FUNDATEC
FUNDACION TECNOLOGICA DE COSTA RICA

Especialidad Data Science

Curso:

Big Data

Profesor:

Felipe Meza

Estudiante:

Sergio Castillo Segura

Proyecto

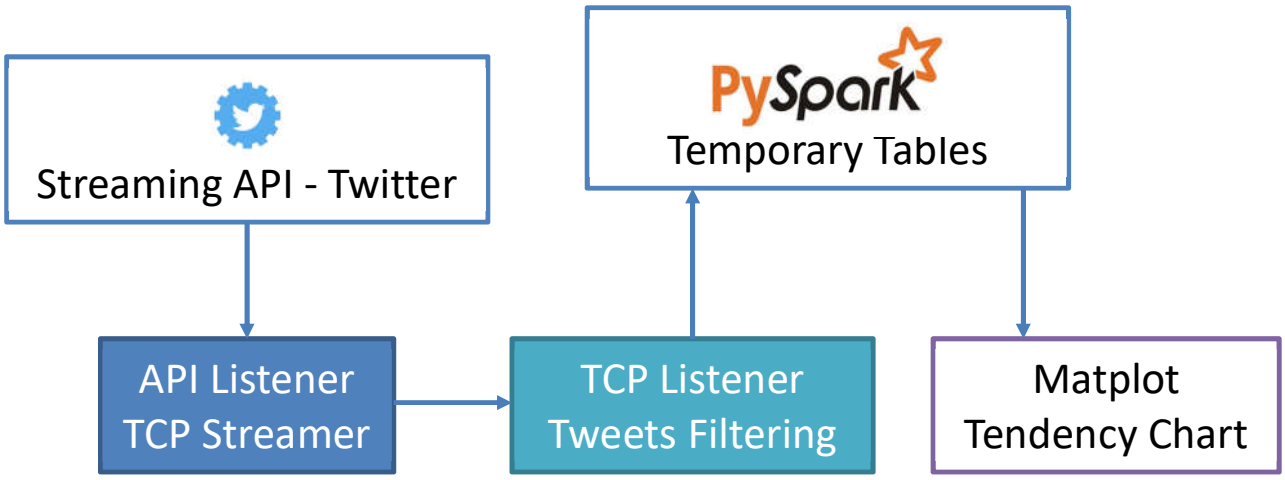
Streaming de tendencias con #hashtags en Twitter

9 Febrero, 2020

Objetivo

El objetivo de este ejercicio será construir una solución basada en python y pyspark para obtener Tweets de la interface Streaming de Twitter en tiempo real, filtrar la palabra clave de interés, buscar los hashtags o palabras que inician con '#' y generar una gráfica de tendencia con las 10 más comunes actualizándose en periodos cercanos a tiempo real.

Arquitectura de la solución



El diagrama anterior representa la arquitectura a alto nivel de la solución de streaming seleccionada para este ejercicio.

Las flechas reflejan el flujo de los datos.

Como se puede notar, la solución se compone de 5 componentes:

1. Twitter API: es el API de Twitter de donde se obtienen los tweets en tiempo real por medio de streaming, para conectarse al API primero se debe hacer un registro de la aplicación en el sitio web para Desarrolladores: https://developer.twitter.com
2. API Listener, este componente consiste en un script de python que realmente tiene 2 funciones: la de conectarse al API de streaming de Twitter para obtener los Tweets y filtrar por aquellos que contienen la palabra clave de búsqueda, y la de crear un servicio de streaming por medio de sockets TCP convencionales. Este último es necesario para el tercer componente a continuación. Para la primera función se utiliza la librería Tweepy que abstrae toda la conexión al API y solo se le debe pasar las credenciales de desarrollador para validar la conexión.

3. El TCP listener es en realidad una notebook de jupyter que permite interactuar con el código más fácilmente, además, el primer módulo debe poder ejecutarse en segundo plano por lo que no era factible ejecutarlo bajo demanda en otra notebook. Se basa en 3 funciones: el objeto Spark se conecta al TCP streamer por medio de sockets en el puerto indicado para obtener los tweets que han sido filtrados, una vez obtenidos se vuelven a filtrar para obtener y ordenar los hash de tendencias y guardarlos en una tabla temporal.
4. Finalmente, utilizando matplotlib se obtienen los hash de la tabla temporal y se dibuja el gráfico. Esta última funcionalidad está identificada como un módulo por aparte en la arquitectura pero en realidad el código es parte de la misma jupyter notebook.
5. El propio Spark, el cual lo utilizamos para recibir y procesar el flujo de tweets y almacenarlos en una tabla temporal en memoria.

Para la ejecución se requiere python 3.7, spark portable y jupyter notebooks.

El primero modulo al ser un script independiente debe correrse directamente desde la línea de comandos con python, mientras que el segundo módulo se ejecuta directamente desde la notebook.

Ambos módulos son dependientes entre sí, el primero no comenzara a recibir tweets hasta que spark haya abierto una conexión, y spark no podrá abrir una conexión si el primer módulo no está ejecutándose recibiendo conexiones por medio de los sockets.

Spark realmente estará ejecutando el procesamiento por medio de micro-batches en vez de atender a cada tweet uno por uno, cada batch contiene los tweets de los últimos 10 segundos. Finalmente, el grafico, para este ejercicio, se estará actualizando cada 2 segundos bajo iteraciones finitas solo con fines ilustrativos mostrando la agregación de toda la data de los últimos 5 minutos.

Referencias

https://developer.twitter.com/

https://developer.twitter.com/en/use-cases/analyze

https://github.com/emiljdd/Tweepy-SparkTwitterI

https://github.com/rajrohan/spark-streaming-twitter

https://towardsdatascience.com/hands-on-big-data-streaming-apache-spark-at-scale-fd89c15fa6b0

https://cognitiveclass.ai/blog/analyze-social-media-data-real-time