Analysis of Social Media Content Using Text Mining On Big Data

Ankita A. Kathane[1], Vivek B. Kute[2]
[1] Student, Department of Computer Engineering, St. Vincent Pallotti College of Engineering Nagpur, India.
[2] Associate professor, Department of Computer Engineering, St. Vincent Pallotti College of Engineering Nagpur, India.

---***---

Abstract -The world is advancing rapidly with the technology changing each day and forming the base of almost every activity of human life. We have advanced so much in this context that almost all the devices that we use today have some elements of artificial intelligence.If we look deep into it, artificial intelligence has many aspects that are being explored and Natural language processing is one of them. Text mining deals with extracting or storing information that is new or was previously unknown to the system. This information can then be used for various purposes that can be specified as per the use of the audience or companies.The purpose of this research paper is to target the audience's queries. We try to fetch the queries from the website's audience and process it so that we extract meaningful, correct, and accurate keywords. Further, these keywords will be sorted to find our most hit keyword and used for article creation.We have used text mining and natural language processing for this purpose.

***Key Words*:**Text mining, text analysis, big data, social media, big data analytics, social media data, data mining, social media analysis.

1. INTRODUCTION

Social media platforms have proved to be an advantage for companies to reach out to their audiences. Therefore, it is critical to identify and rectify users in society. Every day a new concept in lifestyle and technology is introduced. Hence, it is important to get to the basics of the utilization of technology to yield its benefits.[6]
Text mining, a process which is also popular as text analytics, is a technique that uses artificial intelligence by the means of natural language processing (NLP) to transfer and process unstructured data that is in the form of text into structured data that can further be used for analytics by using machine learning algorithms. Most of the time the term text mining is confused with the term data mining. [3] While on a broader aspect it seems the same, it is, however, different. Text mining is related to the process of searching, processing, and storing textual information i.e. unstructured data. [1] On the other hand, data mining deals with data that is in a structured form.
While the web has much of its content in the form or text, this textual data is unstructured and difficult to process. Here NLP- natural language processing steps in. Natural language processing is a crucial part of the text mining process. The algorithms used in NLP such as lemmatization or stemming algorithms are used to extract the words that make sense in the natural language. The words that do not make sense can be discarded, or stored separately as per the need of the users.
The traditional methods of mining into data assume that the data is already in the structured form stored in a relational database. However, the scenario changes when a huge amount of data is to be mined or stored that is in textural form. 80 % of the data present on social media platforms is in the form of unstructured or textual form while the remaining 20 % is in a structured form. [4] Storing the data that is useful and relevant for the organization is crucial because by doing so the organization tends to save the storage space that would otherwise be spent on "junk" data. The concept of information extraction is very crucial and needs to be applied by defining the target data.

Information extraction is an activity that transforms a corpus of textual data into a more structured database which is useful in further data processing.[5] Most of the websites or blogs need content to be published for their readers that are existing or to fetch new readers. However, they tend to publish articles based on the current trend, or most keyword hits on the internet. We aim to build a module that processes the natural language entered by the user in the website's search bar and store it for the editor. This way the list of new articles can be made according to the demand of the audience visiting the website and them getting quality content.

MOTIVATION

Through studying various papers, it was observed that a website is not optimized depending on the need or demands of the audience. This works as a setback for the content writers and developers as well, because eventually, they want to create content keeping in mind what the audience wants. It was thus observed that the audience that visits the website might enter the topic they desire in the search bar of the website. If the search queries of the audience were to be fetched, it would give help the editors to decide what topics they need to have on their website as per the popularity in what topic has been searched. However, running through the "N" number of queries and reading out each word that has been searched by every particular person searching them will be tedious. Moreover, it will be foolish to invest a huge amount of time and energy into a task that could be done easily.
The text mining process gives a solution to such a problem.[6] Following the method of Natural Language Processing, it becomes easy to hit the keywords that are required by the editor to render a suitable title for a new article according to the most popular demand. This reduces the research time of the editor to check what is currently trending. Moreover, the content that is developed

2. THE PROPOSED SYSTEM

2.1. WORKING

In the proposed system, we target the searching queries of the audience. From there we fetch the queries. Here the NLP – natural language processing plays an important role as the fetched queries are then refined through it. The searching queries might contain “stop words” i.e. words that areas such as not useful as they are not keywords. Then by using lemmatization, we try to generate a root keyword out of the many keywords that make the same meaning. For easy of the editor, we then group these keywords as in to make a keyword that would make sense as well as a keyword for the article. Further, on hitting similar keywords we increase the count of the keyword to the number of times it was searched. This will give the popularity of the keyword. Then a list is generated in which at least 3 keywords from the queries entered by the users are displayed.

2.2. ARCHITECTURE OF THE SYSTEM

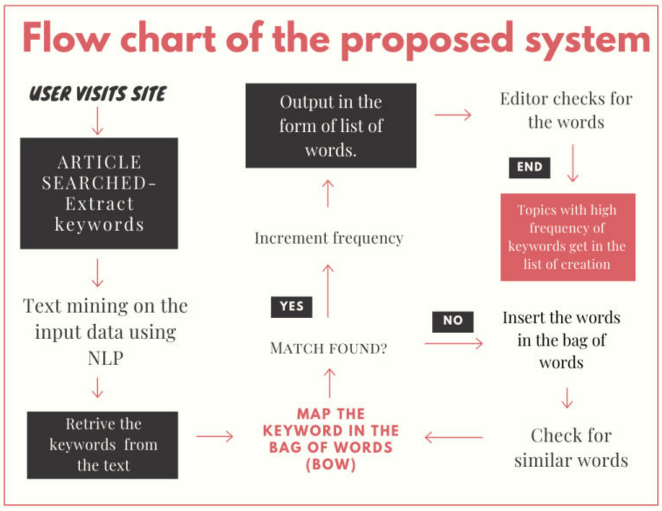
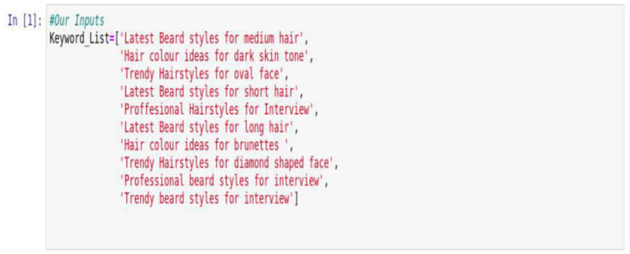

Flow chart of the proposed system
USER VISITS SITE
ARTICLE SEARCHED-Extract keywords
Text mining on the input data using NLP
Retrive the keywords from the text
MAP THE KEYWORD IN THE BAG OF WORDS (BOW)
MATCH FOUND?
YES
Increment frequency
Output in the form of list of words.
Editor checks for the words
END
Topics with high frequency of keywords get in the list of creation
NO
Insert the words in the bag of words
Check for similar words

Fig – 1.1: Flowchart of the proposed system

2.3. STEPWISE WORKING OF THE PROJECT

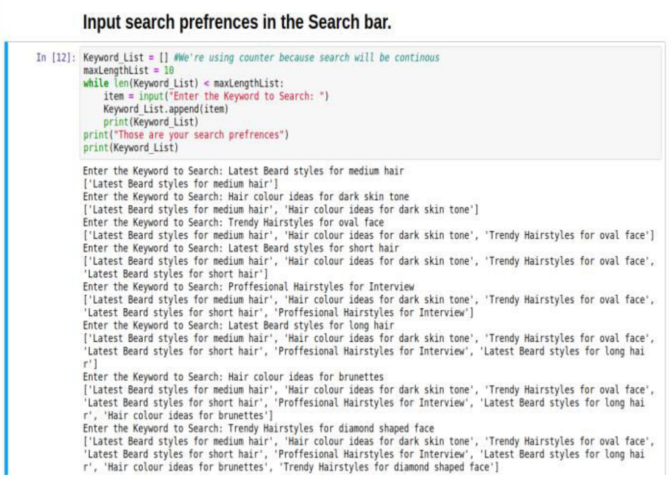
Following is the example of input keywords that will be done by random users at the end user’s side. These can vary from person to person. We have taken 10 inputs for example. However, the system can take the “n” number of data in the input bar.



```
In [1]: #Our Inputs
        Keyword_List=['Latest Beard styles for medium hair',
                      'Hair colour ideas for dark skin tone',
                      'Trendy Hairstyles for oval face',
                      'Latest Beard styles for short hair',
                      'Proffesional Hairstyles for Interview',
                      'Latest Beard styles for long hair',
                      'Hair colour ideas for brunettes ',
                      'Trendy Hairstyles for diamond shaped face',
                      'Professional beard styles for interview',
                      'Trendy beard styles for interview']
```

Fig - 1.2: Sample Input data

Step 1: Here the user inputs the above search data set. We used a counter of max length 10 to track the number of inputs.

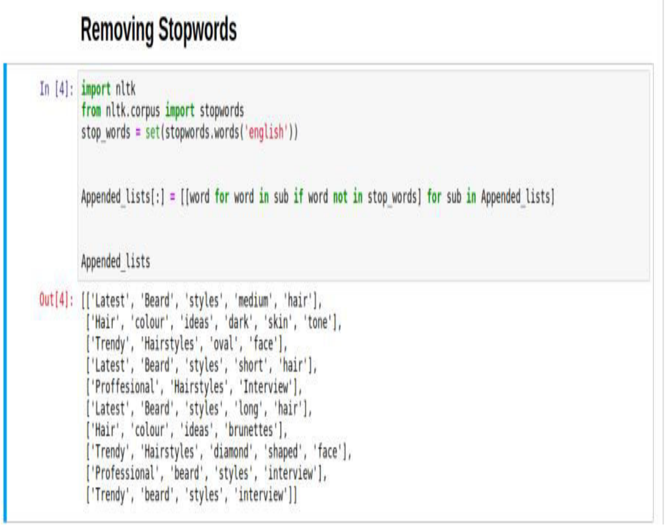


Input search prefrences in the Search bar.

```
In [12]: Keyword_List = [] #We're using counter because search will be continous
         maxLengthList = 10
         while len(Keyword_List) < maxLengthList:
             item = input("Enter the Keyword to Search: ")
             Keyword_List.append(item)
             print(Keyword_List)
         print("Those are your search prefrences")
         print(Keyword_List)

         Enter the Keyword to Search: Latest Beard styles for medium hair
         ['Latest Beard styles for medium hair']
         Enter the Keyword to Search: Hair colour ideas for dark skin tone
         ['Latest Beard styles for medium hair', 'Hair colour ideas for dark skin tone']
         Enter the Keyword to Search: Trendy Hairstyles for oval face
         ['Latest Beard styles for medium hair', 'Hair colour ideas for dark skin tone', 'Trendy Hairstyles for oval face']
         Enter the Keyword to Search: Latest Beard styles for short hair
         ['Latest Beard styles for medium hair', 'Hair colour ideas for dark skin tone', 'Trendy Hairstyles for oval face',
         'Latest Beard styles for short hair']
         Enter the Keyword to Search: Proffesional Hairstyles for Interview
         ['Latest Beard styles for medium hair', 'Hair colour ideas for dark skin tone', 'Trendy Hairstyles for oval face',
         'Latest Beard styles for short hair', 'Proffesional Hairstyles for Interview']
         Enter the Keyword to Search: Latest Beard styles for long hair
         ['Latest Beard styles for medium hair', 'Hair colour ideas for dark skin tone', 'Trendy Hairstyles for oval face',
         'Latest Beard styles for short hair', 'Proffesional Hairstyles for Interview', 'Latest Beard styles for long hai
         r']
         Enter the Keyword to Search: Hair colour ideas for brunettes
         ['Latest Beard styles for medium hair', 'Hair colour ideas for dark skin tone', 'Trendy Hairstyles for oval face',
         'Latest Beard styles for short hair', 'Proffesional Hairstyles for Interview', 'Latest Beard styles for long hai
         r', 'Hair colour ideas for brunettes']
         Enter the Keyword to Search: Trendy Hairstyles for diamond shaped face
         ['Latest Beard styles for medium hair', 'Hair colour ideas for dark skin tone', 'Trendy Hairstyles for oval face',
         'Latest Beard styles for short hair', 'Proffesional Hairstyles for Interview', 'Latest Beard styles for long hai
         r', 'Hair colour ideas for brunettes', 'Trendy Hairstyles for diamond shaped face']
```

Fig- 1.3: List of Data

Step 2:In this step, we attempt to remove the stop words that are used in the input streams and try to focus on the keywords. This will help us focus more on the type of result we need. As a result, we get a list of words that were present in the sentence whilst the stop words are successfully removed.

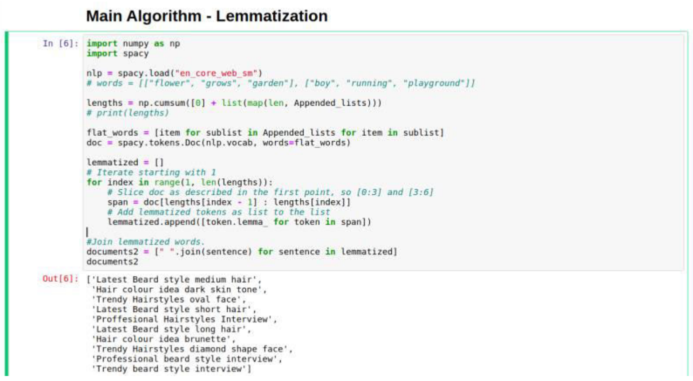


Removing Stopwords

```
In [4]: import nltk
        from nltk.corpus import stopwords
        stop_words = set(stopwords.words('english'))


        Appended_lists[:] = [[word for word in sub if word not in stop_words] for sub in Appended_lists]


        Appended_lists

Out[4]: [['Latest', 'Beard', 'styles', 'medium', 'hair'],
         ['Hair', 'colour', 'ideas', 'dark', 'skin', 'tone'],
         ['Trendy', 'Hairstyles', 'oval', 'face'],
         ['Latest', 'Beard', 'styles', 'short', 'hair'],
         ['Proffesional', 'Hairstyles', 'Interview'],
         ['Latest', 'Beard', 'styles', 'long', 'hair'],
         ['Hair', 'colour', 'ideas', 'brunettes'],
         ['Trendy', 'Hairstyles', 'diamond', 'shaped', 'face'],
         ['Professional', 'beard', 'styles', 'interview'],
         ['Trendy', 'beard', 'styles', 'interview']]
```

Fig- 1.4: Removing stop words

Step 3:While many words have the same root and most of the time we need the root words to get our work done, we use lemmatization to make sense out of the sentence that was input by the user and the list of words that we got after applying the process of removing stop words

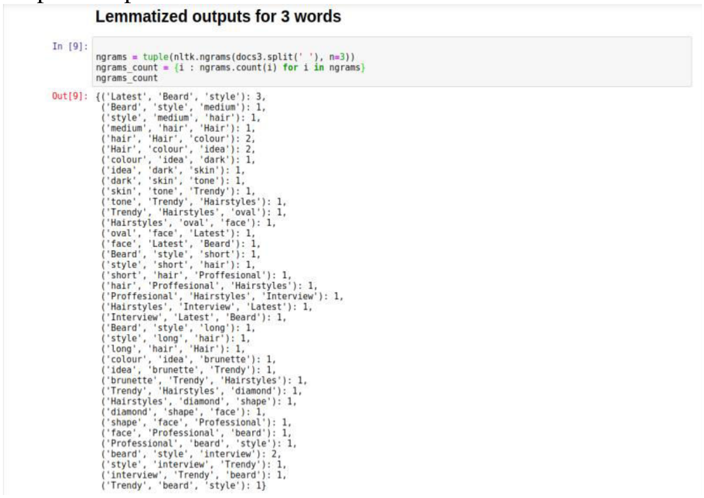


Main Algorithm - Lemmatization

```
In [6]: import numpy as np
        import spacy

        nlp = spacy.load("en_core_web_sm")
        # words = [["flower", "grows", "garden"], ["boy", "running", "playground"]]

        lengths = np.cumsum([0] + list(map(len, Appended_lists)))
        # print(lengths)

        flat_words = [item for sublist in Appended_lists for item in sublist]
        doc = spacy.tokens.Doc(nlp.vocab, words=flat_words)

        lemmatized = []
        # Iterate starting with 1
        for index in range(1, len(lengths)):
            # Slice doc as described in the first point, so [0:3] and [3:6]
            span = doc[lengths[index - 1] : lengths[index]]
            # Add lemmatized tokens as list to the list
            lemmatized.append([token.lemma_ for token in span])

        #Join lemmatized words.
        documents2 = [" ".join(sentence) for sentence in lemmatized]
        documents2

Out[6]: ['Latest Beard style medium hair',
         'Hair colour idea dark skin tone',
         'Trendy Hairstyles oval face',
         'Latest Beard style short hair',
         'Proffesional Hairstyles Interview',
         'Latest Beard style long hair',
         'Hair colour idea brunette',
         'Trendy Hairstyles diamond shape face',
         'Professional beard style interview',
         'Trendy beard style interview']
```

Fig- 1.5: Main algorithm – lemmatization

Step 4: Outputs



Lemmatized outputs for 3 words

```
In [9]:
ngrams = tuple(nltk.ngrams(docs3.split(' '), n=3))
ngrams_count = {i : ngrams.count(i) for i in ngrams}
ngrams_count

Out[9]: {('Latest', 'Beard', 'style'): 3,
 ('Beard', 'style', 'medium'): 1,
 ('style', 'medium', 'hair'): 1,
 ('medium', 'hair', 'Hair'): 1,
 ('hair', 'Hair', 'colour'): 2,
 ('Hair', 'colour', 'idea'): 2,
 ('colour', 'idea', 'dark'): 1,
 ('idea', 'dark', 'skin'): 1,
 ('dark', 'skin', 'tone'): 1,
 ('skin', 'tone', 'Trendy'): 1,
 ('tone', 'Trendy', 'Hairstyles'): 1,
 ('Trendy', 'Hairstyles', 'oval'): 1,
 ('Hairstyles', 'oval', 'face'): 1,
 ('oval', 'face', 'Latest'): 1,
 ('face', 'Latest', 'Beard'): 1,
 ('Beard', 'style', 'short'): 1,
 ('style', 'short', 'hair'): 1,
 ('short', 'hair', 'Proffesional'): 1,
 ('hair', 'Proffesional', 'Hairstyles'): 1,
 ('Proffesional', 'Hairstyles', 'Interview'): 1,
 ('Hairstyles', 'Interview', 'Latest'): 1,
 ('Interview', 'Latest', 'Beard'): 1,
 ('Beard', 'style', 'long'): 1,
 ('style', 'long', 'hair'): 1,
 ('long', 'hair', 'Hair'): 1,
 ('colour', 'idea', 'brunette'): 1,
 ('idea', 'brunette', 'Trendy'): 1,
 ('brunette', 'Trendy', 'Hairstyles'): 1,
 ('Trendy', 'Hairstyles', 'diamond'): 1,
 ('Hairstyles', 'diamond', 'shape'): 1,
 ('diamond', 'shape', 'face'): 1,
 ('shape', 'face', 'Professional'): 1,
 ('face', 'Professional', 'beard'): 1,
 ('Professional', 'beard', 'style'): 1,
 ('beard', 'style', 'interview'): 2,
 ('style', 'interview', 'Trendy'): 1,
 ('interview', 'Trendy', 'beard'): 1,
 ('Trendy', 'beard', 'style'): 1}
```

Fig- 1.6: Sample output.

2.4. COMPARISON GRAPHS

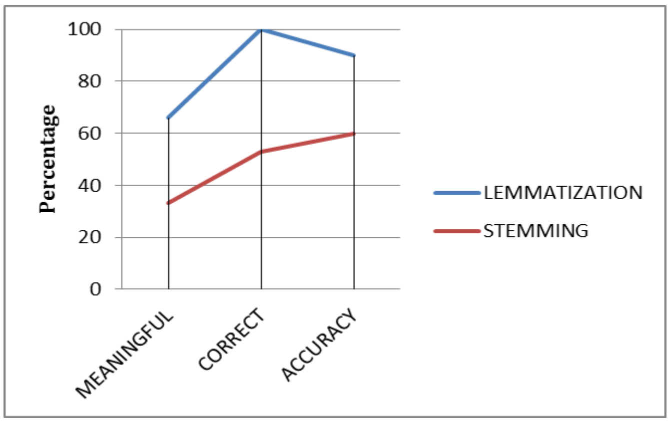


Fig- 1.7: Comparison between text mining algorithms.

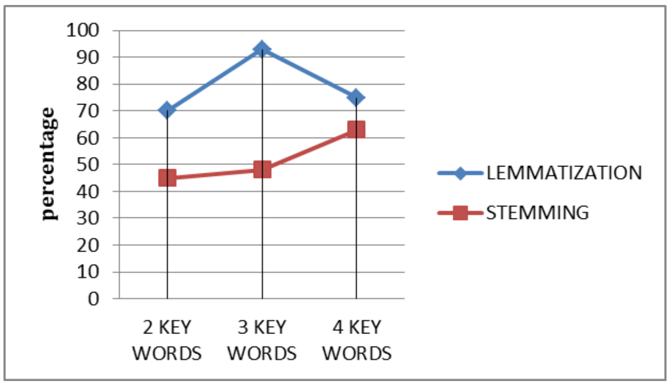


Fig-1.8: Accuracy of keywords depending on length.

3. CONCLUSIONS

We also implemented the stemming algorithm for text mining to compare lemmatization and stemming algorithms of text mining. However, we receive repeated words from the stemming process. Keeping that in mind, we decide to rank the words so that the *content editor* finds it easy to choose the words that are hit often and thus have higher search frequency by the traffic on the website. Initially, we lemmatize and stem for 2 words. However, we find that 2 words are not enough to conclude. Therefore, we move to have 3 words
When we compare both the algorithms, we find that the lemmatization algorithm gives us more useful word extraction than the stemming algorithm. For example, the word "hairstyle" has different bases in both the cases and makes more sense in the example of the lemmatization algorithm than in the stemming algorithm. Also, we find that having 3 keywords in the extraction dictionary is helpful to reach a useful conclusion.

REFERENCES

1. Hotho, A., Nürnberger, A. and Paaß, G. (2005). "A brief survey of text mining". In Ldv Forum, Vol. 20(1), p. 19-62
2. http://people.ischool.berkeley.edu/~hearst/text-mining.html
3. what-text-mining-text-analytics-and-natural-language-processing
4. Zhang, J. Q., Craciun, G., &Shin, D. (2010). When does electronic word-of-mouth matter? A study of consumer product reviews. Journal of Business Research, 63(12), 1336-1341.
5. Text Mining with Information Extraction. Raymond J. Mooney and Un Yong Nahm. Department of Computer Sciences, University of Texas, Austin, TX 78712-1188
6. Analysis of Social Media Content Using Text Mining On Big Data: Literature Review (Survey) Ankita A. Kathane1, Vivek B. Kute2
7. The Power of Social Media Analytics: Text Analytics Based on Sentiment Analysis and Word Clouds on R. Ahmed Imran KABIR1, Ridoan KARIM2, Shah NEWAZ3 , Muhammad Istiaque HOSSAIN4