Informe: Implementación y Pruebas del Algoritmo KMeans Distribuido con mrjob

Descripción del algoritmo implementado

En este proyecto se implementó el algoritmo de clustering KMeans utilizando la biblioteca **mrjob** para simular un entorno de procesamiento distribuido tipo MapReduce con Hadoop. La implementación busca replicar el enfoque presentado en el artículo / paper http://vargas-solar.com/big-data-analytics/wp-content/uploads/sites/35/2015/11/2-parallelkmeansmapreduce_zhao.pdf que propone una versión paralelizable del algoritmo KMeans, apta para grandes volúmenes de datos.

El algoritmo KMeans funciona iterativamente asignando puntos de datos a los centros más cercanos (clusters) y recalculando los centros como la media de los puntos asignados. La implementación en mrjob divide el proceso en dos pasos MapReduce por iteración:

1. **Map (label_mapper)**: Para cada punto, se calcula la distancia a cada centro, y se emite un par clave-valor donde la clave es el cluster más cercano y el valor es el punto mismo.
2. **Reduce (reducer)**: Se recibe la lista de puntos asignados a cada cluster y se calcula el nuevo centro como el promedio de esos puntos.

Este proceso se repite hasta alcanzar el número máximo de iteraciones o hasta que los centros converjan.

Archivos principales

- **generate_data.py**: Script para generar datos de prueba sintéticos para el clustering.
- **pkmeans.py**: Implementación del algoritmo KMeans distribuido con mrjob, gestionando los pasos mapper y reducer.
- **run_kmeans.py**: Script principal para ejecutar el proceso iterativo de KMeans, controla la carga de centros y el ciclo de iteraciones.
- **analyze_results.py**: Script para analizar y visualizar los resultados finales del clustering.

Pruebas realizadas

Se realizaron múltiples ejecuciones del algoritmo con diferentes configuraciones:

- Número de clusters (`k`): 3
- Número máximo de iteraciones (`max_iter`): 10
- Dataset sintético generado con `generate_data.py`

Durante las pruebas se validó:

- La correcta convergencia de los centros a lo largo de las iteraciones.
- La integridad y formato de los datos intercambiados entre etapas.
- La robustez frente a errores en la lectura y escritura de centros.

Se detectaron y corrigieron errores relacionados con el formato de lectura de los centros (líneas vacías) y la forma en que se procesan los puntos en el mapper, mejorando la estabilidad de la ejecución.

La implementación replicó satisfactoriamente el proceso iterativo de KMeans en un entorno MapReduce simulado con mrjob, logrando clusterizar los datos sintéticos generados. Este ejercicio permitió comprender el paralelismo de la etapa de asignación de clusters y agregación de nuevos centros, así como las dificultades prácticas de manejar estados entre iteraciones. La idea del paper fue sentar una base para futuras implementaciones distribuidas y escalables en entornos de Big Data.