FIN42110: HomeWork 4

Ross Kearney

25th March 2022

1 Twitter Topic Modelling

The topic chosen for modelling was the top 10 NFT experts. Their twitter handles are listed below along with the source of the list.

Top 10 NFT Twitter Accounts
https://captainaltcoin.com/nft-twitter-accounts/

Deeze (@DeezeFi)
Beeple (@beeple)
NFT Lately (@NFTLately)
TY Smith (@TyDanielSmith)
Crypto Baristas (@cryptobaristas)
Rac.eth (@RAC)
j1mmy.eth (@j1mmyeth)
RealmissNFT (@RealmissNFT)
Farokh.eth (@farokh)
NFT.NYC (@NFT_NYC)

2 Word Cloud

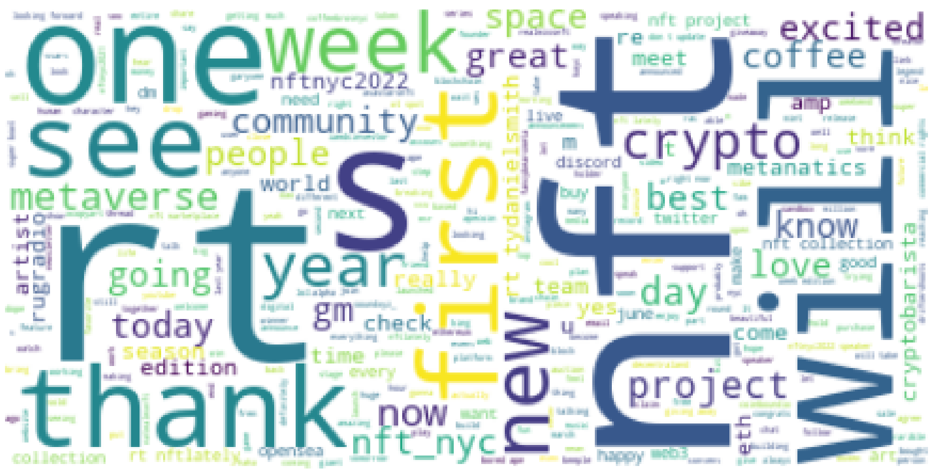

Figure 1: Top 10 Twitter NFT Experts Word Cloud.

The above Word Cloud is a product of the resulting model for 2 topics:

[(0, '0.020*"rt"
+ 0.013*"nft"
+ 0.008*"nfts"
+ 0.005*"one"
+ 0.004*"see"
+ 0.004*"nftnyc"
+ 0.003*"gm"
+ 0.003*"love"
+ 0.003*"metaverse"
+ ' '0.003*"like"'),

(1, '0.019*"rt"
+ 0.013*"nft"
+ 0.006*"nfts"
+ 0.004*"crypto"
+ 0.004*"first"
+ 0.004*"community"
+ 0.004*"week"
+ 0.004*"nftnyc"
+ 0.004*"project"
+ ' '0.003*"going"')]

3 Python Code

```
 #!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Wed Mar  2 16:05:16 2022

@author: rosskearney
"""

# 10 NFT Twitter Accounts
# https://captainaltcoin.com/nft-twitter-accounts/

# Deeze (@DeezeFi)
# Beeple (@beeple)
# NFT Lately (@NFTLately)
# TY Smith (@TyDanielSmith)
# Crypto Baristas (@cryptobaristas)
# Rac.eth (@RAC)
# j1mmy.eth (@j1mmyeth)
# RealmissNFT (@RealmissNFT)
# Farokh.eth (@farokh)
# NFT.NYC (@NFT_NYC)

import tweepy
import sqlite3 as lite
import pandas as pd

import wordcloud
import re
from wordcloud import WordCloud
import gensim.corpora as corpora # Create Dictionary
import gensim
from gensim.utils import simple_preprocess
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
import matplotlib.pyplot as plt
from pprint import pprint


consumer_key = 'XXXXXXXXXXXXXX'
consumer_key_secret = 'XXXXXXXXXXXXX'

access_key = 'XXXXXXXXXXXX'
access_key_secret = 'XXXXXXXXXXXXXX'


con = lite.connect('TWEETi3.db')
cur = con.cursor()
cur.execute("DROP TABLE IF EXISTS TWEETS")
cur.execute("CREATE TABLE TWEETS(author text, created int, tweet text)")

# Authenticate to Twitter
auth = tweepy.OAuthHandler(consumer_key, consumer_key_secret)
auth.set_access_token(access_key,access_key_secret)
api = tweepy.API(auth)

# test authentication
try:
    api.verify_credentials()
    print("Authentication OK")
except:
    print("Error during authentication")

screen_names = ('@DeezeFi',
                '@beeple',
                '@NFTLately',
                '@TyDanielSmith',
                '@cryptobaristas',
                '@RAC',
                '@j1mmyeth',
                '@RealmissNFT',
                '@farokh',
                '@NFT_NYC')

# Cycle through each @ and pull the most recent 200 tweets
for i in range(len(screen_names)):
    public_tweets = api.user_timeline(screen_name = screen_names[i], count=200)

# For each tweet in each user, take the date created, author name, and content of tweet
# Enter these pulled values into the database
    for tweet in public_tweets:
        created = (tweet.created_at.strftime("%d/%m/%Y"))
        author = tweet.author.screen_name
        tweet = tweet.text
        rowi = (author, created, tweet)
        cur.execute("INSERT OR IGNORE INTO TWEETS VALUES( ?, ?, ?)",rowi)
        con.commit()

# Confirm database populated successfully
print('DataBase populated successfully')

# Test query on data base
testquery = """select author, created, tweet from TWEETS"""
testq1_df = pd.read_sql_query(testquery,con)

testq1_df

# Close Data base
con.close()

# For wordcloud, removing tweet content deemed unneccesary
def sent_to_words(sentences):
    for sentence in sentences:
        # deacc=True removes punctuations
        yield(gensim.utils.simple_preprocess(str(sentence), deacc=True))

def remove_stopwords(texts):
    return [[word for word in simple_preprocess(str(doc))
              if word not in stop_words] for doc in texts]

if __name__ == '__main__':

    con = lite.connect('TWEETi3.db')
    tweets = pd.read_sql_query("SELECT * FROM TWEETS", con)

    # Remove punctuation
    tweets['tweet_processed'] = \
    tweets['tweet'].map(lambda x: re.sub('[,\.!?]', '', x))

    # Remove urls
    tweets['tweet_processed'] = \
    tweets['tweet_processed'].map(lambda x: re.sub(r'http\S+', '', x))

    # Convert the titles to lowercase
    tweets['tweet_processed'] = \
    tweets['tweet_processed'].map(lambda x: x.lower())

    # Join the different processed tweets together.
    long_string = ','.join(list(tweets['tweet_processed'].values))

    # Create a WordCloud object
    wordcloud = WordCloud(background_color="white", max_words=5000, \
    contour_width=3, contour_color='steelblue')

    # Generate a word cloud
    wordcloud.generate(long_string)

    # Display the generated image:
    plt.imshow(wordcloud, interpolation='bilinear')
    plt.axis("off")
    plt.show()

#LDA
stop_words = stopwords.words('english')
data = tweets.tweet_processed.values.tolist()
data_words = list(sent_to_words(data))

# remove stop words
data_words = remove_stopwords(data_words)
id2word = corpora.Dictionary(data_words)        # Create Corpus
texts = data_words                # Term Document Frequency
corpus = [id2word.doc2bow(text) for text in texts]

# Number of topics for topic modelling analysis
num_topics = 2      # Build LDA model
lda_model = gensim.models.LdaMulticore(corpus=corpus, id2word=id2word,
                                        num_topics=num_topics)
pprint(lda_model.print_topics())
```