NC STATE UNIVERSITY

ADBI Capstone Project: A Keyword-based querying and heavy-hitter analysis over a real-time twitter stream

Team Members:

Aayush Saxena (asaxena6)
Abhishek Akotiya (aakotiy)
Kartik Maheshwari (kmahesh2)

DATA

- Twitter Streaming API
- Provided by twitter.
- Need a developer account/access token to access and use this.
- Filtered for performance optimization and typical hardware usage.

TOOLS USED

- Apache Spark
- Apache Kafka
- Confluent API
- Elasticsearch

PIPELINE

- Connecting Apache Spark, Kafka, and Elasticsearch
- Ingest real time data from twitter to Kafka.
- Create connection between Kafka Spark and Elasticsearch Server.
- Run python scripts for live keyword based querying and heavy hitter analysis.

DATA INGESTION

- Taking live feeds from twitter, accessing through Stream API.
- Storing them in Kafka Queue in JSON format.
- Created Elasticsearch Indexing based on this real time data ingestion

ELASTICSEARCH QUERIES

Performing types of queries on data using Elasticsearch.
Common queries used:

- Match_All :
- Term :
- Match_phrase :
- Prefix :
- Fuzzy :

BLOOM FILTERS

- A bloom filter is a probabilistic data structure which is used to check whether an element is present in a set of words or not.
- It gives a definite answer if the element is not present.
- If the element is present it confirms it with a high probability.
- We use this for STOP WORDS REMOVAL.

WHAT IS COUNT-MIN SKETCH

- Count Min Sketch is a probabilistic data structure that stores the frequency of elements.
- An element passes through multiple hash functions and the values generated are used to update the sketch data structure by incrementing the values in specific places.
- Count-min sketches providing an efficient way of compressed storing of vital information with certain error bound enables large scope in data compression applications

HEAVY HITTERS/TOP-k ANALYSIS

- Heavy hitters are the most frequently occurring elements in a stream of data.
- For every new element, we fetch count min sketch frequency and compare it with the minimum frequency element in the heavy hitters data structure.
- If the frequency of the current word is greater, we delete the minimum frequency element and add this element as a heavy hitter.