2007).

### 2.1.3 Mano de obra directa

La mano de obra es conocida también como el costo que se paga a los trabajadores por las horas que destinan en la transformación de la materia prima en un producto final como indica (Rojas Medina , 2007).

Se conoce tambien como mano de obra a todos los salarios, prestaciones sociales, y demás conceptos laborales, que son remunerados a los trabajadores que intervienen de forma directa o indirecta en la fabricación de un producto o prestación del servicio (Jiménez Lemus, 2010).

El costo de mano de obra es la retribución que obtiene el personal por el esfuerzo físico o mental ejecutado. La mano de obra se clasifica en dos: la mano de obra directa es aquella que se realiza dentro del proceso de fabricación, aquí se encuentran todos los operarios, ya que son ellos los que tienen contacto directo con la materia prima y la modifican hasta llegar a un producto final, de acuerdo a (Rojas Medina , 2007).

Como menciona Rojas (2007) “la contabilización de la mano de obra en un sistema de costos comúnmente consta de tres actividades: control de tiempo, cálculo de la nómina total y asignación de los costos de la nómina”.

### 2.1.4 Procesos

(Arango Serna, Campuzano Zapata, & Zapata Cortes, 2015) denota que “se necesita contar con producciones eficientes en donde no exista retrasos en la entrega de referencias de un producto, este aspecto es sumamente importante para que las empresas se mantengan activas en el mercado, el cual exige tiempos de entrega rápidos, cumplimiento de cantidad y calidad”.

Por lo tanto, la implementación de sistemas de producción más eficientes ha llegado a ser un factor que es crucial para las plantas de manufactura que busquen eficiencia y eficacia en sus procesos (Arango Serna, Campuzano Zapata, & Zapata Cortes, 2015).

Los procesos se consideran la base operativa de las organizaciones y su papel preponderante se atribuye, en parte, a la necesidad de alinear los resultados organizacionales a las exigencias y expectativas de los clientes (Hernández Nariño, Medina León, Nogueira Rivera, Negrin Sosa, & Marqués León, 2014).

De acuerdo con (Hernández Nariño, Medina León, Nogueira Rivera, Negrin Sosa, & Marqués León, 2014), “un proceso es una secuencia ordenada de actividades repetitivas que se realizan por una persona, grupo o departamento dentro de la organización, estas actividades tienen la capacidad de transformar unas entradas (inputs) en salidas o resultados programados (outputs) para un destinatario ejecutado de una manera eficaz y eficiente para obtener un valor agregado”.

### 2.1.5 Machine Learning

Machine learning es un subcampo de la inteligencia artificial que puede definirse como el conjunto de métodos que detectan patrones automáticamente en un conjunto de datos sin estar programados explícitamente para ello, estos métodos pueden emplearse para predecir datos futuros, o para llevar a cabo otro tipo de decisiones en entornos de incertidumbre (Menasalvas, Rodríguez, Jiménez , & Duque, 2021).

“Se considera al Machine Learning como una posible metodología para dotar a un sistema inteligente la capacidad de realizar una determinada tarea” (Menasalvas, Rodríguez, Jiménez , & Duque, 2021).

#### *2.1.5.1 Algoritmos de Machine Learning*

(Menasalvas, Rodríguez, Jiménez , & Duque, 2021), indica que “dentro de los algoritmos de Machine Learning, existen varias categorías en base a distintos criterios”. Por lo que los autores manifiestan la necesidad de realizar una diferenciación enfocada en la funcionalidad, entre algoritmos descriptivos y predictivos.

(Menasalvas, Rodríguez, Jiménez , & Duque, 2021) señala que un algoritmo descriptivo “tiene como objetivo describir la población mediante la exploración de las propiedades y características de los datos”, dentro de estos algoritmos los autores encuentra a: Clustering, summarization, Reglas de Asociación y Sequence Discovery.

(Menasalvas, Rodríguez, Jiménez , & Duque, 2021) definen que un algoritmo predictivo “tiene como objetivo realizar predicciones sobre eventos futuros desconocidos a partir de datos históricos, en este caso encontramos a algoritmos de clasificación y estimación de valor”.

Comúnmente se utiliza una clasificación en función a sus datos es decir si contamos con datos etiquetados o no etiquetados.

Es fundamental conocer que existen cuatro categorías: Aprendizaje supervisado, aprendizaje no supervisado, aprendizaje semisupervisado y aprendizaje por refuerzo (Menasalvas, Rodríguez, Jiménez , & Duque, 2021).

#### *2.1.5.2 Aprendizaje no Supervisado*

Los algoritmos de aprendizaje no supervisado trabajan con conjunto de datos in etiquetar permitiendo inferir patrones o relaciones existentes a través de la exploración de los datos e inferencias que permiten descubrir estructuras ocultas en los datos (Menasalvas, Rodríguez, Jiménez , & Duque, 2021).

El no contar con datos etiquetados supone la ausencia de un punto de referencia con el que se evalua la calidad del modelo, lo que aumenta la complejidad de la validación del modelo (Menasalvas, Rodríguez, Jiménez , & Duque, 2021).

#### *2.1.5.3 Clustering*

Clustering es una tarea importante en el análisis de datos y las aplicaciones de minería de datos. Los datos se dividen en grupos similares de objetos en función de sus características mediante el proceso de clustering, en donde los objetos que pertenecen a un grupo son muy similares entre sí, y al mismo tiempo son diferentes respecto a los objetos que pertenecen a otros grupos (Pérez-Ortega, y otros, 2018).

Cada grupo de datos con objetos similares son clusters, esto significa que los clusters son el conjunto ordenado de datos que tienen características semejantes. Clustering es un proceso de aprendizaje no supervisado (Yadav & Dhingra, 2016).

### 2.1.6 Algoritmo K-means

Según (López Sánchez , 2019) “K – means es un algoritmo heurístico de clasificación no supervisada que agrupa un conjunto de N datos en k grupos basándose en sus características”. Se dividen los datos en clases o grupos homogéneos de modo que los elementos de la misma clase son tan similares como sea posible, eso se da lugar gracias al agrupamiento en donde se minimiza la distancia intracluster y maximiza la distancia inter-cluster.

“La distancia más utilizada en el algoritmo K- means es la distancia euclideana**”** (López Sánchez , 2019).

Por lo tanto este algoritmo construye una partición de las observaciones en k grupos con la finalidad de minimizar la suma de las distancias cuadráticas (within-cluster sum of squares, WCSS) de cada dato al centroide de su cluster. Es decir, minimiza la función conocida como función de error cuadrático (López Sánchez , 2019).

Entre las principales características de este algoritmo tenemos las siguientes: procesa grandes conjuntos de datos de manera eficiente y funciona solo con variables numéricas (Yadav & Dhingra, 2016).

#### *2.1.6.1 Etapas del algoritmo K – means*

(López Sánchez , 2019) señala que “K- means es un algoritmo iterativo que se divide en 3 etapas Iniciación, Asignación, Actualización”

“Iniciación: Se elige aleatoriamente el número de clusters k y los centroides $\left(\boldsymbol{m}^{(\boldsymbol{k})}\right)$ iniciales del conjunto de datos” (López Sánchez , 2019).

“Asignación: Cada dato es asignado a su centroide más cercano y se calcula la distancia euclideana entre ambos. Cada uno de los centroides define un cluster **“**(López Sánchez , 2019).

“Actualización: Se actualiza la posición del centroide de cada grupo tomando como nuevo centroide el valor medio de todos los datos que define un cluster” (López Sánchez , 2019).

### *2.1.6.2 Método del codo*

Para realizar una elección adecuada del número de clusters dentro del algoritmo K-means se empleará el método del codo.

El “método de codo” (elbow method) utiliza el valor WCSS (within-cluster sum of squares) es decir, “usa la suma de la distancia euclidiana de cada punto de un k cluster con respecto a su centroide” (López Sánchez , 2019). Según (Segovia Ortega, 2021) lo antes descrito, “repite iterativamente partiendo desde un k= i hasta llegar a un valor de $k_n$ , en donde el valor de k varia de uno en uno”.

De acuerdo con (Segovia Ortega, 2021) “Una vez obtenido los valores de WCSS, se realiza una gráfica en función de un rango alto de valores k, el punto donde se observa un cambio brusco en la línea continua a medida que aumenta el parámetro k se conoce como “codo” y es justo para este número de agrupamientos que el modelo obtenido en los resultados es el más optimo”

### *2.1.6.3 Normalización de datos*

El preprocesamiento de datos se usa para mejorar el rendimiento de los resultados y se realiza antes de usar cualquier algoritmo de exploración de datos. La normalización de datos se encuentra entre los procesos de preprocesamiento, en el cual se escalan los datos de los atributos para caer en un pequeño rango especificado (Mohamad & Usman, 2013). “La normalización antes de la clusterización es especialmente necesaria para el cálculo de la distancia euclidiana que es sensible a variaciones dentro de la magnitud o escalas de los atributos” (Mohamad & Usman, 2013).

Como indica (Mohamad & Usman, 2013) “En aplicaciones reales, debido a las variaciones en la selección del valor atributo, un atributo puede dominar a otro**.** La normalización evita que los atributos que poseen números más grandes superen a los atributos que poseen números más pequeños. El objetivo sería igualar las dimensiones o magnitudes y también la variabilidad de las características” por consiguiente el autor manifiesta que las técnicas de preprocesamiento de datos se “aplican a datos sin procesar para que los datos sean limpios, libres de ruido y consistentes”**.**

“La normalización genera clústers de buena calidad y mejora la precisión de los algoritmos de agrupamiento a través de la estandarización de los datos sin procesar

convirtiéndolos a un rango especifico mediante una transformación lineal" es lo expresado por (Mohamad & Usman, 2013).

Para (Mohamad & Usman, 2013) "No existe una regla universalmente definida para normalizar los dataset, por lo tanto, esto queda a criterio del usuario. Los métodos de normalización de datos incluyen Z-score, Min-Max y escala decimal".

**"Z-score:** los valores para un atributo X se estandarizan en función de la media y desviación estándar de X, este método es útil cuando se desconocen el mínimo y máximo reales del atributo X" (Mohamad & Usman, 2013).

$$\boldsymbol{X_{ij} = Z(X_{ij}) = \frac{X_{ij} - \overline{X_J}}{\sigma_j}}$$

**"Min-Max:** transforma el conjunto de datos entre 0.0 y 1.0" (Mohamad & Usman, 2013). Usa la siguiente formula:

$$\boldsymbol{MM(X_{IJ}) = \frac{X_{ij} - X_{min}}{X_{max} - X_{min}}}$$

**"Normalización por escala decimal:** normaliza moviendo el punto decimal de los valores de la característica X**, e**l número de puntos decimales movidos depende del valor absoluto máximo de X" (Mohamad & Usman, 2013). Se usa la siguiente formula:

$$\boldsymbol{DS(X_{ij}) = \frac{X_{ij}}{10^c}}$$

### 2.1.7 Density Based Spatial of Clustering of Application with Noise (DBSCAN)

"El algoritmo DBSCAN usa una estimación de densidad simple media, basada en el umbral para el número de vecinos, minPts dentro de un radio Ɛ (con una medida de distancia arbitraria)" (Schubert, Sander, Ester, Kriegel, & Xu, 2017).

"El objetivo de DBSCAN es localizar aquellas áreas que satisfagan esta densidad mínima y que están separados por zonas de menos densidad" (Schubert, Sander, Ester, Kriegel, & Xu, 2017).

Por razones de eficiencia, DBSCAN no realiza estimaciones de densidad entre puntos (Schubert, Sander, Ester, Kriegel, & Xu, 2017).

Las regiones con una alta densidad de puntos representan la existencia de clústeres mientras que las regiones con una baja densidad de puntos indican grupos de ruido o grupos de valores atípicos (Chakraborty & Nagwani, 2011).

DBSCAN hace que los clústeres se formen en base a un análisis de accesibilidad de densidad y conectividad de densidad (Chakraborty & Nagwani, 2011).

En donde:

“Accesibilidad de densidad: Se dice que un punto p es de densidad accesible desde un punto q si el punto p está dentro de la distancia Ɛ del punto q y q tiene suficiente número de puntos en sus vecinos que están dentro de la distancia Ɛ” (Pastrán Ramírez & Gongora Aya, 2021).

“Conectividad de densidad: Se dice que un punto p y q están conectados a la densidad si existe un punto r que tiene un número suficiente de puntos en sus vecinos y los puntos p y q están dentro de la distancia Ɛ. Este es el proceso de encadenamiento, entonces, si q es vecino de r, r es vecino de s, s es vecino de t, que a su vez es vecino de p y esto implica que q es vecino de p” (Pastrán Ramírez & Gongora Aya, 2021).

En la figura 1 se ilustra el concepto de DBSCAN, donde “objetos con más de minPts vecinos dentro de este radio (incluido el punto de consulta) se consideran un punto central” (Schubert, Sander, Ester, Kriegel, & Xu, 2017). *A es el punto central.*

(Schubert, Sander, Ester, Kriegel, & Xu, 2017) indica que,” todos los vecinos dentro del radio Ɛ de un punto central se considera parte de este cluster que el punto central (llamado densidad directa alcanzable). Si alguno de estos vecinos vuelve a ser un punto central, sus vecinos son incluidos transitivamente (densidad alcanzable)”.

“Los puntos que no son centrales en este conjunto se denominan puntos fronterizos y todos los puntos dentro del mismo conjunto están densamente conectados. *B y C son puntos fronterizos”* (Schubert, Sander, Ester, Kriegel, & Xu, 2017).

Puntos que no son densamente alcanzables desde cualquier punto central se consideran ruido y no pertenecen a ningún cluster. *N son puntos ruido* (Schubert, Sander, Ester, Kriegel, & Xu, 2017).

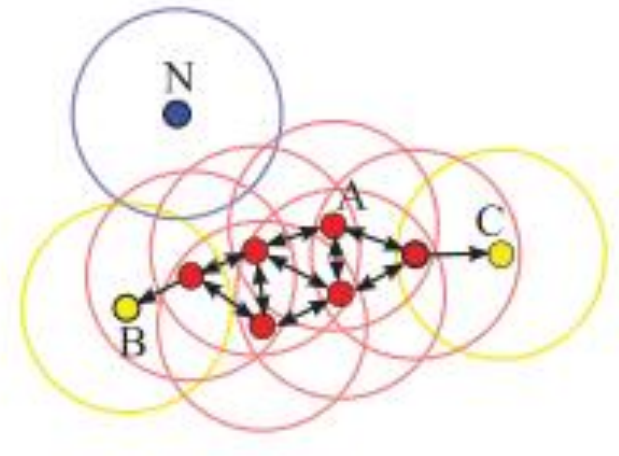


Figura 1: Ilustración de DBSCAN*(Schubert, Sander, Ester, Kriegel, & Xu, 2017)*.

Las flechas indican la accesibilidad a la densidad directa. Los puntos B y C están densamente conectados, porque ambos son densamente alcanzables desde A (Schubert, Sander, Ester, Kriegel, & Xu, 2017).

N no es densamente alcanzables y por lo tanto se considera un punto de ruido (Schubert, Sander, Ester, Kriegel, & Xu, 2017).

La idea clave del agrupamiento basado en la densidad es que para cada objeto de un grupo la vecindad de un radio dado (eps ) tiene que contener en menos un número mínimo de objetos (MinPts), es decir, la cardinalidad de la vecindad tiene que superar algún umbral (Schubert, Sander, Ester, Kriegel, & Xu, 2017).

El algoritmo DBSCAN requiere dos parámetros: ε (eps, radio del cluster) y MinPts (número mínimo de puntos necesarios dentro del cluster) (Karami & Johansson, 2014).

#### *2.1.7.1 Etapas de DBSCAN*

- “Se debe comenzar con un punto de partida arbitrario que no ha sido visitado” (Chakraborty & Nagwani, 2011).

- “Se recupera la vecindad ε de este punto, si contiene suficientes puntos se forma el cluster, de lo contrario, el punto se etiqueta como ruido (más tarde este punto se puede convertir en parte del grupo)” (Chakraborty & Nagwani, 2011).
- “Si se encuentra que un punto es parte del grupo, entonces su ε -vecindad también es parte del grupo y el procedimiento anterior del paso 2 se repite para todos los puntos de ε - vecindario. Esto se repite hasta que se determinen todos los puntos en el grupo” (Pastrán Ramírez & Gongora Aya, 2021).

#### *2.1.7.2 EPS*

Para determinar diferentes rangos de valores eps se necesita realizar un gráfico de k-dist para todos los puntos, el valor k dado es ingresado por el usuario (Gaonkar & Sawant, 2013).

Inicialmente se calcula el promedio de las distancias de cada punto a todos los k de su vecino más cercano (Gaonkar & Sawant, 2013). El uso de la grafico k-dist permite el cálculo eficiente de k-nearest neighbors of a point (k vecino más cercano de un punto) (Gaonkar & Sawant, 2013).

El promediar permite suavizar la curva hacia eliminación de ruido, para una automatización posterior y más fácil de detección de umbrales de densidad (Gaonkar & Sawant, 2013). La grafica k-distances se la realiza en orden ascendente, para ayudar a identificar el ruido con relativa facilidad (Gaonkar & Sawant, 2013).

El objetivo es determinar las rodillas “knees” que corresponde a un umbral, donde un cambio agudo de gradiente ocurre a lo largo de la curva de k-distance (Gaonkar & Sawant, 2013). Esto representa un cambio en la distribución de la densidad entre puntos**, c**ualquier valor inferior a este umbral eps estimado puede agrupar patrones de manera eficiente, donde el promedio de k-distances es menor que eps, lo que implica patrones o puntos pertenecientes a una determinada densidad (Gaonkar & Sawant, 2013).

La trama se verá más como escalera si los objetos están distribuidos regularmente dentro de grupos de diferentes densidades (Gaonkar & Sawant, 2013).

Para conjunto de datos con una densidad muy variada, se presentará alguna variación, dependiendo de la densidad del grupo y la distribución aleatoria de los

puntos, pero para puntos de el mismo nivel de densidad, el rango de variación no será enorme mientras que se espera un cambio brusco para ver entre dos niveles de densidad (Gaonkar & Sawant, 2013).

#### *2.1.7.3 Determinación de MinPts*

Este parámetro decide el tamaño del cluster y también afecta el número de datos ruidosos (Starczewski, Goetzen, & Joo Er, 2020). Además, si el valor MinPts tiene un valor alto, el número de clusters es pequeño (Starczewski, Goetzen, & Joo Er, 2020). Por otro lado, cuando este parámetro es demasiado pequeño el algoritmo de clustering puede crear muchos agrupamientos (Starczewski, Goetzen, & Joo Er, 2020). En general, la elección de este parámetro suele realizarse individualmente dependiendo del conjunto de datos, pero en muchos casos, el Min Pts es igual a 4 o 5 (Starczewski, Goetzen, & Joo Er, 2020). Estos valores aseguran un buen compromiso entre el tamaño de los grupos y la cantidad de datos de ruido (Starczewski, Goetzen, & Joo Er, 2020).

### 2.1.8 Parámetros de estimación del modelo

Como indica (Bojorque Chasi, 2020) “es importante cuantificar el rendimiento del clustering realizado, para lo cual se usan medidas intrínsecas, que miden la bondad de los resultados sin tener en cuenta información externa”.

Para (Pastrán Ramírez & Gongora Aya, 2021) “dentro de la validación intrínseca se distinguen las medidas de cohesión y separación, en donde, las medidas de cohesión determinan el grado de proximidad de los puntos que conforman el clusters, en cambio las medidas de separación determinan el grado de separación de los clusters”.

Las medidas intrínsecas más usadas son:

- *Coeficiente de Silhouette:* es una métrica para evaluar la calidad del agrupamiento. “La evaluación de la calidad de los agrupamientos se realiza utilizando la distancia interna e interdistancia que existen entre los objetos que son parte de una estructura de datos, entre más alto es el valor de este indicador mayor es la probabilidad de que ese sea el número ideal de clústeres

que se deben ingresar como parámetro inicial en el algoritmo” (Segovia Ortega, 2021) .

Se define como:

$$s(i) = \frac{b - a}{\max(a, b)}$$

Donde:

- a: “mide la cohesión como la distancia media de esa instancia al resto de instancias de ese cluster” (Segovia Ortega, 2021) .
- b: “la separación como la distancia media de esa instancia al conjunto de instancias que conforman el cluster más cercano. El cálculo de esta media a nivel de cluster se calculará como la media de este coeficiente para todas sus instancias” (Segovia Ortega, 2021).

“Este coeficiente esta cotado al intervalo [-1,1] donde un valor más alto será indicativo de clusters mejor definidos” (Segovia Ortega, 2021).

# CAPÍTULO III

# 3. METODOLOGÍA

Se usa algoritmos de aprendizaje no supervisado por que no se cuenta con las etiquetas de estos datos.

Los algoritmos que se emplean son K- means que nos permite agrupar datos en base a sus características formando grupos homogéneos es decir que los elementos de este cluster son similares entre sí y DBSCAN que utiliza una estimación de densidad para formar los cluster en función del número de los datos vecinos dentro de un radio.

## 3.1 Recolección de datos

Se cuenta con un archivo en formato txt con información detallada de los procesos de manufactura y productos que se fabrican dentro de la empresa dentro del periodo 2021 y 2022. Adicionalmente se cuenta con la cantidad teórica y la cantidad real de mano de obra asignada para cada proceso de manufactura.

## 3.2 Herramientas y Técnicas

En la elaboración de este proyecto de utilizará el lenguaje Python que es uno de los lenguajes preferidos en el ámbito de Machine Learning por su amplio ecosistema de bibliotecas (scikit-learn, numpy, pandas, keras, tensorflow), la legibilidad del código gracias a su simple sintaxis y su naturaleza multiparadigma y flexible. (Menasalvas, Rodríguez, Jiménez , & Duque, 2021). La versión utilizada en este trabajo corresponde a la 3.10.

Adicionalmente se usará Jupyter Notebook que es una interfaz web de código abierto que permite la inclusión de texto, video, audio, imágenes, así como la ejecución de código a través del navegador en múltiples lenguajes. (Cabrera Granado & Díaz García)

También se utilizará la librería Sci-kit learn que permite la elaboración de modelos para aprendizaje supervisado como no supervisado y cuenta con herramientas que permiten analizar los resultados obtenidos.

Para realizar los respectivos gráficos se empleará la librería Matplotlib que permite crear distintos tipos de figuras y gráficos para facilitar la comprensión de los resultados.

### 3.3 Descripción de los datos

Los datos se encuentran almacenados en un archivo txt, con un total de 167562 registros con **16** campos, que se detallan a continuación:

- **Descripción:** Este campo representa al nombre del producto que se va a realizar.
- **Lote Previsto:** Este campo es un número que se asigna a un conjunto de unidades de venta que lo identifica y confiere trazabilidad.
- **Fase:** Este campo representa a la etapa de fabricación del producto, esta puede ser: preparación (PR) o envase empaque
- **Cantidad planificada- Cabecera:** Este campo es un número que representa la cantidad teórica que se desea fabricar de un producto.
- **Cantidad completada:** Este campo es un número que representa la cantidad real que se fabricó de un producto.
- **Fecha Emisión**: Este campo representa la fecha en la que una orden se crea.
- **Fecha Cierre:** Este campo representa la fecha en la que una orden se terminó de manufacturar.
- **Estado:** Situación en la que se encuentra la orden**.** Se cuenta con tres estados: Cerrada, Planificada y Liberada
- **Número de artículo:** Este campo es un número que identifica a cada uno de los materiales o recursos que se van a utilizar en la elaboración de un producto.
- **Descripción de Articulo:** Este campo contiene el nombre de las materias primas y el recurso asignado en cada fase de manufactura
- **Cantidad planificada – Filas:** Este campo es un número que contiene la cantidad teórica de cada materia prima y recurso asignado en cada fase de manufactura.

- **Cantidad suministrada:** Este campo es un número que contiene la cantidad real de cada materia prima y recurso asignado en cada fase de manufactura.
- **Variación:** Este campo contiene la diferencia de la cantidad planifica y la cantidad suministrada.
- **%Variación:** Este campo contiene el porcentaje de la variación.
- **Cantidad adicional:** Este campo contiene la cantidad de recurso adicional que se usó en el proceso de manufactura de un producto.
- **%Adicional:** Este campo contiene el porcentaje de la cantidad adicional.

## 3.4 Preparación de los Datos

### 3.4.1 Seleccionar los Datos

Se va a utilizar el número de registros que se encuentran dentro de los atributos que se detallan a continuación. Los atributos seleccionados son:

- Descripción
- Fase
- Estado
- Descripción de articulo
- Variación
- % Variación

Los atributos de los cuales se va a prescindir dentro de proyecto son:

- Lote Previsto
- Cantidad planificada-Cabecera
- Cantidad Completada
- Cantidad adicional
- %Adicional

Estos atributos no aportan con ninguna información en este proceso de minería de datos por lo que se va a prescindir de ellos.

Adicionalmente las siguientes columnas se mantendrán dentro de la base de datos como posible fuente de información adicional en el proceso de minería de datos.

- Cantidad planificada – Filas
- Cantidad Suministrada
- Fecha de Emisión
- Fecha cierre

### 3.4.2 Limpieza de datos

Al analizar la base de datos se decidió tomar en cuenta las siguientes consideraciones dentro de la base de datos para poder cumplir con nuestros objetivos.

- Al empezar a trabajar con el atributo Descripción, se determinó que existían dentro de la base de datos producto que se fabricaban para uso interno por lo que estos productos fueron eliminados de la base datos, ya que únicamente nos vamos a centrar en los productos comercializables de la empresa.
- Dentro de la columna descripción de Articulo contamos con registros que se refieren la uso de maquina y CIF, estos registros son eliminados porque no van a ser empleados para cumplir con nuestros objetivos.
- En el Atributo Estado contamos con las siguientes opciones: cerrada, planificada, liberada y registros vacíos. Al revisar estos datos, se observó que los registros que corresponden a planificada, liberada y los registros vacíos no contaban con cantidades planificadas y suministradas que correspondan a la mano de obra. Es decir que estas órdenes no se llevaron a cabo y esta fue la razón para que estos registros fueran eliminados.
- Para poder seleccionar únicamente los productos del año 2021 y 2022 fue necesario separar la fecha del campo emisión para poder seleccionar únicamente los datos que corresponden a esos años para poder realizar nuestro trabajo de investigación

- Una vez que ya contamos con la base de datos que vamos a usar, es necesario separa los datos por proceso de manufactura para poder evaluar cada uno por separado.

## 3.5 Análisis del porcentaje de Variación

El proceso de manufactura en el cual su desviación estándar es mayor que su promedio dentro del atributo %Variación se considerara que el uso de mano de obra dentro de este proceso se encuentra disperso es decir la asignación de recurso no se encuentra controlada.
Los procesos de manufactura son los siguientes:

| **Procesos de Manufactura** |
|---|
| Pesaje |
| Preparación 2 |
| Reacondicionado |

Tabla 1 : Procesos de Manufactura

Se procede a revisar por proceso la variación de recurso tiempo de mano de obra, utilizando la función describe () que proporciona estadísticas descriptivas incluyendo aquellas que resumen la tendencia central, dispersión y la forma de distribución de los datos. (Chacón, 2023)

- **Pesaje**

```
In [14]: dpesajef.describe()
```

Out[14]:

| | Cantidad planificada - Cabecera | Cantidad completada | Cantidad planificada - Filas | Cantidad suministrada | Variacion | %Variacion |
|---|---|---|---|---|---|---|
| count | 3.602000e+03 | 3.602000e+03 | 3602.000000 | 3602.000000 | 3602.000000 | 3602.000000 |
| mean | 5.519875e+05 | 5.508757e+05 | 2.630109 | 3.224506 | -0.594397 | -23.399031 |
| std | 9.921626e+05 | 9.919082e+05 | 0.998358 | 1.971417 | 1.462985 | 55.326881 |
| min | 1.087200e+02 | 1.087200e+02 | 0.500000 | 0.100000 | -9.145000 | -488.230000 |
| 25% | 1.431938e+04 | 1.425938e+04 | 1.825000 | 2.170000 | -0.850000 | -40.000000 |
| 50% | 7.527600e+04 | 7.480800e+04 | 2.375000 | 2.500000 | -0.295000 | -9.370000 |
| 75% | 4.500000e+05 | 4.420000e+05 | 3.200000 | 3.500000 | 0.205000 | 8.630000 |
| max | 1.250000e+07 | 1.249780e+07 | 6.167000 | 13.170000 | 4.420000 | 96.870000 |

Gráfico 1: Estadísticas descriptivas del %Variación en el proceso Pesaje

En el gráfico 1, se observa que existe mucha dispersión de los datos dentro del atributo %Variación, porque la desviación estándar es mayor a la media dentro del atributo.

Al revisar los cuartiles, se determina que el rango Inter cuartil es de 31.37, es un valor que no engloba ni el 50% del total de los datos. Adicionalmente también se observa que existen datos se encuentran como valores extremos (outliers), lo que nos indica que si existe variación en la asignación del recurso tiempo de mano de obra dentro de este proceso.

```
In [16]: dpesajef= dpesajef['%Variacion']

In [22]: dpesajef.plot(kind='box',figsize=(8,6))

         plt.title('Box plot of mopesajef')
         plt.ylabel('% Variacion')

         plt.show()
```

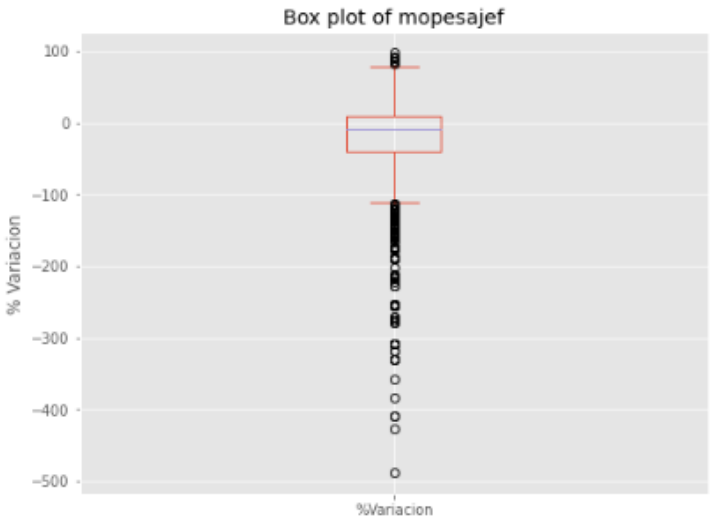


Gráfico 2: Diagrama de caja para %Variación en el proceso Pesaje

- **Preparación 2**

```
In [7]: dprep2.describe()
```

Out[7]:

| | Cantidad planificada - Cabecera | Cantidad completada | Cantidad planificada - Filas | Cantidad suministrada | Variacion | %Variacion |
|---|---|---|---|---|---|---|
| count | 3.940000e+02 | 3.940000e+02 | 394.000000 | 394.000000 | 394.000000 | 394.000000 |
| mean | 2.735347e+06 | 2.733970e+06 | 27.056079 | 7.201508 | 19.854571 | 72.124619 |
| std | 6.638773e+05 | 6.640650e+05 | 5.685775 | 2.082938 | 5.255683 | 12.121377 |
| min | 1.384300e+03 | 1.377580e+03 | 1.011074 | 0.680000 | -1.000000 | -23.630000 |
| 25% | 2.846250e+06 | 2.845820e+06 | 27.770000 | 6.750000 | 19.815000 | 71.620000 |
| 50% | 2.846250e+06 | 2.846020e+06 | 27.770000 | 7.250000 | 20.440000 | 73.640000 |
| 75% | 2.851875e+06 | 2.851445e+06 | 27.815000 | 7.810000 | 20.815000 | 74.830000 |
| max | 1.250000e+07 | 1.249780e+07 | 105.000000 | 26.500000 | 89.500000 | 89.190000 |

Gráfico 3: Estadísticas descriptivas del %Variación en el proceso Preparación2

En el gráfico 3, se observa que existe mucha dispersión de los datos dentro del atributo %Variación, porque el rango intercuartil es de 3.21, es un valor que no engloba ni el 50% del total de los datos. Adicionalmente también se observa que los

datos se encuentran como valores extremos (outliers), lo que nos indica que si existe variación en la asignación del recurso tiempo de mano de obra dentro de este proceso.

```
In [22]: dfpre2= dprep2['%Variacion']

In [23]: dfpre2.plot(kind='box',figsize=(8,6))

         plt.title('Box plot of mopreparacio 2')
         plt.ylabel('% Variacion')

         plt.show()
```

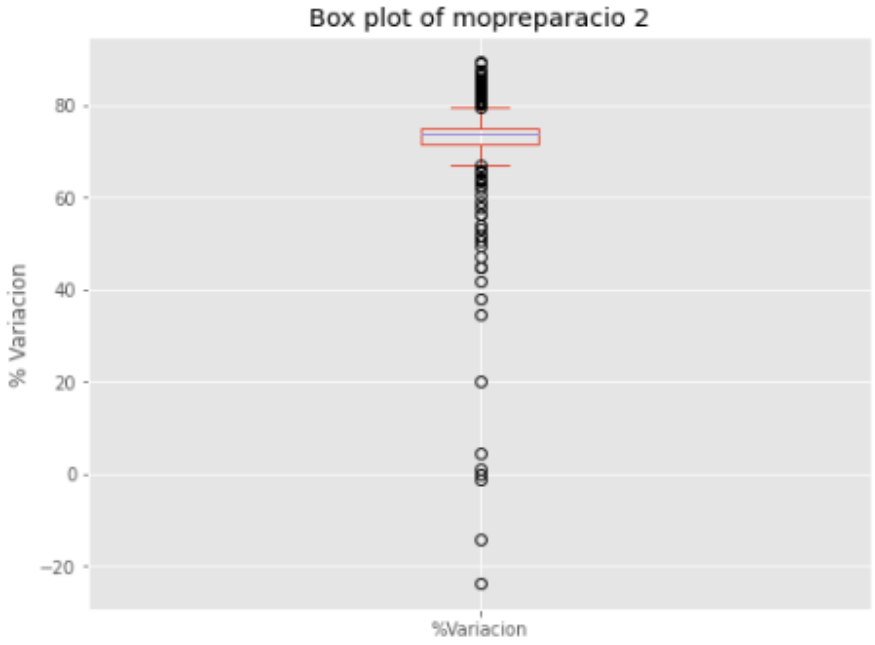


*Gráfico 4:* Diagrama de caja para %Variación en el proceso Preparación 2

- **Recubrimiento**

```
In [8]: drec.describe()
```

Out[8]:

| | Cantidad planificada - Cabecera | Cantidad completada | Cantidad planificada - Filas | Cantidad suministrada | Variacion | %Variacion |
|---|---|---|---|---|---|---|
| **count** | 202.000000 | 202.000000 | 202.000000 | 202.000000 | 202.000000 | 202.000000 |
| **mean** | 135752.741634 | 132633.420149 | 13.427057 | 10.806906 | 2.620151 | 12.879604 |
| **std** | 194820.161720 | 191616.750507 | 7.364675 | 6.913395 | 6.192116 | 42.115949 |
| **min** | 1232.400000 | 1232.400000 | 2.089000 | 1.900000 | -10.800001 | -132.240000 |
| **25%** | 19698.120000 | 19025.000000 | 6.754026 | 6.500000 | -1.500801 | -14.150000 |
| **50%** | 55574.550000 | 55050.000000 | 11.388200 | 9.500000 | 1.744009 | 17.150000 |
| **75%** | 149040.000000 | 143125.000000 | 19.400000 | 13.000000 | 4.519500 | 47.495000 |
| **max** | 900007.200000 | 897100.000000 | 31.356799 | 35.500000 | 22.079999 | 81.530000 |

Gráfico 5: Estadísticas descriptivas del %Variación en el proceso Reacondicionado

En el gráfico 5, se observa que existe dispersión de los datos dentro del atributo %Variación, porque la desviación estándar es mayor que la media dentro del atributo. Adicionalmente al revisar el rango Inter cuartil es de 33.34, es un valor que, si engloba a los datos, pero también existen valores extremos (outliers), lo que nos indica que si existe variación en la asignación del recurso tiempo de mano de obra dentro de este proceso.

```
In [10]: drec= drec['%Variacion']

In [11]: drec.plot(kind='box',figsize=(8,6))

         plt.title('Box plot of mopesajef')
         plt.ylabel('% Variacion')

         plt.show()
```


Gráfico 6: Diagrama de caja para %Variación en el proceso Reacondicionado

## 3.6 Algoritmo K- means

El algoritmo K means, tiene la función de agrupar grandes conjuntos de datos en k clusters y los clusters generados son independientes. Con estas características del modelo podemos agrupar los datos en función del porcentaje de variación y la cantidad de la orden ya que este algoritmo únicamente trabaja con valores numéricos.

### 3.6.1 Generación del plan de prueba

La métrica que se van a utilizar para probar la validez del modelo son métricas de valoración interna, en donde su objetivo es evaluar el cluster formado usando solo cantidades y características inherentes al conjunto de datos. (Pastrán Ramírez & Gongora Aya, 2021). Las métricas que se van a usar son las siguientes:

- Coeficiente de Silueta: es una métrica para evaluar la calidad del agrupamiento obtenido con algoritmos de clustering. Es un indicador del número ideal de clusters. El coeficiente de Silueta es un valor que varía entre -1 y 1.

### 3.6.2 Construcción del modelo

Plantear un modelo para clasificar los productos en cada proceso con el fin de conocer cuáles son los que están demandando mayor cantidad de mano de obra que la cantidad estándar.

En este caso se plantea los siguientes parámetros:

**n_clusters:** el número de clúster que se van a formar, así como el número de centroides para generar

**int:** método de iniciación

**randon_state:** Determina la generación de números aleatorios para la inicialización del centroide.

| Parámetro | Valor |
|---|---|
| n_cluster | El número se obtendrá en función del método del codo |
| int | k-means ++ |
| random_state | 0 |

Tabla 2: Parámetros del algoritmo K- means

El número de clusters se determinará a partir del método del codo en el que se evidencia claramente cuál es el número optimo a través de su gráfica.

El parámetro int elegido es k-means ++ en el que se selecciona los centroides de clúster iniciales utilizando un muestreo basado en una distribución empírica de probabilidad de la contribución de los puntos a la inercia general.

El parámetro random_state elegido es cero

- **PESAJE**

Para aplicar el logaritmo K means al proceso de pesaje se realizó la normalización usando MinMaxScaler y se calculó el número de clusters usando el método del codo.

*Cálculo del número de clusters*

```
# Graficamos el número de clusters y la distancia
plt.plot(range(2,20), objective_function, marker='o')
plt.title('El Método del Codo')
plt.xlabel('Número de K clusters')
plt.ylabel('WCSS')
plt.show()
```

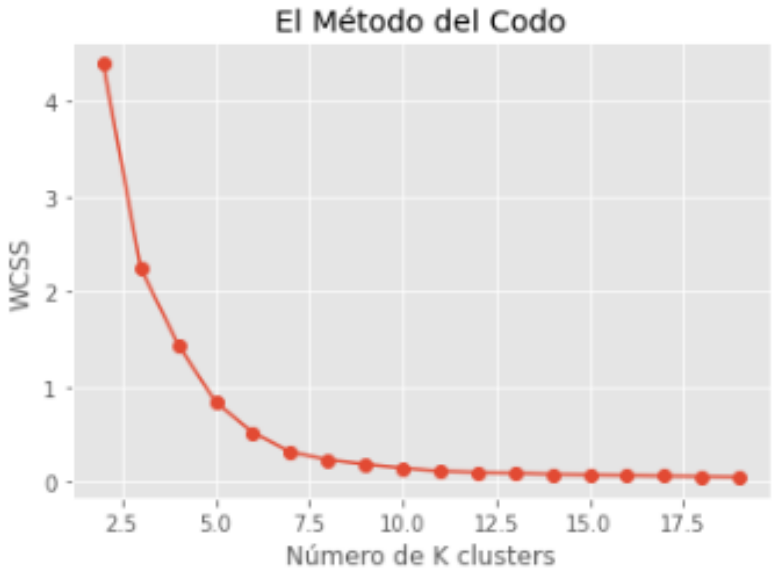


*Gráfico 7: Número de clúster para el proceso Pesaje*

En el gráfico 7 se evidencia que la línea continua presenta un cambio brusco en el número tres de la escala de las abscisas, esto nos lleva a elegir este valor como el número de clúster para este proceso.

*Entrenamiento del modelo y cálculo del número de centroides*

```
# Entrenamos al modelo con el número óptimo de clusters, en este caso es 3
tuned_clustering=KMeans(n_clusters=3,init='k-means++',random_state=0)
labels=tuned_clustering.fit_predict(clus)

# Los centroides calculados son
tuned_clustering.cluster_centers_[:]
```

```
array([[0.99079965, 0.22012409],
       [0.98942755, 0.01146783],
       [0.14345484, 0.06693468]])
```

*Gráfico 8: Entrenamiento del modelo para el proceso Pesaje*

*Gráfico de los clúster formados*

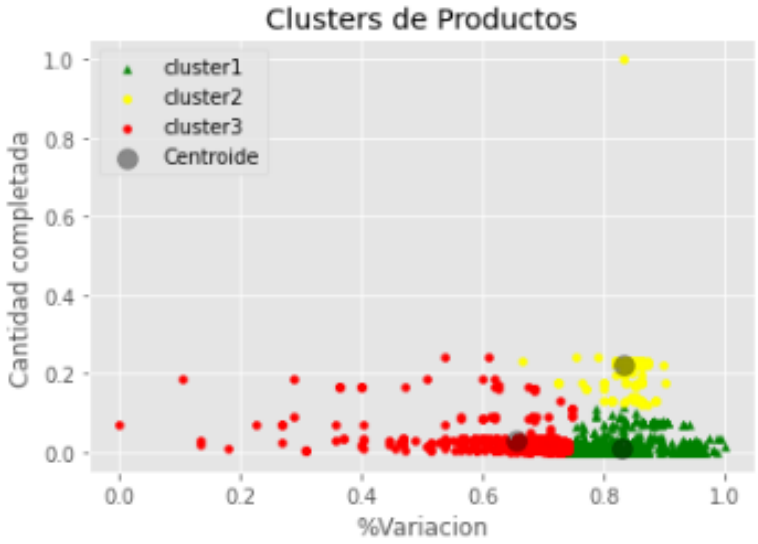


*Gráfico 9: Número de clusters formados para el proceso Pesaje*

*Evaluación del algoritmo*

```
In [68]: from sklearn import metrics
         metrics.silhouette_score(clus,tuned_clustering.labels_,metric='euclidean')
```

```
Out[68]: 0.6111527324383618
```

*Gráfico 10: Evaluación del modelo usando el coeficiente silueta para el proceso Pesaje usando K- means*

- **PREPARACIÓN 2**

Para aplicar el logaritmo K means al proceso de pesaje se realizó la normalización usando MinMaxScaler y se calculó el número de clusters usando el método del codo.

*Cálculo del número de clusters*

```
# Graficamos el número de clusters y la distancia
plt.plot(range(2,20), objective_function, marker='o')
plt.title('El Método del Codo')
plt.xlabel('Número de K clusters')
plt.ylabel('WCSS')
plt.show()
```

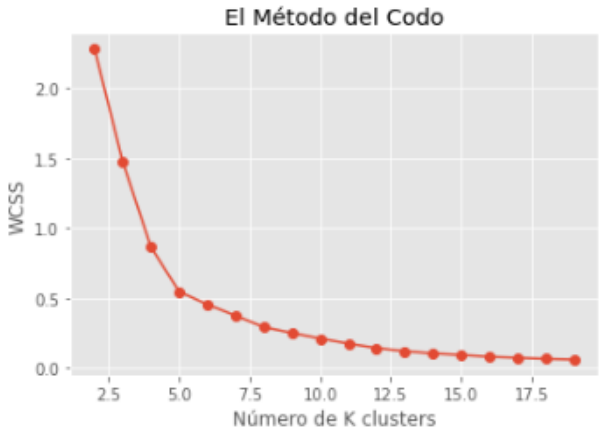


*Gráfico 11: Número de clusters para el proceso Preparación 2*

En el gráfico 11 se evidencia que la línea continua presenta un cambio brusco en el número tres de la escala de las abscisas, esto nos lleva a elegir este valor como el número de clusters para este proceso.

*Entrenamiento del modelo y cálculo del número de centroides*

```
In [35]: # Entrenamos al modelo con el número óptimo de clusters, en este caso es 3
tuned_clustering=KMeans(n_clusters=3,init='k-means++',random_state=0)
labels=tuned_clustering.fit_predict(clus)

# Los centroides calculados son
tuned_clustering.cluster_centers_[:]

Out[35]: array([[0.87119774, 0.22555509],
       [0.19264568, 0.05121704],
       [0.66623382, 0.15093733]])
```

*Gráfico 12: Entrenamiento del modelo para el proceso Preparación 2*

*Gráfico de los clusters formados*

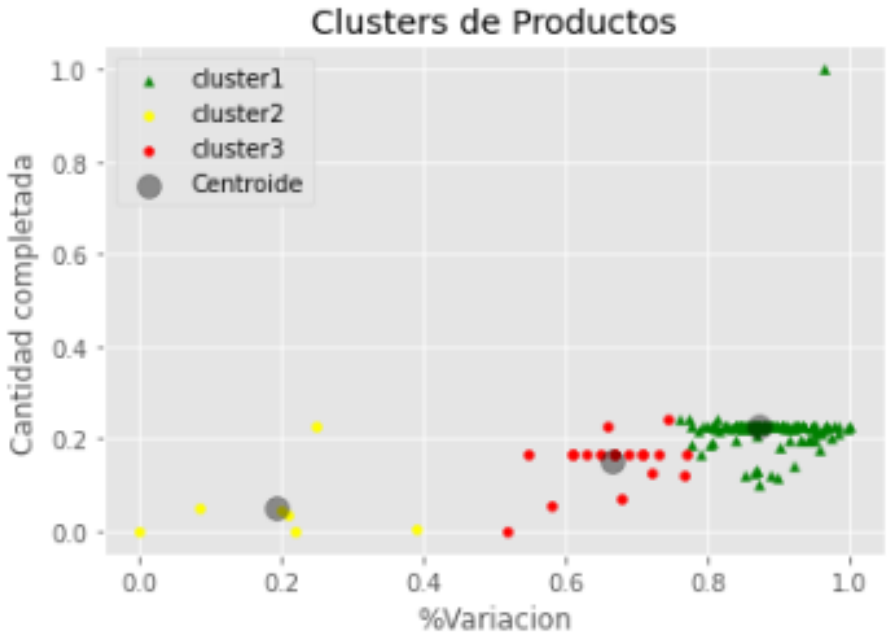


*Gráfico 13: Número de clusters formados para el proceso Preparación 2*

*Evaluación del algoritmo*

```
In [40]: from sklearn import metrics
         metrics.silhouette_score(clus,tuned_clustering.labels_,metric='euclidean')

Out[40]: 0.7810400128902071
```

*Gráfico 14: Evaluación del modelo usando el coeficiente silueta para el proceso Preparación 2 usando K-means*

- **REACONDICIONADO**

Para aplicar el logaritmo K means al proceso de pesaje se realizó la normalización usando MinMaxScaler y se calculó el número de clusters usando el método del codo.

*Cálculo del número de clusters*

```
# Graficamos el número de clusters y la distancia
plt.plot(range(2,20), objective_function,marker='o')
plt.title('El Método del Codo')
plt.xlabel('Número de K clusters')
plt.ylabel('WCSS')
plt.show()
```


*Gráfico 15: Número de clusters para el proceso Reacondicionado*

En el gráfico 15 se evidencia que la línea continua presenta un cambio brusco en el número cinco de la escala de las abscisas, esto nos lleva a elegir este valor como el número de clusters para este proceso.

*Entrenamiento del modelo y cálculo del número de centroides*

In [28]:
```
# Entrenamos al modelo con el número óptimo de clusters, en este caso es 5
tuned_clustering=KMeans(n_clusters=5,init='k-means++',random_state=0)
labels=tuned_clustering.fit_predict(clus)

# Los centroides calculados son
tuned_clustering.cluster_centers_[:]
```

Out[28]:
```
array([[0.61917433, 0.07471745],
       [0.80692687, 0.35137465],
       [0.33176383, 0.01887374],
       [0.51829069, 0.98826557],
       [0.87179926, 0.04533215]])
```

*Gráfico 16: Entrenamiento del modelo para el proceso Reacondicionado*

*Gráfico de los clusters formados*

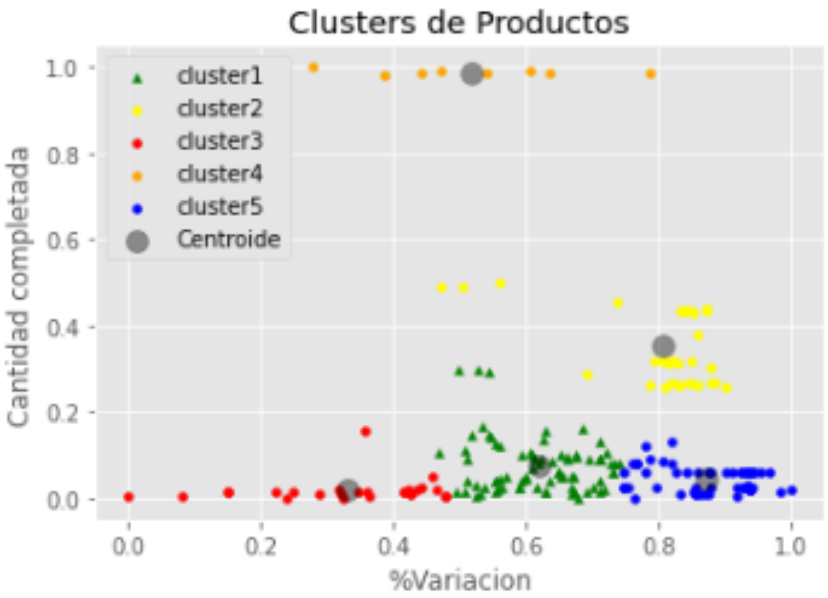


*Gráfico 17: Número de clusters formados para el proceso Reacondicionado*

*Evaluación del algoritmo*

```
In [33]: from sklearn import metrics
         metrics.silhouette_score(clus,tuned_clustering.labels_,metric='euclidean')

Out[33]: 0.5184840849270349
```

*Gráfico 18: Evaluación del modelo usando el coeficiente silueta para el proceso Reacondicionado usando K- means*

## 3.7 Algoritmo DBSCAN

DBSCAN es un algoritmo representativo de clustering basado en la densidad, que define un cluster como el conjunto más grande de puntos densamente conectados (Zhang, 2022). El algoritmo divide la región con cierta densidad en clústeres y considera los clusters como regiones densas separadas por regiones dispersar en el espacio de datos (Zhang, 2022). Este algoritmo puede extraer eficazmente formas arbitrarias de clusters e identificar correctamente valores atípicos en bases de datos ruidosas. (Zhang, 2022)

### 3.7.1 Generación del plan de prueba

La métrica que se van a utilizar para probar la validez del modelo son métricas de valoración interna, en donde su objetivo es evaluar el cluster formado usando solo cantidades y características inherentes al conjunto de datos. (Pastrán Ramírez & Gongora Aya, 2021). La métrica que se va a usar es las siguiente:

- Coeficiente de Silueta: es una métrica para evaluar la calidad del agrupamiento obtenido con algoritmos de clustering. Es un indicador del número ideal de clusters. El coeficiente de Silueta es un valor que varía entre -1 y 1.

### 3.7.2 Construcción del modelo

Plantear un modelo para clasificar los productos en cada proceso con el fin de conocer cuáles son los que están demandando mayor cantidad de mano de obra que la cantidad estándar.
En este caso se plantea los siguientes parámetros:
**Eps:** radio del cluster.
**MinPts:** número mínimo de puntos necesarios dentro del clúster

| Parámetro | Valor |
|---|---|
| Eps | Se toma el valor que indica la gráfica épsilon vs distancia |
| MinPts | 4 |

*Tabla 3*: Parámetros del algoritmo DBSCAN

El valor de eps se determina a partir de la gráfica épsilon vs distancia a partir del método de “knee” en donde un cambio agudo en la curva indica un cambio en la densidad de los puntos.

El valor de MinPts elegido fue 4 en base a bibliografía en donde se especifica que este valor asegura un buen compromiso entre el tamaño de los grupos y la cantidad de datos de ruido. (Starczewski, Goetzen, & Joo Er, 2020)

- **PESAJE**

Para aplicar el logaritmo DBSCAN al proceso de pesaje se calculó el número de eps usando el algoritmo nearest Neighbors y su posterior grafica.

*Determinación de eps*

```
#Algoritmo DBSCAN
# Calculamos la distancia entre puntos usando el algortimo NearestNeighbors
from sklearn.neighbors import NearestNeighbors
# calculando la distancia
neigh=NearestNeighbors(n_neighbors=2)

distance=neigh.fit(clusdb)

# indices y valores de distancia
distances,indices=distance.kneighbors(clusdb)

# Se deben ordenar las distancias en orden incremental
sorting_distances=np.sort(distances,axis=0)


# ordenando las distancias
sorted_distances=sorting_distances[:,1]
print(sorted_distances)
# gráfico entre la distancia vs epsilon
plt.plot(sorted_distances)
plt.xlabel('Distancia')

plt.ylabel('Epsilon')
plt.show()
```

```
[     0.              0.              0.         ...   51100.02415945
   51327.5991819  9498050.00091127]
```

*Gráfico 19: Calculo de la distancia entre puntos usando el algoritmo Nearest Neighbors para el proceso Pesaje*

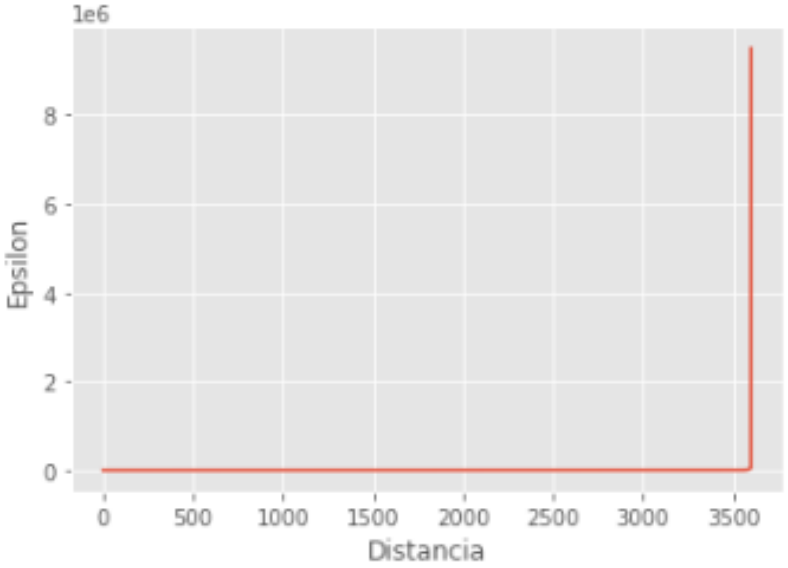


*Gráfico 20: Gráfico de épsilon vs Distancia para el proceso Pesaje*

En el gráfico 20, para el proceso Pesaje no se evidencia ningún cambio en la curva lo que indica que no existe un cambio de densidad en los puntos, es decir todos los puntos se encuentran juntos entre si por lo cual no se puede establecer eps.

- **PREPARACIÓN 2**

Para aplicar el logaritmo DBSCAN al proceso de preparación 2 se calculó el número de eps usando el algoritmo nearest Neighbors y su posterior gráfica.

*Determinación de eps*

```python
#Algoritmo DBSCAN
# Calculamos la distancia entre puntos usando el algortimo NearestNeighbors
from sklearn.neighbors import NearestNeighbors
# calculando la distancia

neigh=NearestNeighbors(n_neighbors=2)

distance=neigh.fit(clus)

# indices y valores de distancia
distances,indices=distance.kneighbors(clus)

# Se deben ordenar las distancias en orden incremental
sorting_distances=np.sort(distances,axis=0)

# ordenando las distancias
sorted_distances=sorting_distances[:,1]
print(sorted_distances)
# gráfico entre la distancia vs epsilon
plt.plot(sorted_distances)
plt.xlabel('Distancia')

plt.ylabel('Epsilon')
plt.show()
```

*Gráfico 21: Calculo de la distancia entre puntos usando el algoritmo Nearest Neighbors para el proceso Preparación 2*

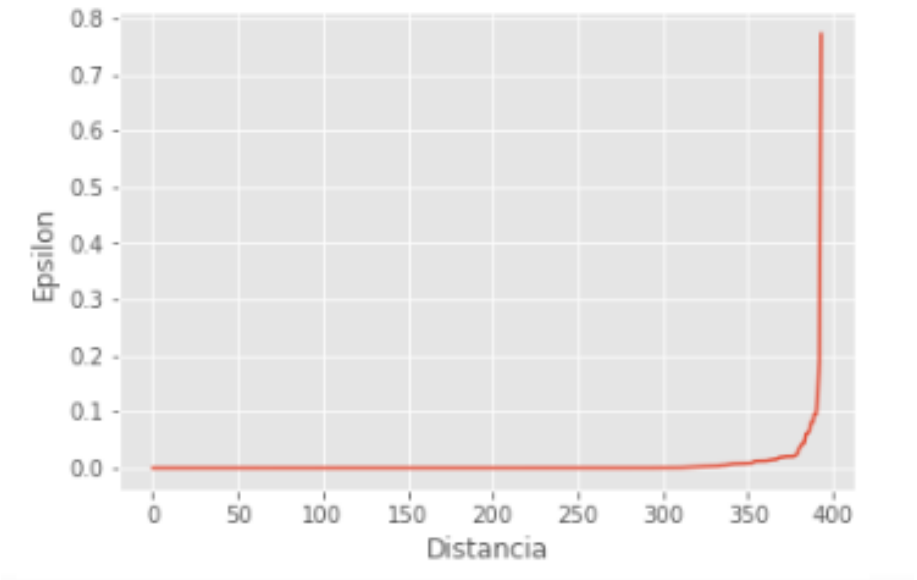


*Gráfico 22: Gráfico de épsilon vs Distancia para el proceso Preparación 2*

En el grafico 22, se evidencia en la curva un cambio en el eje de las ordenadas en el número 0.09, lo que nos indica que en ese punto existe una variación en la densidad de los puntos. Este valor es usado con eps para el proceso de Preparación 2.

*Aplicación del modelo DBSCAN*

```
# inicializando DBSCAN
from sklearn.cluster import DBSCAN

clustering_model=DBSCAN(eps=0.09,min_samples=4)

# fit the model to X
clustering_model.fit(clus)
predicted_labels=clustering_model.labels_
```

*Gráfico 23: Aplicación del modelo para el proceso Preparación 2.*

*Gráfico de los clusters formados*

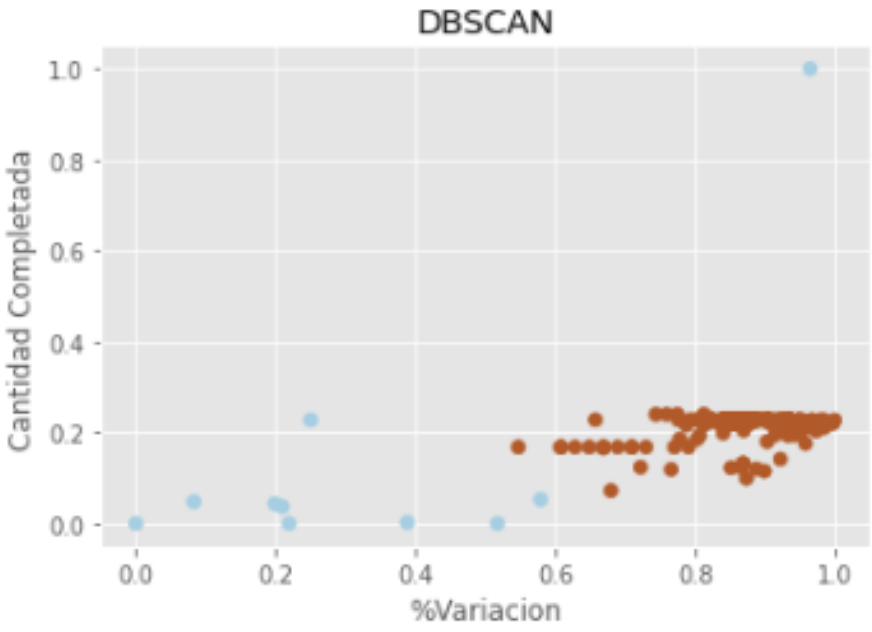


*Gráfico 24: Número de clusters formados para el proceso Preparación 2.*

*Evaluación del algoritmo*

```
In [56]: #Evaluando el modelo
         from sklearn import metrics
         metrics.silhouette_score(clus,predicted_labels)

Out[56]: 0.8921210371563442
```

*Gráfico 25: Evaluación del modelo usando el coeficiente silueta para el proceso Preparación 2 usando DBSCAN*

- **REACONDICIONADO**

*Determinación de eps*

```
#Algoritmo DBSCAN
# Calculamos la distancia entre puntos usando el algortimo NearestNeighbors
from sklearn.neighbors import NearestNeighbors
# calculando la distancia

neigh=NearestNeighbors(n_neighbors=2)

distance=neigh.fit(clus)

# indices y valores de distancia
distances,indices=distance.kneighbors(clus)

# Se deben ordenar las distancias en orden incremental
sorting_distances=np.sort(distances,axis=0)

# ordenando las distancias
sorted_distances=sorting_distances[:,1]
print(sorted_distances)
# gráfico entre la distancia vs epsilon
plt.plot(sorted_distances)
plt.xlabel('Distancia')

plt.ylabel('Epsilon')
plt.show()
```

*Gráfico 26: Calculo de la distancia entre puntos usando el algoritmo Nearest Neighbors para el proceso Reacondicionado.*

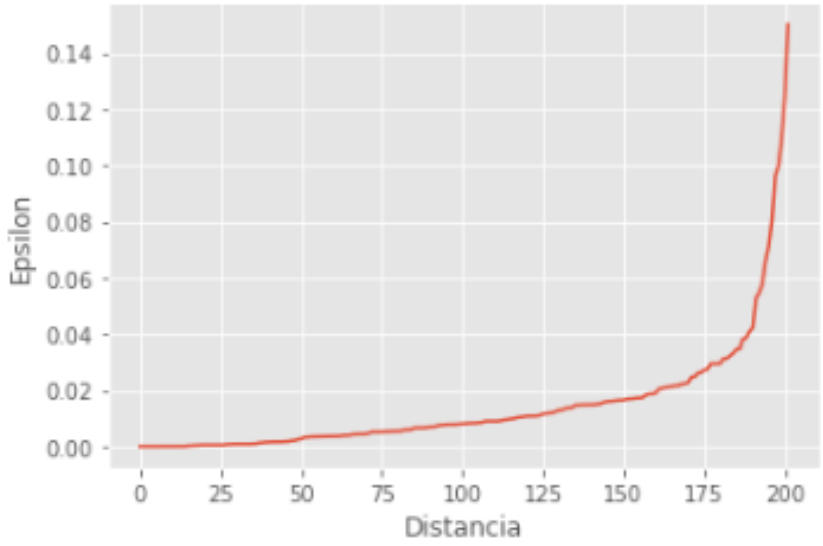


*Gráfico 27: Gráfico de épsilon vs Distancia para el proceso Reacondicionado.*

En el gráfico 27, se evidencia en la curva un cambio en el eje de las ordenadas en el número 0.099, lo que nos indica que en ese punto existe una variación en la densidad de los puntos. Este valor es usado con eps para el proceso de Reacondicionado.

*Aplicación del modelo DBSCAN*

In [59]:
```
# inicializando DBSCAN
from sklearn.cluster import DBSCAN

clustering_model=DBSCAN(eps=0.099,min_samples=4)

# fit the model to X
clustering_model.fit(clus)
predicted_labels=clustering_model.labels_
```

*Gráfico 28: Aplicación del modelo para el proceso Reacondicionado*

*Gráfico de los clúster formados*

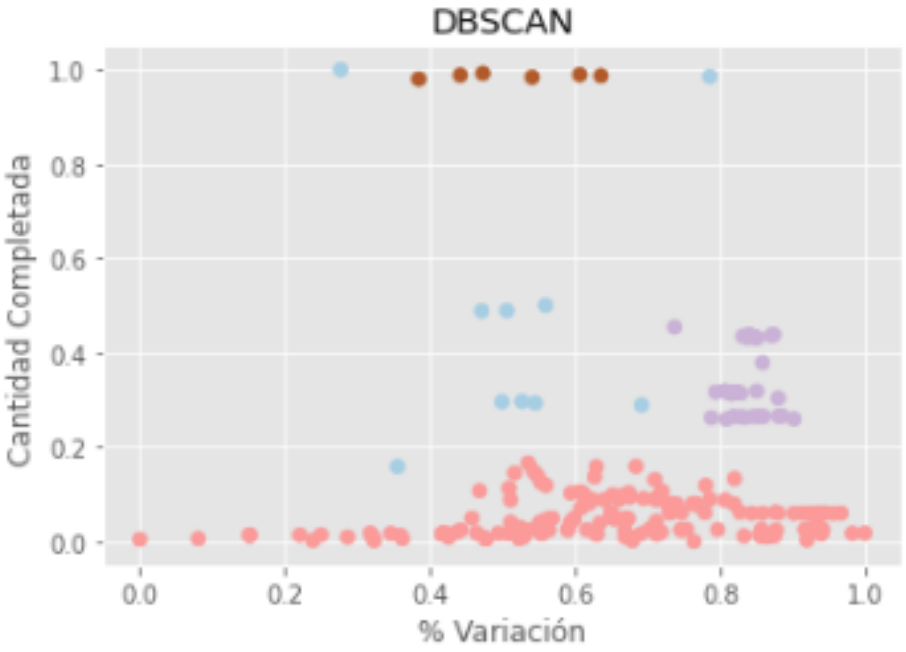


*Gráfico 29: Número de clúster formados para el proceso Reacondicionado.*

*Evaluación del algoritmo*

```
In [60]:  #Evaluando el modelo
          from sklearn import metrics
          metrics.silhouette_score(clus,predicted_labels)

Out[60]: 0.4004281958266067
```

*Gráfico 30: Evaluación del modelo usando el coeficiente silueta para el proceso Reacondicionado usando DBSCAN.*

## 3.8 Resultados

Después de emplear los dos algoritmos de aprendizaje no supervisado a los procesos de Pesaje, Preparación 2 y Reacondicionado se obtienen los siguientes resultados que se detallan en la tabla ## 2 y 3

| Proceso | n_clusters | int | Random_state | Coeficiente de Silueta |
|---|---|---|---|---|
| Pesaje | 3 | k-means ++ | 0 | 0.611 |
| Preparación 2 | 3 | k-means ++ | 0 | 0.781 |
| Reacondicionado | 5 | k-means ++ | 0 | 0.518 |

Tabla 4: Resultados obtenidos al emplear el algoritmo K- means en los tres procesos.

| Proceso | eps | MinPts | Coeficiente de Silueta |
|---|---|---|---|
| Pesaje | 0 | 4 | N/A |
| Preparación 2 | 0.09 | 4 | 0.892 |
| Reacondicionado | 0.099 | 4 | 0.400 |

Tabla 5: Resultados obtenidos al emplear el algoritmo DBSCAN en los tres procesos.

## 3.9 Análisis de los clúster

Para el proceso de pesaje y recubrimiento se usará los clúster formados por el algoritmo K-means y para el proceso Preparación 2 se usará los clúster formados por el algoritmo DBSCAN.

- PESAJE

Para el análisis de clúster se añade una columna adicional al dataset pesaje en donde se especifica a que clúster pertenece cada producto.

Adicionalmente se usa la función group by mediante la cual se puede separar cada uno de los clúster y determinar la siguiente información.

| Clúster | % Variación Min | % Variación Max | Producto |
|---|---|---|---|
| K1 | -53.84 | 96.87 | Saborizante y Sabores |
| K2 | -98.12 | 39.89 | Bebidas hidratantes |
| K3 | -488.23 | -51.19 | Suplementos Alimenticios |

Tabla 6: Información de los clúster formados en el proceso Pesaje

- PREPARACIÓN 2

Para el análisis de clúster se añade una columna adicional al dataset pesaje en donde se especifica a que cluster pertenece cada producto.

Adicionalmente se usa la función group by mediante la cual se puede separar cada uno de los clúster y determinar la siguiente información.

| Cluster | % Variación Min | % Variación Max | Producto |
|---|---|---|---|
| K1 | -23.63 | 85.23 | Sabor Naranja |
| K2 | 38.11 | 89.19 | Bebida Hidratante |

Tabla 7: Información de los clúster formados en el proceso Preparación 2

- REACONDICIONADO

Para el análisis de clúster se añade una columna adicional al dataset pesaje en donde se especifica a que clúster pertenece cada producto.

Adicionalmente se usa la función group by mediante la cual se puede separar cada uno de los clústeres y determinar la siguiente información.

| Clúster | % Variación Min | % Variación Max | Producto |
|---|---|---|---|
| K1 | -32.09 | 25.90 | Galletas con chips de chocolate |
| K2 | -31.47 | 60.62 | Galletas sabor vainilla |
| K3 | -132.24 | -30.29 | Galletas integrales |
| K4 | -73.04 | 35.81 | Galletas sabor vainilla |
| K5 | 27.45 | 81.53 | Galletas estándar |

Tabla 8:Información de los clúster formados en el proceso Reacondicionado

# CAPÍTULO IV

# 4. CONCLUSIONES Y RECOMENDACIONES

## 4.1 CONCLUSIONES

A partir del desarrollo del proyecto de investigación es posible establecer las siguientes conclusiones:

Dentro de una organización es fundamental conocer los procesos con los que se trabaja y la asignación de recursos a cada uno de ellos, esto permitirá tener un control de estos. Al analizar el porcentaje de variación del recurso mano de obra en los procesos se observó lo siguiente: el rango intercuartil de Pesaje y Preparación 2 es de 31.37 y 3.21 respectivamente, estos valores no engloban el 50% de los datos lo que evidencia que los datos se encuentran dispersos entre sí. El proceso Reacondicionado posee un rango intercuartil de 33.34 en el que se encuentran el 50% de los datos, pero en este proceso también se evidencian valores extremos (outliers) los mismos que generan una mayor desviación estándar dentro de los datos.

Los algoritmos de aprendizaje no supervisado empleados fueron K- means y DBSCAN, estos algoritmos permiten clasificar los datos en función de la similitud que posean estos datos entre sí. Se aplicó estos dos algoritmos a los tres procesos, pero el algoritmo K- means se ajustó mejor a Pesaje y Reacondicionado con un coeficiente de silueta de 0.611 y 0.518 respectivamente. Para el proceso Preparación 2 el algoritmo que mejor se ajusto fue DBSCAN con un coeficiente de silueta de 0.781.

Una vez realizada la clasificación de los datos con el algoritmo que mejor se ajusta se determinó que los productos que están demandando mayor cantidad de recurso tiempo de mano de obra son los siguientes: en Pesaje son los suplementos alimenticios, dentro de este grupo se tiene un porcentaje de variación del recurso de mano de obra asignado con un mínimo de -488 y un máximo de -51.19, estos valores negativos nos indican que se asignó mayor cantidad de mano de obra que la estándar por lo cual existe una perdida en el costo de este producto.

Dentro del proceso Preparación 2 el producto sabor naranja es el que está generando mayor cantidad de mano de obra, dentro de este grupo se tiene un

porcentaje de variación del recurso de mano de obra asignado con un mínimo de -23.63 y un máximo de 85.23, el valor negativo nos indica que se asignó mayor cantidad de mano de obra que la estándar por lo cual existe una perdida en el costo de este producto.

Para el proceso Reacondicionado el producto galletas integrales son las que están generando mayor cantidad de mano de obra que el valor estándar lo que aumenta el costo del producto, este grupo posee un porcentaje de variación del recurso de mano de obra asignado con un mínimo de -132.24 y un máximo de -30.29, el valor negativo nos indica que se asignó mayor cantidad de mano de obra que la estándar.

## 4.2 RECOMENDACIONES

- Se recomienda para futuras investigaciones tomar en cuenta si el recurso tiempo de mano de obra posee algún recargo adicional, es decir cuenta con un recargo al 25% o al 100% para conocer con más exactitud que productos son los que están aumentando su costo.
- Respecto a la evaluación de los algoritmos se pueden utilizar otras métricas de validación interna como Índice Davies-Bouldin que permite conocer la distancia de los datos al centro del grupo y el Índice Calinski-Harabasz que permite comparar dos clústeres ente si para conocer cuál de ellos tiene sus grupos mejor definidos. (Robledo Yage, 2019)
- Para conocer aún más a detalle las causas del incremento de recurso tiempo de mano de obra se recomienda hacer una nueva clusterización a partir de los grupos ya formados para conocer que lotes de los productos fueron los que generaron esta variación para así atacar los problemas de raíz.

## BIBLIOGRAFÍA


Aguirre Sajami , C. R., Barona Meza , C. M., & Dávila Dávila , G. (2020). La rentabilidad como herramienta para la toma de decisiones: análisis empirico en una empresa industrial . *Valor Contable* , 50-64.

Aguirre Sajamil , C. R., Barona Meza , C. M., & Dávila Dávila, G. (20 de Septiembre de 2020). La rentabilidad como herramienta para la toma de decisiones: análisis empírico en una empresa industrial. *Revista Valor Contable* , 50-64.

Arango Serna , M., Campuzano Zapata , L., & Zapata Cortes , J. (2015). Mejoramiento de procesos de manufactura utilizando Kanban. *Revista Ingenierías Universidad de Medellín*, 221-233.

Bojorque Chasi , R. X. (2020). *Clustering de sistemas de recomendación mediante técnicas de factorización matricial .* Madrid: Universidad Politécnica de Madrid.

Cabrera Granado , E., & Díaz García , E. (s.f.). *Manual de uso de Jupyter Notebook para aplicaciones docentes.* Madrid: Universidad Complutense de Madrid.

Carollo Limeres , M. (2011). Regresión Lineal.

Carrasquilla Batista , A., Chacón Rodríguez , A., Núñez Montero, K., Gómez Espinoza , O., Valverde , J., & Guerreo Barrantes, M. (2016). Regresión lineal simple y múltiple: aplicacion en la prediccions de variables naturales relacionadas con el crecimiento microalgal. *Tecnología en Marcha, Encuentro de Investigacion y Extension 2016*, 33-45.

Chacón , J. L. (05 de Mayo de 2023). *Introducción a Pandas, la librería de Python para trabajar con datos* . Obtenido de Profile: https://profile.es/blog/pandas-python/

Chakraborty, S., & Nagwani, N. (2011). Analysis and Study of incremental DBSCAN Clustering Algorithm. *International Journal of Enterprise Computing and Business* .

Chavez Valderrama , L., & Salinas Flores, J. W. (2021). Aplicacion del algoritmo K-medoid para la segmentacion de los alumnos ingresantes de una universidad . *Perfiles*, 24-29.

Chirivella Gonzáles, V. (s.f.). *Hipótesis en el modelo de regresión lineal por Mínimos Cuadrados Ordinarios* . Universidad Politecnica de Valencia .

Daza Izquierdo , J. (2016). Crecimiento y rentabilidad empresarial en el sector industrial brasileño . *Contaduría y Administración*, 266-282.

Gal, M., & Daniel L, R. (2019). *Data Standardization.* New York : New York University School of Law.

Gaonkar, M. N., & Sawant, K. (2013). AutoEpsDBSCAN: DBSCAN with Eps Automatic for Large Dataset. *International Journal on Advanced Computer Theory and Engineering*, 11-16.

Hernández Nariño , A., Medina León , A., Nogueira Rivera , D., Negrin Sosa, E., & Marqués León, M. (2014). La caracterización y clasificacion de sistemas, un paso necesario en la gestion y mejora de procesos.Particularidades en organizaciones hospitalarias. *DYNA*, 193-200.

Karami, A., & Johansson, R. (2014). Choosing DBSCAN Parameters Automatically using Differential Evolution. *International Journal of Computer Applications*, 1-11.

Lizcano Álvarez, J. (2004). *Rentabilidad Empresarial. Propuesta Práctica de Análisis y Evaluación .* Imprenta Modelo, S.L.

López Sánchez , V. (2019). *Aplicación y comparativa de cuatro modelos de clustering para datos GTEx.* Barcelona: Universitat Oberta Catalunya.

Menasalvas, E., Rodríguez, A., Jiménez , S., & Duque, S. (s.f.). *Newsletter Trimestral-Algoritmos de Machine Learning.* Obtenido de https://blogs.upm.es/catedra-idanae/wp-content/uploads/sites/698/2021/04/Idanae-1T21.pdf

Milton, J. (1994). *Estadística para Biologia y Ciencias de la Salud .* McGraw Hill.

Mohamad, I., & Usman, D. (2013). Standardization and Its Effects on K-Means Clustering Algorithm. *Research Journal of Applied Sciences, Engineering and Technology*, 3299-3303.

Orellana , L. (2008). *Regresión Lineal Simple.*

Pacheco Bautista , F. A. (2019). *Costos de Producción .* Ediciones Usta .

Pastrán Ramírez , L., & Gongora Aya , S. (2021). *Algoritmo de Selección y Validación del Método de Clusterización óptimo ora datos no supervisados.* Bogotá.

Pérez-Ortega , J., Hidalgo-Reyes, M., Castro-Sánchez, N., Pazos-Rangel, R., Díaz - Parra, O., Olivares-Peregrino, V., & Almanza-Ortega, N. (2018). Una heurística eficiente aplicada al algoritmo K-means para el agrupamiento de grandes instancias altamente agrupadas. *Computación y Sistemass* , 607-619.

Robledo Yage , F. (2019). *Clasificacion de Solanum lycopersicum y parientes silvestres mediante técnicas de aprendizaje automático partiendo de datos genéticos .* Valencia : Universitat Politécnica de Valéncia.

Schubert, E., Sander, J., Ester, M., Kriegel, H.-P., & Xu, X. (2017). DBSCAN Revisited, Revisited: Why and How you should (still) use DBSCAN. *ACM transactions on Databse Systems*.

Segovia Ortega , S. (2021). *Evaluacion de métodos de agrupamiento de perfiles de carga para gestion de demanda en clientes de media tensión*. Guayaquil.

Segovia Ortega , S. (2021). *Evaluación de métodos de agrupamiento de perfiles de carga para gestion de demanda en clientes de media tensión.* Guayaquil: Escuela Superior Politécnica del Litoral.

Starczewski, A., Goetzen, P., & Joo Er, M. (2020). A new method for automatic determining of the DBSCAN parameters. *JAISCR*, 209-221.

Yadav, A., & Dhingra, S. (2016). A review on K-means Clustering Technique. *International Journal of Latest Research in Science and Technology* , 13-16.

Zhang, Y. (2022). DBSCAN Clustering Algorithm Based on Big Data is Applied in Network Information Security Detection. *Hindawi*, 1-8.