learning technics in Python.
Nov 11 · 3 min read

Sentiment Analysis on NBA top players' Twitter Account — Part2 Tweets Data Cleaning

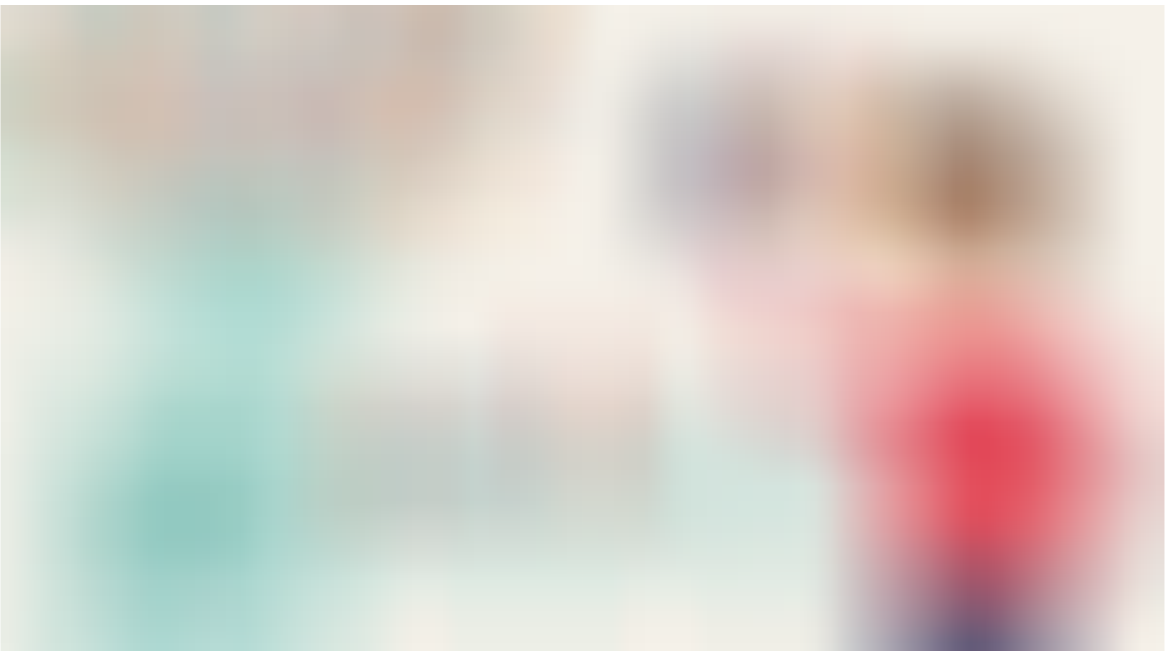


image source: Analytics India Magazine

Abstract

Dive into Text Processing on Social Network

In this section, we are going to clean the tweets since they contain unnecessary texts and we will deal with special cases as well.

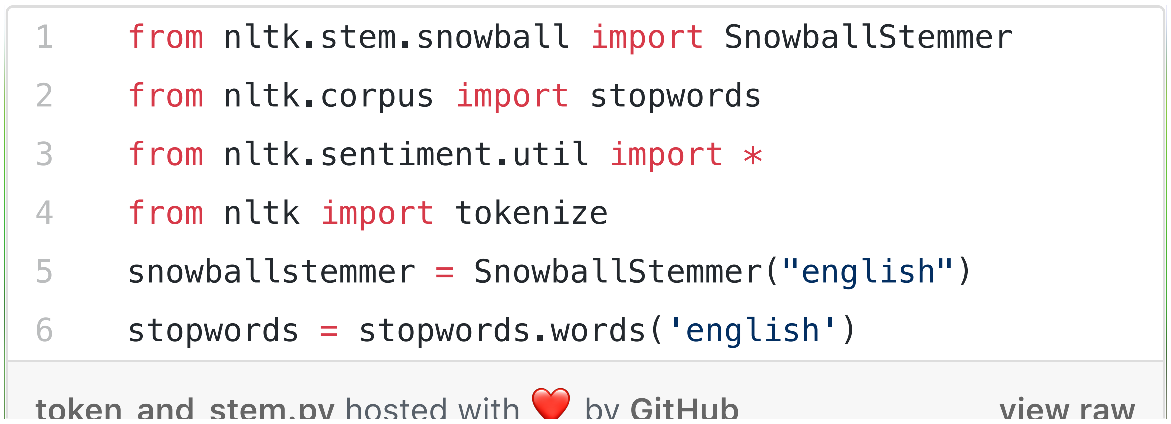
Since people often use special characters, punctuations, or emojis to express their feelings, removing everything except alphabets is not the best way for processing tweets. For example:

> *"In China, it is said that "eat gold soup noodle, then get a gold medal"!! Tomorrow, if I eat 9 in one meal, maybe I can get the MVP in that fortune :). 🍜😋🏀"*

The sentence above is obviously a strong positive reaction not only because of the vocabularies but also the special words like "!!", ":)" and 😋. If wiping out these symbols, the whole expression may become much weaker than before or becomes completely opposite meanings at worst.

Also, many tweets only consist of special characters because they are handy to use. Simply taking out them will cause a huge problem.

Prerequisites for Part 2



```
from nltk.stem.snowball import SnowballStemmer
from nltk.corpus import stopwords
from nltk.sentiment.util import *
from nltk import tokenize
snowballstemmer = SnowballStemmer("english")
stopwords = stopwords.words('english')
```

token_and_stem.py hosted with ❤ by GitHub view raw

Text Processing

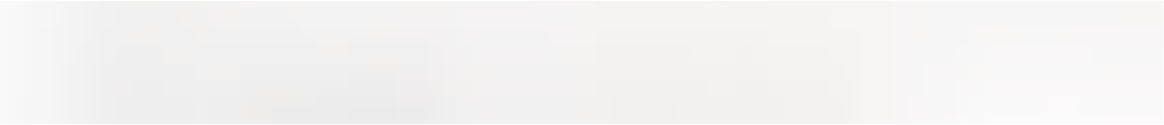
Thus, we only remove some of the words that seem really unnecessary:
1. @’mention: Remove the “@username”(screen name in twitter) no matter it is from retweet or original tweet. We know to tag someone is a way to share the strong emotion but we don’t discuss here.
2. URL: Remove all possible URL links since we are not going to dive deeper into the links
3. stopwords: Remove common words that do not have much meaning
4. punctuations: Removing punctuations is still important but we have to make sure we have kept the important words or symbols first or they will be deleted unconsciously.

Also, we keep the following things in original format because we believe they need to take into the analysis:
1. emoji: 😋, 😢…etc
2. special characters: “:(“, “><”…etc
3. tag of the hashtag: (#) behappy…etc
4. some punctuations: “!!”, “~”, “…”…etc

Next, we will start to handle our text in two part, the first one is just tokenizing and removes the unnecessary words. The second one is applying two stemming method so that we can reduce unique words that actually have the same meaning. Here is the cleaning function I make:

```
def tweets_cleaner(text):
    semiclean_tweet = []
    for tweet in text:
        tweet = re.sub(r'@[A-Za-z0-9_]+','',tweet)
        tweet = re.sub(r"http\S+", "", tweet)
        tweet = re.sub(r"[0-9]*", "", tweet)
        tweet = re.sub(r"(”|“|-|\+|`|#|,|;|\|)*", "",
```

The output if **tweets_cleaner()** looks like this. Here we just remove @'mention, URL, and some punctuation that has very little chance to have special meaning. Note the stopword processing is in the next function because we need to tokenize it before we can remove stopwords.

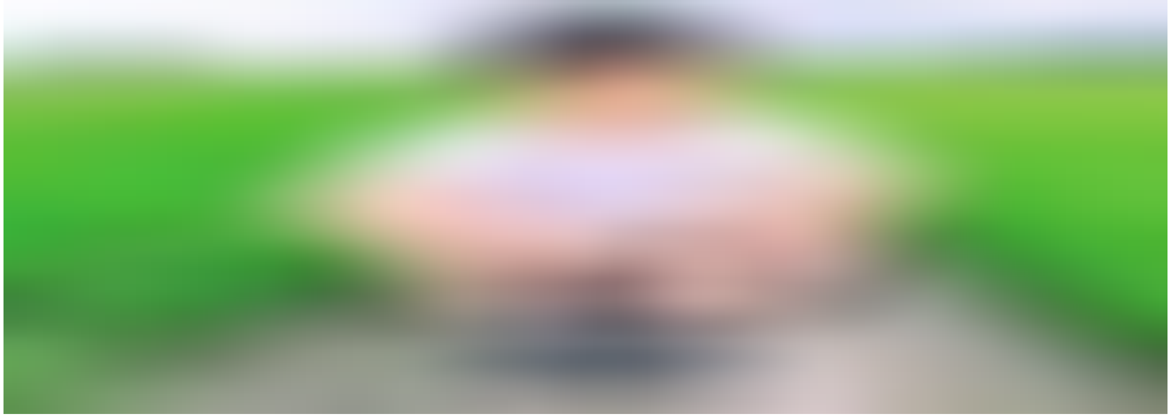


The output if **tokenization_and_stem()** looks like this. You can see the difference between each tokenizing and stemming method. Now we might see some of the tokens seems meaningless such as ":)" becomes ":" and ")". But remember, we will put this back to sentence structure again!

Now, put lists of tokens back into to sentence structure

Finally, we get the cleaned sentence! The back_to_clean_sent() is just concatenating the words back to a sentence.

- sentence_tokenized: Processed sentence with stop words removed.
- sentence_snowstemmeed: Processed sentence with stop words removed and stemming process in **SnowballStemmer** method.



Here is the output from the first player with removing some stopwords, some punctuations, and special symbols that are most likely to be meaningless. Note that it is hard to get the exact clean sentence in NLP, especially when we are not targeting in a particular topic or domain.

That's all for the data cleaning on tweets and chapter 3 we are going to implement the sentiment analysis and data clustering.