Universidad Rey Juan Carlos

MÁSTER EN DATA SCIENCE

Práctica SDPD T3 – 2020

- **Versión**: 1.0.
- **Fecha entrega**: 23:59, lunes, 13 de julio de 2020.
- **Lenguaje de programación**: Python (Jupyter notebook), o Scala (v 2.11).
- **Tecnología**: Apache Spark (Streaming y Structured Streaming), API datos estructurados / SQL; versión 2.4.5 (todas las plataformas).
- **Formato de entrega**: Jupyter notebook (Python o Scala).

1 Dataset

El conjunto de datos seleccionado para esta práctica es UCI Occupancy Detection.

Los datos contienen información de un experimento para intentar predecir los tiempos de ocupación de una estancia, en función de los valores tomados por variables ambientales. Los datos están codificados en formato CSV. El archivo `occupancy_data.csv` contiene 8.143 entradas, una para cada muestra recogida, a intervalos aproximadamente regulares de 1 minuto entre muestras. Los campos incluidos son:

Campo	Descripción
`row`	Identificador de fila.
`date`	Fecha y hora en la que se ha tomado la muestra.
`Temperature`	Temperatura de la estancia, en grados Celsius.
`Humidity`	Humedad ambiental de la estancia (relativa, %).
`Light`	Cantidad de luz, medida en luxes.
`CO2`	Concentración de CO2 en la estancia, en ppm.
`HumidityRatio`	Derivado a partir de la temperatura y humedad relativa de la estancia (kg vapor de agua/kg aire).
`Occupancy`	Indicador de ocupación (0 no ocuapda, 1 ocupada).

Importante: los datos faltantes para cualquier campo del archivo pueden aparecer en el *dataset* com `NaN` (`numpy.nan`). No es imprescindible efectuar imputación de datos antes de resolver cada pregunta, pero sí hay que tener en cuenta los valores faltantes a la hora de realizar las consultas, para evitar errores.

2 Enunciado

El objetivo de la práctica consiste en leer el archivo de datos CSV descrito en el apartado anterior mediante un script productor de Kafka en Python, igual o muy similar al explicado en los ejemplos de clase. Por tanto, se puede partir del mismo código para incluir modificaciones.

Importante: para simular la entrada de datos, el productor debe enviarlos a la **cola de Kafka de entrada de nombre `test`** con un **retardo variable entre muestras** insertadas de **entre 0.6 y 1.3 segundos**.

Después, utilizando la API para programación con **Spark Structured Streaming**, y configurando un intervalo de actualización de *micro-batches* (*triggers*) de 5 segundos, se pide resolver las siguientes consultas:

1. Calcular el promedio de valores de temperatura, humedad relativa y concentración de CO2 para cada *micro-batch*, y el promedio de dichos valores desde el arranque de la aplicación.
2. Calcular el promedio de luminosidad en la estancia en ventanas deslizantes de tamaño 45 segundos, con un valor de deslizamiento de 15 segundos entre ventanas consecutivas.
3. Examinando los datos, podemos apreciar que el intervalo entre muestras originales no es exactamente de 1 minuto en muchos casos. Calcular el número de parejas de muestras consecutivas en cada *micro-batch* entre las cuales el intervalo de separación no es exactamente de 1 minuto.

3 Formato de la entrega

La entrega se realizará mediante un *notebook* de Jupyter . En la entrega, deben aparecer los **resultados de cada una de las consultas,** idealmente junto a una breve explicación si es preciso.