Universidad Internacional de Valencia

Master en Big Data y Data Science

Problema 2: Categoría de video menos vista

Procesamiento de datos masivos: Spark

Autor:
Adrián Hernández Padrón
Julio 2022

1. Código

Una vez hemos leido los datos de entrada, vamos a mapearlos con un flatMap. Debido a la naturaleza del archivo no es necesario leer las filas una a una, podemos añadir las columanas deseadas con un .split y el índice de la columna. Es necesario hacer un try-except para evitar almacenar datos vacíos, con esto solucionamos que el reduceByKey de problemas ya que no sabe como actuar con filas vacías.

```
def PrepareData(linea):
    clavevalor=[]
    #debido a la naturaleza del fichero, simplemente haciendo el .split de la
    #Es necesario hacer un try-except porque la presencia de datos vacíos ari
    #Si los datos existen los guardamos en un array, si existen los ignoramos
    try:
        clave, valor = (linea.split("\t")[3]), int(linea.split("\t")[5])
        clavevalor.append((clave, valor))
    except Exception:
        pass
    return clavevalor
```

Figura 1: Devolvemos la clave-valor con la categoría-tiempo de reproducción.

El reduceByKey que hace la suma de todos los minutos de reproducción para cada categoría, lo hacemos con lambda y después calculamos el valor mínimo entre todas las categorías.

```
#Vamos a resolver el problema con el map de la función y el reduceByKey usando lambda, despu
suma = datosEntrada.flatMap(PrepareData).reduceByKey(lambda x ,y: x + y).min(lambda x:x[1])
#Al hacer el .min() la variable suma pierde su estructura RDD, tras ponerlo de la forma dese
```

El valor devuelto, que es el mínimo, pierde su estructura RDD, por tanto una vez lo coloquemos con la estructura deseada de salida, usamos .parallelize para transformarlo a RDD y poder usar saveAsTextFile.

```
resultado = [str(suma[0])+ ';' + str(suma[1]))]
#Transformamos la variable resultado a RDD usando parallelize y después exportamos
spark.sparkContext.parallelize(list(resultado)).repartition(1).saveAsTextFile(salida)
```

2. Ejecución y resultados

Para lanzar el programa escribimos lo siguiente por la línea de comandos, en donde 0222 es la carpeta con todos los archivos de texto y el *.txt indica que queremos leer todos los archivos txt de la carpeta. $spark - submitCategoriaDeVideosMenosVista.py' file$: $/Users/adrihp/Master/MBID03/scriptsSpark/0222/*.txt' file$: $/Users/adrihp/Master/MBID03/scriptsSpa$
No es necesario eliminar el archivo txt que no contiene datos puesto que con el try-except se ignora sin generar ningún problema.

La entrada tiene la siguiente estrucutra y la salida se guarda en la carpeta salida2 y tiene la siguiente estructura.

Figura 2: Carpeta con la entrada.

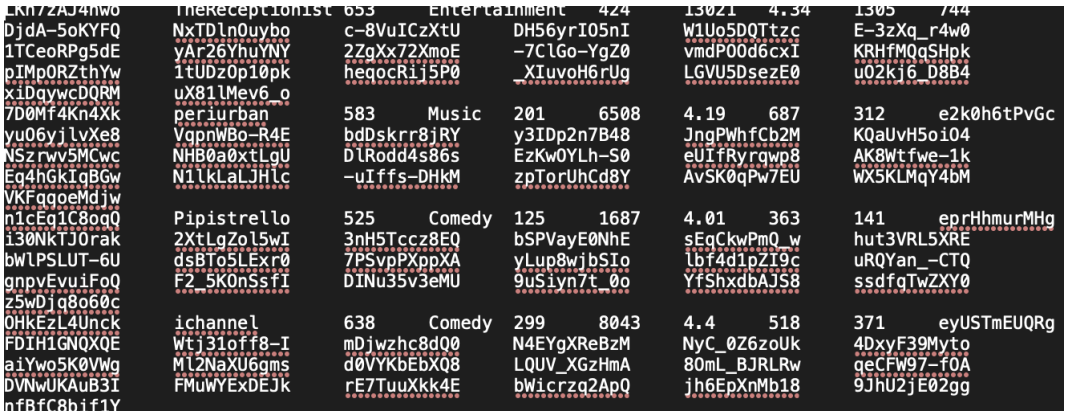

Figura 3: Entrada: ejemplo de un fichero

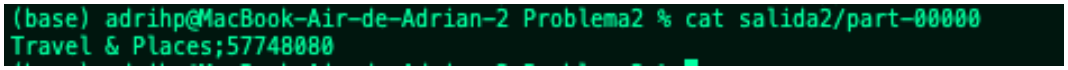

Figura 4: Salida: Categoría de video;Tiempo de reproducción total