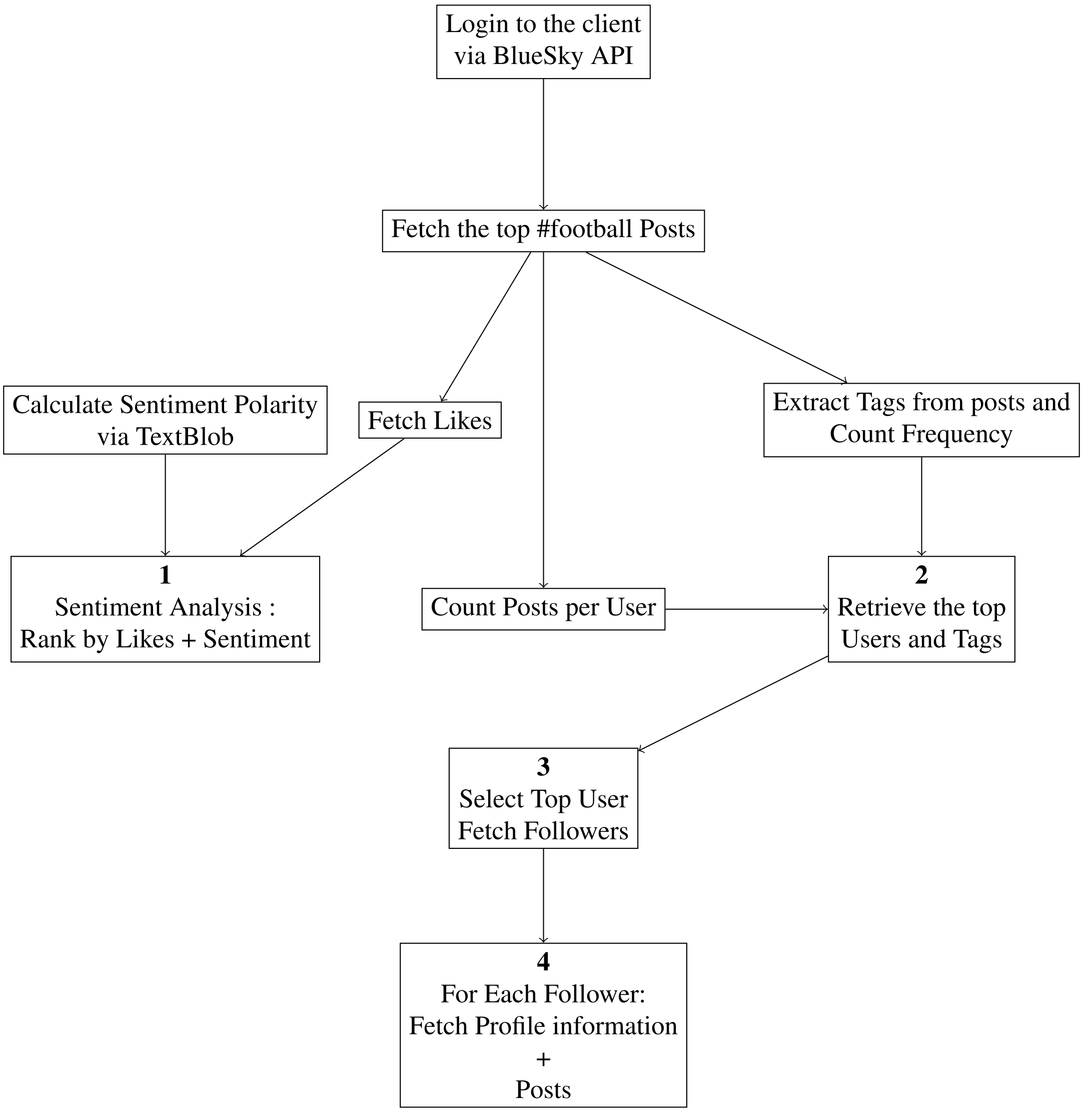
Social Media Data Analysis - Task 2

Stephen BOUCHARDON

June 4, 2025

Flow chart

This part of the project represents the vision and the implementation of the project as of now.
These numbered steps inside the flow chart correspond to the normal continuation of the project and the solutions represented in the next section.


Login to the client via BlueSky API
Fetch the top #football Posts
Calculate Sentiment Polarity via TextBlob
Fetch Likes
Extract Tags from posts and Count Frequency
1 Sentiment Analysis : Rank by Likes + Sentiment
Count Posts per User
2 Retrieve the top Users and Tags
3 Select Top User Fetch Followers
4 For Each Follower: Fetch Profile information + Posts

Solutions

For each subsection, I will present selected parts of the project implementation, with partial solutions. In the joint Jupyter Notebook, you can find the results.

1. Sentiment Analysis

Posts containing the tag #football inside the post's text are retrieved with the Bluesky API. They are collected inside a DataFrame used for all the next steps. We call it the original DataFrame. Then, we analyze the sentiment using the TextBlob library. Each post is assigned a polarity score in the range of [-1.0, 1.0], where -1.0 is considered the least positive and 1.0 the most positive. With that in hand and with the fetched likes, we retrieve those results inside a new DataFrame (for shorter latency). Finally, we sort the new DataFrame by likes and polarities.

2. Top Users per Count and Tags

To resolve the problem of finding the top 10 users per posts and tags, we do as follows :

Find the most repetitive tags: In the column 'text' of the original DataFrame, the tags always start with the symbol #. Every time we encounter a new tag, we will add it to a dictionary as a key with a value representing the number of times we saw it in comments. However, because we use #football as a topic in our search, it is more interesting to remove it and only see the other tags that appear.

Find the most active users: The same logic applies with the column 'author'. We collect inside a dictionary the author's name as a key value and increment as we see it appear in the original DataFrame.

3. Fetching Followers

To make the fetching of followers more interesting, I decided to use the most active user. The Bluesky API offers the possibility to fetch the followers of a user. For this task, we need to import from the original DataFrame the column "author_did" as it is the required parameter of the API's function $get\_followers()$. Then, we can get at most 100 followers of the top user.

4. Follower profile and posts

Based on the results of the last question, we retrieve inside a variable top_user_followers, all the followers of the top user. Then, we can collect and store their profile information in a new DataFrame (name, handle, bio, followers count, following count, posts count, and avatar URL). We will also store inside the DataFrame the posts of the followers by the use of a list type. Example: [post_uri1,post_uri2,...]. This way, with the API's function $get\_posts()$ we can fetch all the posts using the URI.

References

- Bluesky API - Documentation :

  Login to the account :
  https://docs.bsky.app/docs/get-started
  https://docs.bsky.app/docs/starter-templates/bots

  Fetch the posts : https://docs.bsky.app/docs/api/app-bsky-feed-search-posts

  Get the likes : https://docs.bsky.app/docs/api/app-bsky-feed-get-likes

  Get the followers information : https://docs.bsky.app/docs/api/app-bsky-graph-get-followers

- TextBlob - Documentation :
  https://textblob.readthedocs.io/en/dev/quickstart.html#sentiment-analysis

- Applied Data Science with Python - Mr. Lengyel - IN,IT - SoSe 25