Assignment 2: Analyzing Data from Customer Reviews

Learning Objectives:

- Extracting information from unstructured text data.
- Manipulating data structures such as lists, dictionaries, and Pandas DataFrames.
- Parsing text using string methods such as `startswith`, `strip`, `lower` and `split`.
- Automating repetitive tasks using loops and controlling the flow of execution.

Description

One challenge for marketing and sales teams is to analyze large amounts of text data such as customer reviews, and learning to program would give you an edge by reducing the manual labor needed. In this assignment, you will create a python module named `"customer_reviews.py"` containing a function called `analyze`, with 3 input arguments:

- reviewFile: the path to a `.txt` file containing customer reviews. The format is assumed to be the same as the test data sets attached to this assignment.
- stopwordFile: the path to a `.txt` file containing a list of commonly occurring English words to ignore (such as “a”, “the”, “and”, etc). The format is assumed to be the same as the `"stopwords.txt"` file attached to this assignment.
- numWords (default value 20): An integer specifying the number of rows to include in the DataFrame object returned.

The function should return a Pandas DataFrame, in whichs each row is indexed by a word. There are two columns, named `Occurrences` and `Reviews` respectively. For each row corresponding to a word, the column `Occurrences` should specify the number of times the word appears in all of the reviews; the column `Reviews` should specify the number of reviews the word appears in. For example, if the word `perfect` appears 5 times in a single review, this should contribute 5 to the first column and 1 to the second column.

The words contained in the returned DataFrame should not contain any of the words in the stopwordFile. (In other words, you should ignore in the analysis commonly occurring English words such as “a”, “the”, “and”, etc.) Moreover, your code should remove punctuations and white spaces, and put all words in lower case. In other words, the string `"HAPPY"` should be treated as the same word as `"happy"` and `"(PeRfEcT!!!)\n"` should be treated the same as `"perfect"`.

The DataFrame returned should be sorted in decreasing order of the column `Occurrences`. Moreover, you should filter for only the top `numWords` number of rows.

Sample Output

Once you complete the `reviews.py` module and download all of the data sets attached to this assignment and save them all in the same directory, you should be able to run the following code in a Jupyter notebook in the same directory and get the following output. (It is okay if the numbers in your DataFrame are not identically the same as below, but they should be close.)

```
[7]: from reviews import analyze
     stopwordFile='stopwords.txt'
     analyze('sophie_the_girrafe.txt',stopwordFile,10)
```

```
           Occurrences  Reviews
sophie             812      394
toy                651      391
```

```
baby              521      318
loves             395      318
teething          318      244
son               296      224
chew              272      202
months            256      188
teether           252      181
daughter          232      158
```

```
[11]: analyze('baby_einstein_musical_toy.txt',stopwordFile,6)
```

```
        Occurrences  Reviews
toy             615      356
music           480      314
baby            371      251
loves           270      221
months          267      187
lights          265      230
```

```
[4]: ls *.txt
```

```
baby_einstein_musical_toy.txt   sophie_the_girrafe.txt
fisher_price_booster_seat.txt   stopwords.txt
rainforest_jumperoo.txt         summer_infant_changing_pad.txt
```

```
[12]: analyze('fisher_price_booster_seat.txt',stopwordFile,5)
```

```
       Occurrences  Reviews
chair          804      336
tray           596      264
seat           541      258
easy           437      288
high           380      224
```

```
[13]: analyze('rainforest_jumperoo.txt',stopwordFile,5)
```

```
          Occurrences  Reviews
baby              341      181
months            307      184
toys              299      190
loves             279      201
jumperoo          270      167
```

```
[14]: analyze('summer_infant_changing_pad.txt',stopwordFile,5)
```

```
          Occurrences  Reviews
pad               575      287
changing          548      316
baby              261      188
table             197      152
great             182      152
```

Steps

There are many correct ways of solving the assignment and all will be given full credit. However, if you don't know where to start, you can follow the following incremental steps.

1. Create two smaller data files for development purposes, `"reviews1.txt"` and `"reviews3.txt"`. The first file should contain one review (from any of the test data sets) and the second file should contain three reviews. You can use any text editor for this (such as Notepad on Windows or TextEdit on Mac). Save these files in the same directory as the Jupyter notebook you will use for code development.

2. In a Jupyter notebook, write code to print out the content of the file `"reviews1.txt"` that you created, but filtering only for the lines containing the review text. (Do not print the lines starting with `"Review #:"` and `"Date:"`.)

3. Modify the code from the above step to split each line into words (splitting by space) and count the number of occurrences of each word using a dictionary. (See the histogram example from Session 6.) Print the dictionary at the end. (Only run this with `"reviews1.txt"` as otherwise the output would be huge.)

4. Modify the code from the above step so that before you count each word, you first strip away all surrounding white spaces, then strip away all punctuations, then change to lower case. For example

- `"Happy!\n"` should become `"happy"`.
- `"GiRaFFes'\n\r"` should become `"giraffes"`.

To strip away punctuation, you can use the following code.

```
[15]: s='"Happy"!!!?'
      import string
      s=s.strip(string.punctuation)
      s
```

```
'Happy'
```

5. In a separate notebook cell, initialize an empty list called `stopwords`, and use a `for` loop to read each line of the file `stopwords.txt`, stripping away all leading and trailing whitespaces, and appending the line to the list. You should check that afterward, the list is composed of words without the `"\n"` new line character at the end.

6. Modify the code in step 4 so that it will skip words that are in the list `stopwords`. By now, you would have successfully counted the number of occurrences of each word from one review, skipping punctuations and commonly occurring English words.

7. Modify the above code that that the input file is `"reviews3.txt"` and you reset the dictionary and print its content whenever you encounter a new review (i.e. a line starting with `"Review #:"`. Print the content of the dictionary at the end of the program as well, which corresponds to the last review. Reseting the dictionary allows you to process each review separately.

8. Modify the above code so that you have two additional dictionaries, called `occurrrences` and `reviews`. The first counts the total number of occurrences of each word in all reviews, and the second counts the number of reviews each word appears. Each time you encounter a new review or when the program ends, you should update `occurrences` by

adding the number of occurrences of the most recent review to it, and you should update `reviews` by incrementing a word by 1 if the most recent review contains the word. (Both of these tasks can be done using the dictionary of words for each review you have been maintaining in steps 2-7.)

9. Create a DataFrame using the dictionaries `occurrences` and `reviews`, with each row index being a word, and the columns `"Occurrences"` and `"Reviews"` being the content of of the dictionaries.

10. Sort the DataFrame by the column `"Occurrences"` and obtain only the first `numWords` many rows.

11. Test your code on the data sets included with this assignment and check if the answers are close to those in the sample outputs above.

12. Put your code in a function and test the function. Once it's working well, then copy and paste it into a new file named `reviews.py` (using Spyder) and save. Then test the function by importing the module.

13. Write appropriate comments, docstrings in the file `reviews.py`, and walk through the code again to check for logical errors.

Grading Rubric

There are a total of 15 points for this assignment, distributed among the following five categories. Each category is worth 3 points. For every major error, there will be a deduction of 1 point. For every minor error, there will be a deduction of 0.5 points.

- **Syntax:** Code runs without errors. Proper syntax for defining functions and for using lists, dictionaries, and DataFrames. Proper grammar of Python statements.
- **String parsing:** Correct use of `split`, `strip`, `in`, and other string methods for parsing lines, filtering lines, skipping words, removing whitespaces and punctuations. Correctly opening, reading and closing files.
- **"Occurrences" column:** Correct logic for computing the total number of occurrences of each word.
- **"Reviews" column:** Correct logic for computing the number of reviews each word is contained in.
- **Presentation:** Code is reasonably readable with proper variable names, docstrings, and appropriate comments.