

**CSE4022**

**Natural Language Processing**

**Project Report**

**Commentator Trait identification using NLP**

**Slot:** A2+TA2

**Submitted to:** Prof. Sharmila Banu K.

**Team Members:**

Het Patel – 17BCE0567

Shivansh Mishra – 17BCB0112

**GitHub Repo. Link**

<https://github.com/het1999/NLP/tree/master/Project>

**1. Problem Statement**

We plan to use the power of Natural Language processing to our advantage by using it on a dataset of cricket commentary (Cricbuzz/ESPN CricInfo). The first step would be that of finding the keywords from the commentary set. This will be done by classification of most common words and removal of stop words. We will find the top monograms, bigrams and tri grams to do this. After this, we would convert this into a vector table of word counts which would help us find the frequencies of the word. This would finally give us the keywords.

The keywords can be classified into various emotions like angry, sad, happy. A good application of this would be in selecting the best commentators together, or when selecting new personnel to do a sports related segment. Since polar opposites would likely do better, a good mix of emotions will be favourable. We would try finding those people who use the most different class of words and pair them together.

**2. Modules**

**2.1 Text Pre-processing**

Whenever we have textual data, we need to apply several pre-processing steps to the data to transform words into numerical features that work with machine learning algorithms. The pre-processing steps for a problem depend mainly on the domain and the problem itself, hence, we don’t need to apply all steps to every problem. Text pre-processing is used to remove punctuations, say to convert every word into lowercase, to delete unnecessary columns, stopwords removal, stemming, lemmatization, etc.

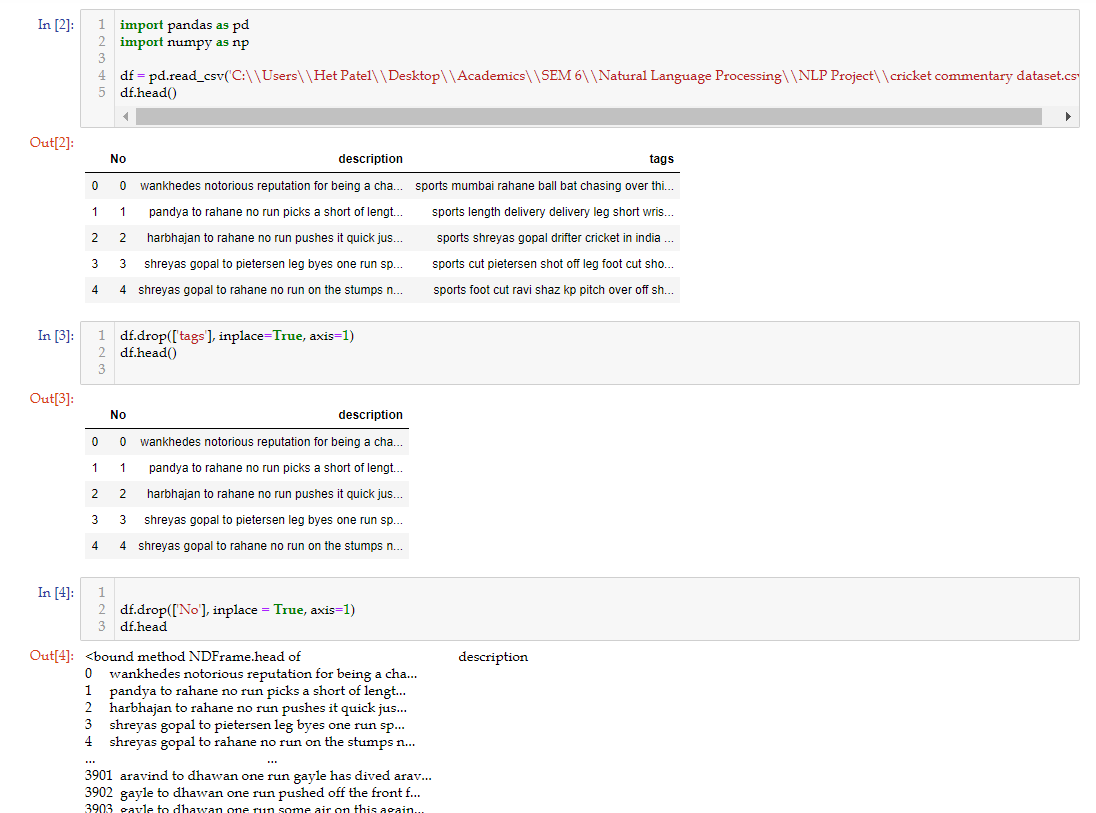


Figure 1. Text Pre-processing

Here, the dataset is initialised and there was one extra column of tags which is not necessary for our project. So, the “tags” column is dropped with the panda framework. And further “No” column is also dropped in order to remove numbering twice.

**2.2 Stop words removal**

A stop word is a commonly used word (such as “the”, “a”, “an”, “in”) that a search engine has been programmed to ignore, both when indexing entries for searching and when retrieving them as the result of a search query. The stop words are removed in text pre-processing to make to model more fit.

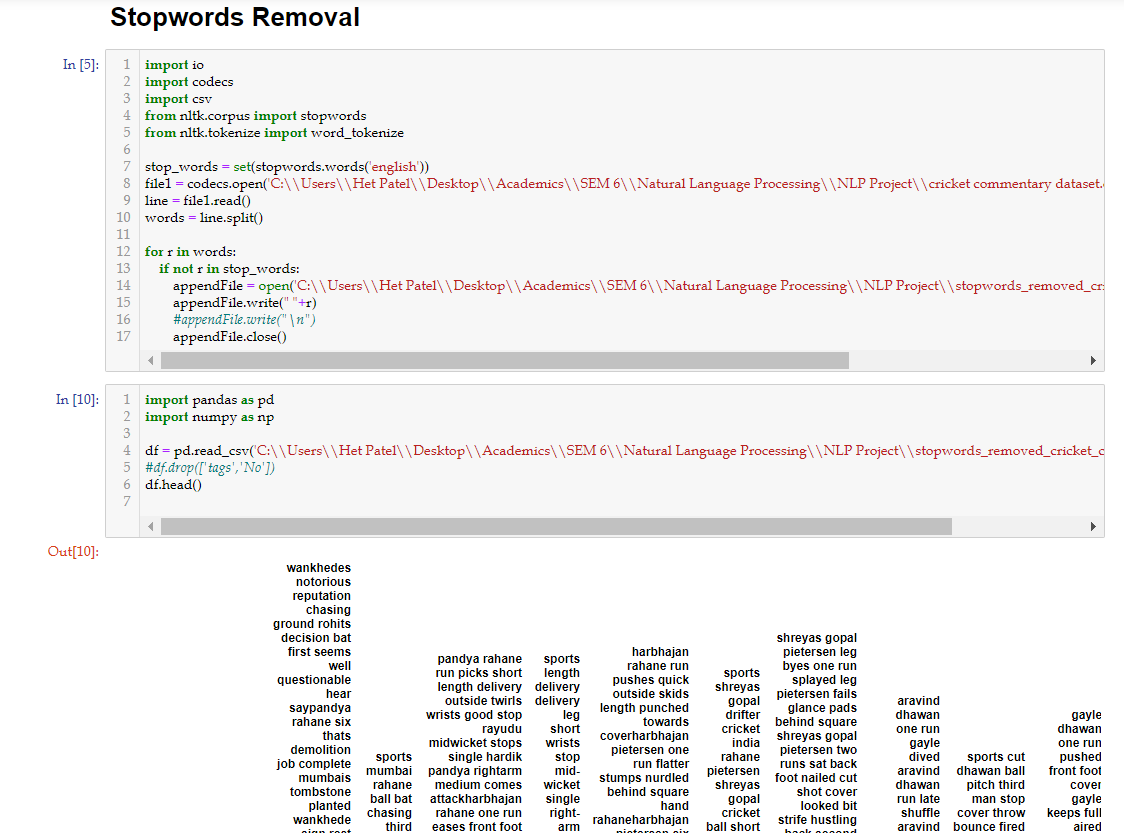


Figure 2. Stop words removal

**2.3 n-gram**

A set that supports searching for members by N-gram string similarity. N-gram gives you the better understanding for keyword extraction and also enhances the effectivity of the model. For example, say {shivansh het}, is considered as 2-gram. Similarly, in the case of 3-gram we will considere every pair of 3 words.

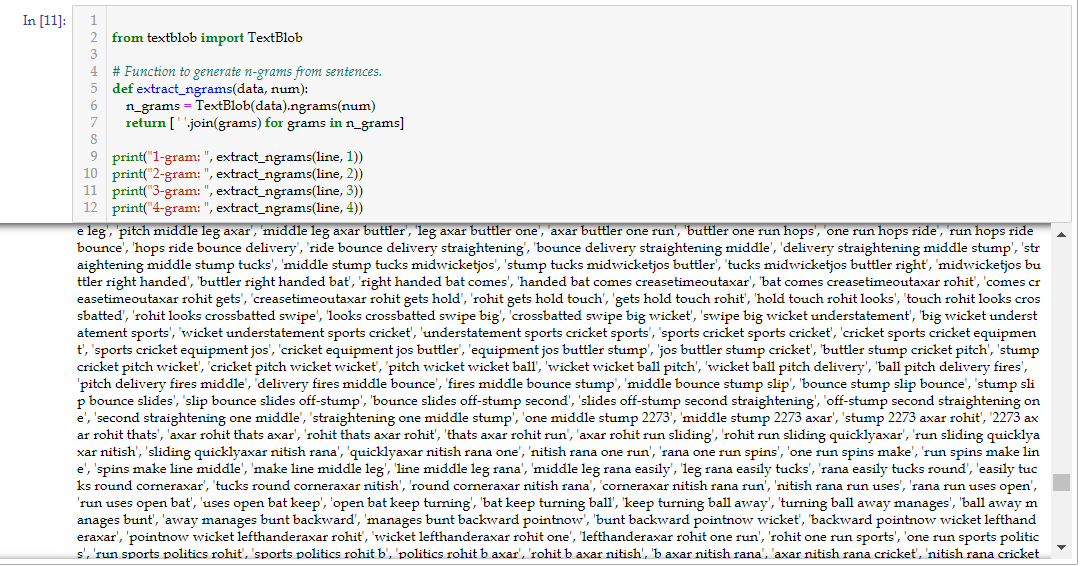


Figure 3. n-gram generation

**2.4 POS tagging**

it may be defined as the process of assigning one of the parts of speech to the given word. It is generally called POS tagging. In simple words, we can say that POS tagging is a task of labelling each word in a sentence with its appropriate part of speech. We already know that parts of speech include nouns, verb, adverbs, adjectives, pronouns, conjunction and their sub-categories.

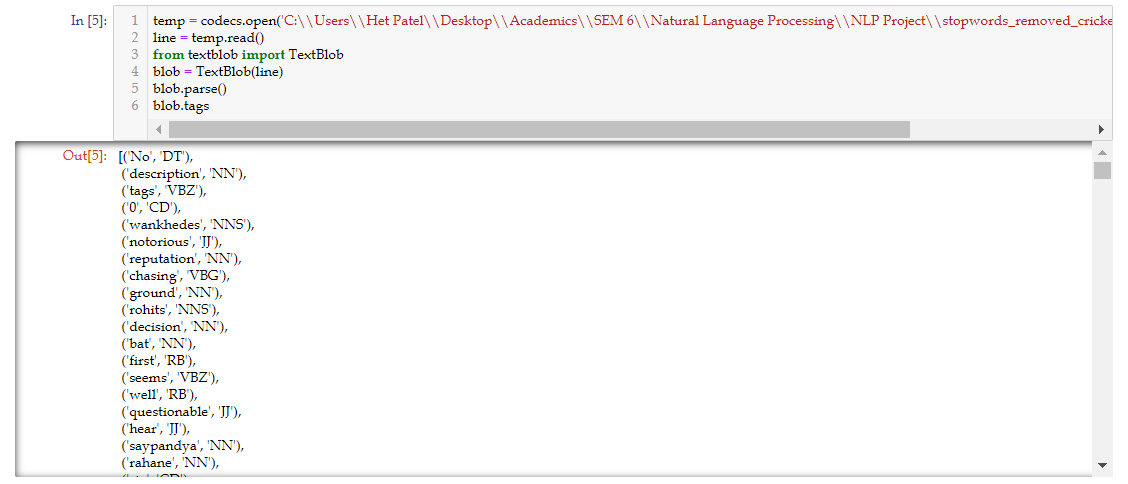


Figure 4. POS tagging

**2.5 Bag of Words**

Bag of Words (BOW) is a method to extract features from text documents. These features can be used for training machine learning algorithms. It creates a vocabulary of all the unique words occurring in all the documents in the training set.

In simple terms, it’s a collection of words to represent a sentence with word count and mostly disregarding the order in which they appear.

BOW is an approach widely used with:

* Natural language processing
* Information retrieval from documents
* Document classifications

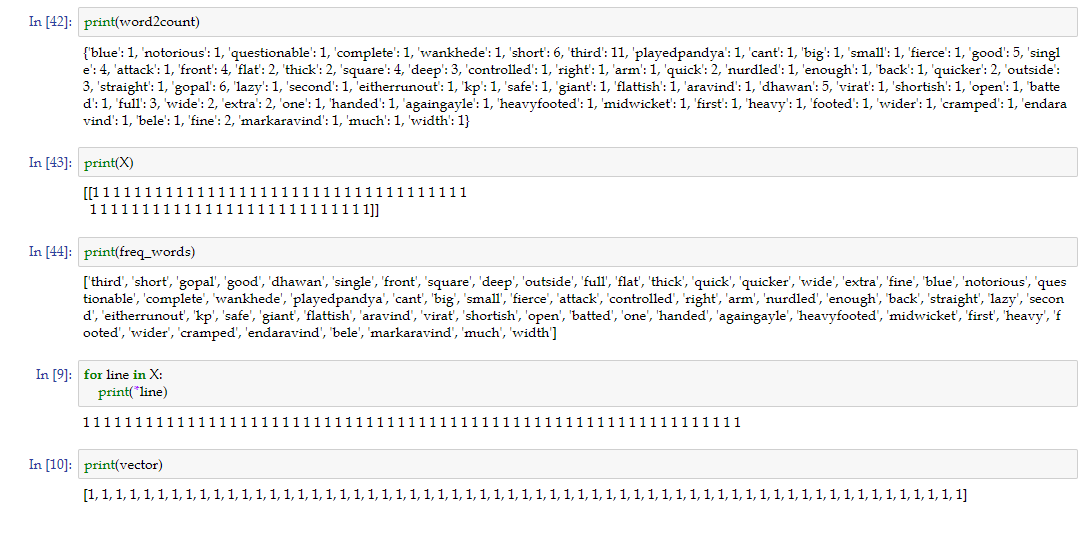


Figure 5. Bag of words

**2.6 Keyword frequency generation**

Keyword frequency refers to how often a keyword appears on a given webpage or within a piece of content. The more frequently a keyword appears in a given page or piece of content, the higher the keyword frequency. Keyword frequency is closely related to keyword density.

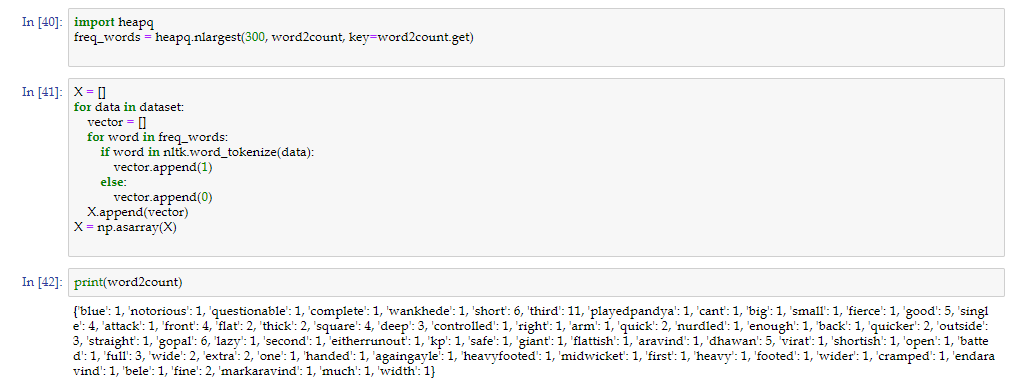


Figure 6. Keyword frequency generator

**2.7 Evaluating quality of commentary by frequency of adjective**

**comparison.**

Here, from the adjective that we got, we will compare it with the adjective dataset and will find the relative score. If the adjectives fetched are there in the dataset that means the quality is good and if the adjectives fetched are not in the enough quantity then the quality seems to be poor.

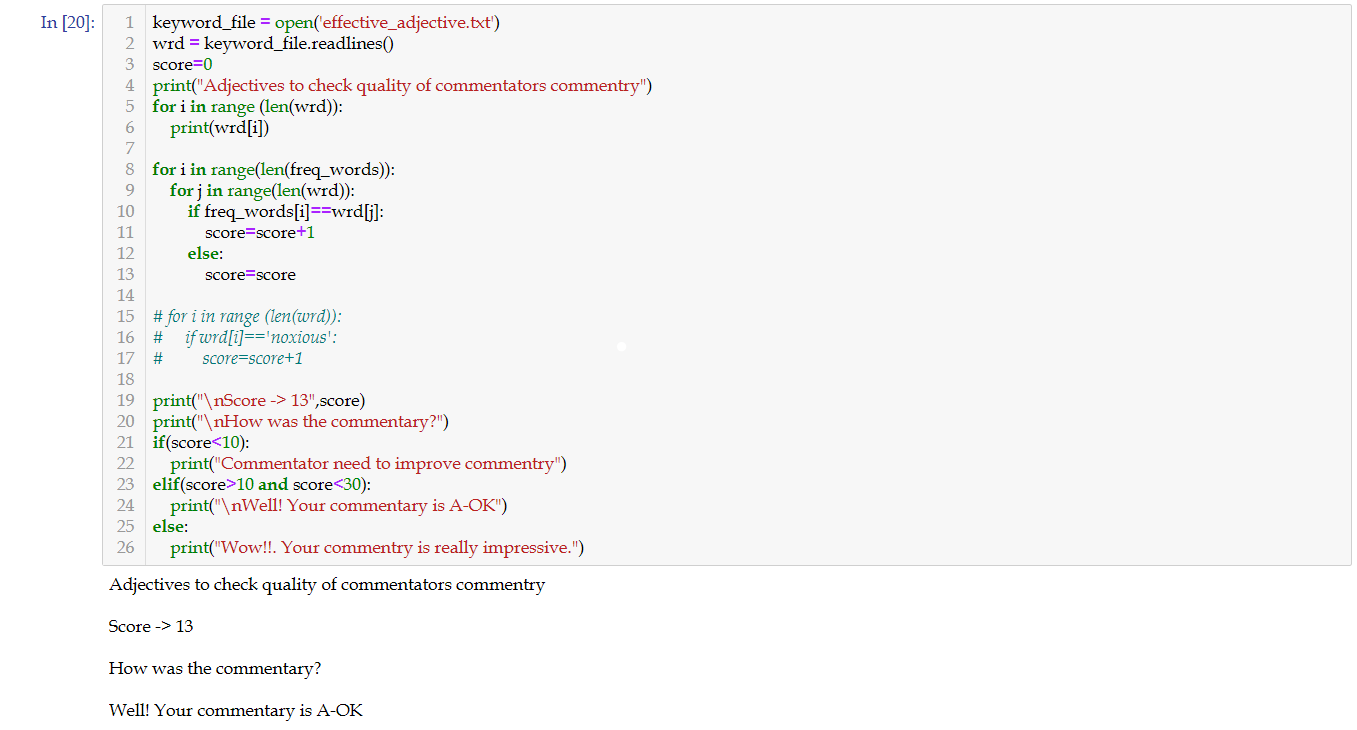


Figure 7. Evaluating quality of commentary by adjective frequency.

**3. List of challenges faced**

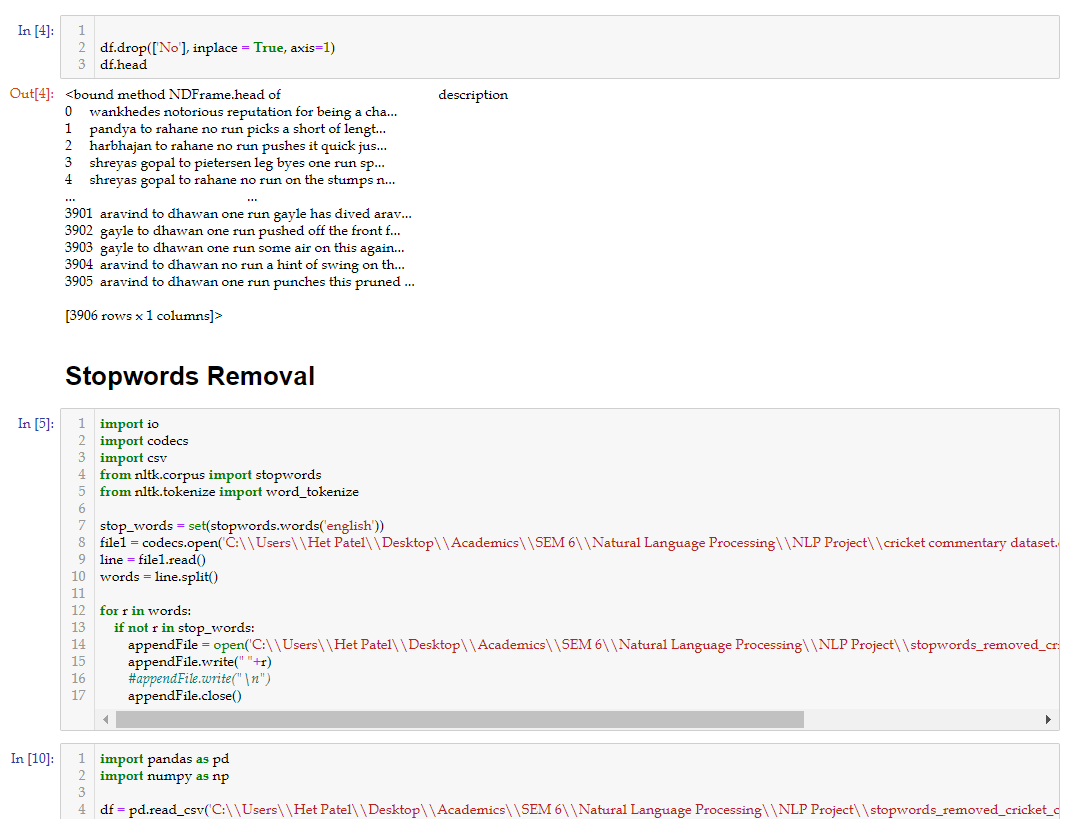
|  |  |  |
| --- | --- | --- |
| No. | Errors | How it was fixed. |
| 1 | While removing stopwords, data was unstructured. | Manually converted unstructured data into structured data by pandas framework. Dropped all the unnecessary columns. |
| 2 | Adjectives were not detected as the data was unstructured. | And found out adjectives manually Using separate POS tagger |
| 3 | By the method taught in class, all the letters of the word were considered as individually tagged NN | Fixed using textblob library. |
| 4 | Notebook data rate limit was initialized as very low. Therefore it was not possible to run a very big dataset. For n-gram processing | Error fixed Execute command in cmd and restart the jupyter --> jupyter notebook --NotebookApp.iopub\_data\_rate\_limit=10000000000(bytes/sec) |
| 5 | WordCloud data not usable | We fixed it using google lenses. Captured in text file. Space separated variables |

**4. Snapshots of errors and fixes**

**4.1**

Error: Unstructured data handling due to stopword removal.

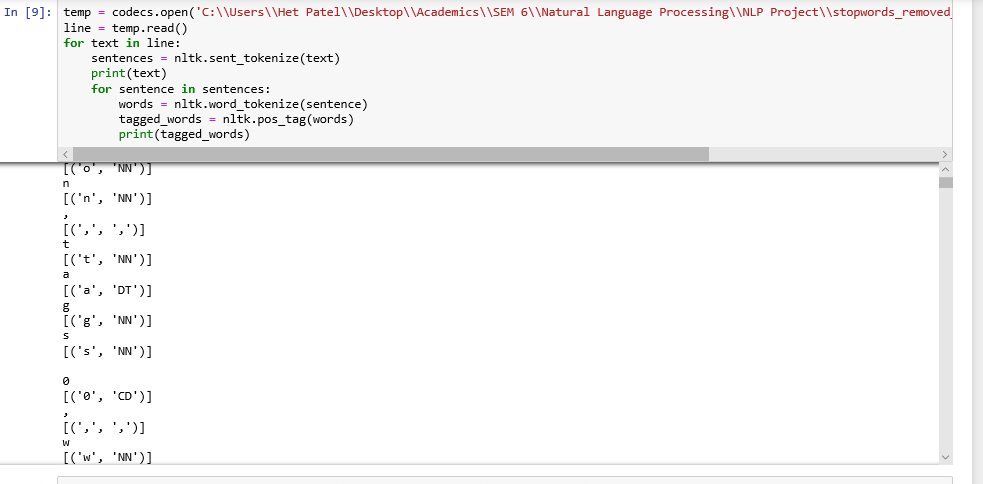
Solution: Fixed using Pandas framework



**4.2**

Error/issue: Adjectives not detected by normal POS tagging

Solution: Used textblob method for POS tagging

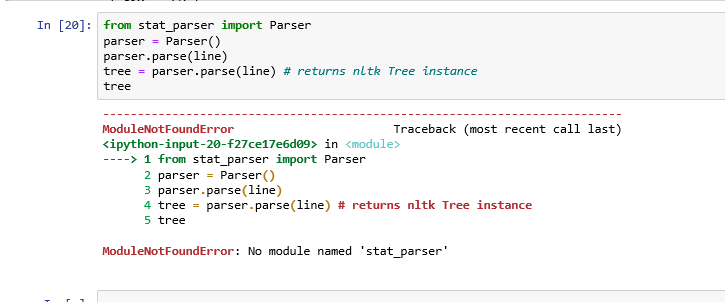




**4.3**

Error: Parsing sentence tree

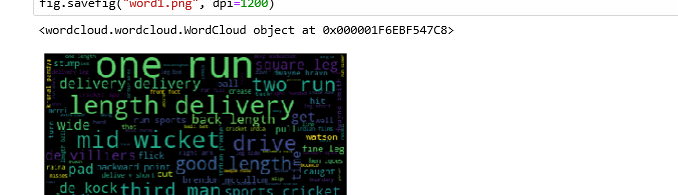
Solution: Use NLTK Parsing framework



**4.4**

Error: Wordcloud Data not usable

Solution: Used Google Lenses to convert into txt file



**4.5**   
Error: Notebook data rate limit was initialized as very low. Therefore it was not possible to run a very big dataset.

For n-gram processing

Solution: **Error fixed Execute command in cmd and restart the jupyter --> jupyter notebook --NotebookApp.iopub\_data\_rate\_limit=10000000000**







**5. References**

1. <https://medium.com/@ageitgey/natural-language-processing-is-fun-part-3-explaining-model-predictions-486d8616813c>

2. <https://www.kdnuggets.com/2019/01/solve-90-nlp-problems-step-by-step-guide.html>

3. <https://towardsdatascience.com/nlp-based-data-preprocessing-method-to-improve-prediction-model-accuracy-30b408a1865f>

4. <https://www.analyticsvidhya.com/blog/2019/04/predicting-movie-genres-nlp-multi-label-classification/>

5. <https://web.eecs.umich.edu/~lahiri/thesis.pdf>

6. <http://charlieojackson.co.uk/python/catergorising-keywords-at-scale.php>

7. <https://medium.com/analytics-vidhya/automated-keyword-extraction-from-articles-using-nlp-bfd864f41b34>

8. <https://programminghistorian.org/en/lessons/extracting-keywords>

9. <https://www.freecodecamp.org/news/an-introduction-to-bag-of-words-and-how-to-code-it-in-python-for-nlp-282e87a9da04/>