GUIÓN – PRESENTACIÓN

1. OBJETIVO:

El objetivo de esta práctica es utilizar los datos facilitados por la empresa BICIMAD de Madrid para la implementación de códigos de Big Data con el fin de etiquetar y ordenar cada estación de bicicletas según su uso recreacional o laboral.

El programa diseñado será capaz de asimilar los archivos disponibles en la base de datos de BICIMAD y sacar la información necesaria para otorgar una puntuación a cada estación de la misma.

Esta puntuación estará comprendida entre los valores 0, correspondiente a una estación puramente recreacional, y 1, correspondiente a una estación puramente laboral.

Además, el programa utilizará estos datos para comparar los usos totales que los usuarios darán a cada estación y también los usos destinados, según los cálculos, hacia labores de ocio o laborales.

Con el fin de presentar una mejor estructura de los datos, estarán disponibles varias opciones de visualización de éstas, véase en forma de tabla, gráfico y coordenadas en mapa interactivo.

Estas puntuaciones están basadas en el sentido común y carecen de un estudio estadístico, pero ofrecen buenos resultados intuitivos que pueden responder a ciertas preguntas. Los usos de estas conclusiones podrían iniciar campañas de marketing enfocadas a estaciones de trabajo u ocio, explorar por qué ciertas estaciones son más utilizadas por usuarios de cada tipo, y, por supuesto, revisar las condiciones de cada puntuación en un futuro y aprovechar el mecanismo del programa para que con un debido estudio estadístico se optimice su funcionamiento.

2. ESTRATEGIA:

La estrategia del programa reside en analizar el comportamiento de cada usuario durante un día y asignar en función de su edad, tipo de usuario (ocasional o anual), número de viajes realizados e intervalos horarios de los mismos, una puntuación por usuario comprendida primeramente entre 0 (usuario puramente de ocio) y 15 (usuario puramente trabajador), para después normalizar dicha cifra en entre 0 y 1.

Una vez otorgada la puntuación por usuario y conseguida una lista de las estaciones con que ha tenido contacto, se atribuirá a dichas estaciones esta puntuación, con un solo uso. Posteriormente, se sumarán las puntuaciones de los demás usuarios a las estaciones con las que han tenido contacto aumentando el número de usos de las mismas, para luego hacer una media total y conseguir así el objetivo del estudio.

Un usuario con puntuación menor de 0.5 será clasificado como "usuario por ocio" y un usuario con puntuación mayor de 0.5, como "usuario por trabajo". Por tanto, para cada estación, según el usuario que haya influido en su puntuación, también se recogerá el uso por ocio o por trabajo.

Asumimos que un usuario con una puntuación de exactamente 0.5 es aquel cuyas características hacen que sea complicado asignarle una categoría, quedando relegado a ofrecer su puntuación a la estación sin formar parte de ningún grupo.

3. Puntuaciones

Como se ha dicho en el anterior punto, a cada cliente primeramente se le atribuirá una puntuación entre 0 y 15. Esta puntuación es el resultado de la suma de tres niveles de puntuación, cada nivel con atribución desde 0 hasta 5 puntos:

1) Nivel de edad:

En este nivel, atribuiremos los siguientes puntos a nuestros clientes según la edad:

- Menores de 16 años: 1 punto (probablemente por ocio)

Aclaración: Consideramos muy poco probable que un menor de 16 años se desplace por trabajo. Sin embargo, en este estudio hemos considerado el desplazamiento por motivos académicos como desplazamiento laboral, ya que es un desplazamiento hacia una actividad de formación para el posterior trabajo.

- Entre 17 y 18 años: 2 puntos (más probable que vaya a estudiar o trabajar).
- Entre 19 y 26 años: 4 puntos (con esta edad los clientes ya empiezan a conducir sus bicicletas por la carretera y tienen a utilizarla para estudiar en la universidad o trabajar)
- Entre 26 y 40 años: 5 puntos (consideramos esta franja la más probable para que un cliente se desplace por trabajo).
- Entre 41 y 65 años: 3 puntos (menos probabilidad del uso de bicicletas para trabajar).
- Mayores de 66 años: 0 puntos (consideramos ínfima la probabilidad de desplazarse por trabajo).

2) Nivel de tipo de usuario (ocasional u anual):

En este nivel, consideramos que si un cliente tiene el abono anual es porque necesitará la bicicleta de forma continuada, y, por tanto, seguramente realice acciones profesionales o académicas:

- Cliente ocasional: 0 puntos
- Cliente anual: 4 puntos

3) Nivel intervalo de horas:

Si un cliente utiliza la bicicleta solamente una vez en un día no podemos establecer con un mínimo de rigor si la ha utilizado para trabajar o para el ocio, aunque sí reconocemos que en intervalos de mañana es más probable que un cliente sea del primer caso.

Si un cliente utiliza la bicicleta dos veces en un día, los intervalos horarios son lo suficientemente alejados y son coherentes a un horario laboral, podemos constatar que hay más probabilidad de que sea por trabajo.

Finalmente, si un cliente utiliza la bicicleta tres o más veces en un día es complicado designar ante qué tipo de cliente estamos (incluso podría haber cogido la bicicleta por la mañana para trabajar y por la tarde para el ocio), así que en estos casos sólo diferenciaremos la puntuación teniendo en cuenta el día de la semana de los mismos (los fines de semana atribuiremos menos puntos).

Los intervalos horarios que utilizaremos para un mismo día serán:

a) Intervalo 1: De 7:00 a.m. a 09:59 a.m. (Horario en el que normalmente se va a trabajar).

b) Intervalo 2: De 10:00 a.m. a 12:59 p.m. (Horario de compras, paseos, etc.).

c) Intervalo 3: De 13:00 p.m. a 15:59 p.m. (Horario de comidas).

d) Intervalo 4: De 16:00 p.m. a 20:59 p.m. (Trabajo de tarde, o también horario de ocio).

e) Intervalo 5: De 21:00 p.m. a 22:59 p.m. (Horario de vuelta del trabajo).

f) Intervalo 6: De 23:00 p.m. a 23:59 p.m. y de 00:00 a.m. a 06:59 a.m. del mismo día.

(Horario nocturno donde los usuarios pueden estar en discotecas y hay poco transporte púbico).

Estos datos sobre los intervalos en combinación con el día de la semana que se produce el viaje (haremos diferencias entre día de entre semana y fines de semana) nos dan las siguientes tablas de puntuaciones:

ENTRE SEMANA – 1 VIAJE

INTERVALO	1	2	3	4	5	6
PUNTUACIÓN	3.5	2.5	2.5	2.5	2.5	2.5

FINES DE SEMANA – 1 VIAJE

INTERVALO	1	2	3	4	5	6
PUNTUACIÓN	2.5	1	1	1	1	1

En las siguientes tablas, referentes a dos viajes, se muestra en la primera columna el intervalo de salida y en la primera fila el intervalo de llegada:

ENTRE SEMANA – 2 VIAJES

INTERVALO	1	2	3	4	5	6
1	0	2	5	5	5	2.5
2	2	0	2	3	4	2.5
3	5	2	0	3	4	2.5
4	5	3	3	0	3	3.5
5	5	4	4	3	0	0
6	1	2.5	2.5	3.5	0	0

FINES DE SEMANA – 2 VIAJES

INTERVALO	1	2	3	4	5	6
1	0	1	1.5	1.5	1.5	0
2	1	0	0	1.5	1.5	1.5
3	1.5	0	0	0	1.5	0
4	1.5	1.5	1	0	1	0
5	1.5	1.5	1.5	1	0	0
6	0	1.5	0	0	0	0

Ante la posibilidad de no poder encontrar en los archivos de datos información sobre edad y/o tipo de usuario, en esos casos la puntuación del mismo irá sobre 10 o sobre 5 puntos, y la media se hará en consecuencia teniendo en cuenta la falta de parámetros.

3. Procedimiento

La estrategia del programa, una vez importados los datos en bruto, se resume en tres pasos:

1) Cargar la RDD en bruto y realizar un proceso de mapeo para transformarla en una RDD cuyas líneas estén formados por diccionarios de la forma:

{"id": ...; "edad": ...; "tipo": ...; "viajes": [(fecha1, origen1, destino1),...] }

Con funciones auxiliares que a partir de una hora en formato "string" devuelven una tupla fecha = ("ES o FS", intervalo_horario) y la extracción y distribución del resto de datos podemos llegar a esta RDD utilizando dos mapeos, una para conseguir pares "clave-valor" y poder agrupar las estaciones que ha utilizado el mismo cliente durante el día, y otro para distribuir de nuevo los datos en el diccionario final.

La RDD que exporta este paso es la llamada RDD 2.

2) Transformar la anterior RDD en otra RDD cuyas líneas estén formados por diccionarios de la forma:

{'id': ... , 'puntos': ... , 'uso_estaciones': ...}

El uso de los datos edad, tipo de usuario, número de viajes e intervalo horario nos sirven para asignar a cada usuario su puntuación utilizando los procedimientos del apartado anterior.

Definiendo las funciones auxiliares correspondientes y utilizando un mapeo podemos llegar a esta nueva RDD, primero sacando los datos necesarios de cada línea y aplicando las funciones para después crear un diccionario con el formato adecuado y añadir los valores a este.

La RDD que exporta este paso es la llamada RDD 3.

3) Transformar la anterior RDD en una lista de diccionarios final con el formato:

[{'estacion': .... , 'media': ...., 'usos': .... , 'usos_ocio': …. , 'usos_trabajo': ….}, …]

Este es el paso más complicado de realizar y el que más recursos consume.

El objetivo de este paso es coger cada estación de una lista y otorgarle la puntuación de las personas que han utilizado dicha estación.

Si dos personas con distintas puntuaciones han utilizado la misma estación, el objetivo es atribuirle a esa estacion la suma de las respectivas puntuaciones de cada persona y dividirla entre dos. Es lógico pensar que esto puede generalizarse a un número finito de personas y conseguir para cada estación su media de puntos, así como su uso (recuento del número de personas que han tenido contacto con ella), su uso de ocio (recuento del número de personas con una puntuación menor que 0.5 que han tenido contacto con ella) y su uso de trabajo (análogo al uso de ocio, pero con personas con puntuación mayor de 0.5).

Sin embargo, una persona puede haber visitado más estaciones. Es decir, ahora es necesario aislar cada estación, atribuir a su puntuación la puntuación del usuario, buscar usuarios con la misma estación y juntar los datos. Por tanto, el objetivo es conseguir un "RDD de mayor tamaño" con el cual podamos separar estaciones de usuarios.

Para ello, hemos dividido este paso en los siguientes procedimientos:

a) Cambiar el formato de la RDD anterior a una nueva RDD cuyos elementos tengan el formato:

[('estacion23', {'puntos': ...,'numero_personas': 1}), ...]

Esta es una manera de conseguir que cada estación sea una clave, pero como cada linea de la RDD anterior contiene una lista de estaciones, cada linea de la nueva RDD con este formato será una lista.

'numero_personas' es siempre 1 en este formato, ya que la puntuación de la estación sólo ha dependido de la puntuación que le ha otorgado la persona.

La RDD que exporta este paso es la llamada RDD 4.

b) Transformar una RDD de líneas en formato:

línea 1 -> [('estacion23', {'puntos': ..., 'numero_personas': 1}),('estacion47', {'puntos': ..., 'numero_personas': 1}), ...]

en una RDD de líneas en formato:

línea 1 -> ('estacion23', {'puntos': ..., 'numero_personas': 1})

línea 2 -> ('estacion47', {'puntos': ..., 'numero_personas': 1})

línea 3 -> ...

Esto significa transformar una RDD de "un tamaño" en otra RDD de "tamaño más grande". Para este procedimiento seguiremos los siguientes pasos:

- Conseguir el número de elementos que tiene la lista más grande de todas las líneas de la RDD original. Para esto aplicamos una pequeña reducción. Este número será n.
- Crear una lista de n RDD's donde cada línea de estas RDD's esté formada por el n-ésimo componente de la lista correspondiente a cada línea de la RDD original. Las RDD llegarán en algún momento a incluir líneas de tuplas en blanco, ya que muchas listas de la RDD original tienen un tamaño pequeño.
- Unir todas estas RDD's en una gran RDD formada, ahora sí, por las líneas de formato ('estacion23', {'puntos': ..., 'numero_personas': 1}) y tuplas vacías. Filtrar después todas estas tuplas en blanco para expulsarlas.

Con un mapper tal que a cada línea de la antigua RDD le extraiga el elemento n-ésimo y un bucle de unión de RDD's llegamos finalmente a la RDD que nos permite esto.

La RDD que exporta este paso es la llamada RDD 5.

c) Transformar la RDD anterior en una RDD de líneas en formato:

('estacion23', [{'puntos': 0.1 , 'numero_personas': 1}, {'puntos': 0.7 , 'numero_personas': 1}, ...]

Realizando una simple operación de groupKey().

La RDD que exporta este paso es la llamada RDD 6.

d) Transformar la RDD anterior en una lista de diccionarios de la forma:

{'estacion': ... , 'media': ..., 'usos': ... , 'usos_ocio': ... , 'usos_trabajo': ...}

donde:

'media' = puntos en media de la estación (entre 0 y 1)

'usos' = número de personas que han tenido contacto con la estación

'usos_ocio' = número de personas con una puntuación menor de 0.5 que han tenido contacto con la estación

'usos_trabajo' = número de personas con una puntuación mayor de 0.5 que han tenido contacto con la estación

Esto es posible gracias a un mapper que separa la clave sacando la información de la id de la estación y junta todos los elementos de la lista de valores en un gran diccionario del que saca la media y suma los usos necesarios.

La RDD que exporta este paso es la llamada RDD 7, y contiene los valores finales que convertiremos a una lista de diccionarios y finalmente exportaremos en un archivo nuevo que luego necesitaremos para llevar a cabo el siguiente apartado.

4. OPCIONES DE VISUALIZACIÓN

Hemos implementado una serie de funciones adicionales que, tomando como base el archivo producido por el apartado anterior, devuelven una visualización de los datos obtenidos. La visualización se muestra en tres formatos distintos, atendiendo a los parámetros que solicite el usuario. Estos formatos son:

- *Formato de tabla:* El programa devuelve una tabla con las estaciones en las filas y los distintos parámetros en las columnas. Se le pide al usuario que especifique con respecto a qué parámetro se quiere ordenar la tabla y si quiere que el orden sea creciente o decreciente. También se le pide especificar si quiere mostrar todas las estaciones o únicamente las N primeras.

- *Formato de mapa:* Se muestra un mapa de Madrid con marcadores indicando la posición de las 10 primeras estaciones de la tabla (según el parámetro y orden especificados). Los marcadores muestran el número y dirección de la estación al deslizar el ratón sobre ellos, y el nombre de la misma al hacer clic.

- *Formato de gráfico:* Un gráfico de barras con las estaciones en el eje X (tantas como se muestran en la tabla, en el mismo orden). Cada estación tiene tres barras indicando los usos totales, los usos por trabajo y los usos por ocio.

Pasamos ahora a desgranar el código en detalle:

1) Funciones auxiliares:

Estas dos funciones preparan los datos del archivo cargado para ser manejados por las funciones de visualización. Son, a saber:

- *import_coord_catalog():*

  Esta función lee el archivo Excel de las coordenadas BICIMAD importado desde la web con un comando anterior y transforma los datos en una lista de diccionarios de formato:

  {'id_est': ... , 'nombre_est': ..., 'dir_est': ... , 'lat_est': ... , 'lon_est': ...}.

  Para ello sigue los siguientes pasos:

  - Importa el archivo Excel y filtra las columnas innecesarias con métodos de la librería pandas de análisis de datos.
  - Parsea el DataFrame obtenido en el paso anterior con un bucle while para crear la lista de diccionarios.
  - Depura la lista de diccionarios con un bucle for y varios condicionales. Elimina las ids repetidas, y corrige las coordenadas y direcciones ausentes o incompletas para que no causen errores.
  - Ordena la lista de diccionarios por id de la estación para un resultado más limpio.

- *v_table(lista_tabla, opcion, orden, n):*

  Esta función toma la lista de diccionarios formada por elementos cargados del archivo raíz y una de las 5 opciones de parámetro, 2 de orden y n elementos. Estos parámetros se obtienen como input en el programa principal, que veremos más adelante. Con la lista y los parámetros confecciona la tabla, el mapa y el gráfico que conforman la visualización. Sigue los siguientes pasos:

- Ordena la lista rdd con métodos de Spark, en función de un parámetro u otro según el argumento "opción" de la función.

- Invierte la lista o no en función del argumento "orden" de la función, para seguidamente crear un DataFrame con pandas con los n primeros elementos, según especifica el argumento correspondiente.

- Crea una lista con las estaciones utilizadas, para usarlas al representar el mapa y el eje X del gráfico. También deshecha las puntuaciones medias, ya que no las representamos en el gráfico.

- Muestra la tabla con la función tabulate, de pandas.

- Con un bucle while y obteniendo primero las coordenadas de todas las estaciones (con la función anterior), crea una lista con las coordenadas de las primeras 10 estaciones de la tabla. Si hay menos de 10, lanza un aviso.

- Comprueba que todas las estaciones a representar tengan coordenadas asociadas. Para las que no (porque no figuren en el Excel), lanza un aviso.

- Usando la librería folium, crea y muestra el mapa con los marcadores correspondientes.

- Crea un DataFrame con los datos para el gráfico y lo muestra con el método plot.bar()

Adicionalmente, descargamos el Excel con las coordenadas directamente de la página web de Bicimad con el comando wget.

2) Función principal:

Se encarga de cargar el archivo lista_trabajo_ocio.json, resultado del apartado anterior, y de pedir los argumentos para llamar a la función v_table y mostrar la visualización. Sigue los siguientes pasos:

- Comprueba que el archivo json existe, y en caso contrario lanza un aviso.

- Convierte el json en un rdd y calcula su tamaño (número de líneas) con una reducción, para después reconvertirlo en una lista con collect().

- Pide valores para los argumentos "opción" y "orden" de v_table mediante inputs, con condicionales para gestionar texto no admitido.

- Pide un número de filas a mostrar (que se asegura sea igual o menor que el total consultando el tamaño del rdd obtenido antes), de nuevo manejando excepciones y distintas clases de input con condicionales.

- Confirma la visualización solicitada con un print(), y llama a la función v_table con los argumentos introducidos.

5. CONLCUSIONES

Algunas conclusiones preliminares:

- Hemos observado que ninguna estación llega a una puntuación media de 0.5, quedándose todas del lado del ocio en mayor o menor medida. Concluimos que, según nuestros criterios, el uso recreativo de BICIMAD es predominante en toda la red, y que no hay ninguna estación cuyo uso primario sea laboral.

- Las estaciones más utilizadas se concentran en el casco histórico (zona de Sol) y zonas recreativas destacadas (Retiro, Matadero). Las menos utilizadas están también en la zona de Sol (posiblemente por cercanía a otras mejor situadas), y en zonas poco céntricas y de alto volumen de tráfico rodado (Argüelles, Nuevos Ministerios, Chamartín norte), lo que explicaría la reticencia de los usuarios a utilizarlas.

- Las estaciones con puntuación media más inclinada al ocio coinciden con las más utilizadas en general, consecuencia directa de los dos puntos anteriores. Las más inclinadas al uso laboral se concentran alrededor de la zona de Azca y Nuevos Ministerios, y de la estación de Atocha y el eje Prado-Recoletos. Esto indica una tendencia al uso multimodal del transporte público (encadenar la bicicleta con el tren/cercanías/bus interurbano) por parte de muchos usuarios, que quizás interesaría potenciar.

En definitiva, hemos observado que BICIMAD responde a un uso principalmente recreativo u ocasional, pero que su uso para trayectos laborales sigue unos patrones muy marcados con potencial de crecimiento. Un estudio más profundo de los datos y/o un dataset más detallado podrían arrojar más luz al respecto, y podría ser el objetivo de este proyecto en el futuro.