**Task: Text Pre-Processing**

In this task, you are required to write Python code to preprocess a set of articles about user reviews and convert them into numerical representations (which are suitable for input into recommender-systems/ information-retrieval algorithms)

The data set that we provide is a single PDF file that contains many user reviews on different products. Your task is to extract and transform the information from the PDF file by performing the following tasks:

1. Generate the corpus vocabulary with the same structure as **sample\_vocab.txt**. Please note that the vocabulary must be sorted alphabetically.

2. For each product ID, generate the sparse representation (i.e., doc-term matrix) of the PDF file according to the structure of the **sample\_countVec.txt.**

The following steps must be performed (**not necessarily in the same order**) to complete the assessment. Please note that the order of preprocessing matters and will result in different vocabulary and hence different count vectors. **It is part of the assessment to figure out the correct order of preprocessing which makes the most sense**

1. The word tokenization must use the following regular expression,

**"[a-zA-Z]+(?:[-'][a-zA-Z]+)?"**

1. The context-independent and context-dependent stopwords must be removed from the vocabulary.

For context-independent, The provided context-independent stop words list (i.e, **stopwords\_en.txt**) must be used.

For context-dependent stopwords, you must set the threshold to more than **ceil(Number\_of\_PIDs / 2**).

1. Tokens should be stemmed using the **Porter** stemmer.
2. Rare tokens (with the threshold set to less than **10 PIDs (i.e. 10 PIDs)**) must be removed from the vocab.
3. Creating the sparse matrix using countvectorizer.
4. Tokens with a length less than 3 should be removed from the vocab.
5. First 200 meaningful bigrams (i.e., collocations) must be included in the vocab using **PMI** measure.
6. Calculate the vocabulary containing both unigrams and bigrams