MLB Sentiment Test

***This is our current test phase of extracting data from our data source, which we continue to experiment with for a successful extraction and analysis of the data and/or featured terms within the data.**

In [1]:
```python
import os
import datetime
import re

# for the lyrics scrape section
import requests
import time
from bs4 import BeautifulSoup
from collections import defaultdict, Counter
import random
import shutil
```

The two teams I (Andrew Kim) have chosen are the Chicago White Sox and the Oakland Athletics.

In [11]:
```python
MLB = {'white sox':"https://www.mlbtraderumors.com/chicago-white-sox",
       'oakland athletics':"https://www.mlbtraderumors.com/oakland-athletics"}
```

In [12]:
```python
# Let's set up a dictionary of lists to hold our links
MLB_pages = defaultdict(list)

# Set base_url for website link to lyrics
base_url = 'https://www.mlbtraderumors.com/'
url_list = []

for MLB, MLB_page in MLB.items() :
    # request the page and sleep
    r = requests.get(MLB_page)
    time.sleep(5 + 10*random.random())

    #Use Beautiful Soup to extract data from HTML lyrics url
    soup = BeautifulSoup(r.text, 'html.parser')

    # now extract the links to lyrics pages from this page
    # store the links `lyrics_pages` where the key is the artist and the
    # value is a list of links.

    for link in soup.find_all('a'):
        # acquire the links
        temp_href = str(link.get('href'))
        temp_url = base_url + temp_href
        url_list.append(temp_url)

        # retain lyric links
        substring = MLB + '/'
        url_filtered = [i for i in url_list if substring in i]
        url_filtered

        # save and store the links to 'lyrics_pages'
        MLB_pages[MLB] = url_filtered
```

In [22]:
```python
for MLB, lp in MLB_pages.items() :
     assert(len(set(lp)) > 20)
```

```
---------------------------------------------------------------------------
AssertionError                            Traceback (most recent call last)
<ipython-input-22-3527a691eb2b> in <module>
      1 for MLB, lp in MLB_pages.items() :
----> 2      assert(len(set(lp)) > 20)

AssertionError:
```

In [23]:
```python
for MLB, links in MLB_pages.items() :
    print(f"For {MLB} we have {len(links)}.")
    print(f"The full pull will take for this artist will take {round(len(links)*10/3600,2)} hours.")
```

```
For white sox we have 0.
The full pull will take for this artist will take 0.0 hours.
For oakland athletics we have 0.
The full pull will take for this artist will take 0.0 hours.
```

In [ ]:
```python
```

In [24]:
```python
def generate_filename_from_link(link) :

    if not link :
        return None

    # drop the http or https and the html
    name = link.replace("https","").replace("http","")
    name = link.replace(".html","")
    # create empty folder in repo "teams"
    name = name.replace("/teams/","")

    # Replace useless chareacters with UNDERSCORE
    name = name.replace("://","").replace(".","_").replace("/","_")

    # tack on .txt
    name = name + ".txt"

    return(name)
```

In [25]:
```python
# Make the teams folder here. If you'd like to practice your programming, add functionality
# that checks to see if the folder exists. If it does, then use shutil.rmtree to remove it and
# create a new one.

if os.path.isdir("teams") :
    shutil.rmtree("teams/")

os.mkdir("teams")
```

In [29]:
```python
url_stub = "https://www.mlbtraderumors.com/"
start = time.time()

for MLB in MLB_pages :
    # Use this space to carry out the following steps:
    # 1. Build a subfolder for the artist
    subfold = "MLB/" + MLB
    if os.path.isdir(subfold):
        shutil.rmtree(subfold + '/')
    os.mkdir(subfold)

    # 2. Iterate over the team pages
    i = 0
    urls = MLB_pages[team]

    while i < total_pages:
        temp_url = urls[i]

        # 3. Request the team page.
        # Don't forget to add a line like `time.sleep(5 + 10*random.random())`
        # to sleep after making the request
        r = requests.get(temp_url)
        time.sleep(5 + 10*random.random())

        soup = BeautifulSoup(r.text, 'html.parser')

        # 4. Extract the title and team from the page.
        title = soup.find_all('b')[1].get_text()
        teams = soup.find_all('div')[22].get_text()

        song = title + '\n\n' + lyrics

        # 5. Write out the title, two returns ('\n'), and the lyrics. Use `generate_filename_from_url`
        # to generate the filename.
        wd = os.getcwd()
        folders = "\\team\\" + MLB
        filepath = wd + folders
        filename = generate_filename_from_link(temp_url)
        pathname = filepath + '\\' + filename

        text_file = open(pathname, "w")
        text_file.write(team)
        text_file.close()

        i +=1
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
<ipython-input-29-fa2d08ad8878> in <module>
      8     if os.path.isdir(subfold):
      9         shutil.rmtree(subfold + '/')
---> 10     os.mkdir(subfold)
     11
     12     # 2. Iterate over the team pages

FileNotFoundError: [WinError 3] The system cannot find the path specified: 'MLB/white sox'
```

In [ ]:
```python
print(f"Total run time was {round((time.time() - start)/3600,2)} hours.")
```

In [ ]:
```python
```