Analysis of Different Text Summarization Approaches

A project report submitted for the partial fulfilment of the

Bachelor of Technology Degree in

Computer Science & Engineering under

Maulana Abul Kalam Azad University of Technology by

Sonakshi Kumari

Roll No: 10400317083, Registration Number: 171040110442

&

Ananya Mukherjee

Roll No: 10400317204, Registration Number: 171040110321

Academic Session: 2017-2021

Under the Supervision of

Prof. Soma Das

Department of Computer Science and Engineering Institute of Engineering & Management

Y-12, Salt Lake, Sector 5, Kolkata, Pin 700091, West Bengal, India

Affiliated To

Maulana Abul Kalam Azad University of Technology

BF 142, BF Block, Sector 1, Kolkata, West Bengal 700064

June, 2021

CERTIFICATE

TO WHOM IT MAY CONCERN

This is to certify that the project report titled **"Analysis of Different Text Summarization Approaches"**, submitted by **Sonakshi Kumari, Roll No: 10400317083**, **Ananya Mukherjee, Roll No: 10400317204**, students of **Institute of Engineering & Management** in partial fulfillment of requirements for the award of the degree of **Bachelor of Technology in Computer Science & Engineering**, is a bonafide work carried out under the supervision of **Prof. Soma Das** during the final year of the academic session of 2017-2021. The content of this report has not been submitted to any other university or institute for the award of any other degree.

It is further certified that the work is entirely original and the performance has been found to be satisfactory.

Prof. Soma Das
Assistant Professor
Department of Computer Science and Engineering
Institute of Engineering & Management

Prof.(Dr.) Mohuya Chakraborty
H.O.D.
Department of Computer Science and Engineering
Institute of Engineering & Management

Prof.(Dr.) Amlan Kusum Nayak
Principal
Institute of Engineering & Management

INSTITUTE OF ENGINEERING & MANAGEMENT

DECLARATION

FOR NON-COMMITMENT OF PLAGIARISM

We, Ananya Mukherjee, Sonakshi Kumari, students of B.Tech in the Department of Computer Science and Engineering, Institute of Engineering & Management have submitted the project report in partial fulfillment of the requirements to obtain the above noted degree. We declare that we have not committed plagiarism in any form or violated copyright while writing the report and have acknowledged the sources and/or the credit of other authors wherever applicable. If subsequently it is found that we have committed plagiarism or violated copyright, then the authority has full right to cancel/reject/revoke our degree.

Name of the Student: Ananya Mukherjee

Full Signature: ______________________________

Name of the Student: Sonakshi Kumari

Full Signature: ______________________________

Date: 25.6.2021

Contents

ABSTRACT

Natural language processing (NLP) is a subfield of linguistics, computer science, and artificial intelligence that helps computers understand, interpret and manipulate human language.NLP in artificial language helps making human language accessible to computers. The goal of artificial intelligence is to build software and robots with the same range of abilities as humans and NLP has helped in several ways in achieving this. The capacity for language is one of the central features of human intelligence, and is therefore a prerequisite for artificial intelligence. Also, much of artificial intelligence research is dedicated to the development of systems that can reason from premises to a conclusion, but such algorithms are only as good as what they know. NLP can act as the “knowledge bottleneck”, by acquiring knowledge from texts and conversations.

Text summarization is one of those applications of Natural Language Processing (NLP) which is bound to have a huge impact on our lives. This is an incipient practice for verdict out the summary of the text article. In this digital era, people do not have much time to go through the whole textual data. Reduction of text is also a very complex problem. In this project different approaches for text summarization has been discussed and implemented. For the same text document the output summary of different approaches vary.

Acknowledgement:

We must not forget to acknowledge everyone who has provided constant support to us during our B.Tech course. First and foremost, we would like to express sincere gratitude to our supervisor **Prof. Soma Das** for her continuous support and motivation in fueling the pursuance of carrying out this project endeavor. Without her guidance and persistent encouragement, this project work would not have been possible. She has been a tremendous mentor for us throughout this academic journey. Many of her academic advises about our career growth have been priceless.

We would like to convey sincere gratitude to **Prof.(Dr.)Mohuya Chakraborty** for providing us constant inspiration to stand firm against several setbacks throughout the course. Additionally, we would like to thank all the technical, non-technical and office staffs of our department for extending facilitating cooperation wherever required. We also express gratitude to all of our friends in the department for providing the friendly environment to work on the project work.

We would also like to thank our Director **Prof. Satyajit Chakraborti** for providing us an outstanding platform in order to develop our academic career. In addition, we also preserve a very special thankful feeling about our Principal **Prof. Amlan Kusum Nayak** for being a constant source of inspiration.

A special thank is due to our family. Words cannot express how grateful we are to our parents for all the sacrifices that they have made while giving us necessary strength to stand on our own feet.

Finally, we would like to thank everybody who has provided assistance, in whatever little form, towards successful realization of this project but with an apology that we could not mention everybody’s name individually.

INTRODUCTION

Text-Summarization is done with the help of NLP. It is a method to get most relevant information from a larger text document reducing its size.

Text summarization can broadly be divided into two categories — **Extractive Summarization** and **Abstractive Summarization**.

1. **Extractive Summarization:** These methods rely on extracting several parts, such as phrases and sentences, from a piece of text and stack them together to create a summary. Therefore, identifying the right sentences for summarization is of utmost importance in an extractive method. No new text is generated in this approach.

2. **Abstractive Summarization:** These methods use advanced NLP techniques to generate an entirely new summary. Some parts of this summary may not even appear in the original text

SOURCE TEXT:- Ana and Alina took a taxi to attend night party in the city. While in the party Alina collapsed and was rushed to the hospital.

Extractive Summary:- Ana Alina attend party. Alina collapsed rushed hospital.

Abstractive Summary:- Alina was hospitalized after attending a party with Ana.

There are various Algorithms through which implementations of these Methods are done.

SCOPE AND PURPOSE

Various Situations where text summarizers are used:-

1. News Headline generation.

2. Product Review summary.

3. Information Extraction from Research Paper, Financial reports, Bio-medical data, Literature etc.

Overview Different Text Summarization Algorithms:-

There are different text summarization algorithms. Here we will discuss and implement them and try to find out the best suitable algorithm for different contexts.

1.Sentence Scoring based on Word Frequency

This is the simplest of all approaches. After preprocessing the text, we assign weights to each word based on the frequency of the word in the passage. For example, if “Soccer” occurs 4 times within the passage, it will have a weight of 4. using the weights assigned to each word, we will create a score for each sentence. To create the summary, we will take all the sentence that has a score that exceeds a threshold.

2.TextRank

This is a derivative of the famous PageRank created by the Google Cofounders. In PageRank, they generated a matrix that calculates the probability that a user will move from one page to another. In the case of TextRank, we generate a cosine similarity matrix where we have the similarity of each sentence to each other. The similarity matrix is then converted into a graph, with sentences as vertices and similarity scores as edges, for sentence rank calculation. The PageRank algorithm is then applied to this graph to evaluate the importance of each sentence. The top N sentences will then be used to generate the summary.

3. Skip Thought Vectors

Here is the overall pipeline of this method:
- Text Cleaning
- Encoder/Decoder
- K means Clustering
- Extract summary.

4.Few other approaches

LexRank summarizer uses google’s**page rank algorithm** which is basically an unsupervised approach using connectivity matrix and page rank score (that is basically the probability of visiting a page by the user). This is used for ranking web pages in the online search.

The Reduction algorithm is another graph-based model which values sentences according to the sum of the weights of their edges to other sentences in the document. This weight is computed in the same way as it is in the **TextRank model**.

In day to day life text summarizer is used in various fields. The inshorts app also uses text summarization for reducing the textual data. There are a lot of parameters to tweak before making a final judgement. Some might perform better on shorter summaries, some on longer summaries. The style of writing could make a difference.

Implementations

Here text summarization method based on sentence scoring has been implemented:-

Implementation of Sentence Scoring:

Procedure:-

1. **Convert the paragraph into sentences**
   First, the paragraph is split into its corresponding sentences. The best way of doing the conversion is to extract a sentence whenever a period appears.
2. **Text Pre-processing**
   Next, text processing is done by removing the stop words (extremely common words with little meaning such as “and” and “the”), numbers, punctuation, and other special characters from the sentences.Performing the filtering assists in removing redundant and insignificant information which may not provide any added value to the text’s meaning.
3. **Tokenization**
   Each word of the sentence is converted into individual token that is list of string in python.
4. **Evaluated the weighted occurrence frequency of the words**
   Thereafter, the weighted occurrence frequency of all the words is calculated. To achieve this, the occurrence frequency of each of the words is divided by the frequency of the most recurrent word in the paragraph.
5. **Substitute words with their weighted frequencies**
   Each word found in original sentence is substituted with their weighted frequencies, and then sum of each sentence is calculated. The sentences whose score tops are extracted as summary. Mostly top 30% of total sentences are extracted as summary.

CODE:-

```
from nltk.corpus import stopwords
from nltk import sent_tokenize, word_tokenize
from nltk.stem import PorterStemmer


ps = PorterStemmer()


def text_preprocessing(sentences: list) -> list:
    """
    Pre processing text to remove unnecessary words.
    """
    print('Preprocessing text')

    stop_words = set(stopwords.words('english'))

    clean_words = []

    for sent in sentences:
        # Tokenizing words.
        words = word_tokenize(sent.lower())
        # Removing non alphabetic and numeric words.
        words = [ps.stem(word) for word in words if word.isalnum()]
        # Removing stopwords
        clean_words += [word for word in words if word not in stop_words]
    return clean_words


def create_word_frequency_table(words: list) -> dict:
    """
    Creating word frequency table which contains frequency of each word used in the text.
    """
    print('Creating word frequency table')

    freq_table = dict()

    for word in words:
        if word in freq_table:
            freq_table[word] += 1
        else:
            freq_table[word] = 1

    return freq_table


def create_sentence_score_table(sentences: list, freq_table: dict) -> dict:
    """
    Creating a dictionary to keep the score of each sentence.
    Sore is the sum of frequency of words used in the sentence.
    """
    print('Creating sentence score table')


    sent_value = dict()
    for sentence in sentences:
        for word, freq in freq_table.items():
            if ps.stem(word) in sentence.lower():
                if sentence in sent_value:
                    sent_value[sentence] += freq
                else:
                    sent_value[sentence] = freq


    return sent_value


def create_summary(sentence_values : dict,summary_sentence_count) -> dict:
  summary = dict(sorted(sentence_values.items(), key=lambda item: item[1], reverse=True))
  output = ""
  c = 0
  for pairs in summary:
    if c < summary_sentence_count:
        output += pairs


    else:
        break


    c = c+1;
  return output


print("process is starting")
f = open("inputfile.txt", encoding="utf8")
text = ""
for x in f:
    print(x)
    text += x


# tokenize the sentences


sentences = sent_tokenize(text.strip())
print('Sentences', len(sentences), sentences)


clean_words = text_preprocessing(sentences)
print('Clean Words', len(clean_words), clean_words)


freq_table = create_word_frequency_table(clean_words)
print('Frequency Table', freq_table)


sent_values = create_sentence_score_table(sentences, freq_table)
```

print('Sentence values', sent_values)

total_sentence_count = len(sent_values)

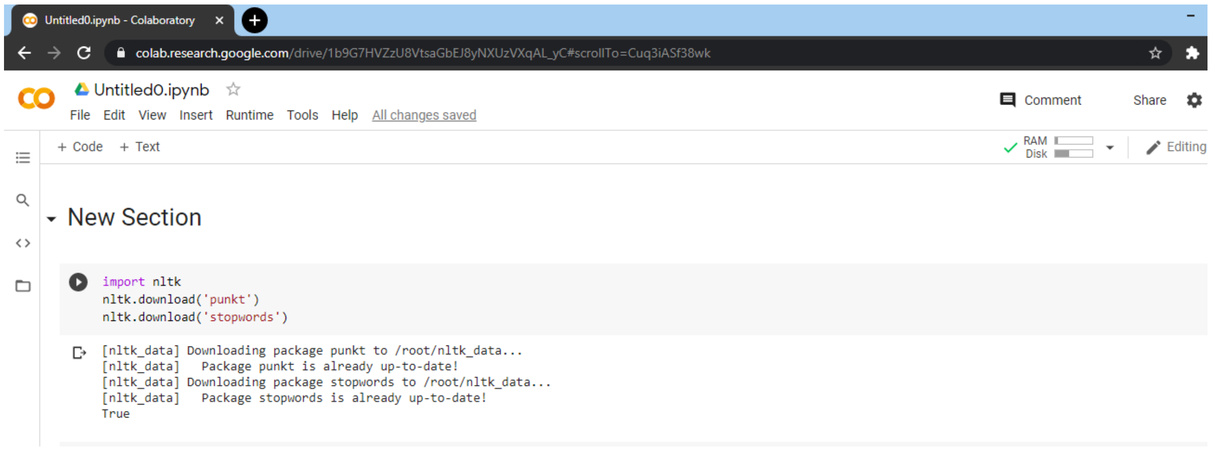
summary_sentence_count = total_sentence_count//4

summary = create_summary(sent_values,summary_sentence_count)

print(summary)

Explanation

1. Natural language toolkit(nltk) is imported and Punkt and stopword Tokenizer Models are downloaded. Punkt is used to make nltk.tokenize.word_tokenize work.nltk**.**download**("**stopwords**")** ,this will store the stopwords corpus under the nltk_data.

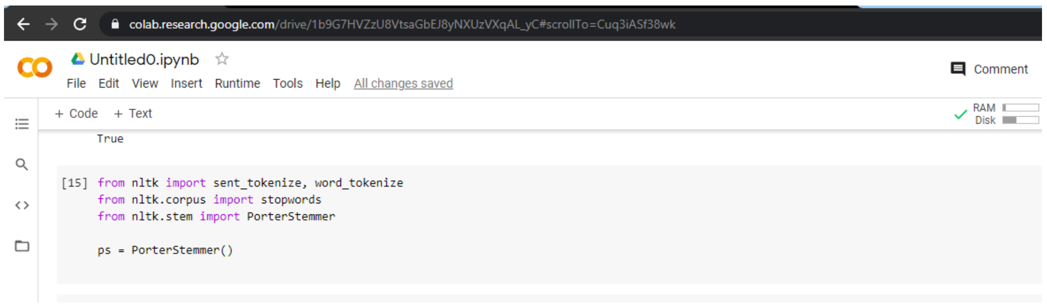


Untitled0.ipynb - Colaboratory
colab.research.google.com/drive/1b9G7HVZzU8VtsaGbEJ8yNXUzVXqAL_yC#scrollTo=Cuq3iASf38wk
Untitled0.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
New Section

```
import nltk
nltk.download('punkt')
nltk.download('stopwords')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
True
```

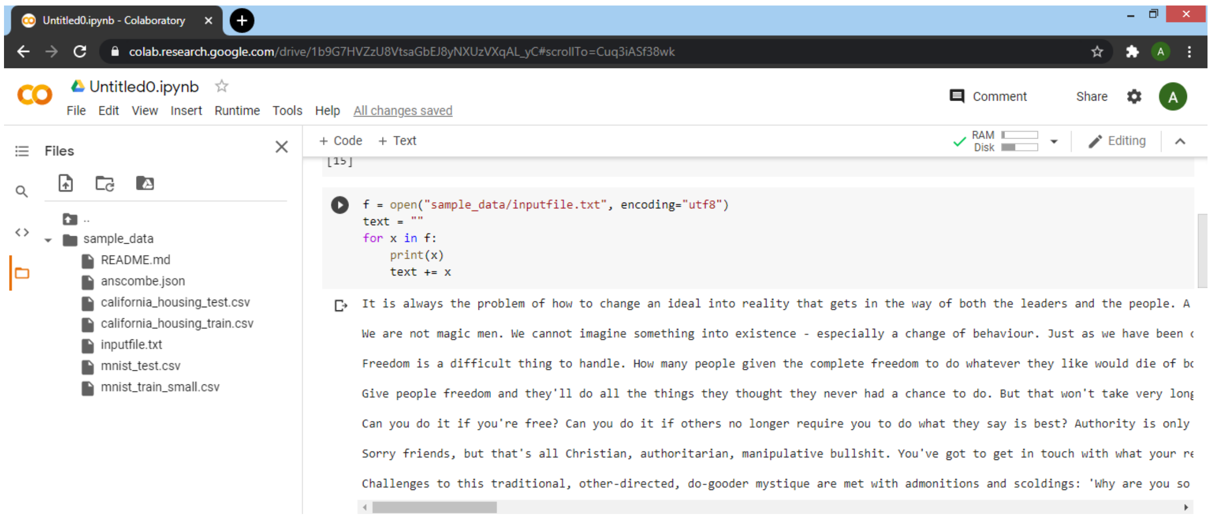
2. Then all the important packages are imported.The Porter stemming algorithm (or 'Porter stemmer') is a process for removing the commoner morphological and inflexional endings from words in English.



colab.research.google.com/drive/1b9G7HVZzU8VtsaGbEJ8yNXUzVXqAL_yC#scrollTo=Cuq3iASf38wk
Untitled0.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
+ Code + Text
RAM Disk

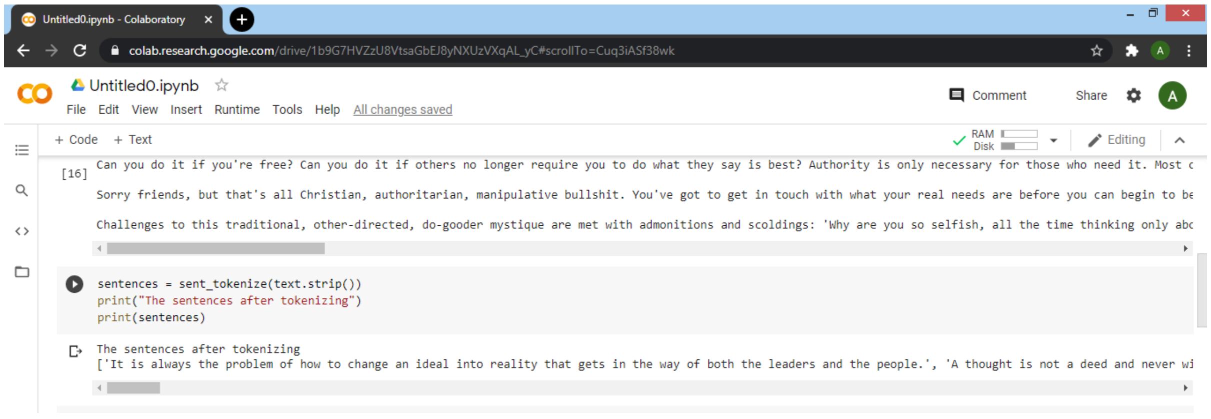
```
True

[15] from nltk import sent_tokenize, word_tokenize
     from nltk.corpus import stopwords
     from nltk.stem import PorterStemmer

     ps = PorterStemmer()
```

3. A sample input file is read.



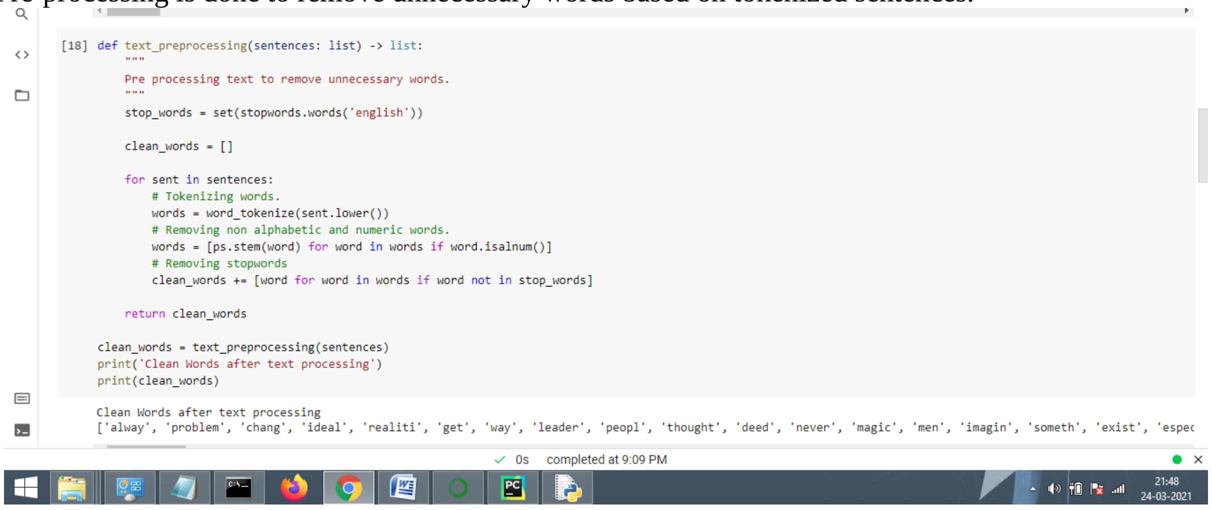
```
[15]

f = open("sample_data/inputfile.txt", encoding="utf8")
text = ""
for x in f:
    print(x)
    text += x

It is always the problem of how to change an ideal into reality that gets in the way of both the leaders and the people. A

We are not magic men. We cannot imagine something into existence - especially a change of behaviour. Just as we have been c

Freedom is a difficult thing to handle. How many people given the complete freedom to do whatever they like would die of bc

Give people freedom and they'll do all the things they thought they never had a chance to do. But that won't take very long

Can you do it if you're free? Can you do it if others no longer require you to do what they say is best? Authority is only

Sorry friends, but that's all Christian, authoritarian, manipulative bullshit. You've got to get in touch with what your re

Challenges to this traditional, other-directed, do-gooder mystique are met with admonitions and scoldings: 'Why are you so
```

4. Sentences are tokenized.



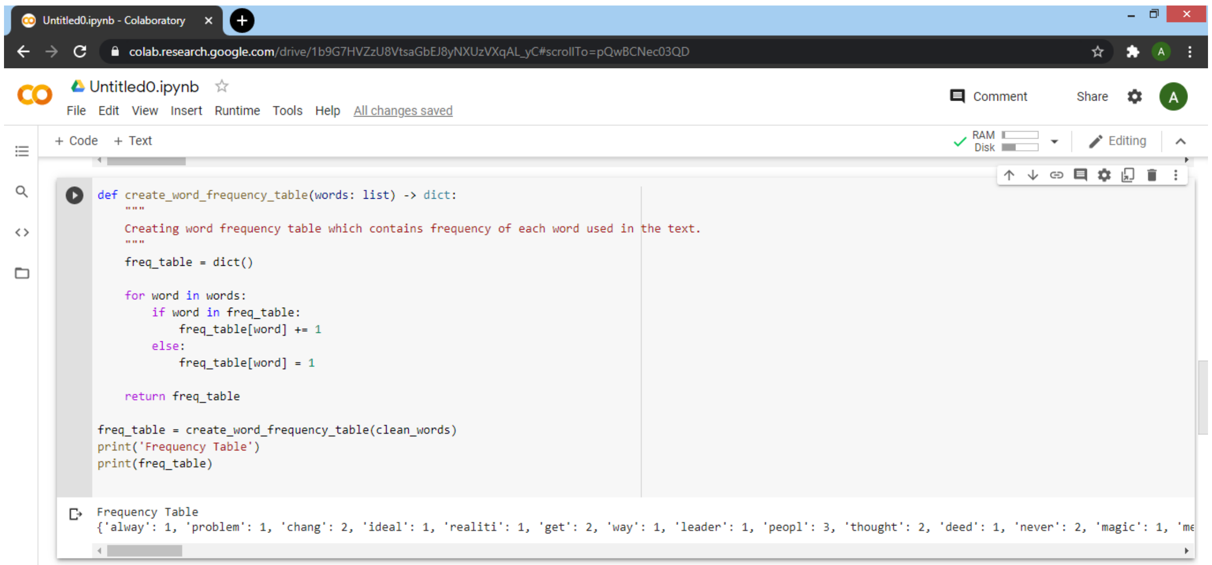
```
[16] Can you do it if you're free? Can you do it if others no longer require you to do what they say is best? Authority is only necessary for those who need it. Most c

     Sorry friends, but that's all Christian, authoritarian, manipulative bullshit. You've got to get in touch with what your real needs are before you can begin to be

     Challenges to this traditional, other-directed, do-gooder mystique are met with admonitions and scoldings: 'Why are you so selfish, all the time thinking only abc

sentences = sent_tokenize(text.strip())
print("The sentences after tokenizing")
print(sentences)

The sentences after tokenizing
['It is always the problem of how to change an ideal into reality that gets in the way of both the leaders and the people.', 'A thought is not a deed and never wi
```

5. Pre-processing is done to remove unnecessary words based on tokenized sentences.



```
[18] def text_preprocessing(sentences: list) -> list:
        """
        Pre processing text to remove unnecessary words.
        """
        stop_words = set(stopwords.words('english'))

        clean_words = []

        for sent in sentences:
            # Tokenizing words.
            words = word_tokenize(sent.lower())
            # Removing non alphabetic and numeric words.
            words = [ps.stem(word) for word in words if word.isalnum()]
            # Removing stopwords
            clean_words += [word for word in words if word not in stop_words]

        return clean_words

    clean_words = text_preprocessing(sentences)
    print('Clean Words after text processing')
    print(clean_words)

    Clean Words after text processing
    ['alway', 'problem', 'chang', 'ideal', 'realiti', 'get', 'way', 'leader', 'peopl', 'thought', 'deed', 'never', 'magic', 'men', 'imagin', 'someth', 'exist', 'espec
```

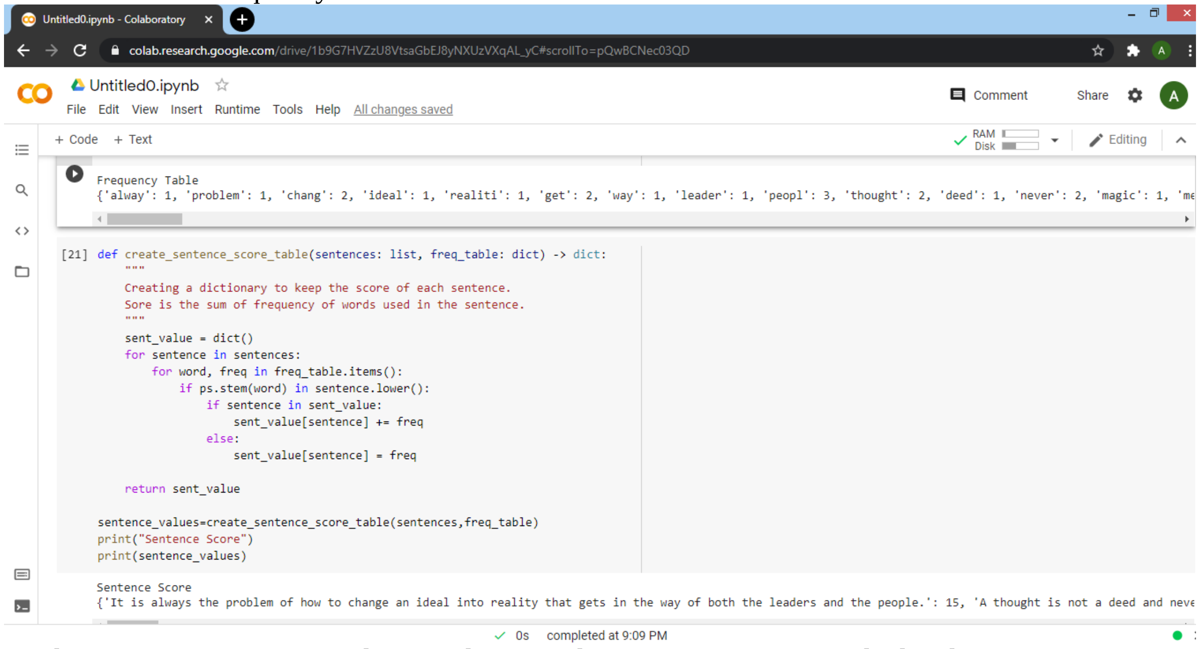
6. A word frequency table is created to keep a count of the frequency of each clean_words.



```python
def create_word_frequency_table(words: list) -> dict:
    """
    Creating word frequency table which contains frequency of each word used in the text.
    """
    freq_table = dict()

    for word in words:
        if word in freq_table:
            freq_table[word] += 1
        else:
            freq_table[word] = 1

    return freq_table

freq_table = create_word_frequency_table(clean_words)
print('Frequency Table')
print(freq_table)
```

```
Frequency Table
{'alway': 1, 'problem': 1, 'chang': 2, 'ideal': 1, 'realiti': 1, 'get': 2, 'way': 1, 'leader': 1, 'peopl': 3, 'thought': 2, 'deed': 1, 'never': 2, 'magic': 1, 'me
```

7. Based on the frequency table sentence score of each table is calculated. Sentence score is the summation of the frequency of each clean word of each sentence.



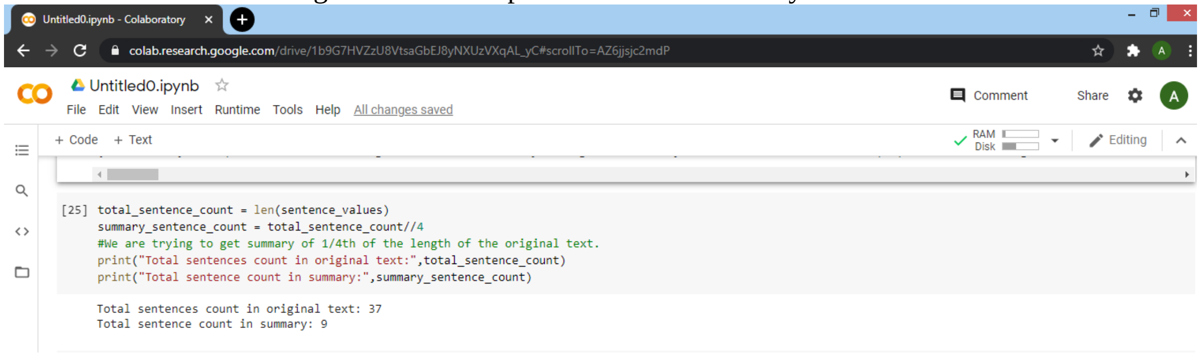
```
Frequency Table
{'alway': 1, 'problem': 1, 'chang': 2, 'ideal': 1, 'realiti': 1, 'get': 2, 'way': 1, 'leader': 1, 'peopl': 3, 'thought': 2, 'deed': 1, 'never': 2, 'magic': 1, 'me
```

```python
[21] def create_sentence_score_table(sentences: list, freq_table: dict) -> dict:
    """
    Creating a dictionary to keep the score of each sentence.
    Sore is the sum of frequency of words used in the sentence.
    """
    sent_value = dict()
    for sentence in sentences:
        for word, freq in freq_table.items():
            if ps.stem(word) in sentence.lower():
                if sentence in sent_value:
                    sent_value[sentence] += freq
                else:
                    sent_value[sentence] = freq

    return sent_value

sentence_values=create_sentence_score_table(sentences,freq_table)
print("Sentence Score")
print(sentence_values)
```

```
Sentence Score
{'It is always the problem of how to change an ideal into reality that gets in the way of both the leaders and the people.': 15, 'A thought is not a deed and neve
```

✓ 0s completed at 9:09 PM

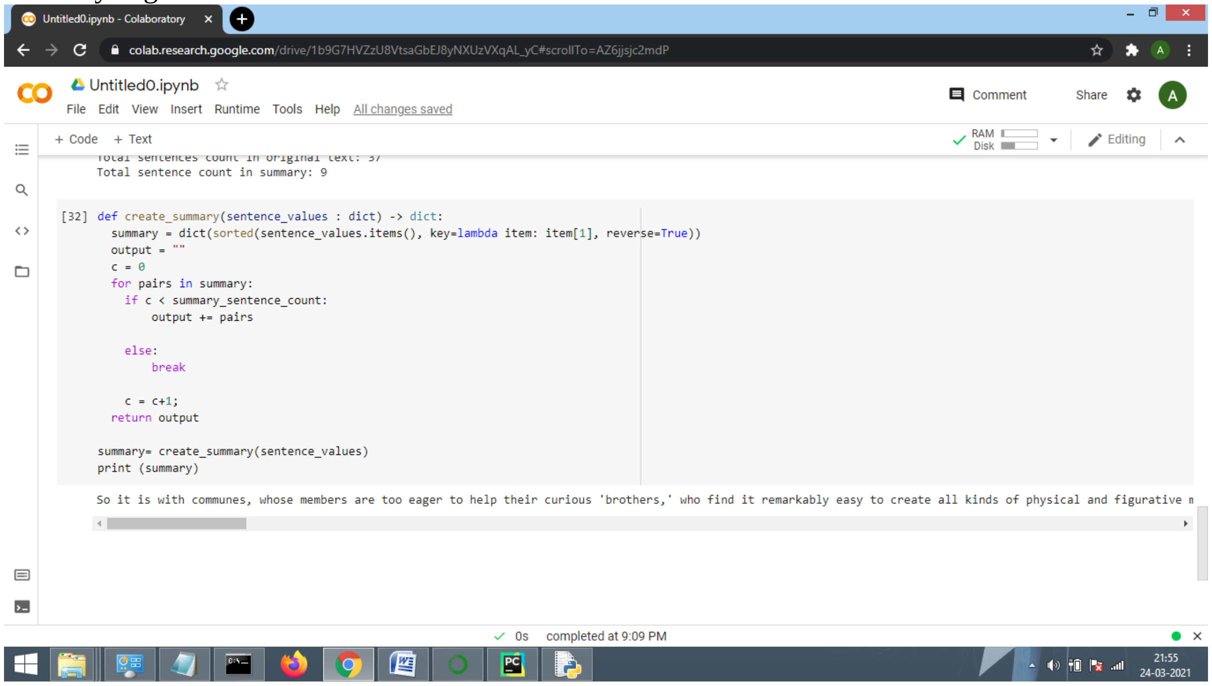
8. Total sentence count in original text and expected count in summary is calculated.



```python
[25] total_sentence_count = len(sentence_values)
summary_sentence_count = total_sentence_count//4
#We are trying to get summary of 1/4th of the length of the original text.
print("Total sentences count in original text:",total_sentence_count)
print("Total sentence count in summary:",summary_sentence_count)
```

```
Total sentences count in original text: 37
Total sentence count in summary: 9
```

9. Lastly top summary-sentence-count sentences with maximum sentence score are selected. And summary is generated.



```
Total sentences count in original text: 37
Total sentence count in summary: 9
```

```
[32] def create_summary(sentence_values : dict) -> dict:
       summary = dict(sorted(sentence_values.items(), key=lambda item: item[1], reverse=True))
       output = ""
       c = 0
       for pairs in summary:
         if c < summary_sentence_count:
             output += pairs

         else:
             break

         c = c+1;
       return output

     summary= create_summary(sentence_values)
     print (summary)
```

```
So it is with communes, whose members are too eager to help their curious 'brothers,' who find it remarkably easy to create all kinds of physical and figurative m
```

Implementation of TextRank Algorithm:

TextRank summarization method is based on Google's PageRank Algorithm.
PageRank assumes that the rank of a webpage W depends on the importance of a webpage suggested by other web pages in terms of links to the page i.e if a webpage ‘X’ has a link to webpage ‘W’, ‘X’ contributes to the importance of ‘W’.
In the similar way, for the task of automated summarization TextRank models any document as a graph using sentences as nodes.
A function to compute similarity of sentences is needed to build edges in between. This function is used to weight the graph edge, the higher the similarity between sentences the more important the edge between them will be in the graph.
TextRank determines the relation of similarity between two sentences based on content that they both share. This overlap is calculated simply as number of common lexical tokens between them, divided by the length of each to avoid promoting long sentences.

Procedure:-

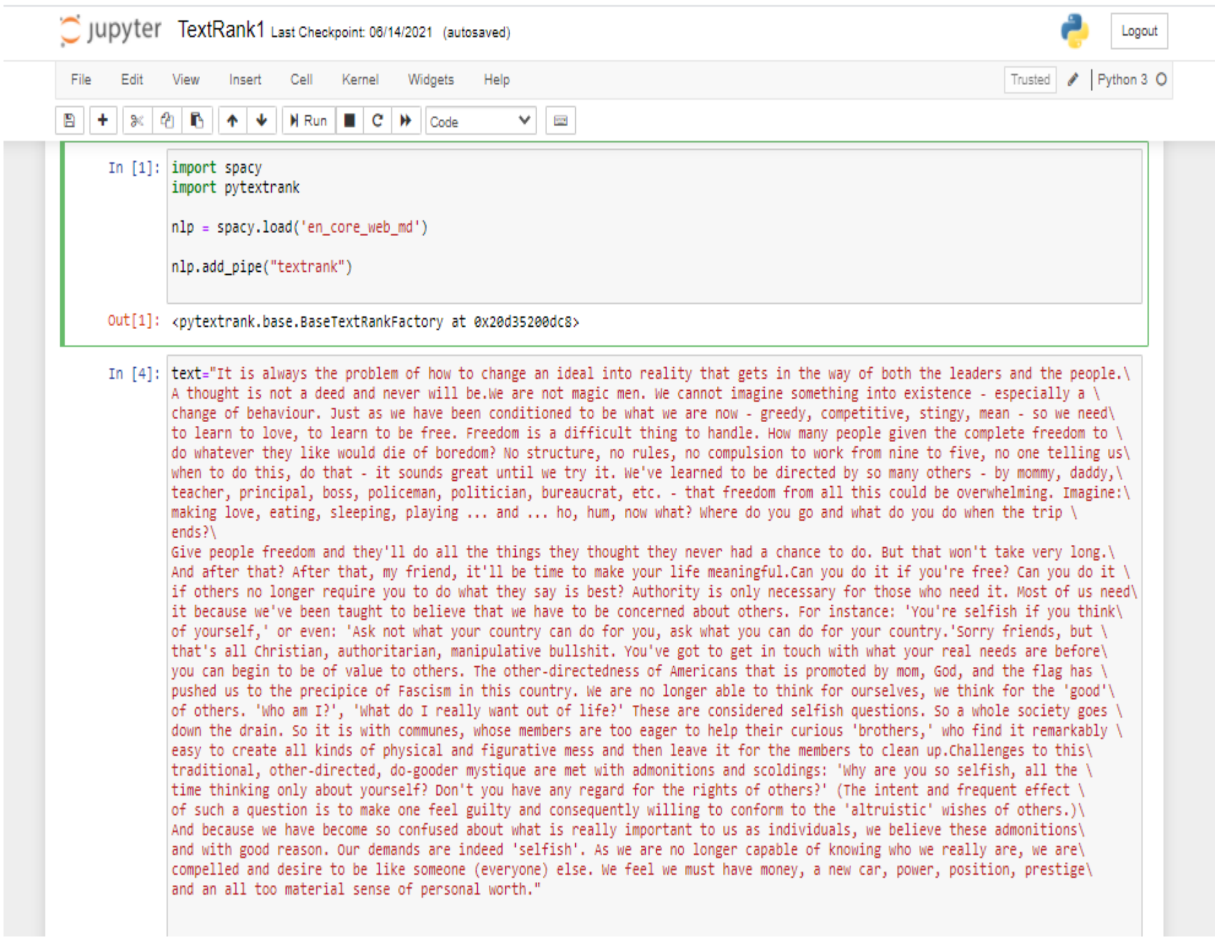
1. **Text Pre-processing**:- Take one paragraph, Remove stop words such as this, that, is, was, a, an, the. .etc. and convert words of sentences in tokens, then lemmatize the tokens. Whatever we get then is called pre-processed text.
2. **Formation of Similarity Matrix**:-In the next step, vector representation (word embedding) for each and every sentence is done .Similarities between sentence vectors are then calculated and stored in a matrix. The similarity matrix is then converted into a graph, with sentences as vertices and similarity scores as edges, for sentence rank calculation
3. **Summary Extraction**:-Finally, a certain number of top-ranked sentences form the final summary.

CODE:-

```
import spacy
# pytextrank is python implementation of textrank as spacy pipeline extension, we need to import that.
import pytextrank
# you need to download English Spacy Model
nlp = spacy.load('en_core_web_md')

# add initializing textrank pipeline.
nlp.add_pipe("textrank")
# put input text into text.
text=” ”

doc = nlp(text)
# examine the top-ranked phrases in the document
for p in doc._.phrases:
    print('{:.4f} {:5d}  {}'.format(p.rank, p.count, p.text))
    print(p.chunks)
# select top scored sentences
for sent in doc._.textrank.summary(limit_phrases=15, limit_sentences=5):
    print(sent)
```

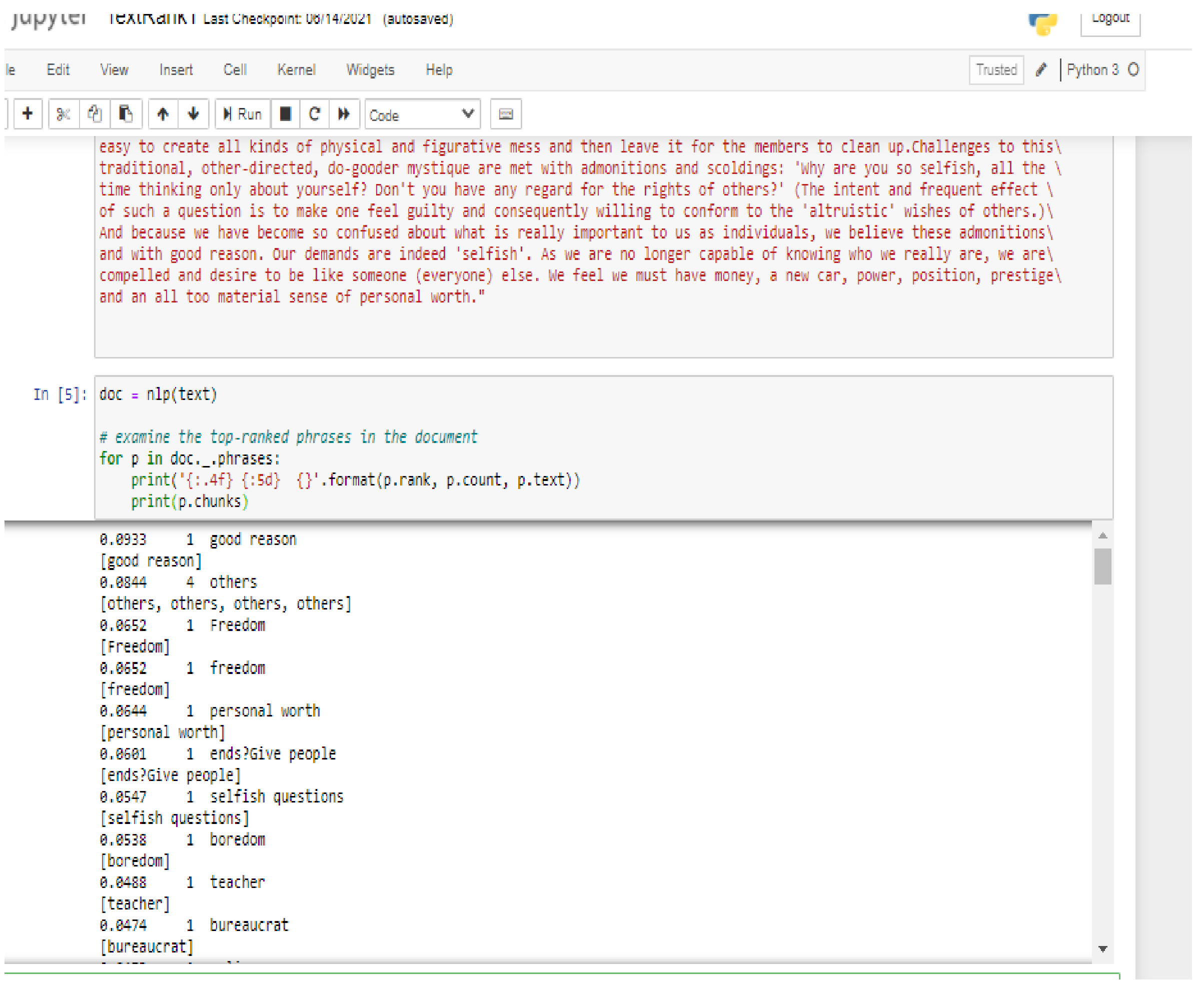
Explanation

1. Necessary libraries such as spacy and pytextrank, that is python implementation of TextRank as a spaCy pipeline extension, has been imported. Then English pipeline optimizer “en_core_web_sm” , that does all text pre-processing, tokenization, lemmatization has been loaded.TextRank algorithm has been initialized by calling “nlp.add_pipe”.
2. In the Text field Input text has been inserted.


jupyter TextRank1 Last Checkpoint: 06/14/2021 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted Python 3
Run Code

```
In [1]: import spacy
        import pytextrank

        nlp = spacy.load('en_core_web_md')

        nlp.add_pipe("textrank")

Out[1]: <pytextrank.base.BaseTextRankFactory at 0x20d35200dc8>

In [4]: text="It is always the problem of how to change an ideal into reality that gets in the way of both the leaders and the people.\
A thought is not a deed and never will be.We are not magic men. We cannot imagine something into existence - especially a \
change of behaviour. Just as we have been conditioned to be what we are now - greedy, competitive, stingy, mean - so we need\
to learn to love, to learn to be free. Freedom is a difficult thing to handle. How many people given the complete freedom to \
do whatever they like would die of boredom? No structure, no rules, no compulsion to work from nine to five, no one telling us\
when to do this, do that - it sounds great until we try it. We've learned to be directed by so many others - by mommy, daddy,\
teacher, principal, boss, policeman, politician, bureaucrat, etc. - that freedom from all this could be overwhelming. Imagine:\
making love, eating, sleeping, playing ... and ... ho, hum, now what? Where do you go and what do you do when the trip \
ends?\
Give people freedom and they'll do all the things they thought they never had a chance to do. But that won't take very long.\
And after that? After that, my friend, it'll be time to make your life meaningful.Can you do it if you're free? Can you do it \
if others no longer require you to do what they say is best? Authority is only necessary for those who need it. Most of us need\
it because we've been taught to believe that we have to be concerned about others. For instance: 'You're selfish if you think\
of yourself,' or even: 'Ask not what your country can do for you, ask what you can do for your country.'Sorry friends, but \
that's all Christian, authoritarian, manipulative bullshit. You've got to get in touch with what your real needs are before\
you can begin to be of value to others. The other-directedness of Americans that is promoted by mom, God, and the flag has \
pushed us to the precipice of Fascism in this country. We are no longer able to think for ourselves, we think for the 'good'\
of others. 'Who am I?', 'What do I really want out of life?' These are considered selfish questions. So a whole society goes \
down the drain. So it is with communes, whose members are too eager to help their curious 'brothers,' who find it remarkably \
easy to create all kinds of physical and figurative mess and then leave it for the members to clean up.Challenges to this\
traditional, other-directed, do-gooder mystique are met with admonitions and scoldings: 'Why are you so selfish, all the \
time thinking only about yourself? Don't you have any regard for the rights of others?' (The intent and frequent effect \
of such a question is to make one feel guilty and consequently willing to conform to the 'altruistic' wishes of others.)\
And because we have become so confused about what is really important to us as individuals, we believe these admonitions\
and with good reason. Our demands are indeed 'selfish'. As we are no longer capable of knowing who we really are, we are\
compelled and desire to be like someone (everyone) else. We feel we must have money, a new car, power, position, prestige\
and an all too material sense of personal worth."
```

3. Input text has been fed into nlp pipeline, that implements textRank Algorithm. Top ranked phrases in the sentence has been printed.

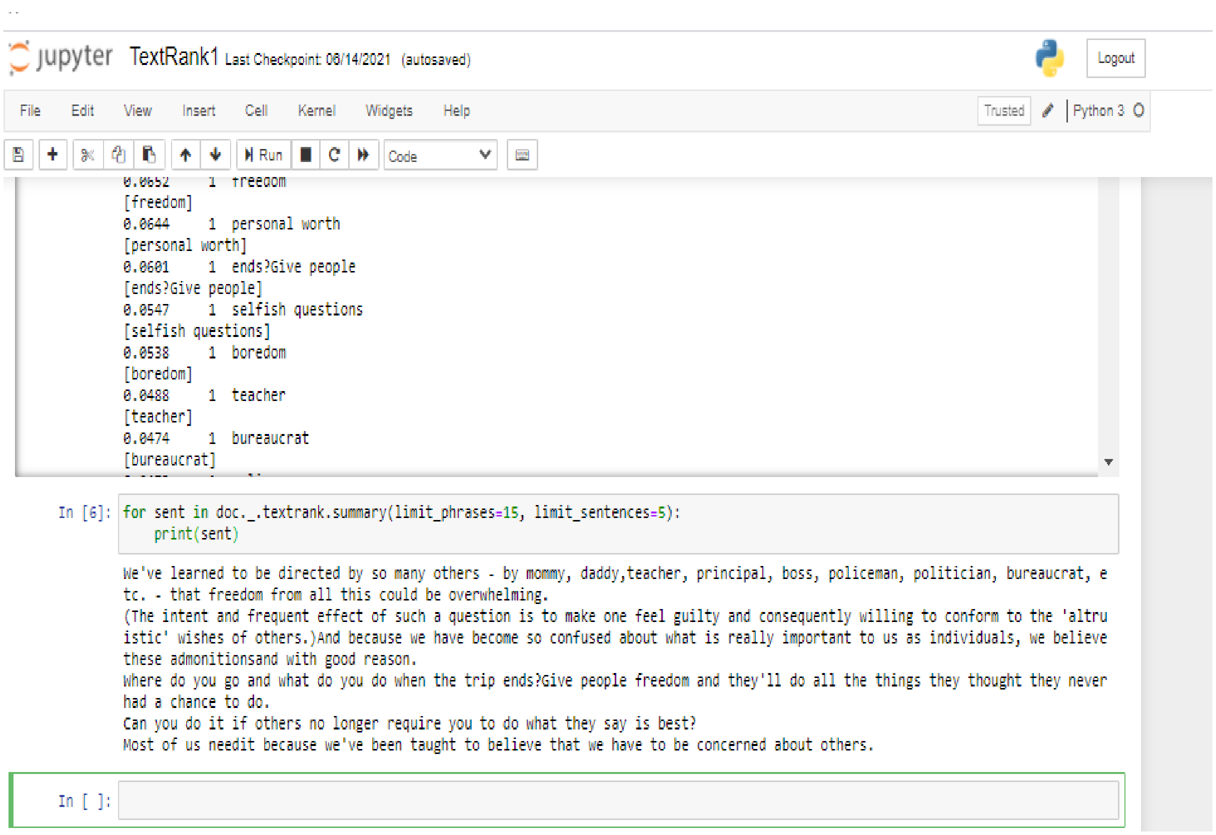

jupyter TextRankT Last Checkpoint: 08/14/2021 (autosaved)
Logout
le Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

```
easy to create all kinds of physical and figurative mess and then leave it for the members to clean up.Challenges to this\
traditional, other-directed, do-gooder mystique are met with admonitions and scoldings: 'Why are you so selfish, all the \
time thinking only about yourself? Don't you have any regard for the rights of others?' (The intent and frequent effect \
of such a question is to make one feel guilty and consequently willing to conform to the 'altruistic' wishes of others.)\
And because we have become so confused about what is really important to us as individuals, we believe these admonitions\
and with good reason. Our demands are indeed 'selfish'. As we are no longer capable of knowing who we really are, we are\
compelled and desire to be like someone (everyone) else. We feel we must have money, a new car, power, position, prestige\
and an all too material sense of personal worth."
```

```
In [5]: doc = nlp(text)

        # examine the top-ranked phrases in the document
        for p in doc._.phrases:
            print('{:.4f} {:5d}  {}'.format(p.rank, p.count, p.text))
            print(p.chunks)
```

```
0.0933     1  good reason
[good reason]
0.0844     4  others
[others, others, others, others]
0.0652     1  Freedom
[Freedom]
0.0652     1  freedom
[freedom]
0.0644     1  personal worth
[personal worth]
0.0601     1  ends?Give people
[ends?Give people]
0.0547     1  selfish questions
[selfish questions]
0.0538     1  boredom
[boredom]
0.0488     1  teacher
[teacher]
0.0474     1  bureaucrat
[bureaucrat]
```

4. Top 5 sentences has been printed as summary.


jupyter TextRank1 Last Checkpoint: 06/14/2021 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

```
0.0652     1  freedom
[freedom]
0.0644     1  personal worth
[personal worth]
0.0601     1  ends?Give people
[ends?Give people]
0.0547     1  selfish questions
[selfish questions]
0.0538     1  boredom
[boredom]
0.0488     1  teacher
[teacher]
0.0474     1  bureaucrat
[bureaucrat]
```

```
In [6]: for sent in doc._.textrank.summary(limit_phrases=15, limit_sentences=5):
            print(sent)
```

```
We've learned to be directed by so many others - by mommy, daddy,teacher, principal, boss, policeman, politician, bureaucrat, e
tc. - that freedom from all this could be overwhelming.
(The intent and frequent effect of such a question is to make one feel guilty and consequently willing to conform to the 'altru
istic' wishes of others.)And because we have become so confused about what is really important to us as individuals, we believe
these admonitionsand with good reason.
Where do you go and what do you do when the trip ends?Give people freedom and they'll do all the things they thought they never
had a chance to do.
Can you do it if others no longer require you to do what they say is best?
Most of us needit because we've been taught to believe that we have to be concerned about others.
```

```
In [ ]:
```

Implementation of Skip-thought vector:

Procedure:-

1. **Sentence-tokenizing**
   First, the paragraph is split into its corresponding sentences. The best way of doing the conversion is to extract a sentence whenever a period appears.
2. **Encoder/Decoder Network**
   The encoder is typically a GRU-RNN which generates a fixed length vector representation *h*(i) for each sentence *S(i)* in the input. The encoded representation *h(i)* is obtained by passing final hidden state of the GRU cell (i.e. after it has seen the entire sentence) to multiple dense layers. The decoder network takes this vector representation *h*(i) as input and tries to generate two sentences — *S(i-1)* and *S(i+1)*, which could occur before and after the input sentence respectively. Separate decoders are implemented for generation of previous and next sentences, both being GRU-RNNs. The vector representation *h(i)* acts as the initial hidden state for the GRUs of the decoder networks.
3. **K-means clustering**
   Each cluster will have some center point which, in the vector space, would indicate the point which closely represents the theme of that cluster. With this in mind, when trying to create a summary, we should only need the sentence which is the closest to the center of that cluster. The key here is choosing the correct number of clusters to do a good job of summarizing the content.

4. **Extract sentences**
   Extract the sentences which are closest to the centre of the cluster.

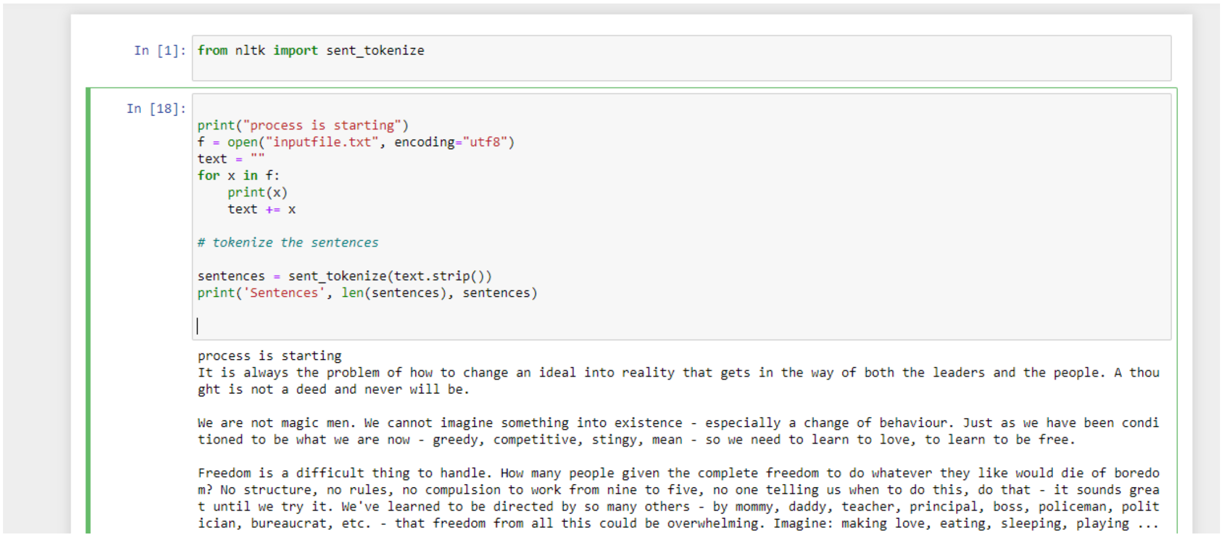
CODE:-

```
from nltk import sent_tokenize, word_tokenize


print("process is starting")
f = open("inputfile.txt", encoding="utf8")
text = ""
for x in f:
    print(x)
    text += x

# tokenize the sentences

sentences = sent_tokenize(text.strip())
print('Sentences', len(sentences), sentences)


import skipthoughts

# You would need to download pre-trained models first
model = skipthoughts.load_model()
encoder = skipthoughts.Encoder(model)
encoded =  encoder.encode(sentences)



import numpy as np
from sklearn.cluster import KMeans

n_clusters =int (np.ceil(len(encoded)**0.5))
kmeans = KMeans(n_clusters=n_clusters)
kmeans = kmeans.fit(encoded)


from sklearn.metrics import pairwise_distances_argmin_min

avg = []
for j in range(n_clusters):
    idx = np.where(kmeans.labels_ == j)[0]
    avg.append(np.mean(idx))

closest, _ = pairwise_distances_argmin_min(kmeans.cluster_centers_, encoded)
ordering = sorted(range(n_clusters), key=lambda k: avg[k])
summary = ' '.join([sentences[closest[idx]] for idx in ordering])


print(summary)
```

Explanation

In this case we have used a pre-trained model available in the github.The end result is an off-the-shelf encoder that can produce highly generic sentence representations that are robust and perform well in practice. (github repository link:https://github.com/ryankiros/skip-thoughts)

1.Sentences are tokenized

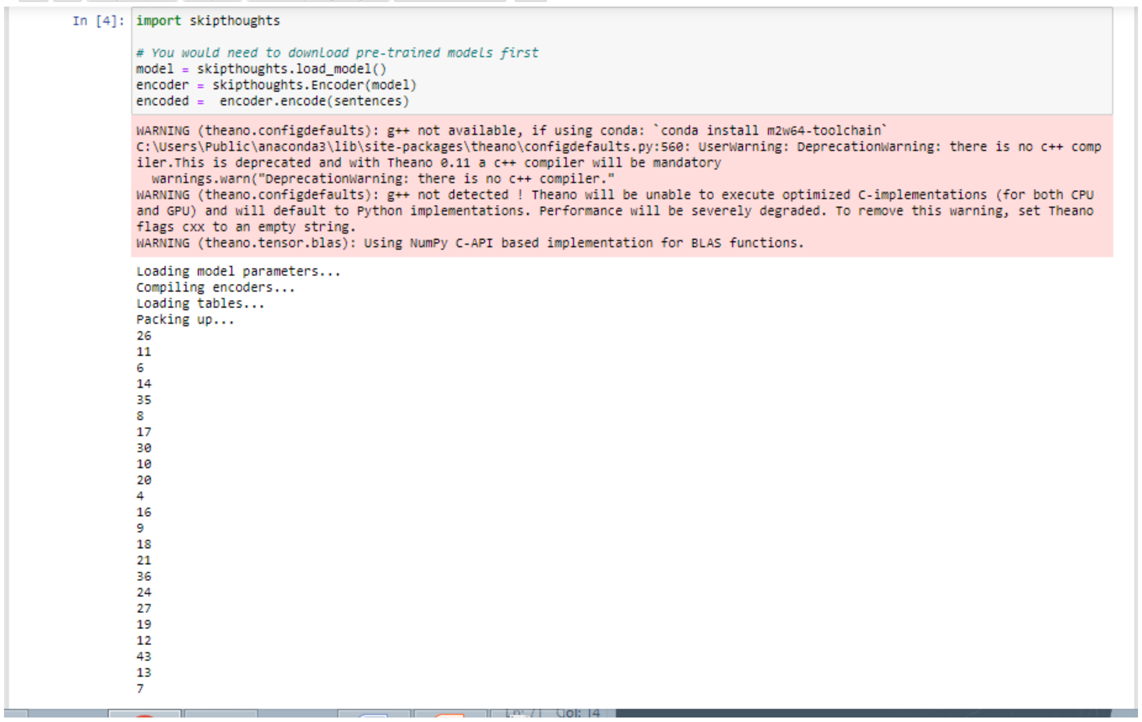


```
In [1]: from nltk import sent_tokenize
```

```
In [18]:
print("process is starting")
f = open("inputfile.txt", encoding="utf8")
text = ""
for x in f:
    print(x)
    text += x

# tokenize the sentences

sentences = sent_tokenize(text.strip())
print('Sentences', len(sentences), sentences)
```

```
process is starting
It is always the problem of how to change an ideal into reality that gets in the way of both the leaders and the people. A thou
ght is not a deed and never will be.

We are not magic men. We cannot imagine something into existence - especially a change of behaviour. Just as we have been condi
tioned to be what we are now - greedy, competitive, stingy, mean - so we need to learn to love, to learn to be free.

Freedom is a difficult thing to handle. How many people given the complete freedom to do whatever they like would die of boredo
m? No structure, no rules, no compulsion to work from nine to five, no one telling us when to do this, do that - it sounds grea
t until we try it. We've learned to be directed by so many others - by mommy, daddy, teacher, principal, boss, policeman, polit
ician, bureaucrat, etc. - that freedom from all this could be overwhelming. Imagine: making love, eating, sleeping, playing ...
```

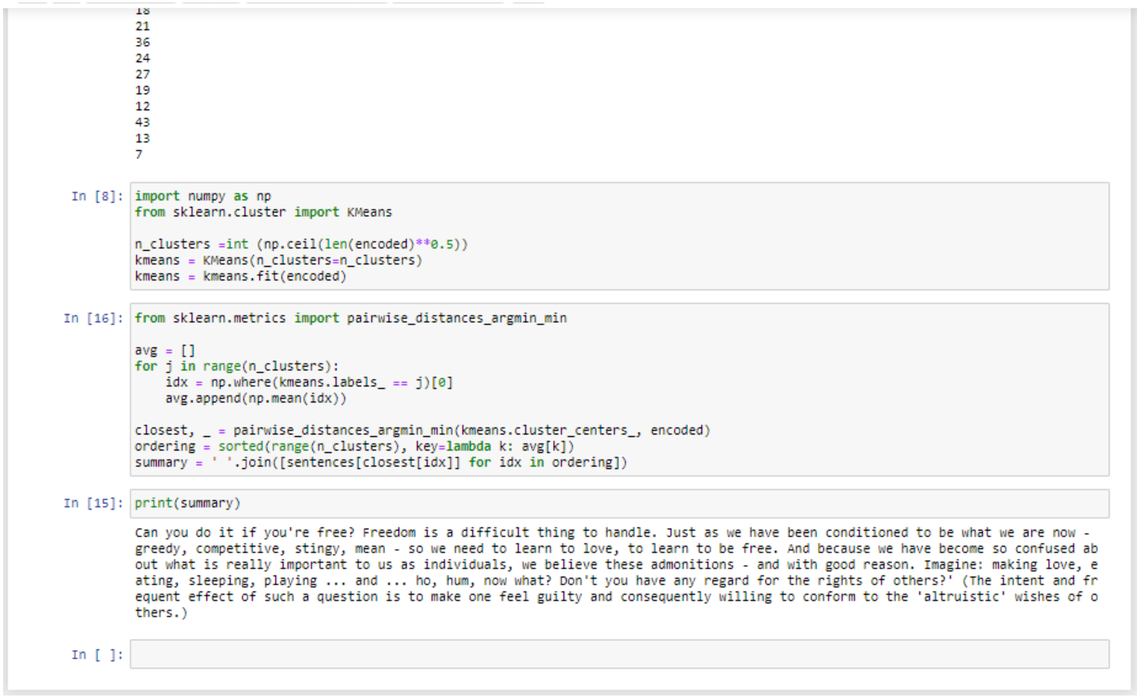
2.Using the pre-defined skipthought module the sentences are encoded.



```
In [4]: import skipthoughts

# You would need to download pre-trained models first
model = skipthoughts.load_model()
encoder = skipthoughts.Encoder(model)
encoded =  encoder.encode(sentences)
```

```
WARNING (theano.configdefaults): g++ not available, if using conda: `conda install m2w64-toolchain`
C:\Users\Public\anaconda3\lib\site-packages\theano\configdefaults.py:560: UserWarning: DeprecationWarning: there is no c++ comp
iler.This is deprecated and with Theano 0.11 a c++ compiler will be mandatory
  warnings.warn("DeprecationWarning: there is no c++ compiler."
WARNING (theano.configdefaults): g++ not detected ! Theano will be unable to execute optimized C-implementations (for both CPU
and GPU) and will default to Python implementations. Performance will be severely degraded. To remove this warning, set Theano
flags cxx to an empty string.
WARNING (theano.tensor.blas): Using NumPy C-API based implementation for BLAS functions.

Loading model parameters...
Compiling encoders...
Loading tables...
Packing up...
26
11
6
14
35
8
17
30
10
20
4
16
9
18
21
36
24
27
19
12
43
13
7
```

3.Using the K-means clustering algorithm sentences closest to the centre of the cluster are extracted.



```
18
21
36
24
27
19
12
43
13
7
```

```
In [8]: import numpy as np
        from sklearn.cluster import KMeans

        n_clusters =int (np.ceil(len(encoded)**0.5))
        kmeans = KMeans(n_clusters=n_clusters)
        kmeans = kmeans.fit(encoded)
```

```
In [16]: from sklearn.metrics import pairwise_distances_argmin_min

         avg = []
         for j in range(n_clusters):
             idx = np.where(kmeans.labels_ == j)[0]
             avg.append(np.mean(idx))

         closest, _ = pairwise_distances_argmin_min(kmeans.cluster_centers_, encoded)
         ordering = sorted(range(n_clusters), key=lambda k: avg[k])
         summary = ' '.join([sentences[closest[idx]] for idx in ordering])
```

```
In [15]: print(summary)

         Can you do it if you're free? Freedom is a difficult thing to handle. Just as we have been conditioned to be what we are now -
         greedy, competitive, stingy, mean - so we need to learn to love, to learn to be free. And because we have become so confused ab
         out what is really important to us as individuals, we believe these admonitions - and with good reason. Imagine: making love, e
         ating, sleeping, playing ... and ... ho, hum, now what? Don't you have any regard for the rights of others?' (The intent and fr
         equent effect of such a question is to make one feel guilty and consequently willing to conform to the 'altruistic' wishes of o
         thers.)
```

```
In [ ]:
```

Output

Input Text:

It is always the problem of how to change an ideal into reality that gets in the way of both the leaders and the people. A thought is not a deed and never will be.
We are not magic men. We cannot imagine something into existence - especially a change of behaviour. Just as we have been conditioned to be what we are now - greedy, competitive, stingy, mean - so we need to learn to love, to learn to be free.
Freedom is a difficult thing to handle. How many people given the complete freedom to do whatever they like would die of boredom? No structure, no rules, no compulsion to work from nine to five, no one telling us when to do this, do that - it sounds great until we try it. We've learned to be directed by so many others - by mommy, daddy, teacher, principal, boss, policeman, politician, bureaucrat, etc. - that freedom from all this could be overwhelming. Imagine: making love, eating, sleeping, playing ... and ... ho, hum, now what? Where do you go and what do you do when the trip ends?
Give people freedom and they'll do all the things they thought they never had a chance to do. But that won't take very long. And after that? After that, my friend, it'll be time to make your life meaningful.
Can you do it if you're free? Can you do it if others no longer require you to do what they say is best?
Authority is only necessary for those who need it. Most of us need it because we've been taught to believe that we have to be concerned about others. For instance: 'You're selfish if you think of yourself,' or even: 'Ask not what your country can do for you, ask what you can do for your country.'
Sorry friends, but that's all Christian, authoritarian, manipulative bullshit. You've got to get in touch with what your real needs are before you can begin to be of value to others. The other-directedness of Americans that is promoted by mom, God, and the flag has pushed us to the precipice of Fascism in this country. We are no longer able to think for ourselves, we think for the 'good' of others. 'Who am I?', 'What do I really want out of life?' These are considered selfish questions. So a whole society goes down the drain. So it is with communes, whose members are too eager to help their curious 'brothers,' who find

it remarkably easy to create all kinds of physical and figurative mess and then leave it for the members to clean up.
Challenges to this traditional, other-directed, do-gooder mystique are met with admonitions and scoldings: 'Why are you so selfish, all the time thinking only about yourself? Don't you have any regard for the rights of others?' (The intent and frequent effect of such a question is to make one feel guilty and consequently willing to conform to the 'altruistic' wishes of others.) And because we have become so confused about what is really important to us as individuals, we believe these admonitions - and with good reason. Our demands are indeed 'selfish'. As we are no longer capable of knowing who we really are, we are compelled and desire to be like someone (everyone) else. We feel we must have money, a new car, power, position, prestige, and an all too material sense of personal worth.

Summary generated by sentence scoring:

So it is with communes, whose members are too eager to help their curious 'brothers,' who find it remarkably easy to create all kinds of physical and figurative mess and then leave it for the members to clean up.Challenges to this traditional, other-directed, do-gooder mystique are met with admonitions and scoldings: 'Why are you so selfish, all the time thinking only about yourself?No structure, no rules, no compulsion to work from nine to five, no one telling us when to do this, do that - it sounds great until we try it.Give people freedom and they'll do all the things they thought they never had a chance to do.Just as we have been conditioned to be what we are now - greedy, competitive, stingy, mean - so we need to learn to love, to learn to be free.We feel we must have money, a new car, power, position, prestige, and an all too material sense of personal worth.How many people given the complete freedom to do whatever they like would die of boredom?And because we have become so confused about what is really important to us as individuals, we believe these admonitions - and with good reason.The other-directedness of Americans that is promoted by mom, God, and the flag has pushed us to the precipice of Fascism in this country.

Summary generated by TextRank:

We've learned to be directed by so many others - by mommy, daddy,teacher, principal, boss, policeman, politician, bureaucrat, etc. - that freedom from all this could be overwhelming.
(The intent and frequent effect of such a question is to make one feel guilty and consequently willing to conform to the 'altruistic' wishes of others.)And because we have become so confused about what is really important to us as individuals, we believe these admonitionsand with good reason.
Where do you go and what do you do when the trip ends?Give people freedom and they'll do all the things they thought they never had a chance to do.
Can you do it if others no longer require you to do what they say is best?
Most of us needit because we've been taught to believe that we have to be concerned about others.

Summary generated by Skip-thought vector:
Can you do it if you're free? Freedom is a difficult thing to handle. Just as we have been conditioned to be what we are now - greedy, competitive, stingy, mean - so we need to learn to love, to learn to be free. And because we have become so confused about what is really important to us as individuals, we believe these admonitions - and with good reason. Imagine: making love, eating, sleeping, playing ... and ... ho, hum, now what? Don't you have any regard for the rights of others?' (The intent and frequent effect of such a question is to make one feel guilty and consequently willing to conform to the 'altruistic' wishes of others.)

Decision

- This Sentence scoring based on word frequency approach gives good output but drawback here is that this method is heavily reliant on the vocabulary in the article. Some similar word such as “people” and “person” are counted separately when they should be considered same as they have same intent. So to overcome this drawback we move to next approaches.

- Text rank- TextRank implementations tend to be lightweight and can run fast even with limited memory resources. The only disadvantage is that it omit the keywords which has lower chance to appear though being meaningful in context.

- Skipthought vector results in an impressive summary between three of the used approaches as it seems to capture the idiomatic meaning of the words

Conclusion

In this project we implement three different text summarization algorithms and observer how their results vary with each other. Based on different use case different algorithms will be used. But the best among them is skip thought vector implementation that gives us most unbiased summary and focus on each aspect of given text sentences.

Reference

1. https://towardsdatascience.com/text-summarization-on-the-books-of-harry-potter-5e9f5bf8ca6c
2. https://ieeexplore.ieee.org/document/7508049
3. https://www.analyticsvidhya.com/blog/2018/11/introduction-text-summarization-textrank-python
4. https://arxiv.org/abs/1506.06726
5. https://github.com/ryankiros/skip-thoughts
6. Natural Language Processing by Jacob Eisenstein , November 13, 2018
7. A Review Paper on Text Summarization Deepali K. Gaikwad1 and C. Namrata Mahender2(https://www.ijarcce.com/upload/2016/march-16/IJARCCE%2040.pdf)
8. International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056 Volume: 04 Issue: 04 | Apr -2017 p-ISSN: 2395-0072 © 2017, IRJET | Impact Factor value: 5.181 | ISO 9001:2008 Certified Journal | Page 1777 Text Summarization using Sentence Scoring Method T. Sri Rama Raju, Bhargav Allarpu(https://www.irjet.net/archives/V4/i5/IRJET-V4I5493.pdf)
9. Skip-Thought Vectors Ryan Kiros 1 , Yukun Zhu 1 , Ruslan Salakhutdinov 1,2 , Richard S. Zemel 1,2 Antonio Torralba 3 , Raquel Urtasun 1 , Sanja Fidler 1 University of Toronto 1 Canadian Institute for Advanced Research 2 Massachusetts Institute of Technology (https://papers.nips.cc/paper/2015/file/f442d33fa06832082290ad8544a8da27-Paper.pdf)
10. TextRank: Bringing Order into Texts Rada Mihalcea and Paul Tarau Department of Computer Science University of North Texas (https://web.eecs.umich.edu/~mihalcea/papers/mihalcea.emnlp04.pdf)

SIGNATURE OF STUDENTS:

1. **Sonakshi Kumari**
2. **Ananya Mukherjee**

SIGNATURE OF MENTOR:

DATE :June 25, 2021.