UNIVERSIDAD NACIONAL DE SAN ANTONIO ABAD DEL CUSCO

FACULTAD DE INGENIERÍA ELÉCTRICA, ELECTRÓNICA, INFORMÁTICA Y MECÁNICA

ESCUELA PROFESIONAL DE INGENIERÍA INFORMÁTICA Y DE SISTEMAS

Proyecto de investigación semestral titulado:

"Algoritmo K-means con pyspark"

Asignatura:

Aprendizaje Automático

Docente:

MONTOYA CUBAS, Carlos Fernando

Estudiantes:

- **CASILLA PERCCA, Vladimir Dante** **174908**
- **INCA CRUZ, Carlos Eduardo** **174912**
- **HUAMAN HERMOZA, Antony Isaac** **170434**
- **PEREIRA CHINCHERO, Richard Mikhael** **171916**
- **QUISPE CHAMBILLA, Carlos Enrique** **174447**
- **QUISPE PALOMINO, Luiyi Antony** **174914**

Cusco – Perú
2021

1. Introducción

Un desafío grande en ciencias de la computación es llegar a diseñar programas de computadoras que sean capaces de aprender, dichos avances abren una amplia gama de nuevas aplicaciones.

El presente proyecto aborda la temática del análisis de datos en un entorno big data, a su vez,se implementa el algoritmo de k-means utilizando pyspark para la clasificación de imágenes que contienen números escritos, con el fin de que cualquier persona interesada tenga las nociones necesarias para implementar este ambiente utilizando las herramientas de Hadoop y Apache Spark. En cuanto al análisis de datos, se busca la construcción de un clasificador de imágenes mediante el método de K-means, el cual es un algoritmo de clustering que fue utilizado con el API que ofrece la librería Apache Spark para Python.

Cabe mencionar, que la base de datos utilizada fue descargada de Internet en formato CSV y al trabajar con ETL no se realizó un proceso de depuración, solo se utilizó para cargar el dataset a HDFS. Al no contar con los equipos necesarios, el proyecto se desarrolló de forma stand-alone y además, debido a las características del equipo, la base de datos utilizada no fue de gran tamaño. Los resultados obtenidos al utilizar K-means se muestran solamente por consola.

2. Estado del Arte

En la actualidad, existen numerosos estudios e investigaciones relacionados con el campo de la clasificación de imágenes utilizando diferentes técnicas como grafos, redes neuronales, K-means. En este trabajo nos centraremos en este último algoritmo que es un algoritmo de clasificación no supervisada (clusterización) que agrupa objetos en k grupos basándose en sus características.

Recientes estudios en el campo de la investigación de clasificación de imágenes muestran, que formular los problemas de K-means en tiempo continuo configurado de una manera correcta, evita una conversión predefinida y permite encontrar de manera más rápida unas soluciones globales a través de Convex Optimization.

Sin embargo, los algoritmos de K-means son definidos de la forma que se busca obtener un tiempo continuo, existen muchas propuestas para mejorar el rendimiento como la eficiencia, pero podemos afirmar basándonos en trabajos relacionados, que esto todavía inmerso en investigaciones y desarrollos, ya que existen otras técnicas que le hacen frente a este algoritmo.

Diseño de k-means inteligentes basados en spark para clustering de big data

El crecimiento de los datos nos ha llevado a la generación de grandes datos donde la cantidad de datos no se puede calcular utilizando un entorno convencional. Hay muchos entornos computacionales que se han desarrollado para calcular big data, uno de ellos es Hadoop, que tiene el sistema de archivos distribuidos y el marco MapReduce. Spark es un nuevo marco que se puede combinar con Hadoop y ejecutar sobre él. En este artículo, diseñamos k-means inteligentes basados en Spark para la agrupación de big data. Nuestro diseño utiliza un lote de datos en lugar de utilizar un conjunto de datos distribuido resistente (RDD) original. Comparamos nuestro diseño con la implementación que usa RDD original de datos. El resultado del experimento muestra que la implementación usando lotes de datos es más rápida que la implementación usando RDD original.

Paralelización de la agrupación en clústeres basada en K-Means en Spark

K-means es, de hecho, una familia de algoritmos de agrupamiento con diferentes funciones de distancia y una variedad de extensiones, por ejemplo, agrupamiento difuso y agrupamiento de consenso. Sin embargo, los algoritmos de agrupamiento basados en K-medias emplean el procedimiento iterativo de dos fases similar que incluye el cálculo de la distancia y la actualización de los centroides. Por lo tanto, explorar las implementaciones paralelas de este procedimiento iterativo de dos fases en Spark no solo es universal para una gran cantidad de algoritmos de agrupamiento, sino que también satisface las necesidades prácticas que abordan los grandes datos. Este documento contribuye a revelar los detalles de implementación para paralelizar el agrupamiento basado en K-means en Spark. En particular, primero presentamos el límite de la denominada agrupación en clústeres basada en K-means, y luego presentamos el marco paralelizable general en Spark. Discutimos la barrera técnica y sus estrategias alternativas para cada paso. Los resultados experimentales tanto en conjuntos de datos UCI a gran escala como en conjuntos de datos de texto demuestran la eficacia y eficiencia de nuestras implementaciones.

3. Marco Teórico

3.1 Clustering

El *aprendizaje de máquina* estudia el aprendizaje automático a partir de datos para conseguir hacer predicciones precisas a partir de observaciones con datos previos.
La clasificación automática de objetos o datos es uno de los objetivos del aprendizaje de máquina. Tenemos tres tipos de clasificacion clasificacion supervisada, semi supervisada y clasificación no supervisada, en este caso será clasificación no supervisada:

> En ***clasificación no supervisada*** los datos no tienen etiquetas (o no queremos utilizarlas) y estos se clasifican a partir de su estructura interna (propiedades, características).

3.2 Algoritmo K-means

K-means es un algoritmo de clasificación no supervisada (clusterización) que agrupa objetos en *k* grupos basándose en sus características. El agrupamiento se realiza minimizando la suma de distancias entre cada objeto y el centroide de su grupo o cluster. Se suele usar la distancia cuadrática.

El algoritmo consta de tres pasos.

***Inicialización*:** una vez escogido el número de grupos, *k*, se establecen *k*=centroides en el espacio de los datos, por ejemplo, escogiendo aleatoriamente.

***Asignación objetos a los centroides**:* cada objeto de los datos es asignado a su centroide más cercano.

***Actualización centroides**:* se actualiza la posición del centroide de cada grupo tomando como nuevo centroide la posición del promedio de los objetos pertenecientes a dicho grupo.

Se repiten los pasos 2 y 3 hasta que los centroides no se mueven, o se mueven por debajo de una distancia umbral en cada paso.
El algoritmo *k-means* resuelve un problema de optimización, siendo la función a optimizar (minimizar) la suma de las distancias cuadráticas de cada objeto al centroide de su cluster.

Los objetos de se representan con vectores reales de d dimensiones $(x_1, x_2, \dots, x_n)$ y el algoritmo k-means construye k grupos donde minimiza la suma de distancias de los objetos, dentro de cada grupo $S = \{S_1, S_2, \dots, S_k\}$, a su centroide. El problema se puede formular de la siguiente forma:

$$\min_{\mathbf{S}} E(\boldsymbol{\mu_i}) = \min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x}_j \in S_i} \|\mathbf{x}_j - \boldsymbol{\mu}_i\|^2$$

donde S es el conjunto de datos cuyos elementos son los objetos x_j representados por vectores, donde cada uno de sus elementos representa una característica o atributo. Tendremos k grupos o clusters con su correspondiente centroide μ_i.
En cada actualización de los centroides, desde el punto de vista matemático, imponemos la condición necesaria de extremo a la función $E(\mu_i)$ que, para la función cuadrática anterior es:

$$\frac{\partial E}{\partial \boldsymbol{\mu}_i} = 0 \implies \boldsymbol{\mu}_i^{(t+1)} = \frac{1}{\left|S_i^{(t)}\right|} \sum_{\mathbf{x}_j \in S_i^{(t)}} \mathbf{x}_j$$

y se toma el promedio de los elementos de cada grupo como nuevo centroide.

Las principales ventajas del método k-means son que es un método sencillo y rápido. Pero es necesario decidir el valor de k y el resultado final depende de la inicialización de los centroides. En principio no converge al mínimo global sino a un mínimo local.

3.2.1 Casos de uso del algoritmo k-means

Los algoritmos no supervisados de clustering como k-means pueden ser usados para encontrar grupos ocultos en los datos, o intuidos pero no etiquetados. Pueden servir para confirmar o descartar algún error sobre los datos. También son usados para descubrir relaciones entre los datos, que de manera manual no podríamos haber obtenido.

Una vez se ejecuta el algoritmo y obtenidos sus grupos o etiquetas, se puede pasar a un problema de aprendizaje supervisado. Es decir, asignando a cada grupo una clase distinta.

3.2.2 Elección de k

Aunque el algoritmo k-means pertenece a los algoritmos denominados como no supervisados, es necesario seleccionar un valor k del número de grupos en los que se agrupan los datos. En general, no hay una forma exacta de determinar el número de grupos, pero se pueden usar ciertas reglas o estadísticos que nos ayudan a estimar el número de grupos:

El método del codo:

La idea básica de los algoritmos de clustering es la minimización de la varianza intra-cluster y la maximización de la varianza inter-cluster. Es decir, queremos que cada observación se encuentre muy cerca a las de su mismo grupo y los grupos lo más lejos posible entre ellos.

El método del codo utiliza la distancia media de las observaciones a su centroide. Es decir, se fija en las distancias intra-cluster. Cuanto más grande es el número de clusters k, la varianza intra-cluster tiende a disminuir. Cuanto menor sea la distancia intra-cluster mejor, ya que significa que los clústers son más compactos. El método del codo busca el valor k que satisfaga que un incremento de k, no mejore sustancialmente la distancia media intra-cluster.

3.2.3 Elección de la distancia

La elección de la distancia para los problemas de clustering es de gran importancia ya que cambiar la medida de similaridad entre elementos impacta en el cálculo de los clústers.

Las distancias más clásicas usadas en algoritmos de clusters son la distancia euclidiana, la distancia manhattan y la distancia coseno

Distancia euclidiana

En general, la distancia euclidiana entre los puntos X=(x_1,x_2,...,x_n) y Y=(y_1,y_2 ,...,y_n), del espacio euclídeo n-dimensional, se define como:

$$d_{euc}(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

3.2.3.4 Desventajas del K-Means

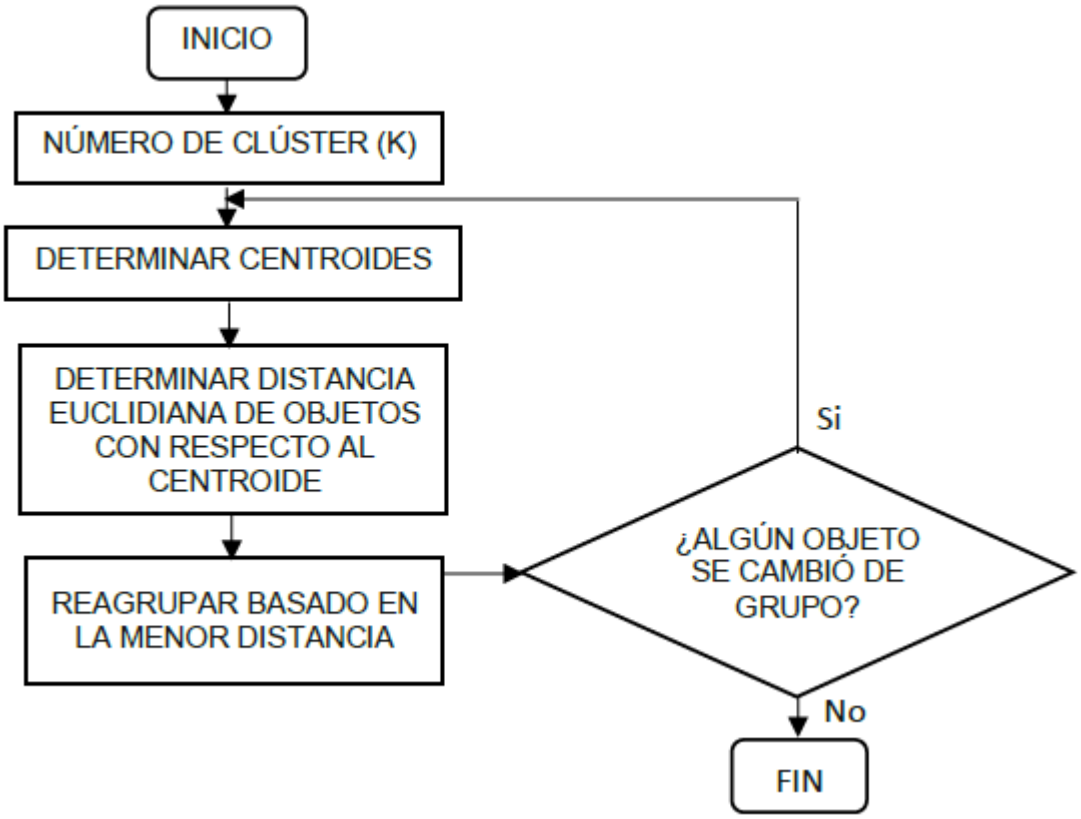
Ya hemos visto la potencia que tiene este algoritmo. Por lo sencillo que es de aplicar y la valiosa información sobre nuestros datos que nos aporta. Como no es oro todo lo que reluce, tengo que comentaros también las desventajas que ofrece:

- Tenemos que elegir "*k*" nosotros mismos. Es muy posible que nosotros cometamos un error, o que sea imposible escoger una k óptima.
- Es sensible a outliers. Los casos extremos hacen que el clúster se vea afectado. Aunque esto puede ser algo positivo a la hora de detectar anomalías.
- Es un algoritmo que sufre de la maldición de la dimensionalidad.

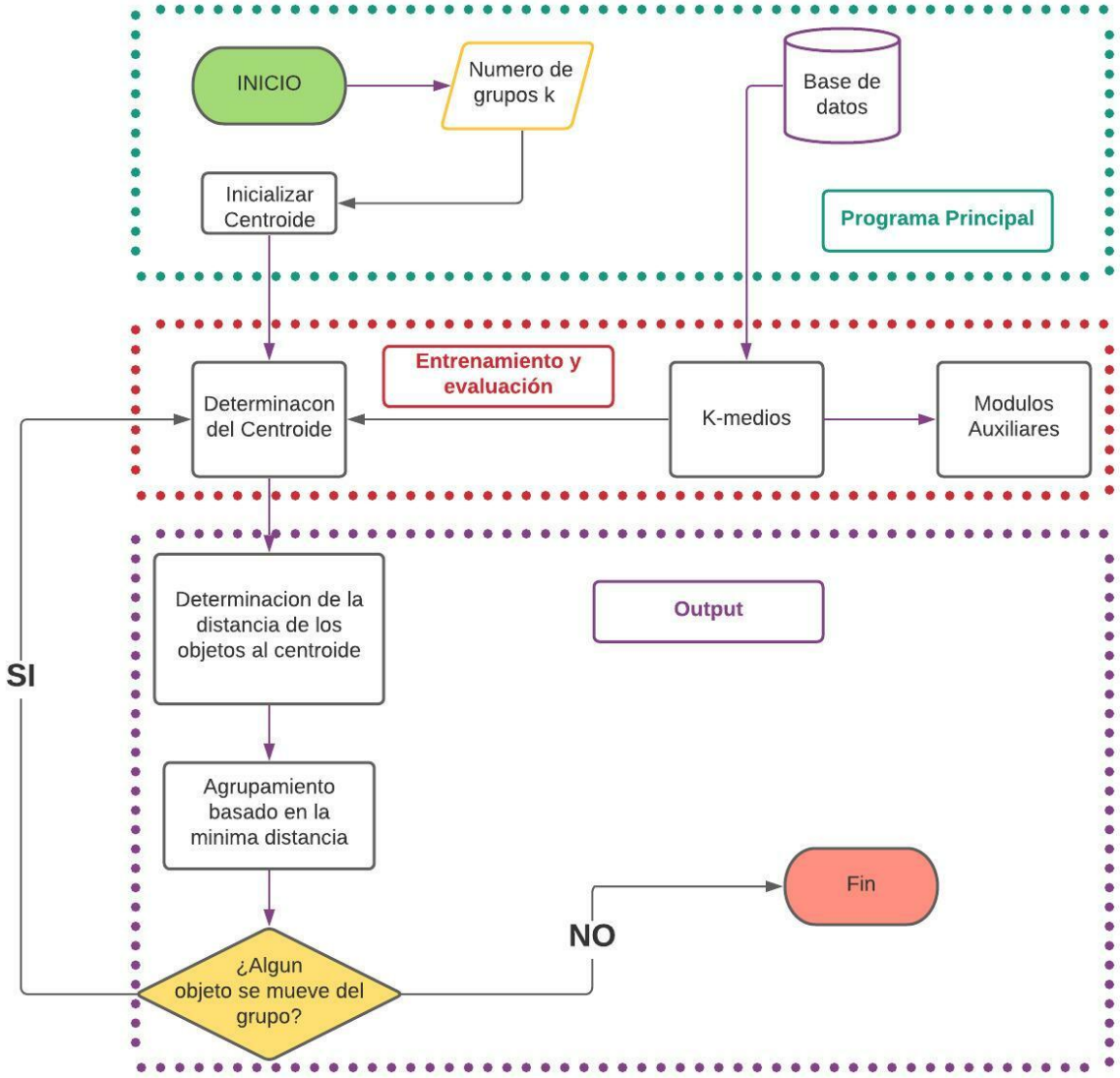
4. Implementación

Antes del proceso de la implementación se tiene que tomar en cuenta la estructura del algoritmo, en este caso hacemos una breve comparación de la estructura inicial del algoritmo k-means en **4.1.** con la estructura mejorada del algoritmo implementada en PySpark mostrado en **4.7 .**
La principal diferencia es el paralelismo la cual el algoritmo k means original no posee, se explicará a profundidad en items posteriores como se puede mejorar dicha implementacion

4.1.Diagrama de flujo:


INICIO
NÚMERO DE CLÚSTER (K)
DETERMINAR CENTROIDES
DETERMINAR DISTANCIA EUCLIDIANA DE OBJETOS CON RESPECTO AL CENTROIDE
REAGRUPAR BASADO EN LA MENOR DISTANCIA
¿ALGÚN OBJETO SE CAMBIÓ DE GRUPO?
Si
No
FIN

4.2 Diagrama de flujo del programa



4.3. Librerías utilizadas

-**sys**.- Utilizado para lectura de archivos

-**time**.- Utilizado para calcular el tiempo de convergencia

-**numpy**.- Utilizado para usar arreglos y normalizar vectores

-**google.colab.**- Utilizado para formatear tablas en google colab

https://pypi.org/project/numpy/

–**matplotlib**.- Utilizado para visualizar gráficos estadísticos

https://pypi.org/project/matplotlib/

-**Pandas**.- Utilizado para mostrar tablas.

https://pypi.org/project/pandas/

-**PIL**.- Utilizado para para visualizar archivos

https://pypi.org/project/Pillow/

-**pyspark**.- Spark es un motor de análisis unificado para el procesamiento de datos a gran escala.

https://pypi.org/project/pyspark/

4.4. Instalación de Pyspark en google colab

https://colab.research.google.com/

```
!pip install pyspark
from pyspark.sql import SparkSession
spark = SparkSession.builder.appName("PySpark en Google Colab").getOrCreate()
sc=spark.sparkContext
```

4.5. Implementar algoritmo k means

-eleccion(p).-Módulo que genera una muestra aleatoria de [0, len(p)),donde p[i] es la probabilidad asociada con i.

-kmeans_inicializacion(rdd, K).-Módulo que selecciona conjuntos `RUNS` de puntos iniciales para `K`-means

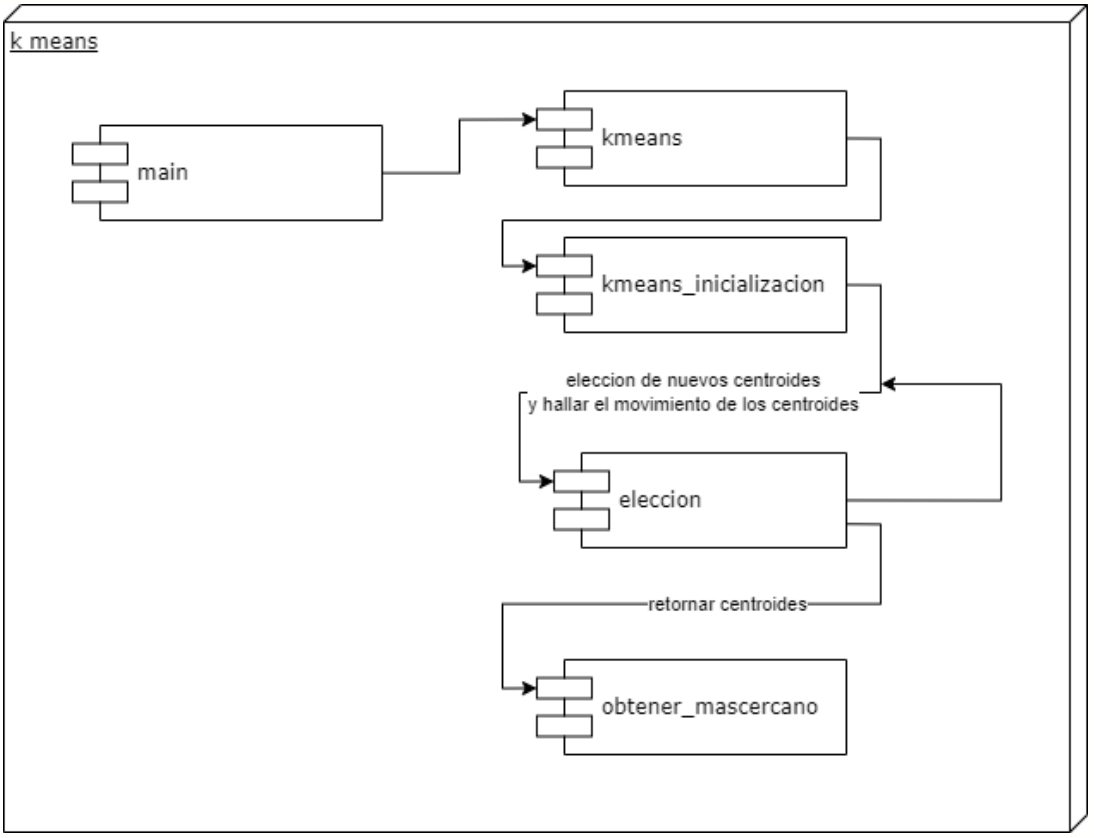
-obtener_mascercano(p, centers).-Módulo que devuelve los índices a los centroides más cercanos de 'p'.'centers' contiene conjuntos de centroides, donde 'centers[i]' es el i-ésimo conjunto de centroides.

-kmeans(rdd, K, converge_dist=0.1).-Módulo que ejecute el algoritmo K-means en 'rdd'

-Comprobar(palabra).-Módulo que comprueba si una cadena es número o texto

-kmeans_fit(rdd,K).-Módulo que devuelve un np.array con los clusters del rdd.

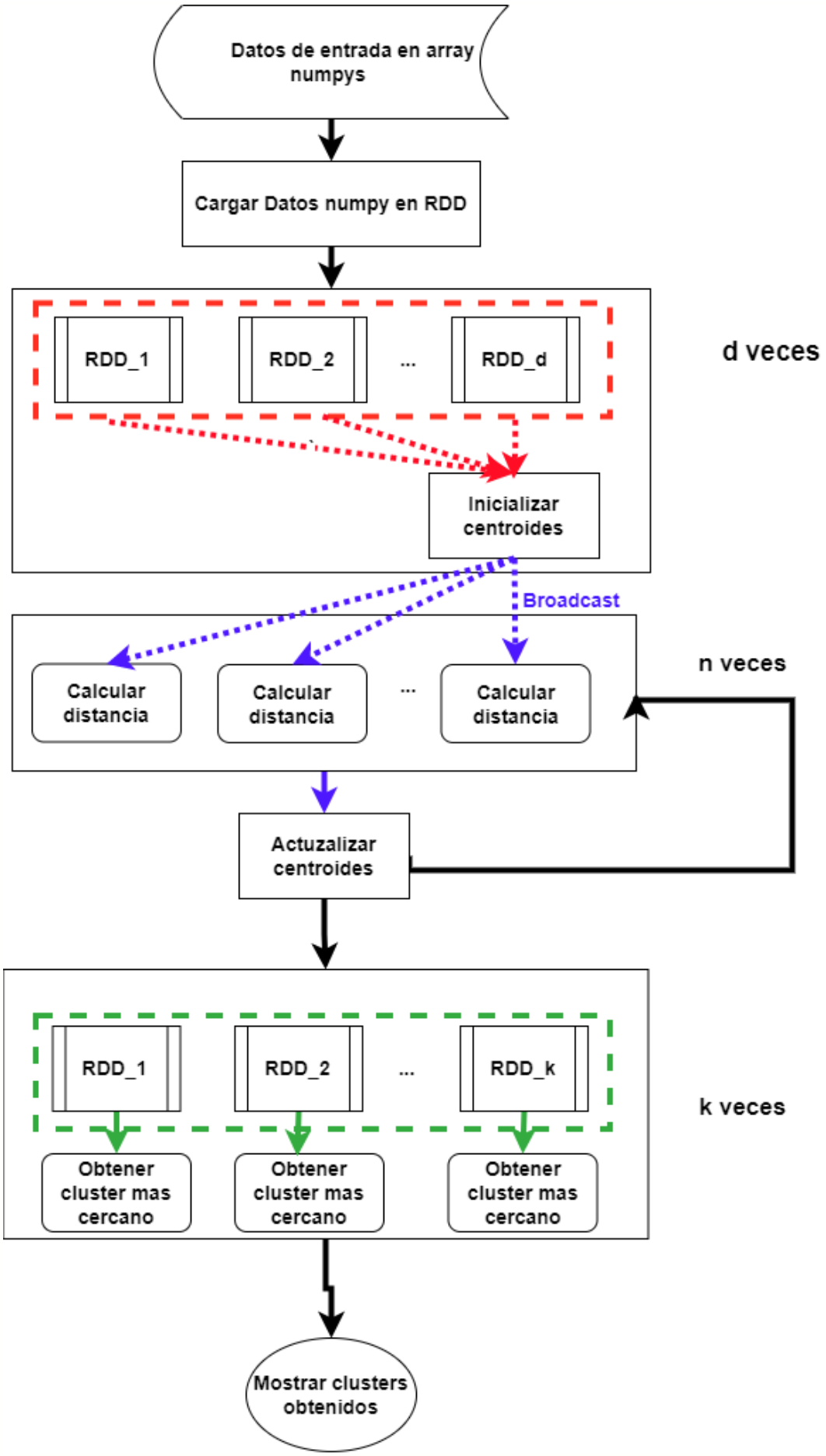
4.6. Diagrama de componentes



4.7. K means en pyspark

Se muestra la siguiente diagrama de flujo que describe el comportamiento del algoritmo en un ambiente big data (pyspark framework).Siendo n el número de datos de nuestro dataset, k el número de clusters, d la dimensión del vector de características, y k el número de clusters que pasamos como parámetro al invocar el método k means fit.

Al distribuir la información y la aplicación de los módulos calcular distancia en distintas instancias se busca aprovechar las bondades de pyspark framework, la cual es paralelizar procesos.


Datos de entrada en array numpys
Cargar Datos numpy en RDD
RDD_1
RDD_2
...
RDD_d
d veces
Inicializar centroides
Broadcast
Calcular distancia
Calcular distancia
...
Calcular distancia
n veces
Actuzalizar centroides
RDD_1
RDD_2
...
RDD_k
k veces
Obtener cluster mas cercano
Obtener cluster mas cercano
Obtener cluster mas cercano
Mostrar clusters obtenidos

5. Conclusiones

- En primer lugar, destacar cómo en la mayoría de las imágenes reales, el número de etiquetas de la imagen segmentada, cuyos valores son máximos, no coinciden con la imagen de referencia. Esto es debido a que el ground truth no es totalmente realista al color y por lo tanto, existen problemas de coincidencia de píxeles.
- Dado que el algoritmo de k-medias se calcula en función de la distancia euclidiana, el algoritmo de k-medias es más sensible al rango de datos, por lo que antes de usar el algoritmo de k-medias, los datos deben estandarizarse para garantizar que el algoritmo de k-medias no se vea afectado por La influencia de las dimensiones de las características.
- El clustering es una técnica muy popular para problemas sin etiqueta y también para tareas de EDA. El K-Means es el rey de esta técnica por su sencillez tanto de entender como de aplicar. Se basa en agrupar los datos según la distancia entre ellos. Aunque como todo en esta vida tiene peros.

6. Bibliografía

- **Singh, P. (2018).** ***Machine Learning with PySpark: With Natural Language Processing and Recommender Systems*****. Apress.**
- ***Jun Yin*** **(2018). Parallelizing K-Means-Based Clustering on Spark** **https://www.researchgate.net/publication/312487404_Parallelizing_K-Means-Based_Clustering_on_Spark**
- **Z. Chen y S. Xia, "Algoritmo de agrupamiento de medios K con centro inicial mejorado",** ***Segundo taller internacional sobre descubrimiento de conocimientos y minería de datos*** **de 2009, págs. 790-792, 2009.**
- **J. Wang y X. Su, "Un algoritmo de agrupamiento K-Means mejorado",** ***3.ª Conferencia internacional de IEEE sobre software y redes de comunicación 2011*** **, págs. 44-46, 2011.**
- ***I. Kusuma, M. A. Ma'sum, N. Habibie, W. Jatmiko and H. Suhartanto, "Design of intelligent k-means based on spark for big data clustering," 2016 International Workshop on Big Data and Information Security (IWBIS), 2016, pp. 89-96, doi: 10.1109/IWBIS.2016.7872895.***
- ***B. Wang, J. Yin, Q. Hua, Z. Wu and J. Cao, "Parallelizing K-Means-Based Clustering on Spark," 2016 International Conference on Advanced Cloud and Big Data (CBD), 2016, pp. 31-36, doi: 10.1109/CBD.2016.016.***