CS 196: Data Hackerspace

Assignment 2: APIs and Data Visualization

October 2, 2018

Instructions: This homework will be due electronically on Monday, October 15th at 11:59pm as a submission on Github. Please write your answers as a `.py` file (Jupyter notebooks will not be graded) and commit it to the same repository as last week (called `data_hackerspace_homework`). Skeleton code is given as `hw2.py`. Do not edit anything outside of the given functions.

1 APIs

1.1 Music API

For this question, we will be using the Lyrics.ovh API to retrieve song lyrics for an artist and a song. Your goal is to write the function `lyrics_word_count_easy` which accepts three string parameters: artist, song, and phrase, and returns the count of the phrase in the lyrics. To make it simple, write the function to not be case sensitive and allow the phrase to be a substring of a word in the lyrics. If the song is not found, return -1.
Example: If the function is called with these parameters: `lyrics_word_count("Rick Astley", "Never Gonna Give You Up", "never")`, the output should be 40.
Hints / Useful Links:

1. Python Regular Expressions
2. Lyrics.ovh API Hack (only use if you are really stuck)

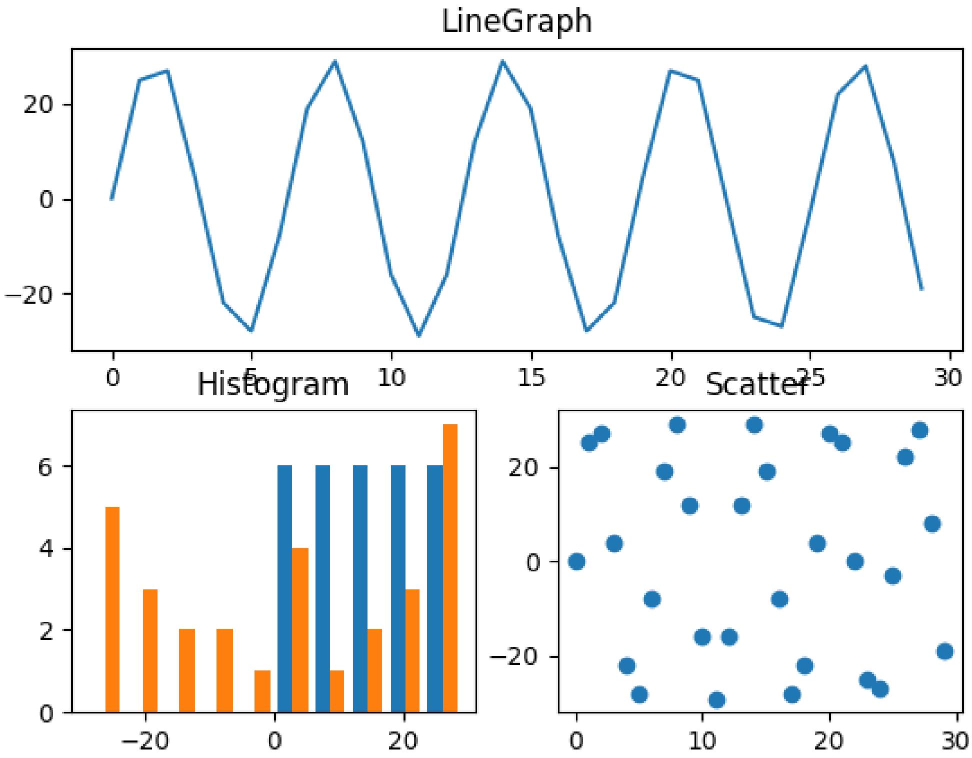
1.2 Music API 2.0

This question is an extension to the previous question. In this question, your goal is to write the function `lyrics_word_count` which accepts two string parameters: artist and phrase, and returns the count of the phrase across all of artist's songs. Return -1 if artist is not found.
Note: This is an open-ended question. You are allowed to use any resources/APIs you would like in order to solve this problem. You might find that using more than one API is helpful, in which case feel free to do that. Also, we understand that all APIs might not have the same list of songs for each artist, and we will take that into consideration while grading your labs. Here are some APIs you can use to solve this problem: MusicGraph (recommended), Musixmatch, Spotify, Lyrics.ovh.

2 Data Visualization

Your goal is to write the function `visualize` which creates a `matplotlib` with three subplots in a 2x2 grid, where the first subplot is a line graph which spans the entire top row, the second is a histogram on the bottom left, and the third is a scatter plot on the bottom right. The data for each plot is given below: `x = np.array([0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10., 11., 12., 13., 14., 15., 16., 17., 18., 19., 20., 21., 22., 23., 24., 25., 26., 27., 28., 29.]), y = np.array([0., 25., 27., 4., -22., -28., -8., 19., 29., 12., -16., -29., -16., 12., 29., 19., -8., -28., -22., 4., 27., 25., -0., -25., -27., -3., 22., 28., 8., -19.])` Each plot uses both `x` and `y`. The line graph and scatter plot should have `x` on the horizontal axis and `y` on the vertical axis, and the histogram should show both `x` and `y` on the same plot. Use the default style, X and Y limits, colours, and bin size. Additionally, add titles to each plot: `LineGraph`, `Histogram`, `Scatter`.

This function takes in no parameters and should return `plt.show()`. In the development environment, this should always return `None`, but we will be using a custom matplotlib which allows for autograding. Your plot should look like this:


LineGraph
Histogram
Scatter