Using hive data base to store the tweets because it is schema on read. It means based on the structure of the files, we can create our schema.

Select distinct tweetid, tweetcontent

From tweetsinfo;

Select count(distinct tweetcontent)

From tweetsinfo;

Current Pipe line:

I have created a batch process pipe line. The tweets consumed using twitter API will be stored to files on our local file system.

For the initial process, we can move the files to hdfs location and under hive directory and create tables on top of files. Once we have structures created, we can query the tables.

For ongoing process, we can schedule a job using a scheduling tool like control M, that runs every midnight. This job uses a shell script that moves files from local files system to hdfs location and repairs the table and invalidates the metadata.

The table structure can be partitioned on tweet\_post\_year and tweet\_post\_month (depending on the volume of the data).

Risks involved –

* We don’t know the volume of data will be synchronous. Even after running the job for couple of weeks, we don’t know whether we can expect the same amount of data or not.
* There should not be small files problem. If we are creating small files, it would be an overhead on namenode.
* Also, once we know the volume of data, only then we can set up the better partitioning and bucketing strategy on hive tables.
* For this, we need to understand the machine learning team’s requirements on how they are going to use the data.
* I would also explore the option to process tweets in real time mode than in a batch mode.

Graphical user interface, application

Description automatically generated with medium confidence

How would you roll out the pipeline going from proof-of-concept to a production-ready solution?

Will automate everything before rolling out to production. Things that will be automated:

* Moving the file from local file system to hdfs (1st step).
* Creating the hive table structures, defining partitioning on it.
* Compress small files.

What is the level of effort required to deliver each phase of the solution?

* Data Modelling – Need to work with ML team and understand how they are going to utilize the data. Incorporate all those metrics into your code to consume them from API.

Level of Effort – Medium to hard

* Analyse the data and provide them the sample data to make sure the data works for ML team.

Level of Effort – Low to Medium

* Perform if there are any other transformations or cleanup required.

Level of Effort – Low to medium

* Automate the batch process of loading tweets into tables.

Level of Effort – Low to medium

* Setup a monitoring process :
  + To merge the small files if there are any

Level of Effort – Low to medium

* + To make sure API credentials are working and the java program is running.

Level of Effort – Low to Medium

* Test the entire process end to end.

Level of Effort – Medium to hard.

What is your estimated timeline for delivery for a production-ready solution?

It’s hard to come up with a timeline for production- ready solution. Ideally, from my past experience, most of the time will be consumed in development environment and the next one will be the migration from development to the (next higher) integration test environment. From then onwards, we don’t expect major issues and we expect the process to be smooth. The current process that I have been following is

DEV->SIT->Perf Test->UAT->Prod.

Because we have so many deployments and migrations happening before production, we will mostly know all the issues that we expect. If at any stage any part of our code/requirement breaks, we will start from dev and work upwards.

I agree that this might not be the case in every company, but I expect to take more time in development.

Expected time to develop this complete solution – 2-3 sprints in agile model.