Spring 2023 CSE 532 Final Programming Portion

Data Description: The four attached json files (**sub_basketball.json**, **sub_machinelearning.json**, **sub_personalfinance.json**, **sub_louisville.json**), represent four separate classes of top 100 submissions from subreddits with the corresponding suffix. For example, **sub_basketball.json** has the top 100 submissions taken from the basketball subreddit.
Each submission has up to 5 characteristics (stored as key-value pairs): **Title**, **id**, **Score**, **Num_Comments**, **Date***.
* Many submissions are missing one or more these characteristics: see instructions below.

Instructions (Five parts in total):

Part 1. Load each json file into Python (dictionaries or data-frames are recommended) and perform the following:
a. discard any submissions that lack one or more of the 5 characteristics (DataFrames may require finesse here)
b. after doing so, discard the **id** characteristic, as it will *not* be used with future tasks.

For each collection, save the modified list of submissions *back* into a **new** json file with the name **prep_class#.json**, where # matches the order of json files cited above (0=…basketball, 1= …machinelearning, 2=...personalfinance, 3=…louisville). You should have files **prep_class0.json**, **prep_class1.json**, **prep_class2.json**, and **prep_class3.json** at the end of the process, all of which have the modified submissions.

Part 2. For **each** *modified* collection of submissions (i.e. after the transformation from part 1) use textblob to calculate the sentiment *polarity* of each submission's **title** (ignore the subjectivity, and don't use the NaiveBayesAnalyzer) Create a 2x2 (subplot) grid of bar plots, where each bar plot includes the # submissions per subreddit that have negative (polarity<-0.25), neutral (-0.25≤polarity≤0.25) , and positive (polarity>0.25) sentiment. You should have 3 bars per plot (one bar for negative, one for neutral, one for positive), and 4 bar plots total (one per subreddit).

Part 3. Pool together all *modified* submissions into a single collection (list, DataFrame, etc.) , but maintain a combined *secondary list* that labels each submission by its class (0, 1, 2, or 3). Ex: If there are 96 submissions from the basketball subreddit in the pooled list of tweets, the first 96 elements of the secondary list should be 0.

Part 4. Assume your combined collections have a length of **n** submissions in total. Your next goal is to construct a **n x 5 numpy feature array** suited for machine learning, where each array row matches the corresponding index in your collections, and the 5 array columns represent the features for the submission at that position as follows:
Feature 1: The character length of the submission's title.
Feature 2: The sentiment of the submission's title.
Feature 3: The submission's **Score**
Feature 4: The submission's number of comments (**Num_Comments**).
Feature 5: The *year* of the submission (note that if you are reading your json file into a DataFrame, you may want to look into using *convert_dates=False*, as it may make producing this feature easier.)

For example, the first row in your feature array may look like the below:

```
[46.   , -0.2   , 1356.0 ,  249.0, 2020.   ]
```

Part 5. Convert your secondary list of classes into a 1D array, and then perform 10-fold cross-validation using two distinct classification estimators (either the ones we used in class, or those of your own choosing) to determine the accuracy available in using the features from part 4 in predicting the class of a given submission. For full credit, you should produce both classifier accuracies and a confusion matrix for the most accurate classifier (in similar vein to Part 1 of HW 4) . Write a paragraph (comments or a separate text file are fine for this) discussing whether or not you think the features and chosen classifiers provide acceptable accuracy for the task.