ADS 509 Assignment 2.1: Tokenization, Normalization, Descriptive Statistics

This notebook holds Assignment 2.1 for Module 2 in ADS 509, Applied Text Mining. Work through this notebook, writing code and answering questions where required.

In the previous assignment you pulled lyrics data on two artists. In this assignment we explore this data set and a pull from the now-defunct Twitter API for the artists Cher and Robyn. If, for some reason, you did not complete that previous assignment, data to use for this assignment can be found in the assignment materials section of Canvas.

This assignment asks you to write a short function to calculate some descriptive statistics on a piece of text. Then you are asked to find some interesting and unique statistics on your corpora.

General Assignment Instructions

These instructions are included in every assignment, to remind you of the coding standards for the class. Feel free to delete this cell after reading it.

One sign of mature code is conforming to a style guide. We recommend the Google Python Style Guide. If you use a different style guide, please include a cell with a link.

Your code should be relatively easy-to-read, sensibly commented, and clean. Writing code is a messy process, so please be sure to edit your final submission. Remove any cells that are not needed or parts of cells that contain unnecessary code. Remove inessential `import` statements and make sure that all such statements are moved into the designated cell.

Make use of non-code cells for written commentary. These cells should be grammatical and clearly written. In some of these cells you will have questions to answer. The questions will be marked by a "Q:" and will have a corresponding "A:" spot for you. *Make sure to answer every question marked with a* `Q:` *for full credit.*

In [1]:
```
import os
import re
import emoji
import pandas as pd
import numpy as np

from collections import Counter, defaultdict
from nltk.corpus import stopwords
from string import punctuation
import nltk
sw = stopwords.words("english")
```

In [2]:
```
# Add any additional import statements you need here
import string
import matplotlib.pyplot as plt
```

In [3]:
```
# change `data_location` to the location of the folder on your machine.
data_location = "/Users/amyou/Desktop/ADS 509"

# These subfolders should still work if you correctly stored the
# data from the Module 1 assignment
twitter_folder = "twitter/"
lyrics_folder = "lyrics/"
```

In [4]:
```
def descriptive_stats(tokens, num_tokens = 5, verbose=True) :
    """
        Given a list of tokens, print number of tokens, number of unique token
        number of characters, lexical diversity (https://en.wikipedia.org/wiki/
        and num_tokens most common tokens. Return a list with the number of to
        of unique tokens, lexical diversity, and number of characters.

    """

    # Fill in the correct values here.
    num_tokens = len(tokens)
    num_unique_tokens = len(set(tokens))
    lexical_diversity = num_unique_tokens / num_tokens if num_tokens > 0 else 
    num_characters = sum(len(token) for token in tokens)

    if verbose :
        print(f"There are {num_tokens} tokens in the data.")
        print(f"There are {num_unique_tokens} unique tokens in the data.")
        print(f"There are {num_characters} characters in the data.")
        print(f"The lexical diversity is {lexical_diversity:.3f} in the data."

        # print the five most common tokens
        common_tokens = Counter(tokens).most_common(num_tokens)
        print("Most common tokens:")
        for token, count in common_tokens:
            print(f"{token}: {count}")

    return([num_tokens, num_unique_tokens,
            lexical_diversity,
            num_characters])
```

In [5]:
```
text = """here is some example text with other example text here in this text"""
assert(descriptive_stats(text, verbose=True)[0] == 13)
assert(descriptive_stats(text, verbose=False)[1] == 9)
assert(abs(descriptive_stats(text, verbose=False)[2] - 0.69) < 0.02)
assert(descriptive_stats(text, verbose=False)[3] == 55)
```

```
There are 13 tokens in the data.
There are 9 unique tokens in the data.
There are 55 characters in the data.
The lexical diversity is 0.692 in the data.
Most common tokens:
text: 3
here: 2
example: 2
is: 1
some: 1
with: 1
other: 1
in: 1
this: 1
```

Q: Why is it beneficial to use assertion statements in your code?

A: Assertion statements are beneficial because it improves hte reliability, clarity, and correctness of the code.

Data Input

Now read in each of the corpora. For the lyrics data, it may be convenient to store the entire contents of the file to make it easier to inspect the titles individually, as you'll do in the last part of the assignment. In the solution, I stored the lyrics data in a dictionary with two dimensions of keys: artist and song. The value was the file contents. A data frame would work equally well.

For the Twitter data, we only need the description field for this assignment. Feel free all the descriptions read it into a data structure. In the solution, I stored the descriptions as a dictionary of lists, with the key being the artist.

In [6]:
```
# Read in the lyrics data

# Specify the full path to the directory containing the lyrics data
lyrics_directory = "/Users/amyou/Desktop/ADS 509/ADS 509 Module 2/M1 Results/ly

# Initialize an empty dictionary to store lyrics data
lyrics_data = {}

# Iterate over the artists' folders
for artist_folder in os.listdir(lyrics_directory):
    artist = artist_folder.lower()  # Use lowercase for consistency

    # Initialize an empty dictionary for each artist to store song lyrics
    lyrics_data[artist] = {}

    # Construct the full path to the artist's folder
    artist_folder_path = os.path.join(lyrics_directory, artist_folder)

    # Check if the artist's folder is a directory
    if os.path.isdir(artist_folder_path):
        # Iterate over the files in the artist's folder
        for song_file in os.listdir(artist_folder_path):
```

```
            # Construct the full path to the song file
            song_file_path = os.path.join(artist_folder_path, song_file)

            # Check if the item is a file
            if os.path.isfile(song_file_path):
                # Extract the song name from the file name (remove the ".txt" e
                song = os.path.splitext(song_file)[0]

                # Read the contents of the song file with explicit encoding spe
                with open(song_file_path, "rb") as file:
                    # Decode the binary data using the appropriate encoding
                    lyrics = file.read().decode(encoding="utf-8", errors="igno

                # Store the lyrics content in the dictionary
                lyrics_data[artist][song] = lyrics

# Inspect the structure of the lyrics_data dictionary

print(lyrics_data)
```

{'.ds_store': {}, 'robyn': {'robyn_includemeout': '"Include Me Out"\n\n\n\r\nI t is really very simple\r\nJust a single pulse, repeated at a regular interval \n\r\nMmm, hmm\r\nDon\'t include me out, no\r\nDon\'t include me out\n\r\nOne time for the records and the hits\r\nTwo for your money-maker, shake, boom\r\n Three times for the lucky and the dead\n\r\nOne time for the sorry and safe\r \nTwo for the beggar and his company\r\nThree times for the sinner and the sai nt\n\r\nYeah, bow down all you wicked and the vain\r\nBow to the miracle, the em, na, na\r\nThree times and the devil will be gone\n\r\nOne time for the fir e, bring it on\r\nTwo for the boogie, gotta bang the beat\r\nThree times for t he ladies, show me some love\n\r\nTalking \'bout everyone, every day, all day \n\r\nAnd if your world should fall apart\r\nThere\'s plenty room inside my he art\r\nJust don\'t include me out\r\nDon\'t include me out\n\r\nAnd if your wo rld should fall apart\r\nI still got room inside my heart\r\nJust don\'t inclu de me out\r\nDon\'t include me out, d-d-don\'t include me out\n\r\nAll hail to the mamas who hold it down\r\nHail to the pillar of the family\r\nThis one\'s for the granny, take a bow\n\r\nOne time for the crazy and the bent\r\nCome o n, all you trannies click your heels for me\r\nAll praise the fugeses and the gems\n\r\nTalking \'bout everyone, every day, all day, oh yeah\n\r\nAnd if you r world should fall apart\r\nThere\'s plenty room inside my heart\r\nJust don \'t include me out\r\nJust don\'t include me out\n\r\nAnd if your world should fall apart\r\nI still got room inside my heart\r\nJust don\'t include me out\r \nDon\'t include me out, d-d-don\'t include me out\n\r\nCan I get a beat, beat for all of my watchamacallits\r\nDoing whatever and with whoever they like?\r \nCan I get a beat, beat for all of my watchamacallits\r\nDoing whatever and w ith whoever they like?\n\r\nCan I get a bam, bam for all of my watchamacallits \r\nDoing whatever and with whoever they like?\r\nCan I get a bam, bam for all of my watchamacallits\r\nDoing whatever and with whoever they like?\n\r\nI\'m talking about everyone, every day, all day, hey\n\r\nAnd if your world should fall apart\r\nThere\'s plenty room inside my heart\r\nJust don\'t include me o ut\r\nDon\'t include me out\n\r\nAnd if your world should fall apart\r\nI stil l got room inside my heart\r\nJust don\'t include me out\r\nDon\'t include me out\n\r\nAnd if your world should fall apart\r\nThere\'s plenty room inside my heart\r\nJust don\'t include me out, hey, hey\n\r\nAnd if your world should fa ll apart\r\nI still got room inside my heart, yeah\r\nJust don\'t include me o ut, hey\n', 'robyn_electric': '"Electric"\n\n\n\r\nElectric...\n\r\nIt\'s elec tric\r\nIt\'s a natural high\r\nElectric\r\nWe don\'t always know why\r\nElect ric\r\nKeep your ego aside\r\nWell it\'s electric\r\nIt\'s a thing you can\'t deny\n\r\nBlood boils without fire\r\nDay come, day go\r\nWithout your desire \r\nDisturbs the flow\r\nWhen in denial\r\nNight falls,\r\nFalls to take you h igher\n\r\nIt\'s electric\r\nIt\'s a natural high\r\nElectric\r\nWe don\'t alw ays know why\r\nElectric\r\nKeep your ego aside\r\nWell it\'s electric\r\nIt \'s the thing you can\'t deny\n\r\nA hard question\r\nNeeds an easy answer\r\n Recognize, accept no need to censor\r\nThe harder the fall\r\nThe higher the b ounce\r\nSmooth moves\r\nKeeps your buoyancy\r\nYeah, keeps your balance\r\nOh h ah\n\r\nIt\'s electric\r\nIt\'s a natural high\r\nElectric\r\nWe don\'t alwa ys know why\r\nElectric\r\nKeep your ego aside\r\nWell it\'s electric\r\nIt\'s the thing you can\'t deny\n\r\nOhh\r\nElectric\r\nElectric (can\'t deny that i t\'s so)\r\nElectric (it\'s electric)\r\nElectric (oooh yeah yeah yeah)\n\r\nI n the eyes of a child\r\nAnd the love that moves\r\nThe sun and the stars abov e\r\nThat race your heart\r\nSomebody dies\r\nAnd you have to cry\r\nWhen you think this is it\r\nAnd then some other\r\nShit just happens\r\nYeah, it just happens, happens\r\nThat\'s when\n\r\nIt\'s electric\r\nIt\'s a natural high\r \nElectric\r\nWe don\'t always know why\r\nElectric\r\nKeep your ego aside\r\n Well it\'s electric\r\nIt\'s the thing you can\'t deny\n\r\nIt\'s electric\r\n It\'s a natural high\r\nElectric\r\nWe don\'t always know why\r\nElectric\r\nK eep your ego aside\r\nWell it\'s electric\r\nIt\'s the thing you can\'t deny\n \r\nElectric....\n', 'robyn_beach2k20': '"Beach 2K20"\n\n\n\r\n(So you wanna g o out?\nHow you gonna get there?\nOK?\nShould we call someone?\nAlright\nSo... OK)\n\n(Can\'t wait to go)\n(OK\nOK\nSo... OK\nOK\nOK)\n\n(So you wanna go ou t?)\nTo this cute place on the beach\nThey do really nice food\n(How you gonna

get there?)\nI mean, it\'s right on the beach\nCome through, it\'ll be cool\n(Should we call someone?)\n(Hmm, OK)\n\n(This place on the beach\nI gotta tell ya\nThis place on the beach\nIt\'s a party, baby\nIt\'s a party)\n\n(So you wanna go out?\nHow you gonna get there?\nOK?\nShould we call someone?\nAlright\nSo... OK)\n\n(Come down\nDon\'t wait too long\nWhat you wanna do, baby?\nOh, yeah, yeah...\nOK)\n\nTo this cute place on the beach\nThey do really nice food\nI mean, it\'s right on the beach\nCome through, it\'ll be cool\nTo this cute place on the beach\nThey do really nice food\nI mean, it\'s right on the beach\nCome through, it\'ll be cool\n\n(So you wanna go out?\nOK\n(Let\'s go party)\nShould we call someone?\nAlright\n(Let\'s go party)\nOK)\n\nTo this cute place on the beach\nThey do really nice food\n(How you gonna get there?)\nI mean, it\'s right on the beach\nCome through, it\'ll be cool\n(Should we call someone?)\nTo this cute place on the beach\nThey do really nice food\nI mean, it\'s right on the beach\nCome through, it\'ll be cool\n(OK)\n\n(Party, party, party, party\nThis place on the beach)\n\nLet\'s go party\n(Let\'s go party)\n(Party)\n(Party)\n\nLet\'s go party\n(Oh, yeah)\nLet\'s go party\nLet\'s go party\n(Oh, yeah)\nLet\'s go party\n\n(Let\'s go party)\n(Let\'s go party)\n(Oh, yeah)\n(Party, baby)\n', 'robyn_lovekills': '"Love Kills"\n\n\n\r\nIf you\'re looking for love\r\nGet a heart made of steel \'cus you know that love kills\r\nDon\'t go messing with love\r\nIt\'ll hurt you for real, don\'t you know that love kills\n\r\nIf you\'re looking for love\r\nGet a heart made of steel \'cus you know that love kills\r\nDon\'t go messing with love\r\nIt\'ll hurt you for real, don\'t you know that love kills\n\r\nProtect yourself, \'cus you\'ll wreck yourself\r\nIn this cold, hard world, so check yourself\r\nYou conceal your dreams and you shield yourself\r\n\'Til that one kind soul reveals itself\n\r\nProtect yourself, \'cus you\'ll wreck yourself\r\nIn this cold, hard world, so check yourself\r\nYou conceal your dreams and you shield yourself\r\n\'Til that one kind soul reveals itself\n\r\nIf you\'re looking for love\r\nGet a heart made of steel \'cus you know that love kills\r\nDon\'t go messing with love\r\nIt\'ll hurt you for real, don\'t you know that love kills\n\r\nIf you\'re looking for love\r\nGet a heart made of steel \'cus you know that love kills\r\nDon\'t go messing with love\r\nIt\'ll hurt you for real, don\'t you know that love kills\n\r\nMm, and I know when you\'re in to deep you still think of me, sometimes\r\nStockholm syndrome and misery, there\'s a penalty for love crimes\n\r\nProtect yourself, \'cus you\'ll wreck yourself\r\nIn this cold, hard world, so check yourself\r\nYou conceal your dreams and you shield yourself\r\n\'Til that one kind soul reveals itself\n\r\nProtect yourself, \'cus you\'ll wreck yourself\r\nIn this cold, hard world, so check yourself\r\nYou conceal your dreams and you shield yourself\r\n\'Til that one kind soul reveals itself\n\r\nProtect yourself\r\nSo check yourself\n\r\nMm, and I know when you\'re in to deep, is to think of me, sometimes\r\nStockholm syndrome and misery, there\'s a penalty for love crimes\n\r\nProtect yourself, \'cus you\'ll wreck yourself\r\nIn this cold, hard world, \'cus you know that love kills\r\nYou conceal your dreams and you shield yourself\r\nIn this cold, hard world, don\'t you know that love kills\n\r\nProtect yourself, \'cus you\'ll wreck yourself\r\nIn this cold, hard world, \'cus you know that love kills\r\nYou conceal your dreams and you shield yourself\r\nIn this cold, hard world, don\'t you know that love kills\n\r\nProtect yourself, \'cus you\'ll wreck yourself\r\nIn this cold, hard world, \'cus you know that love kills\r\nYou conceal your dreams and you shield yourself\r\nIn this cold, hard world, don\'t you know that love kills\n', 'robyn_timemachine': '"Time Machine"\n\n\n\r\nHey, what did I do?\r\nCan\'t believe the fit I just threw\r\nStupid\r\nWanted the reaction\n\r\nI remember the words\r\nHow I said them, so they would hurt\r\nBut then, I regret my actions\n\r\nIf I could press rewind\r\nRewind the take\r\nRedefine the line\r\nWe make mistakes\r\nTake it back in time, \r\nJust one day ... hey\n\r\nSo all I need is a time machine\r\nA one way track cause\r\nI\'m taking it back, taking it back\r\nAll I want is a DeLorean\r\nIf I could go just like that\r\nI\'ll be taking it back, taking it back\n\r\nYeah who\'s laughing now?\r\nDidn\'t think you\'d actually go\r\nStupid\r\nMe and all my bitching\n\r\nSecond thoughts and regrets\r\n?? but then\r\nThis ain\'t science fiction\n\r\nIf I co

ack on my feet again\n\nHere I am, I\'m back on the street again\nHere I am, I \'m back on my feet again\n', 'cher_loveonarooftop': '"Love On A Rooftop"\n\n \n\r\nWe used to talk forever on a dime\nNow we live together never find the t ime\nWe used to walk as lovers on the sand\nNow we\'re workin\' full-time on o ur lifetime plan\n\nWe never stopped to see the moon at night\nWe\'re just too busy leadin\' complicated lives\n\nI remember love on a rooftop\nWe couldn\'t make the love stop\nWe were givin\' all that we got\n\nI remember holdin\' you so tight\nWhen kissin\' lasted all night\nLove on a rooftop\n\nLook at us now, we\'re all grown up\nWe got it all together, got it all sewn up\nBut is this a ll it all was leadin\' to\nDo we just run out of dreams when all our dreams co me true\n\nWhatever happened to those endless nights\nWhen we were happy livin \' young and foolish lives\n\nI remember love on a rooftop\nWe couldn\'t make the love stop\nWe were givin\' all that we got\n\nI remember holdin\' you so t ight\nWhen kissin\' lasted all night\nLove on a rooftop\n\nWe never stopped to see the moon at night\nWe\'re just too busy leadin\' complicated lives\n\nI re member love on a rooftop\nWe couldn\'t make the love stop\nWe were givin\' all that we got\n\nI remember holdin\' you so tight\nWhen kissin\' lasted all nigh t\nLove on a rooftop\n\nI remember love on a rooftop\nWe couldn\'t make the lo ve stop\nWe were givin\' all that we got\n', 'cher_hardenoughgettingoveryou': '"Hard Enough Getting Over You"\n\n\n\r\nFor the first time, in such a long ti me\nI\'ve been feeling there\'s some reason\nTo hold on through the night\nAnd for the first time since\nYou broke this heart of mine\nI find myself believin g\nIt\'s really gonna be alright\n\nNow you\'re asking me to let\nYou walk bac k into my life\nSomething I can\'t bring myself to do\n\n\'Cause it\'s been ha rd enough getting over you\nYou kept me holding on till the end\nOh it\'s been hard enough getting\' over you\nI don\'t think that I could say goodbye again \n\nI could trust you, but it\'s out of my hands\nI once believed your promise s\nBelieved in every word you said\nI still love you but my heart can\'t take the chance\nThat all the pain I\'ve left behind me\nMight be waiting for me up ahead\n\nJust remember it was you\nWho walked right out of my life\nNo you don \'t know what I\'ve been through\n\n\'Cause it\'s been hard enough getting ove r you\nYou kept me holding on till the end\nOh it\'s been hard enough getting \' over you\nI don\'t think that I could say goodbye again\n\nThen I made a pr omise to myself\nAnd this one I\'m not gonna break\nI\'ve made it without you and I\'m not about to\nTurn around and make the same mistake\n\n\'Cause it\'s been hard enough getting over you\nYou kept me holding on till the end\nOh, it \'s been hard enough getting over you\nI don\'t think that I could say goodbye \nI don\'t think I could say goodbye\nI know that I can\'t say goodbye again \n', 'cher_takeitfromtheboys': '"Take It From The Boys"\n\n\n\r\nSo scared I n ever been\nToo hard to keep it in\nGood girl, independent citizen\nBut you can \'t go back\nNo they never let you back\nSame dance different hall\nSome guys they get it all\nBut you can\'t go back\nAnd they\'re ready to attack\nAnd you can\'t go back\nAnd you\'re terrified\nYou\'re gonna fall\n\nTake it from the boys in the street\nTake it from the boys\nTake it from the boys that you meet \nMaking all that noise\nThey got you sipping\nOn a dry martini on a rack\nThe y\'re slipping out\nThe back door no slack\nMaybe it\'s the same\nWith every m an you meet\nYou might as well be wise\nAnd take it from the boys\nIn the stre ets\nTake it from the boys\nTake it from the boys\nYou might as well be wise\n And take it from the boys\nIn the streets\nTake it from the boys\nTake it from the boys\nYou might as well be wise\nAnd take it from the boys\n\nDo this and promise that\nWell that\'s how they\'re all about\nWatch out, you\'re dealing with a photostat\nWhen they say they care it\'s a bonify affair\nLook out, bew are of the businessman\nExpress American, he\'ll be gone so fast\nCause he tra vels first class\nWith his now and powie stash\nAnd his heart unofficial glass \n\nTake it from the boys in the street\nTake it from the boys\nTake it from t he boys that you meet\nMaking all that noise\nThey got you sipping\nOn a dry m artini on a rack\nThey\'re slipping out\nThe back door no slack\nMaybe it\'s t he same\nWith every man you meet\nYou might as well be wise\nAnd take it from the boys\nIn the streets\nTake it from the boys\nTake it from the boys\nYou mi ght as well be wise\nAnd take it from the boys\nIn the streets\nTake it from t

he boys\nTake it from the boys\nYou might as well be wise\nAnd take it from th
e boys\n\nTake it from the boys in the street\nTake it from the boys\nTake it
from the boys that you meet\nMaking all that noise\nThey got you sipping\nOn a
dry martini on a rack\nThey\'re slipping out\nThe back door no slack\nMaybe it
\'s the same\nWith every man you meet\nYou might as well be wise\nAnd take it
from the boys\nIn the streets\nTake it from the boys\nTake it from the boys\nY
ou might as well be wise\nAnd take it from the boys\nIn the streets\nTake it f
rom the boys\nTake it from the boys\nYou might as well be wise\nAnd take it fr
om the boys\n', 'cher_dreambaby': '"Dream Baby"\n\n\n\r\nI found the boy for m
e\nHe\'s my dream baby\n\nHe doesn\'t now that I\nThat I almost die when\nWhen
he walks by and says hello\nHe\'s not like the guys in books\nHe\'s not really
great on looks, but\nBut I don\'t care, I love him so\n\nAnd I feel so good\nW
henever he\'s around\nFeel so good when he\'s around\n\nOh, oh, oh\nI dream of
him each night\nI dream that he holds me tight\nAnd somehow I\'ll make that\nD
ream come true\n\nI\'m gonna pray for the\nPray for the day he says\n"Hey, pre
tty baby, I love you"\n\nAnd I feel so good\nWhenever he\'s around\nFeel so go
od when he\'s around\n\nI dream of him each night\nI dream that he holds me ti
ght\nAnd somehow I\'ll make that\nDream come true\n\nI\'m gonna pray for the\n
Pray for the day he says\n"Baby, I love you"\nOh, oh, woo oh, oh\nI love you
\n', 'cher_pleasedonttellme': '"Please Don\'t Tell Me"\n\n\n\r\nYa shook the o
verride \nWhy\'d I get this far this time \nA ton of more I mean \nRidin\' hel
l and the mile this time I see \n\nDrift machine ago \nIt\'s one you\'ll never
know \n\nAn hour lates too soon \nSurround the stop right through \nA sound fa
r too long \nExplode, fail and see goin\' to and I\'m gone \n\nDrift machine a
go \nIt\'s one you\'ll never know\n', 'cher_ihopeyoufindit': '"I Hope You Find
It"\n\n\n\r\nThese clouds aren\'t going nowhere, darling\nRain keeps coming do
wn\nI just thought I\'d try to call you\nFor you got too far outta town\nAnd I
hope that you get this message that I\'m leaving for you\n\'Cause I\'d hate th
at you left without hearing the words that I needed you to\n\nAnd I hope you f
ind it\nWhat you\'re looking for\nI hope it\'s everything you dreamed your lif
e could be\nAnd so much more\n\nAnd I hope you\'re happy, wherever you are\nI
wanted you to know that\nAnd nothing\'s gonna change that\nI hope you find it
\n\nAm I supposed to hang around and wait forever?\nLast words that I said\nBu
t that was nothing but a broken heart talking, darling\nYou know it wasn\'t wh
at I meant\nCall me up, let me know that you got this message that I\'m leavin
g for you\n\'Cause I\'d hate that you left without hearing the words that I ne
eded you to\n\nAnd I hope you find it\nWhat you\'re looking for\nI hope it\'s
everything you dreamed your life could be\nAnd so much more\n\nAnd I hope you
\'re happy, wherever you are\nI wanted you to know that\nAnd nothing\'s gonna
change that\nI hope you find it\n\nWhatever it is out there that you were miss
ing here\n\nWell, I hope you find it\nWhat you\'re looking for\nI hope it\'s e
verything you dreamed your life could be\nAnd so much more\n\nAnd I hope you
\'re happy wherever you are\nI wanted you to know that\nAnd nothing\'s gonna c
hange that\nI hope you find it\nI hope you find it\nOh, oh\n', 'cher_classifie
d1a': '"Classified 1A"\n\n\n\r\nI know now how much I love you\nI knew it sure
ly when I saw my blood\nI cried with my wet eyes\nI said "I love you"\nI said
"I love you" dying in the mud\nIt\'s funny you know I\'m not the one that feel
s bad\nSome guy is gonna knock at our front door\nHoney he\'s gonna try and te
ll you in a nice way\nThat Mrs., you\'re not Mrs. anymore.\n\nJust one more ti
me I wish that you could see you\nJust one more time I wish that you were near
\nJust one more time I wish that you could hear me\nBut bitterness won\'t make
me reappear\nI love you, my God how I love you\nI see you all around me\nIt\'s
time now it\'s time to say boodnight\nBut this time my love,\nI won\'t have to
get up and fight\nI love you\n'}}

In [7]:
```
# Read in the twitter data
import os

# Specify the directory containing the Twitter data
twitter_directory = "/Users/amyou/Desktop/ADS 509/ADS 509 Module 2/M1 Results/
```

```
# Initialize variables to count the number of artists and descriptions
num_artists = 0
total_descriptions = 0

# Iterate over the files in the Twitter directory
for filename in os.listdir(twitter_directory):
    filepath = os.path.join(twitter_directory, filename)
    # Check if the file is a text file
    if filename.endswith(".txt"):
        # Open the file and read its contents
        with open(filepath, "r", encoding="utf-8") as file:
            # Iterate over each line in the file
            for line in file:
                # Increment the total number of descriptions
                total_descriptions += 1
    # Increment the number of artists
    num_artists += 1

# Print the summary
print("Number of artists:", num_artists)
print("Total number of descriptions:", total_descriptions)
```

```
Number of artists: 5
Total number of descriptions: 8706875
```

Data Cleaning

Now clean and tokenize your data. Remove punctuation chacters (available in the `punctuation` object in the `string` library), split on whitespace, fold to lowercase, and remove stopwords. Store your cleaned data, which must be accessible as an interable for `descriptive_stats`, in new objects or in new columns in your data frame.

In [8]:
```
punctuation = set(punctuation) # speeds up comparison
```

In [9]:
```
# create your clean twitter data here
# Function to clean and tokenize the text
def clean_tokenize_text(text):
    # Remove punctuation characters
    text = text.translate(str.maketrans("", "", string.punctuation))
    # Split the text on whitespace and convert to lowercase
    tokens = text.lower().split()
    # Remove stopwords
    tokens = [token for token in tokens if token not in sw]
    return tokens

# Initialize an empty dictionary to store cleaned and tokenized descriptions
cleaned_data = {}

# Iterate over the files in the Twitter directory
for filename in os.listdir(twitter_directory):
    filepath = os.path.join(twitter_directory, filename)
    # Check if the file is a text file
    if filename.endswith(".txt"):
        # Initialize a list to store cleaned and tokenized descriptions for ea
        artist_descriptions = []
        # Open the file and read its contents
```

```
        with open(filepath, "r", encoding="utf-8") as file:
            # Iterate over each line in the file
            for line in file:
                # Clean and tokenize the text
                tokens = clean_tokenize_text(line)
                # Append the tokens to the list of descriptions
                artist_descriptions.append(tokens)
        # Store the cleaned and tokenized descriptions for the artist
        artist_name = os.path.splitext(filename)[0]
        cleaned_data[artist_name] = artist_descriptions

# Print the first few cleaned and tokenized descriptions for each artist
for artist, descriptions in cleaned_data.items():
    print(f"Artist: {artist}")
    for i, description in enumerate(descriptions[:3]):
        print(f"Description {i+1}: {description}")
    print()
```

```
Artist: cher_followers_data
Description 1: ['screenname', 'name', 'id', 'location', 'followerscount', 'fri
endscount', 'description']
Description 2: ['hsmcnp', 'country', 'girl', '35152213', '1302', '1014']
Description 3: ['horrormomy', 'jeny', '742153090850164742', 'earth', '81', '51
4', 'Proud', 'supporter', 'of', 'messy', 'buns', 'leggings']

Artist: robynkonichiwa_followers_data
Description 1: ['screenname', 'name', 'id', 'location', 'followerscount', 'fri
endscount', 'description']
Description 2: ['angelxoarts', 'angelxo', '1424055675030806529', 'zacatlan',
'puebla', 'mexico', '29', '535', 'love', 'chill', '•facebook', 'instagram', 's
oundcloud', 'angelxoarts•', 'httpstco447okklkza…']
Description 3: ['songsfornikola', 'johnny', '1502717352575651840', '6', '318',
'books', 'movies', 'music', 'nature', 'tv', 'shows', 'og', 'sweetee', 'since',
'12', 'thanks', 'youtube', 'recommending', 'feeling', 'homepage', '♥']

Artist: cher_followers
Description 1: ['id']
Description 2: ['35152213']
Description 3: ['742153090850164742']

Artist: robynkonichiwa_followers
Description 1: ['id']
Description 2: ['1424055675030806529']
Description 3: ['1502717352575651840']
```

In [10]:

```
# create your clean lyrics data here

# Function to clean and tokenize the text
def clean_tokenize_text(text):
    # Remove punctuation characters
    text = text.translate(str.maketrans("", "", string.punctuation))
    # Split the text on whitespace and convert to lowercase
    tokens = text.lower().split()
    # Remove stopwords
    tokens = [token for token in tokens if token not in sw]
    return tokens

# Initialize an empty dictionary to store cleaned and tokenized lyrics
cleaned_lyrics = {}
```

```
# Iterate over the artists' folders
for artist_folder in os.listdir(lyrics_directory):
    # Ignore non-directory files like .DS_Store
    if not os.path.isdir(os.path.join(lyrics_directory, artist_folder)):
        continue

    artist = artist_folder.lower()  # Use lowercase for consistency

    # Initialize an empty list to store cleaned and tokenized lyrics for each a
    artist_lyrics = []

    # Iterate over the song files in the artist's folder
    for song_file in os.listdir(os.path.join(lyrics_directory, artist_folder)):
        # Read the contents of the song file
        with open(os.path.join(lyrics_directory, artist_folder, song_file), "r"
            lyrics = file.read()

        # Clean and tokenize the lyrics
        tokens = clean_tokenize_text(lyrics)

        # Append the tokens to the list of lyrics for the artist
        artist_lyrics.append(tokens)

    # Store the cleaned and tokenized lyrics for the artist
    cleaned_lyrics[artist] = artist_lyrics

# Print the first few cleaned and tokenized lyrics for each artist
for artist, lyrics in cleaned_lyrics.items():
    print(f"Artist: {artist}")
    for i, song_lyrics in enumerate(lyrics[:3]):
        print(f"Song {i+1} lyrics: {song_lyrics}")
    print()
```

```
Artist: robyn
Song 1 lyrics: ['include', 'really', 'simple', 'single', 'pulse', 'repeated',
'regular', 'interval', 'mmm', 'hmm', 'dont', 'include', 'dont', 'include', 'on
e', 'time', 'records', 'hits', 'two', 'moneymaker', 'shake', 'boom', 'three',
'times', 'lucky', 'dead', 'one', 'time', 'sorry', 'safe', 'two', 'beggar', 'co
mpany', 'three', 'times', 'sinner', 'saint', 'yeah', 'bow', 'wicked', 'vain',
'bow', 'miracle', 'em', 'na', 'na', 'three', 'times', 'devil', 'gone', 'one',
'time', 'fire', 'bring', 'two', 'boogie', 'gotta', 'bang', 'beat', 'three', 't
imes', 'ladies', 'show', 'love', 'talking', 'bout', 'everyone', 'every', 'da
y', 'day', 'world', 'fall', 'apart', 'theres', 'plenty', 'room', 'inside', 'he
art', 'dont', 'include', 'dont', 'include', 'world', 'fall', 'apart', 'still',
'got', 'room', 'inside', 'heart', 'dont', 'include', 'dont', 'include', 'dddon
t', 'include', 'hail', 'mamas', 'hold', 'hail', 'pillar', 'family', 'ones', 'g
ranny', 'take', 'bow', 'one', 'time', 'crazy', 'bent', 'come', 'trannies', 'cl
ick', 'heels', 'praise', 'fugeses', 'gems', 'talking', 'bout', 'everyone', 'ev
ery', 'day', 'day', 'oh', 'yeah', 'world', 'fall', 'apart', 'theres', 'plent
y', 'room', 'inside', 'heart', 'dont', 'include', 'dont', 'include', 'world',
'fall', 'apart', 'still', 'got', 'room', 'inside', 'heart', 'dont', 'include',
'dont', 'include', 'dddont', 'include', 'get', 'beat', 'beat', 'watchamacallit
s', 'whatever', 'whoever', 'like', 'get', 'beat', 'beat', 'watchamacallits',
'whatever', 'whoever', 'like', 'get', 'bam', 'bam', 'watchamacallits', 'whatev
er', 'whoever', 'like', 'get', 'bam', 'bam', 'watchamacallits', 'whatever', 'w
hoever', 'like', 'im', 'talking', 'everyone', 'every', 'day', 'day', 'hey', 'w
orld', 'fall', 'apart', 'theres', 'plenty', 'room', 'inside', 'heart', 'dont',
'include', 'dont', 'include', 'world', 'fall', 'apart', 'still', 'got', 'roo
m', 'inside', 'heart', 'dont', 'include', 'dont', 'include', 'world', 'fall',
'apart', 'theres', 'plenty', 'room', 'inside', 'heart', 'dont', 'include', 'he
y', 'hey', 'world', 'fall', 'apart', 'still', 'got', 'room', 'inside', 'hear
t', 'yeah', 'dont', 'include', 'hey']
Song 2 lyrics: ['electric', 'electric', 'electric', 'natural', 'high', 'electr
ic', 'dont', 'always', 'know', 'electric', 'keep', 'ego', 'aside', 'well', 'el
ectric', 'thing', 'cant', 'deny', 'blood', 'boils', 'without', 'fire', 'day',
'come', 'day', 'go', 'without', 'desire', 'disturbs', 'flow', 'denial', 'nigh
t', 'falls', 'falls', 'take', 'higher', 'electric', 'natural', 'high', 'electr
ic', 'dont', 'always', 'know', 'electric', 'keep', 'ego', 'aside', 'well', 'el
ectric', 'thing', 'cant', 'deny', 'hard', 'question', 'needs', 'easy', 'answe
r', 'recognize', 'accept', 'need', 'censor', 'harder', 'fall', 'higher', 'boun
ce', 'smooth', 'moves', 'keeps', 'buoyancy', 'yeah', 'keeps', 'balance', 'oh
h', 'ah', 'electric', 'natural', 'high', 'electric', 'dont', 'always', 'know',
'electric', 'keep', 'ego', 'aside', 'well', 'electric', 'thing', 'cant', 'den
y', 'ohh', 'electric', 'electric', 'cant', 'deny', 'electric', 'electric', 'el
ectric', 'oooh', 'yeah', 'yeah', 'yeah', 'eyes', 'child', 'love', 'moves', 'su
n', 'stars', 'race', 'heart', 'somebody', 'dies', 'cry', 'think', 'shit', 'hap
pens', 'yeah', 'happens', 'happens', 'thats', 'electric', 'natural', 'high',
'electric', 'dont', 'always', 'know', 'electric', 'keep', 'ego', 'aside', 'wel
l', 'electric', 'thing', 'cant', 'deny', 'electric', 'natural', 'high', 'elect
ric', 'dont', 'always', 'know', 'electric', 'keep', 'ego', 'aside', 'well', 'e
lectric', 'thing', 'cant', 'deny', 'electric']
Song 3 lyrics: ['beach', '2k20', 'wanna', 'go', 'gonna', 'get', 'ok', 'call',
'someone', 'alright', 'ok', 'cant', 'wait', 'go', 'ok', 'ok', 'ok', 'ok', 'o
k', 'wanna', 'go', 'cute', 'place', 'beach', 'really', 'nice', 'food', 'gonn
a', 'get', 'mean', 'right', 'beach', 'come', 'itll', 'cool', 'call', 'someon
e', 'hmm', 'ok', 'place', 'beach', 'gotta', 'tell', 'ya', 'place', 'beach', 'p
arty', 'baby', 'party', 'wanna', 'go', 'gonna', 'get', 'ok', 'call', 'someon
e', 'alright', 'ok', 'come', 'dont', 'wait', 'long', 'wanna', 'baby', 'oh', 'y
eah', 'yeah', 'ok', 'cute', 'place', 'beach', 'really', 'nice', 'food', 'mea
n', 'right', 'beach', 'come', 'itll', 'cool', 'cute', 'place', 'beach', 'reall
y', 'nice', 'food', 'mean', 'right', 'beach', 'come', 'itll', 'cool', 'wanna',
'go', 'ok', 'lets', 'go', 'party', 'call', 'someone', 'alright', 'lets', 'go',
'party', 'ok', 'cute', 'place', 'beach', 'really', 'nice', 'food', 'gonna', 'g
```

```
et', 'mean', 'right', 'beach', 'come', 'itll', 'cool', 'call', 'someone', 'cut
e', 'place', 'beach', 'really', 'nice', 'food', 'mean', 'right', 'beach', 'com
e', 'itll', 'cool', 'ok', 'party', 'party', 'party', 'party', 'place', 'beac
h', 'lets', 'go', 'party', 'lets', 'go', 'party', 'party', 'party', 'lets', 'g
o', 'party', 'oh', 'yeah', 'lets', 'go', 'party', 'lets', 'go', 'party', 'oh',
'yeah', 'lets', 'go', 'party', 'lets', 'go', 'party', 'lets', 'go', 'party',
'oh', 'yeah', 'party', 'baby']

Artist: cher
Song 1 lyrics: ['come', 'stay', 'ill', 'send', 'away', 'false', 'pride', 'il
l', 'forsake', 'life', 'yes', 'ill', 'true', 'true', 'youll', 'come', 'stay',
'lovers', 'past', 'ill', 'leave', 'behind', 'theyll', 'never', 'another', 'min
d', 'ill', 'youll', 'feel', 'free', 'youll', 'come', 'stay', 'promise', 'mad
e', 'faithfully', 'ill', 'keep', 'still', 'decide', 'leave', 'ill', 'try', 'se
e', 'need', 'youll', 'come', 'stay', 'yes', 'ill', 'true', 'true', 'youll', 'c
ome', 'stay', 'live', 'life', 'others', 'ever', 'known', 'know', 'think', 'i
m', 'hardly', 'grown', 'oh', 'thank', 'god', 'last', 'finally', 'see', 'your
e', 'gonna', 'stay', 'see', 'youre', 'gonna', 'stay']
Song 2 lyrics: ['pirate', 'hell', 'sail', 'summer', 'wind', 'blows', 'day', 'e
verybody', 'calls', 'pirate', 'dark', 'handsome', 'way', 'fire', 'eyes', 'li
t', 'fire', 'inside', 'soon', 'feeling', 'much', 'wind', 'waves', 'sea', 'pira
te', 'im', 'gonna', 'take', 'soul', 'want', 'right', 'love', 'know', 'sea', 'w
ont', 'let', 'go', 'pirate', 'love', 'chain', 'know', 'much', 'love', 'turn',
'ship', 'around', 'every', 'time', 'hed', 'sail', 'back', 'wed', 'fall', 'lov
e', 'face', 'would', 'fill', 'wonder', 'places', 'hes', 'knew', 'sweetest', 'l
ove', 'song', 'heard', 'trade', 'winds', 'blow', 'loved', 'way', 'much', 'tel
l', 'secret', 'know', 'pirate', 'im', 'gonna', 'take', 'soul', 'want', 'righ
t', 'love', 'know', 'sea', 'wont', 'let', 'go', 'pirate', 'love', 'chain', 'kn
ow', 'much', 'love', 'turn', 'ship', 'around', 'watch', 'silence', 'another',
'young', 'man', 'goes', 'sea', 'silhouette', 'stirrin', 'painful', 'memory',
'know', 'heart', 'set', 'sail', 'mine', 'set', 'cry', 'cause', 'feel', 'way',
'day', 'daddy', 'said', 'goodbye', 'told', 'pirate', 'im', 'gonna', 'take', 's
oul', 'want', 'right', 'love', 'know', 'sea', 'wont', 'let', 'go', 'pirate',
'love', 'chain', 'know', 'much', 'love', 'turn', 'ship', 'around']
Song 3 lyrics: ['stars', 'never', 'one', 'saying', 'really', 'feel', 'except',
'tonight', 'im', 'bringing', 'everything', 'know', 'thats', 'real', 'stars',
'come', 'go', 'come', 'fast', 'come', 'slow', 'go', 'like', 'last', 'light',
'sun', 'blaze', 'see', 'glory', 'hey', 'gets', 'lonely', 'theres', 'one', 'sha
re', 'shake', 'away', 'hear', 'story', 'people', 'ask', 'fame', 'like', 'athle
tes', 'game', 'break', 'collarbones', 'come', 'swinging', 'us', 'downed', 'u
s', 'crowned', 'lost', 'never', 'found', 'seen', 'live', 'lives', 'sad', 'cafe
s', 'music', 'halls', 'always', 'come', 'singing', 'make', 'theyre', 'young',
'world', 'done', 'dirty', 'job', 'later', 'someone', 'say', 'youve', 'day', 'm
ust', 'make', 'way', 'never', 'know', 'pain', 'living', 'name', 'youd', 'neve
r', 'many', 'years', 'forgetting', 'know', 'well', 'ones', 'get', 'crown', 'le
t', 'yet', 'try', 'make', 'amends', 'without', 'defending', 'perhaps', 'preten
ding', 'never', 'saw', 'eyes', 'grown', 'men', 'twentyfive', 'follow', 'walk',
'ask', 'autographs', 'kiss', 'cheek', 'never', 'believe', 'really', 'love', 'm
ake', 'theyre', 'old', 'perhaps', 'soul', 'theyre', 'afraid', 'bear', 'perhap
s', 'theres', 'nothing', 'stars', 'come', 'go', 'come', 'fast', 'come', 'slo
w', 'go', 'like', 'last', 'light', 'sun', 'blaze', 'see', 'glory', 'seen', 'li
ve', 'lives', 'sad', 'cafes', 'music', 'halls', 'always', 'come', 'singing',
'singing', 'singing', 'singing']
```

Basic Descriptive Statistics

Call your `descriptive_stats` function on both your lyrics data and your twitter data and for both artists (four total calls).

In [11]:
```
# calls to descriptive_stats here

# list the lists
flatten_lyrics = [token for sublist in cleaned_lyrics.values() for token_list :
# descriptive stats for lyrics
print("Descriptive stats for all artists (lyrics data):")
stats = descriptive_stats(flatten_lyrics)
print(f"Number of tokens: {stats[0]}")
print(f"Number of unique tokens: {stats[1]}")
print(f"Lexical diversity: {stats[2]:.3f}")
print(f"Number of characters: {stats[3]}")

# descriptive stats for twitter
description_data = []

for descriptions in description_data:
    print("Descriptive stats for Twitter data:")
    stats = descriptive_stats(descriptions)
    print(f"Number of tokens: {stats[0]}")
    print(f"Number of unique tokens: {stats[1]}")
    print(f"Lexical diversity: {stats[2]:.3f}")
    print(f"Number of characters: {stats[3]}")
    print()
```

```
Descriptive stats for all artists (lyrics data):
There are 51143 tokens in the data.
There are 4664 unique tokens in the data.
There are 246421 characters in the data.
The lexical diversity is 0.091 in the data.
Most common tokens:
love: 1279
im: 812
know: 794
dont: 741
baby: 541
like: 503
youre: 502
got: 495
time: 424
never: 408
go: 369
oh: 366
see: 365
gonna: 348
one: 345
come: 342
cant: 329
heart: 328
take: 322
get: 300
say: 300
want: 295
back: 271
right: 265
cause: 265
way: 263
make: 259
ill: 243
away: 235
ive: 232
feel: 227
man: 225
could: 222
still: 220
let: 220
theres: 214
tell: 202
night: 193
every: 187
better: 187
world: 184
wanna: 181
well: 180
gone: 176
dance: 175
ever: 174
keep: 170
beat: 163
life: 163
good: 163
day: 162
wont: 162
need: 160
yeah: 157
```

```
long: 157
give: 157
think: 156
believe: 156
really: 148
thing: 143
find: 143
something: 142
look: 142
always: 141
girl: 139
around: 139
killing: 137
enough: 136
gotta: 133
would: 133
mind: 130
stop: 126
eyes: 125
walk: 123
show: 118
youll: 117
chorus: 117
home: 116
said: 114
nothing: 114
thats: 113
aint: 113
even: 111
youve: 110
fall: 109
made: 102
little: 101
hold: 100
hear: 97
cry: 94
place: 94
without: 93
stay: 93
work: 91
free: 91
bang: 89
mine: 89
another: 88
hes: 87
hey: 86
face: 86
alone: 86
try: 84
song: 84
strong: 82
kiss: 81
run: 78
inside: 77
true: 77
things: 76
us: 76
somebody: 75
hard: 74
wait: 74
```

```
remember: 73
live: 73
though: 73
tonight: 73
turn: 73
ooh: 73
times: 72
someone: 72
whats: 72
hurt: 71
knew: 71
much: 71
door: 70
going: 68
light: 68
lets: 67
people: 67
sleep: 66
set: 66
ah: 65
dreams: 64
everything: 64
music: 64
woman: 64
hope: 63
new: 63
body: 63
put: 63
boy: 63
forever: 63
left: 63
kind: 62
tears: 62
rain: 62
shes: 62
lonely: 62
dancing: 61
leave: 61
till: 61
looking: 60
coming: 60
honey: 60
two: 59
sun: 59
sometimes: 59
far: 59
break: 59
wish: 58
many: 58
thought: 58
bad: 57
name: 57
deep: 56
last: 56
young: 56
boom: 55
wrong: 55
seen: 55
move: 55
play: 55
```

```
ships: 1
frisco: 1
remain: 1
restin: 1
funnyface: 1
puppy: 1
akward: 1
filling: 1
chairs: 1
baltimore: 1
vow: 1
wawanna: 1
insecure: 1
bein: 1
starin: 1
fourleaf: 1
bouquet: 1
disease: 1
ageless: 1
sublime: 1
savor: 1
mere: 1
mortals: 1
3x: 1
resistance: 1
fieldday: 1
carrying: 1
mostly: 1
hooray: 1
paranoia: 1
strikes: 1
disco: 1
scenes: 1
highs: 1
lows: 1
tipped: 1
datell: 1
due: 1
disappointed: 1
celebrate: 1
celebrity: 1
cupid: 1
chastitys: 1
manner: 1
speech: 1
mends: 1
hurtins: 1
defends: 1
bothered: 1
hmmmmm: 1
pleasing: 1
byebye: 1
baked: 1
tray: 1
muffins: 1
wetwash: 1
drank: 1
dishes: 1
scrubbed: 1
actin: 1
```

```
nows: 1
private: 1
forward: 1
bathed: 1
fearful: 1
sometmes: 1
satsify: 1
sunrise: 1
remembers: 1
honesty: 1
dancer: 1
sorries: 1
daisy: 1
confessions: 1
hyena: 1
howling: 1
roaming: 1
soldier: 1
marches: 1
bleed: 1
stab: 1
common: 1
fairytale: 1
weavin: 1
jokes: 1
workin: 1
fulltime: 1
sewn: 1
citizen: 1
terrified: 1
photostat: 1
bonify: 1
businessman: 1
american: 1
travels: 1
powie: 1
stash: 1
unofficial: 1
woo: 1
shook: 1
override: 1
whyd: 1
ridin: 1
lates: 1
fail: 1
arent: 1
classified: 1
1a: 1
reappear: 1
boodnight: 1
Number of tokens: 51143
Number of unique tokens: 4664
Lexical diversity: 0.091
Number of characters: 246421
```

Q: How do you think the "top 5 words" would be different if we left stopwords in the data?

A: It will affect the frequency of common words and impact meaningful words. If we left stopwords in the data, meaningful words that convey specific information may not appear in

the top 5 words. With stopwords included, important words are easily overseen.

Q: What were your prior beliefs about the lexical diversity between the artists? Does the difference (or lack thereof) in lexical diversity between the artists conform to your prior beliefs?

A: My prior beliefs regarding the diversity of vocabulary usage among artists would be influenced by factors such as their unique styles and lyrical themes. Whether the difference in lexical diversity between artists aligns with my expectations would depend on the specific characteristics of the artists and the results of the analysis. If an artist known for diverse themes and complex lyrics indeed demonstrates higher lexical diversity, it would confirm my expectations.

Specialty Statistics

The descriptive statistics we have calculated are quite generic. You will now calculate a handful of statistics tailored to these data.

1. Ten most common emojis by artist in the twitter descriptions.
2. Ten most common hashtags by artist in the twitter descriptions.
3. Five most common words in song titles by artist.
4. For each artist, a histogram of song lengths (in terms of number of tokens)

We can use the `emoji` library to help us identify emojis and you have been given a function to help you.

In [12]:
```
assert(emoji.is_emoji("❤"))
assert(not emoji.is_emoji(":-)"))
```

Emojis 😁

What are the ten most common emojis by artist in the twitter descriptions?

In [13]:
```
# Your code here

# Define the directory containing the Twitter data files
twitter_directory = "/Users/amyou/Desktop/ADS 509/ADS 509 Module 2/M1 Results/

# Initialize an empty dictionary to store descriptions data
descriptions_data = {}

# Iterate over each file in the directory
for filename in os.listdir(twitter_directory):
    if filename.endswith(".txt"):
        artist = filename[:-4]
        file_path = os.path.join(twitter_directory, filename)

        # Load Twitter data from the file
        with open(file_path, "r") as file:
```

```
            # Initialize a list to store descriptions for the current artist
            descriptions_data[artist] = []

            # Iterate over each line in the file
            for line in file:
                description = line.strip()

                # Append the description to the list for the current artist
                descriptions_data[artist].append(description)

# Function to check if a character is an emoji
def is_emoji(character):
    return character in emoji.EMOJI_DATA

# Function to extract emojis from a text
def extract_emojis(text):
    return [c for c in text if is_emoji(c)]

# Dictionary to store emojis for each artist
artist_emojis = {}

# Iterate over each artist's Twitter descriptions
for artist, descriptions in descriptions_data.items():
    # List to store emojis for the current artist
    artist_emojis[artist] = []

    # Iterate over each description for the current artist
    for description in descriptions:
        # Extract emojis from the description
        emojis = extract_emojis(description)
        artist_emojis[artist].extend(emojis)

# Dictionary to store the top ten emojis for each artist
top_ten_emojis_by_artist = {}

# Iterate over each artist's emojis
for artist, emojis in artist_emojis.items():
    # Count the occurrences of each emoji
    emoji_counts = Counter(emojis)

    # Sort the emojis based on their frequency
    sorted_emojis = sorted(emoji_counts.items(), key=lambda x: x[1], reverse=T

    # Retrieve the top ten most common emojis
    top_ten_emojis = sorted_emojis[:10]

    # Store the top ten emojis for the current artist
    top_ten_emojis_by_artist[artist] = top_ten_emojis

# Print the top ten emojis by artist
for artist, top_emojis in top_ten_emojis_by_artist.items():
    print(f"Top ten emojis for {artist}:")
    for emoji_char, count in top_emojis:
        print(f"{emoji_char} - {count} occurrences")
    print()
```

```
Top ten emojis for cher_followers_data:
❤ - 94506 occurrences
🌈 - 66291 occurrences
♥ - 48059 occurrences
🏳 - 47174 occurrences
✨ - 45846 occurrences
🌊 - 31234 occurrences
💙 - 31050 occurrences
🏻 - 25195 occurrences
✌ - 21963 occurrences
💜 - 21571 occurrences

Top ten emojis for robynkonichiwa_followers_data:
🌈 - 6086 occurrences
❤ - 5635 occurrences
🏳 - 4641 occurrences
♥ - 4249 occurrences
✨ - 3217 occurrences
🏻 - 1751 occurrences
✌ - 1495 occurrences
♀ - 1347 occurrences
🏽 - 1340 occurrences
💙 - 1200 occurrences

Top ten emojis for cher_followers:

Top ten emojis for robynkonichiwa_followers:
```

Hashtags

What are the ten most common hashtags by artist in the twitter descriptions?

In [14]:

```python
# Your code here
# Iterate over each file in the directory
for filename in os.listdir(twitter_directory):
    if filename.endswith(".txt"):
        artist = filename[:-4]
        file_path = os.path.join(twitter_directory, filename)

        # Load Twitter data from the file
        with open(file_path, "r", encoding="utf-8") as file:
            # Initialize a list to store descriptions for the current artist
            descriptions_data[artist] = []

            # Iterate over each line in the file
            for line in file:
                description = line.strip()

                # Append the description to the list for the current artist
                descriptions_data[artist].append(description)

# Function to extract hashtags from a text
def extract_hashtags(text):
    return re.findall(r"#(\w+)", text)

# Dictionary to store hashtags for each artist
artist_hashtags = {}
```

```
# Iterate over each artist's Twitter descriptions
for artist, descriptions in descriptions_data.items():
    # List to store hashtags for the current artist
    artist_hashtags[artist] = []

    # Iterate over each description for the current artist
    for description in descriptions:
        # Extract hashtags from the description
        hashtags = extract_hashtags(description)
        artist_hashtags[artist].extend(hashtags)

# Dictionary to store the top ten hashtags for each artist
top_ten_hashtags_by_artist = {}

# Iterate over each artist's hashtags
for artist, hashtags in artist_hashtags.items():
    # Count the occurrences of each hashtag
    hashtag_counts = Counter(hashtags)

    # Sort the hashtags based on their frequency
    sorted_hashtags = sorted(hashtag_counts.items(), key=lambda x: x[1], revers

    # Retrieve the top ten most common hashtags
    top_ten_hashtags = sorted_hashtags[:10]

    # Store the top ten hashtags for the current artist
    top_ten_hashtags_by_artist[artist] = top_ten_hashtags

# Print the top ten hashtags by artist
for artist, top_hashtags in top_ten_hashtags_by_artist.items():
    print(f"Top ten hashtags for {artist}:")
    for hashtag, count in top_hashtags:
        print(f"#{hashtag} - {count} occurrences")
    print()
```

```
Top ten hashtags for cher_followers_data:
#BLM - 10100 occurrences
#Resist - 6161 occurrences
#BlackLivesMatter - 4888 occurrences
#resist - 3860 occurrences
#FBR - 3330 occurrences
#1 - 3111 occurrences
#TheResistance - 3044 occurrences
#blacklivesmatter - 2738 occurrences
#Resistance - 1953 occurrences
#RESIST - 1878 occurrences

Top ten hashtags for robynkonichiwa_followers_data:
#BlackLivesMatter - 356 occurrences
#BLM - 345 occurrences
#1 - 228 occurrences
#blacklivesmatter - 222 occurrences
#music - 175 occurrences
#Music - 114 occurrences
#EDM - 87 occurrences
#LGBTQ - 76 occurrences
#blm - 60 occurrences
#TeamFollowBack - 59 occurrences

Top ten hashtags for cher_followers:

Top ten hashtags for robynkonichiwa_followers:
```

Song Titles

What are the five most common words in song titles by artist? The song titles should be on the first line of the lyrics pages, so if you have kept the raw file contents around, you will not need to re-read the data.

In [15]:
```
# Your code here

# Define the directory containing the lyrics files
lyrics_directory = "/Users/amyou/Desktop/ADS 509/ADS 509 Module 2/M1 Results/ly

# Initialize an empty dictionary to store titles data
titles_data = {}

# Function to clean and tokenize text
def clean_and_tokenize(text):
    # Convert text to lowercase
    text = text.lower()
    # Remove punctuation and special characters
    text = re.sub(r'[^\w\s]', '', text)
    # Split text into words
    words = text.split()
    return words

# Iterate over each artist folder in the directory
for artist_folder in os.listdir(lyrics_directory):
    # Construct the full path to the artist folder
    artist_folder_path = os.path.join(lyrics_directory, artist_folder)
```

```
    # Check if the item in the directory is a folder
    if os.path.isdir(artist_folder_path):
        # Initialize the list to store titles for the current artist
        titles_data[artist_folder] = []

        # Iterate over each file in the artist folder
        for filename in os.listdir(artist_folder_path):
            if filename.endswith(".txt"):  # Only process text files
                file_path = os.path.join(artist_folder_path, filename)

                # Load lyrics data from the file
                with open(file_path, "r", encoding="utf-8") as file:
                    # Read the first line (title) of the file
                    title = file.readline().strip()

                    # Append the title to the list for the current artist
                    titles_data[artist_folder].append(title)

# Dictionary to store the top five words for each artist
top_five_words_by_artist = {}

# Iterate over each artist's titles
for artist, titles in titles_data.items():
    # List to store words for the current artist
    words = []

    # Iterate over each title for the current artist
    for title in titles:
        # Clean and tokenize the title
        tokens = clean_and_tokenize(title)
        words.extend(tokens)

    # Count the occurrences of each word
    word_counts = Counter(words)

    # Sort the words based on their frequency
    sorted_words = sorted(word_counts.items(), key=lambda x: x[1], reverse=Tru

    # Retrieve the top five most common words
    top_five_words = sorted_words[:5]

    # Store the top five words for the current artist
    top_five_words_by_artist[artist] = top_five_words

# Print the top five words by artist
for artist, top_words in top_five_words_by_artist.items():
    print(f"Top five words in song titles for {artist}:")
    for word, count in top_words:
        print(f"{word} - {count} occurrences")
    print()
```

```
Top five words in song titles for robyn:
me - 11 occurrences
you - 8 occurrences
the - 8 occurrences
my - 8 occurrences
love - 6 occurrences

Top five words in song titles for cher:
the - 54 occurrences
you - 41 occurrences
love - 38 occurrences
i - 32 occurrences
to - 28 occurrences
```

Song Lengths

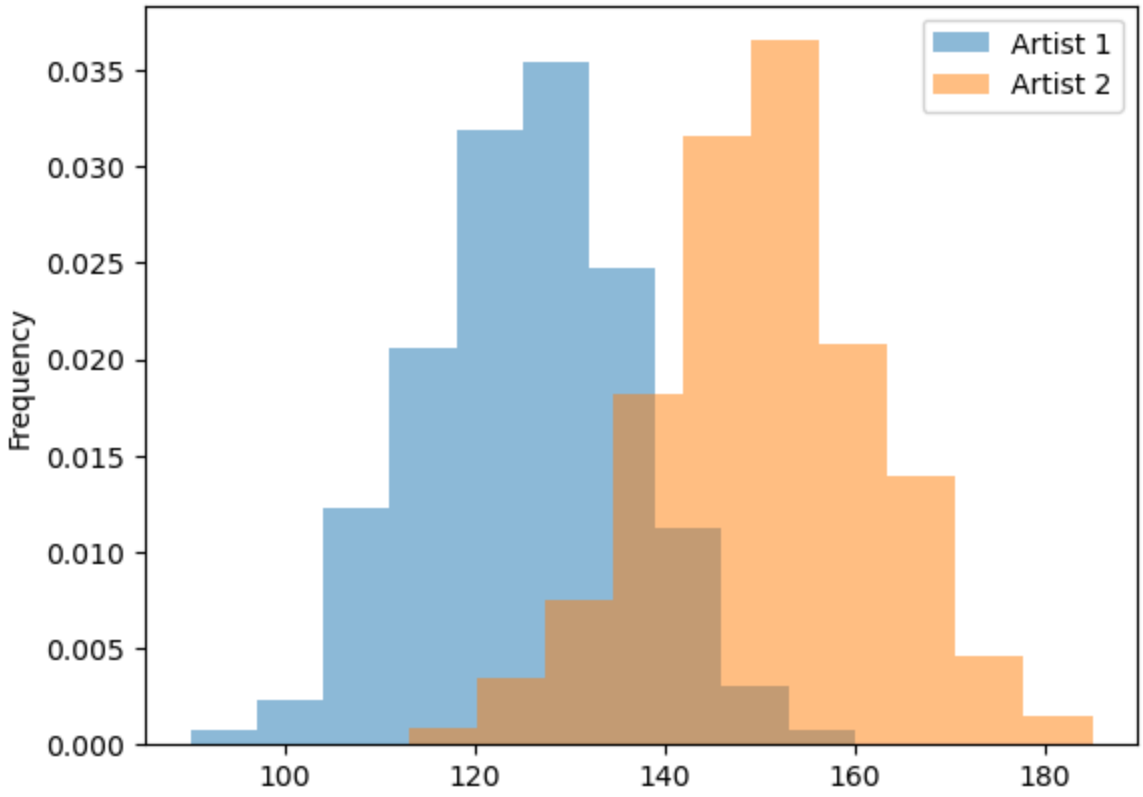
For each artist, a histogram of song lengths (in terms of number of tokens). If you put the song lengths in a data frame with an artist column, matplotlib will make the plotting quite easy. An example is given to help you out.

In [16]:
```
num_replicates = 1000

df = pd.DataFrame({
    "artist" : ['Artist 1'] * num_replicates + ['Artist 2']*num_replicates,
    "length" : np.concatenate((np.random.poisson(125,num_replicates),np.random
})

df.groupby('artist')['length'].plot(kind="hist",density=True,alpha=0.5,legend=
```

Out[16]:
```
artist
Artist 1    AxesSubplot(0.125,0.11;0.775x0.77)
Artist 2    AxesSubplot(0.125,0.11;0.775x0.77)
Name: length, dtype: object
```



Since the lyrics may be stored with carriage returns or tabs, it may be useful to have a function that can collapse whitespace, using regular expressions, and be used for splitting.

Q: What does the regular expression `'\s+'` match on?

A: \s+ will match on any sequences of whitespace characters of single or multiple spaces, tabs, any length, and newlines.

In [17]:
```
collapse_whitespace = re.compile(r'\s+')

def tokenize_lyrics(lyric) :
    """strip and split on whitespace"""
    return([item.lower() for item in collapse_whitespace.split(lyric)])
```

In [18]:
```
# Your lyric length comparison chart here.

# Function to read lyrics from a file and return the number of words
def count_words_in_lyrics(file_path):
    with open(file_path, 'r') as file:
        lyrics = file.read()
    tokens = tokenize_lyrics(lyrics)
    return len(tokens)

# Function to tokenize lyrics
collapse_whitespace = re.compile(r'\s+')
def tokenize_lyrics(lyric):
    """Strip and split on whitespace"""
    return [item.lower() for item in collapse_whitespace.split(lyric)]

# Paths to the artist folders
lyrics_base_path = '/Users/amyou/Desktop/ADS 509/ADS 509 Module 2/M1 Results/ly
```

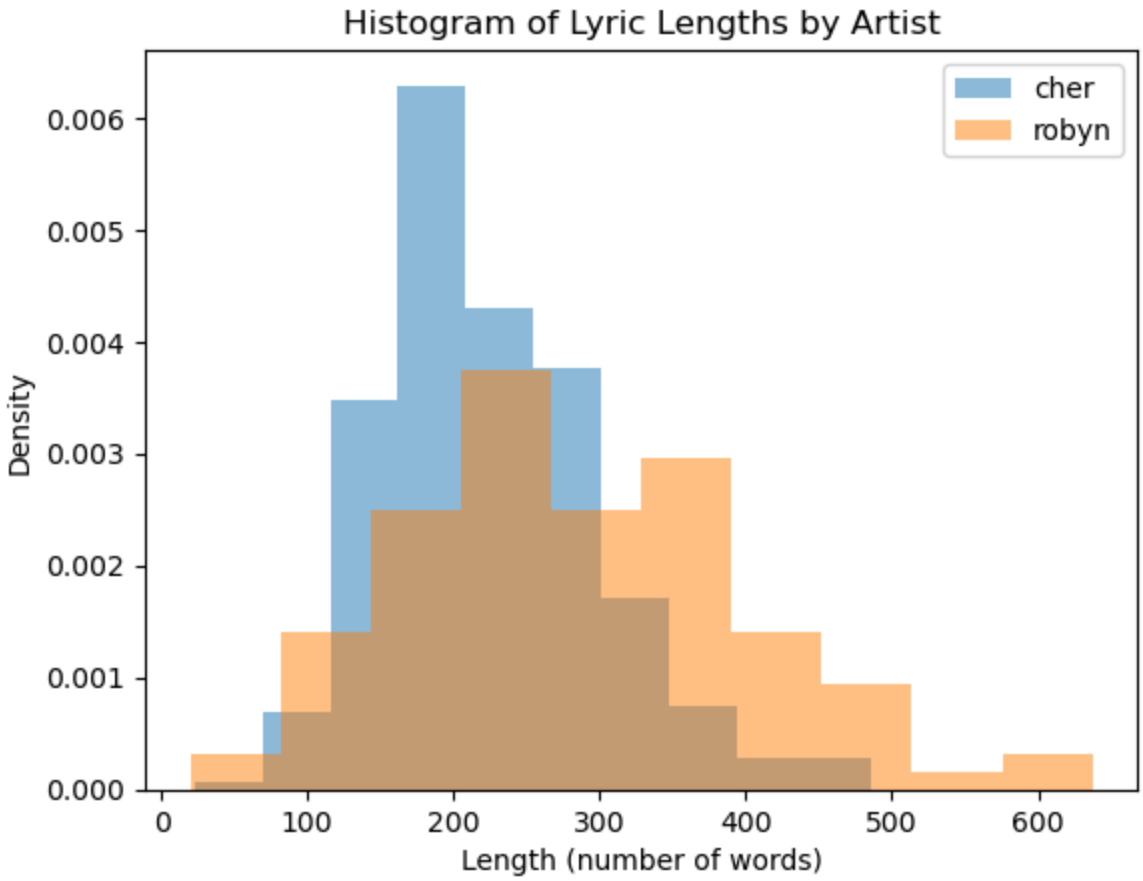
```
artist_folders = ['cher', 'robyn']

# Initialize lists to store lyric lengths
lyric_lengths = {'cher': [], 'robyn': []}

# Process each artist's folder
for artist in artist_folders:
    artist_path = os.path.join(lyrics_base_path, artist)
    for file_name in os.listdir(artist_path):
        file_path = os.path.join(artist_path, file_name)
        if os.path.isfile(file_path):
            length = count_words_in_lyrics(file_path)
            lyric_lengths[artist].append(length)

# Convert to DataFrame
data = []
for artist, lengths in lyric_lengths.items():
    data.extend([(artist, length) for length in lengths])
df = pd.DataFrame(data, columns=['artist', 'length'])

# Plot histograms
df.groupby('artist')['length'].plot(kind='hist', density=True, alpha=0.5, leger
plt.xlabel('Length (number of words)')
plt.ylabel('Density')
plt.title('Histogram of Lyric Lengths by Artist')
plt.legend()
plt.show()
```



In [ ]: