AIM 5001 Module 12 Assignment

Text Mining

***** You may work in small groups of no more than three (3) people for this Assignment *****

As we've learned, many organizations rely on sentiment analysis algorithms to help them gauge the opinions of both existing and potential customers. For example, companies such as Amazon, TripAdvisor, Booking.com, WalMart, and Yelp (amongst others) apply sentiment analysis algorithms to the online product/service reviews provided by their customers to better understand how the public perceives competing products and services.

Your task for the **Module 12 Assignment** is to prepare a collection of text documents for use within a sentiment analysis algorithm. The data set you will be working with is sourced from this site: http://www.cs.cornell.edu/people/pabo/movie-review-data/. Specifically, you will be working with the **polarity** dataset **v2.0**, which is comprised of 1000 positive and 1000 negative movie reviews. Each movie review is in the form of free-form text captured from web site postings. To complete this assignment you will need to make use of a fair amount of pre-processing techniques to prepare the content of the reviews for use within a classification model (e.g., strip out punctuation, stop words, "tokenize" the data, etc.).

Get started on the Assignment as follows:

1) Download the **review_polarity.tar.gz** file to your local environment and decompress its contents. The compressed file contains two directories: **neg** which contains 1000 negative movie reviews; and **pos** which contains 1000 positive movie reviews.
2) Load the **neg** and **pos** directories to your AIM 5001 Github Repository. You need to keep the content of the directories separated since the directories themselves serve as the labels for the classification of the reviews.
3) Then, using a Jupyter Notebook, construct an algorithm (**DO NOT USE scikit-learn COUNTVECTORIZER**) that will read the content of each individual movie review from your new Github directories and convert that content into a properly labeled (i.e., POS / NEG or some appropriate proxy thereof) entry within a Pandas dataframe that encompasses all of the possible words contained within the 2000 movie reviews. When finished, the contents of your Pandas dataframe will constitute a term-document matrix for the movie review data. While constructing this term-document matrix within your Pandas dataframe, you should ensure that you remove any punctuation or stop words from the reviews. How you choose to manage the construction and proper labeling of the term-document matrix is up to you as the text mining / Python practitioner to decide.
4) Convert the cumulative frequency count data content of your newly created Pandas dataframe into a NumPy array.
5) Using the NumPy array, calculate the **sparsity** of the term-document matrix. What percentage of the entries in your term-document matrix contain zeroes?
6) Next, using the content of the Pandas dataframe, plot the frequency distribution for the 30 words which occur most frequently in the **positive** reviews. What insights can you derive from the plot?

7) Then, once again using the content of the Pandas dataframe, plot the frequency distribution for the 30 words which occur most frequently in the **negative** reviews. What insights can you derive from the plot?

8) Now that you have successfully constructed and properly labeled the term-document matrix entries for each of the 2000 individual movie reviews, randomly sample 75% of the vectors contained within the term-document matrix for use as a model training data subset while leaving the remaining 25% of the vectors for the model testing data subset. How you choose to split the data is up to you as the data science / Python practitioner to decide. Be sure to display samples of your training and testing subsets to a reader of your work. Also, tell us how many documents are contained within your training subset? How many documents are contained in your testing subset? How many **positive** and **negative** reviews are contained within each subset? Does the mix of positive and negative reviews appear to be relatively balanced within each of the subsets? Be sure to provide a suitable explanatory narrative in the form of formatted Markdown cells.

Your deliverable for this Assignment is your Jupyter Notebook. It should contain a combination of Python code cells and explanatory narratives contained within properly formatted Markdown cells. The Notebook should contain (at a minimum) the following sections (including the relevant Python code for each section):

1) **Introduction (5 Points)**: Summarize the problem + explain the steps you plan to take to address the problem

2) **Data Preparation (40 Points)**: Describe + show the steps you have taken to load + transform the provided data into properly labeled count vectors within a Pandas-based Term-Document matrix. This section should include any Python code used for Data Preparation as well as an appropriate explanatory narrative.

3) **Calculate Matrix Sparsity (20 Points)**: Describe + show the steps you have taken to transform your Pandas dataframe to a NumPy array. Describe + show the steps you have taken to calculate the sparsity of your term-document matrix. This section should include any Python code used for Data Preparation as well as an appropriate explanatory narrative.

4) **Frequency Distribution Plots (20 Points)**: Explain + present your word count frequency distribution plots for the positive and negative reviews. This section should include any Python code used for creating the plots as well as an appropriate explanatory narrative.

5) **Sentiment Analysis Model Preparation (15 Points)**: Explain + present the process by which you separated the count vectors into training and testing subsets. This section should include any Python code used for creating the training + testing as well as an appropriate explanatory narrative.

Your Jupyter Notebook deliverable should be similar to that of a publication-quality / professional caliber document and should include clearly labeled graphics, high-quality formatting, clearly defined section and sub-section headers, and be free of spelling and grammar errors. Furthermore, your Pythion code should include succinct explanatory comments.

Upload / submit your Jupyter Notebook within the provided M12 Assignment Canvas submission portal. Be sure to save your Notebook using the following nomenclature: **first initial_last name_M12_assn**" (e.g.,

J_Smith_M12_assn). ***Small groups should identity all group members at the start of the Jupyter Notebook and each team member should submit their own copy of the team's work within Canvas.***