UNIVERSIDAD COMPLUTENSE DE MADRID

FACULTAD DE CIENCIAS MATEMÁTICAS

ANÁLISIS DE DATOS CON SPARK

BICIMAD

CECILIA SÁNCHEZ PLAZA, SERGIO RODRIGO ANGULO,
PABLO SIERRA ERICE

Curso 2022/2023

Asignatura Programación Paralela

1. Introducción

Este trabajo consiste en diseñar e implementar una solución a un problema planteado de análisis de datos utilizando Spark. Trabajaremos sobre el dataset que proporciona el ayuntamiento de Madrid del uso de bicicletas del préstamo BICIMAD (sistema de alquiler público de bicicletas eléctricas de la ciudad de Madrid gestionado por la EMT). A continuación, ponemos en contexto el tema de estudio de nuestro trabajo.

Recientemente hemos oído hablar mucho del "nuevo Bicimad" en todos los medios de comunicación. Esto es porque están cambiando todo el sistema y se espera que para julio de este año se hayan incorporado 4500 nuevas bicicletas y 350 nuevas estaciones. Estos cambios son debido a que este servicio llevaba en decadencia desde el 2020, que fue el momento cumbre para bicimad. Vamos a suponer para nuestra práctica que el ayuntamiento de Madrid nos pide un estudio de datos para poder ver cómo ha sido la evolución de este servicio desde que empezaron la legislatura, es decir desde las elecciones municipales que tuvieron lugar en mayo de 2019, hasta un año después, que coincide con el momento histórico de mayor utilización del servicio bicimad. Es por ello que los datos que hemos seleccionado son los correspondientes a mayo de 2019 hasta junio, y los equivalentes meses en 2020.

2. Material y metodología

Todos los datos han sido extraídos del portal de datos abiertos de la EMT, disponibles en el siguiente enlace: https://opendata.emtmadrid.es/Datos-estaticos/Datos-generales-(1).

Los datos vienen dados en formato JSON, y cada uno de estos ficheros contiene la siguiente información:

- **_id**: Identificador del movimiento.
- **user_day_code**: Código del usuario. Para una misma fecha, todos los movimientos de un mismo usuario, tendrán el mismo código, con el fin de poder realizar estudios estadísticos de las tendencias diarias de los usuarios.
- **idunplug_station**: Número de la estación de la que se desengancha la bicicleta.
- **idunplug_base**: Número de la base de la que se desengancha la bicicleta.
- **idplug_station**: Número de la estación en la que se engancha la bicicleta.
- **idplug_base**: Número de la base en la que se engancha la bicicleta.
- **unplug_hourTime**: Franja horaria en la que se realiza el desenganche de la bicicleta. Por cuestiones de anonimato, se facilita la hora de inicio del movimiento, sin la información de minutos y segundos. Todos los movimientos iniciados durante la misma hora, tendrán el mismo dato de inicio.
- **travel_time**: Tiempo total en segundos, entre el desenganche y el enganche de la bicicleta.
- **track**: Detalle del trayecto realizado por la bicicleta entre la estación de partida y la de destino, en formato GeoJSON.
- **user_type**: Número que indica el tipo de usuario que ha realizado el movimiento. Sus posibles valores son:
    - 0: No se ha podido determinar el tipo de usuario
    - 1: Usuario anual (poseedor de un pase anual)
    - 2: Usuario ocasional
    - 3: Trabajador de la empresa

- **ageRange**: Número que indica el rango de edad del usuario que ha realizado el movimiento. Sus posibles valores son:

    0: No se ha podido determinar el rango de edad del usuario

    1: El usuario tiene entre 0 y 16 años

    2: El usuario tiene entre 17 y 18 años

    3: El usuario tiene entre 19 y 26 años

    4: El usuario tiene entre 27 y 40 años

    5: El usuario tiene entre 41 y 65 años

    6: El usuario tiene 66 años o más

- **zip_code**: Texto que indica el código postal del usuario que ha realizado el movimiento.

Para nuestro estudio nos interesa coger la información en cuanto al ageRange, travel_time, y user_day_code. A continuación, explicamos todo lo relacionado con el código implementado.

3. Implementación

Con el fin de gestionar los datos que necesitamos recoger para el estudio recurriremos a programación paralela y por tanto organizaremos los datos recogidos en varios rdd.

Para el análisis de tiempos en función de la edad y el estudio de la media de tiempo de cada trayecto usaremos rdd19 y rdd20 que recoge todos los datos de cada año respectivamente, uniendo los archivos descargados de cada mes.

Sin embargo, para la comparativa del número de usuarios que usan biciMad por mes, necesitamos poder gestionar los datos de cada mes de forma independiente. Para ello, creamos urdd19 y urdd20 cuyos elementos son los rdds independientes a cada mes.

Además de todo lo anterior, en el main podemos encontrar como clasificamos los archivos en función de los nombres proporcionados de esta manera por la página web oficial de biciMad y así poder leerlos y agruparlos en el orden que necesitamos.

El primer estudio consiste en clasificar los datos recogidos en función del rango de edad de los usuarios. Para cada línea del rdd recogemos los datos en una tupla (rango_edad,tiempo_viaje) mediante la función mapper_edad.

A continuación, usamos groupByKey para agrupar los datos en función del grupo de edad y con la nueva tupla obtenida (rango_edad, lista de tiempos de viaje), hacemos la media aplicando sum a la lista y dividiendo entre su longitud.

Para representar los datos extraídos de esta función implementamos un gráfico de barras con eje x los rangos de edad y eje y los tiempos en minutos de cada rango. Además, comparamos los dos años en el mismo gráfico para que se vea de una manera más evidente las diferencias entre un año y otro.

Nuestra segunda función implementa un estudio del número de usuarios que hay al mes dentro de las muestras seleccionadas. Aquí trabajamos con la lista urrd19 y urdd20 donde cada elemento de dichas listas se refiere a un mes y, mediante la función mapper_usuario_unico

transformamos cada línea en el usar_day_code de cada usuario. Usando ahora groupByKey, eliminamos los códigos repetidos, ya que solo estamos interesados en usuarios distintos, y con la longitud de la lista de códigos obtenemos el número de usuarios que hay en cada uno de los meses del muestreo.

Además, en esta función realizamos, de manera análoga a la función anterior, un gráfico de barras para poder comparar los datos de los meses entre mayo y octubre de ambos años. Siendo los ejes de las x los distintos meses del estudio y los ejes de las y el número de usuarios que utilizan biciMad en cada mes.

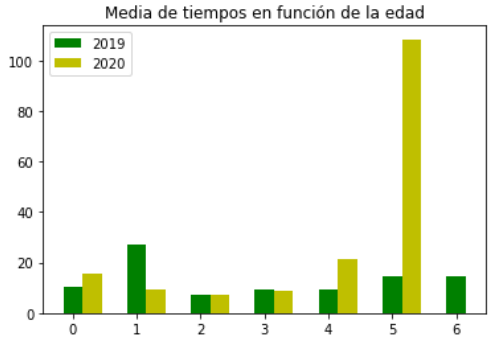
Nuestra última función calcula la media del tiempo de cada trayecto por año. Implementamos la función recogiendo los datos sobre las listas rdd19 y rdd20. A partir de ahí transformamos cada línea en la duración de cada viaje recogido en la muestra y, agrupando todos los datos de cada año con groupByKey, hacemos la media de estos sumándolos y dividiendo entre la longitud de la lista.

Por último, la función proceso ejecuta las tres funciones para así poder obtener los datos que queríamos conseguir.

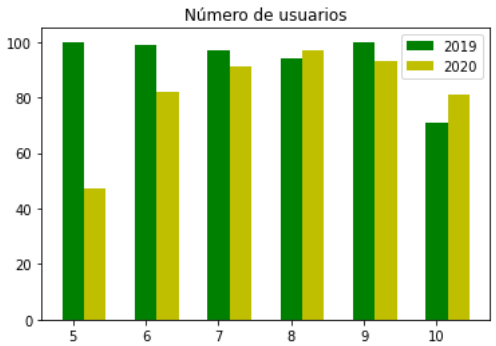
4. Evaluación de resultados.

Antes de evaluar los resultados cabe destacar que hemos tenido problemas a la hora de trabajar con ficheros tan grandes en nuestros ordenadores, ya que por la potencia del CPU no eran capaces de abrir o leer archivos tan grandes. Por ello decidimos tomar una muestra más pequeña de datos, y aunque sabemos que los gráficos que salen no son significativos y no reflejan la realidad, como el código programado sí es correcto bastaría con descargarse los archivos del enlace de arriba correspondientes a nuestros meses y ejecutar el programa en un ordenador más potente, para así obtener gráficas y resultados más completos y acordes a la realidad que queremos estudiar.

4.1. Comparativa del **tiempo medio** de uso de la bicicleta en función de las edades.
En general podemos observar que entre años los resultados son similares, a excepción del grupo de edad 5, donde el resultado se puede haber alterado si hay un número bajo de usuarios con ese rango en nuestra muestra. Para conseguir resultados que se ajusten más con la realidad bastaría con ampliarla a un mayor número de datos.



4.2. En la gráfica para el **número de usuarios** por mes podemos observar como en la mayoría de los meses el número de usuarios disminuye en 2020 respecto a 2019. Lo que hace ver que, en esa diferencia de tiempo, un gran número de clientes dejaron de utilizar los servicios ofrecidos por esta plataforma.



4.3. En cuanto a la última función del código, obtenemos que la **duración media de los trayectos** en 2019 fueron 10 minutos, y en 2020 fueron 28 minutos.
Estos datos nos servirán para proponer una solución al problema de la pérdida de clientes activos en este servicio. La idea sería ofrecer un periodo de tiempo diario que sea gratuito lo que atraerá a más usuarios. Utilizando los tiempos medios de trayecto de 2019 y 2020, el ayuntamiento podrá determinar qué periodo de tiempo es el más adecuado para que esta oferta sea rentable pudiendo estudiar los tiempos para cada rango de edad y como esta media evoluciona a lo largo del tiempo.

5. Conclusión

Con esta práctica hemos conseguido adentrarnos en el inmenso mundo del Big Data a través de Spark y de la biblioteca de Python *pyspark*. Creemos que esto nos ha ofrecido grandes beneficios para nuestro futuro, dada la importancia que tiene poder extraer conocimientos valiosos de grandes volúmenes de datos, permitiéndonos así tomar decisiones informadas y obtener una ventaja competitiva en el mundo actual impulsado por los datos.