Analyzing Racist Tweets on Soccer Players

Report 1

Aravind Padmanabhan
Computer Science
Binghamton University
Binghamton, New York
apadman1@binghamton.edu

Chethana Gopinath
Computer Science
Binghamton University
Binghamton, New York
cgopina1@binghamton.edu

Ridhuvarshan Natarajan
Computer Science
Binghamton University
Binghamton, New York
rnatara2@binghamton.edu

INTRODUCTION

Our project, as mentioned in the proposal is focused on analyzing racist tweets made on soccer players. We used the "Tweepy" library for python and streamlined the live tweets with the help of Twitter APIs.

CHALLENGES

Our project, as mentioned in the proposal is focused on analyzing racist tweets made on soccer players. Our initial methodology of data collection was to collect historical tweets using the tweepy.cursor() method. But the dataset collected as a result of this, failed to produce enough racist content to perform any reasonable analysis. Many of the collected tweets didn't hold much relevance to our objective of collecting racist data. This inference was mainly drawn due to Twitter's moderation policy of removing abusive tweets. Due to the irrelevancy and sparseness of the collected historical data, the consensus was to start collecting live tweets from Twitter as and when a match happens. As the data was being collected, the trend observed was that many tweets just had headlines of articles without actual proof of racist verbal attacks happening on any particular player from any club. Without actual proof of racism, there wouldn't be a metric which decided if the information in the tweet was veritable or not. Finally, we ended up collecting all the tweets based on hashtags of clubs and tournaments who we wanted to follow. This was done during the period of the match from Twitter. This would give a wider dataset for performing sentiment analysis as next steps. This dataset could be refined and cleaned further to derive the results that are required.

METHODOLOGY

In order to collect tweets pertaining to the requirement, a certain number of hashtags were isolated and based on them, tweets were collected. Using the real-time tweet streamlining API, we were able to gather around 400,000 tweets that matched the hashtags which we used as filters for the data collection. The gathering of data was done on match days - before, during and after the match. Our choice of hashtags for the tweets collection was based on tournaments, clubs and keywords that we wanted to closely follow. We chose three tournaments namely English Premier League, UEFA Champions League and UEFA Europa league and monitored tweets that were posted during the matches in which these clubs were participating. The hashtags used for the soccer matches were of the format #SHUARS (for a match between Sheffield United and Arsenal), #UEL (for Europa league games) etc. We also collected tweets that contained the hashtag #NoRoomForRacism which is an official hashtag released by the English Premier League to call out racist behavior. Though tweets containing this hashtag may not be racist, they may be helpful for deriving some valuable observations such as analyzing racist incidents by fans. This in turn, can provide further information to collect even more tweets relating to those particular incidents. In addition to this, we used our Professor's suggestion of expanding our scope to include NBA games as well. As the new season started last week, collecting these tweets would enlarge our dataset adding new data which can be analyzed along with our existing data. For the NBA games, we used hashtags such as #NBA, #LAKERS etc.

IMPLEMENTATION

At first, we ran our code on one of our local machines and collected data there. This was done in python using the Jupyter Notebook. We used the "Tweepy" Library for streamlining the live tweets and stored the collected data in a json file. This was later moved to the mongo database that was installed in the virtual machine.

A) First, we copied the consumer keys and access keys to a json file so that it is easy to refer the keys and ping the API. The code snippet below is used to ping Twitter API every time we run the program.

The code snippet below simply receives the incoming stream of tweets and saves them in a json file.

```python
twitter_cred['CONSUMER KEY'] = 'UPeI5WvTfpJPBUBxmk0pqJv0p'
twitter_cred['CONSUMER_SECRET'] = 'VtRNMpaYVU36Ztfiu5wU4Yk7qBu1r5h4Gi1jaZAD21fA1UYB9Q'
twitter_cred['ACCESS KEY'] = '1175127666577592323-oVohjZ7E6cdIePo4G7zP3SDvIDktWj'
twitter_cred['ACCESS_SECRET'] = 'XHfu50AydSPWE9MRCpznNC0HkoFKIiZmSzmw51PoOrikS'


# In[4]:


with open('twitter_credentials.json','w') as secret_info:
    json.dump(twitter_cred, secret_info, indent = 4, sort_keys= True)


# In[5]:


with open('twitter_credentials.json') as cred_data:
    info = json.load(cred_data)
    print(info)
    consumer_key = info["CONSUMER KEY"]
    consumer_secret = info["CONSUMER_SECRET"]
    access_key = info["ACCESS KEY"]
    access_secret = info["ACCESS_SECRET"]


# In[6]:


auth = tweepy.OAuthHandler(consumer_key,consumer_secret)
auth.set_access_token(access_key,access_secret)
api = tweepy.API(auth)
```

A

```python
class MyListener(StreamListener):

    def on_data(self, data):
        try:
            with open('race.json', 'a') as f:
                f.write(data)
                return True
        except BaseException as e:
            print("Error on_data: %s" % str(e))
        return True

    def on_error(self, status):
        print(status)
        return True

twitter_stream = Stream(auth, MyListener())
twitter_stream.filter(track=['#MUNLIV','#UCL','#SHUARS','#NoRoomForRacism',
    '#Lakers','#UEL','#NBA','#MCIAVL','#BURCHE','#SOULEI','#NORMUN','#ARSCRY'])
```

B

REFERENCES

http://docs.tweepy.org/en/latest/

DATA STORAGE

```
rnatara2@CS480-05:/home/PODS/apadman1$ mongo
MongoDB shell version v3.6.3
connecting to: mongodb://127.0.0.1:27017
MongoDB server version: 3.6.3
Server has startup warnings:
2019-10-25T14:37:43.836-0400 I STORAGE  [initandlisten]
2019-10-25T14:37:43.836-0400 I STORAGE  [initandlisten] ** WARNING: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine
2019-10-25T14:37:43.836-0400 I STORAGE  [initandlisten] **          See http://dochub.mongodb.org/core/prodnotes-filesystem
2019-10-25T14:37:44.768-0400 I CONTROL  [initandlisten]
2019-10-25T14:37:44.768-0400 I CONTROL  [initandlisten] ** WARNING: Access control is not enabled for the database.
2019-10-25T14:37:44.768-0400 I CONTROL  [initandlisten] **          Read and write access to data and configuration is unrestricted.
2019-10-25T14:37:44.768-0400 I CONTROL  [initandlisten]
> use mydb
switched to db mydb
> show collections
tweets
> db.tweets.find()
```


{ "_id" : ObjectId("5db793bbf58d92caf3fbbff5"), "created_at" : "Mon Oct 21 22:54:18 +0000 2019", "id" : NumberLong("1186415441578790914"), "id_str" : "1186415441578790914", "text" : "RT @aww_ghostTears: Is this your Van Dijk? 😂😂😂 \nRashford made him crawl like a toddler 😂😂\n#MUNLIV https://t.co/jLJjltFkeU", "source" : "Twitter for iPhone", "truncated" : false, "in_reply_to_status_id" : null, "in_reply_to_status_id_str" : null, "in_reply_to_user_id" : null, "in_reply_to_user_id_str" : null, "in_reply_to_screen_name" : null, "user" : { "id" : NumberLong("1110499735532699648"), "id_str" : "1110499735532699648", "name" : "boredoms.definition", "screen_name" : "willy_d0n", "location" : "Somewhere", "url" : null, "description" : "⚫⚪🔴🇯🇲", "translator_type" : "none", "protected" : false, "verified" : false, "followers_count" : 83, "friends_count" : 222, "listed_count" : 0, "favourites_count" : 31848, "statuses_count" : 12039, "created_at" : "Tue Mar 26 11:12:23 +0000 2019", "utc_offset" : null, "time_zone" : null, "geo_enabled" : false, "lang" : null, "contributors_enabled" : false, "is_translator" : false, "profile_background_color" : "F5F8FA", "profile_background_image_url" : "", "profile_background_image_url_https" : "", "profile_background_tile" : false, "profile_link_color" : "1DA1F2", "profile_sidebar_border_color" : "C0DEED", "profile_sidebar_fill_color" : "DDEEF6", "profile_text_color" : "333333", "profile_use_background_image" : true, "profile_image_url" : "http://pbs.twimg.com/profile_images/1171588651718983682/RlzrXv6N_normal.jpg", "profile_image_url_https" : "https://pbs.twimg.com/profile_images/1171588651718983682/RlzrXv6N_normal.jpg", "profile_banner_url" : "https://pbs.twimg.com/profile_banners/1110499735532699648/1553636540", "default_profile" : true, "default_profile_image" : false, "following" : null, "follow_request_sent" : null, "notifications" : null }, "geo" : null, "coordinates" : null, "place" : null, "contributors" : null, "retweeted_status" : { "created_at" : "Sun Oct 20 16:35:30 +0000 2019", "id" : NumberLong("1185957725827670016"), "id_str" : "1185957725827670016", "text" : "Is this your Van Dijk? 😂😂😂 \nRashford made him crawl like a toddler 😂😂\n#MUNLIV https://t.co/jLJjltFkeU", "display_text_range" : [0, 77], "source" : "Twitter Web App", "truncated" : false, "in_reply_to_status_id" : null, "in_reply_to_status_id_str" : null, "in_reply_to_user_id" : null, "in_reply_to_user_id_str" : null, "in_reply_to_screen_name" : null, "user" : { "id" : NumberLong("1115567985513652224"), "id_str" : "1115567985513652224", "name" : "🇬🇭rmah 🕊", "screen_name" : "aww_ghostTears", "location" : "Accra", "url" : null, "description" : "A stoner's dreamland☘\nYoung Thug🐍\nM.anifest👑\nugly but smart\n🐂Messi F.C🔴\n\nMy tweets, my opinion. You got yours 💛", "translator_type" : "none", "protected" : false, "verified" : false, "followers_count" : 28690, "friends_count" : 20089, "listed_count" : 4, "favourites_count" : 8556, "statuses_count" : 7748, "created_at" : "Tue Apr 09 10:51:48 +0000 2019", "utc_offset" : null, "time_zone" : null, "geo_enabled" : false, "lang" : null, "contributors_enabled" : false, "is_translator" : false, "profile_background_color" : "F5F8FA", "profile_background_image_url" : "", "profile_background_image_url_https" : "", "profile_background_tile" : false, "profile_link_color" : "1DA1F2", "profile_sidebar_border_color" : "C0DEED", "profile_sidebar_fill_color" : "DDEEF6", "profile_text_color" : "333333", "profile_use_background_image" : true, "profile_image_url" : "http://pbs.twimg.com/profile_images/1163753190812262400/tLjOtDHf_normal.jpg", "profile_image_url_https" : "https://pbs.twimg.com/profile_images/1163753190812262400/tLjOtDHf_normal.jpg", "profile_banner_url" : "https://pbs.twimg.com/profile_banners/1115567985513652224/1570262240", "default_profile" : true, "default_profile_image" : false, "following" : null, "follow_request_sent" : null, "notifications" : null }, "geo" : null, "coordinates" : null, "place" : null, "contributors" : null, "is_quote_status" : false, "quote_count" : 482, "reply_count" : 339, "retweet_count" : 2862, "favorite_count" : 9299, "entities" : { "hashtags" : [{ "text" : "MUNLIV", "indices" : [70, 77] }], "urls" : [ ], "user_mentions" : [ ], "symbols" : [ ], "media" : [{ "id" : NumberLong("11

We imported the json file into the mongodb present in the virtual machine. Hereafter we can pull the data from it and use it for further analysis. The future plans include fetching the stored data and doing sentiment analysis with it using tools such as Google Perspective and hatebase.