In [3]:
```
#!pip install spacy

#!python -m spacy download en_core_web_sm
```

```
Requirement already satisfied: spacy in c:\users\91996\anaconda3\lib\site-packages (3.5.
0)
Requirement already satisfied: typer<0.8.0,>=0.3.0 in c:\users\91996\anaconda3\lib\site-
packages (from spacy) (0.7.0)
Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in c:\users\91996\anaconda3\l
ib\site-packages (from spacy) (1.0.4)
Requirement already satisfied: pathy>=0.10.0 in c:\users\91996\anaconda3\lib\site-packag
es (from spacy) (0.10.1)
Requirement already satisfied: wasabi<1.2.0,>=0.9.1 in c:\users\91996\anaconda3\lib\site
-packages (from spacy) (1.1.1)
Requirement already satisfied: requests<3.0.0,>=2.13.0 in c:\users\91996\anaconda3\lib\s
ite-packages (from spacy) (2.28.1)
Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.11 in c:\users\91996\anaconda3\l
ib\site-packages (from spacy) (3.0.12)
Requirement already satisfied: smart-open<7.0.0,>=5.2.1 in c:\users\91996\anaconda3\lib
\site-packages (from spacy) (5.2.1)
Requirement already satisfied: langcodes<4.0.0,>=3.2.0 in c:\users\91996\anaconda3\lib\s
ite-packages (from spacy) (3.3.0)
Requirement already satisfied: srsly<3.0.0,>=2.4.3 in c:\users\91996\anaconda3\lib\site-
packages (from spacy) (2.4.6)
Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in c:\users\91996\anaconda3\lib\site-
packages (from spacy) (4.64.1)
Requirement already satisfied: preshed<3.1.0,>=3.0.2 in c:\users\91996\anaconda3\lib\sit
e-packages (from spacy) (3.0.8)
Requirement already satisfied: jinja2 in c:\users\91996\anaconda3\lib\site-packages (fro
m spacy) (2.11.3)
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in c:\users\91996\anaconda3\lib
\site-packages (from spacy) (1.0.9)
Requirement already satisfied: setuptools in c:\users\91996\anaconda3\lib\site-packages
(from spacy) (63.4.1)
Requirement already satisfied: pydantic!=1.8,!=1.8.1,<1.11.0,>=1.7.4 in c:\users\91996\a
naconda3\lib\site-packages (from spacy) (1.10.5)
Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in c:\users\91996\anaconda3\lib\s
ite-packages (from spacy) (2.0.8)
Requirement already satisfied: packaging>=20.0 in c:\users\91996\anaconda3\lib\site-pack
ages (from spacy) (21.3)
Requirement already satisfied: cymem<2.1.0,>=2.0.2 in c:\users\91996\anaconda3\lib\site-
packages (from spacy) (2.0.7)
Requirement already satisfied: thinc<8.2.0,>=8.1.0 in c:\users\91996\anaconda3\lib\site-
packages (from spacy) (8.1.8)
Requirement already satisfied: numpy>=1.15.0 in c:\users\91996\anaconda3\lib\site-packag
es (from spacy) (1.21.5)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in c:\users\91996\anaconda3\lib
\site-packages (from packaging>=20.0->spacy) (3.0.9)
Requirement already satisfied: typing-extensions>=4.2.0 in c:\users\91996\anaconda3\lib
\site-packages (from pydantic!=1.8,!=1.8.1,<1.11.0,>=1.7.4->spacy) (4.3.0)
Requirement already satisfied: idna<4,>=2.5 in c:\users\91996\anaconda3\lib\site-package
s (from requests<3.0.0,>=2.13.0->spacy) (3.3)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\91996\anaconda3\lib\sit
e-packages (from requests<3.0.0,>=2.13.0->spacy) (1.26.11)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\91996\anaconda3\lib\site-p
ackages (from requests<3.0.0,>=2.13.0->spacy) (2022.9.14)
Requirement already satisfied: charset-normalizer<3,>=2 in c:\users\91996\anaconda3\lib
\site-packages (from requests<3.0.0,>=2.13.0->spacy) (2.0.4)
Requirement already satisfied: blis<0.8.0,>=0.7.8 in c:\users\91996\anaconda3\lib\site-p
ackages (from thinc<8.2.0,>=8.1.0->spacy) (0.7.9)
Requirement already satisfied: confection<1.0.0,>=0.0.1 in c:\users\91996\anaconda3\lib
\site-packages (from thinc<8.2.0,>=8.1.0->spacy) (0.0.4)
Requirement already satisfied: colorama in c:\users\91996\anaconda3\lib\site-packages (f
rom tqdm<5.0.0,>=4.38.0->spacy) (0.4.6)
```

```
Requirement already satisfied: click<9.0.0,>=7.1.1 in c:\users\91996\anaconda3\lib\site-
packages (from typer<0.8.0,>=0.3.0->spacy) (8.0.4)
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\91996\anaconda3\lib\site-pac
kages (from jinja2->spacy) (2.0.1)
Collecting en-core-web-sm==3.5.0
  Downloading https://github.com/explosion/spacy-models/releases/download/en_core_web_sm
-3.5.0/en_core_web_sm-3.5.0-py3-none-any.whl (12.8 MB)
     ---------------------------------------- 12.8/12.8 MB 2.2 MB/s eta 0:00:00
Requirement already satisfied: spacy<3.6.0,>=3.5.0 in c:\users\91996\anaconda3\lib\site-
packages (from en-core-web-sm==3.5.0) (3.5.0)
Requirement already satisfied: setuptools in c:\users\91996\anaconda3\lib\site-packages
(from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (63.4.1)
Requirement already satisfied: packaging>=20.0 in c:\users\91996\anaconda3\lib\site-pack
ages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (21.3)
Requirement already satisfied: langcodes<4.0.0,>=3.2.0 in c:\users\91996\anaconda3\lib\s
ite-packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (3.3.0)
Requirement already satisfied: wasabi<1.2.0,>=0.9.1 in c:\users\91996\anaconda3\lib\site
-packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (1.1.1)
Requirement already satisfied: jinja2 in c:\users\91996\anaconda3\lib\site-packages (fro
m spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (2.11.3)
Requirement already satisfied: requests<3.0.0,>=2.13.0 in c:\users\91996\anaconda3\lib\s
ite-packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (2.28.1)
Requirement already satisfied: smart-open<7.0.0,>=5.2.1 in c:\users\91996\anaconda3\lib
\site-packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (5.2.1)
Requirement already satisfied: cymem<2.1.0,>=2.0.2 in c:\users\91996\anaconda3\lib\site-
packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (2.0.7)
Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.11 in c:\users\91996\anaconda3\l
ib\site-packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (3.0.12)
Requirement already satisfied: srsly<3.0.0,>=2.4.3 in c:\users\91996\anaconda3\lib\site-
packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (2.4.6)
Requirement already satisfied: pathy>=0.10.0 in c:\users\91996\anaconda3\lib\site-packag
es (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (0.10.1)
Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in c:\users\91996\anaconda3\l
ib\site-packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (1.0.4)
Requirement already satisfied: pydantic!=1.8,!=1.8.1,<1.11.0,>=1.7.4 in c:\users\91996\a
naconda3\lib\site-packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (1.10.5)
Requirement already satisfied: preshed<3.1.0,>=3.0.2 in c:\users\91996\anaconda3\lib\sit
e-packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (3.0.8)
Requirement already satisfied: thinc<8.2.0,>=8.1.0 in c:\users\91996\anaconda3\lib\site-
packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (8.1.8)
Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in c:\users\91996\anaconda3\lib\s
ite-packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (2.0.8)
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in c:\users\91996\anaconda3\lib
\site-packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (1.0.9)
Requirement already satisfied: numpy>=1.15.0 in c:\users\91996\anaconda3\lib\site-packag
es (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (1.21.5)
Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in c:\users\91996\anaconda3\lib\site-
packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (4.64.1)
Requirement already satisfied: typer<0.8.0,>=0.3.0 in c:\users\91996\anaconda3\lib\site-
packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (0.7.0)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in c:\users\91996\anaconda3\lib
\site-packages (from packaging>=20.0->spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (3.0.
9)
Requirement already satisfied: typing-extensions>=4.2.0 in c:\users\91996\anaconda3\lib
\site-packages (from pydantic!=1.8,!=1.8.1,<1.11.0,>=1.7.4->spacy<3.6.0,>=3.5.0->en-core
-web-sm==3.5.0) (4.3.0)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\91996\anaconda3\lib\sit
e-packages (from requests<3.0.0,>=2.13.0->spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0)
(1.26.11)
Requirement already satisfied: charset-normalizer<3,>=2 in c:\users\91996\anaconda3\lib
\site-packages (from requests<3.0.0,>=2.13.0->spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.
0) (2.0.4)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\91996\anaconda3\lib\site-p
ackages (from requests<3.0.0,>=2.13.0->spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (202
2.9.14)
Requirement already satisfied: idna<4,>=2.5 in c:\users\91996\anaconda3\lib\site-package
```

```
s (from requests<3.0.0,>=2.13.0->spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (3.3)
Requirement already satisfied: blis<0.8.0,>=0.7.8 in c:\users\91996\anaconda3\lib\site-p
ackages (from thinc<8.2.0,>=8.1.0->spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (0.7.9)
Requirement already satisfied: confection<1.0.0,>=0.0.1 in c:\users\91996\anaconda3\lib
\site-packages (from thinc<8.2.0,>=8.1.0->spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0)
(0.0.4)
Requirement already satisfied: colorama in c:\users\91996\anaconda3\lib\site-packages (f
rom tqdm<5.0.0,>=4.38.0->spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (0.4.6)
Requirement already satisfied: click<9.0.0,>=7.1.1 in c:\users\91996\anaconda3\lib\site-
packages (from typer<0.8.0,>=0.3.0->spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (8.0.4)
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\91996\anaconda3\lib\site-pac
kages (from jinja2->spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (2.0.1)
Installing collected packages: en-core-web-sm
Successfully installed en-core-web-sm-3.5.0
[+] Download and installation successful
You can now load the package via spacy.load('en_core_web_sm')
```

In [10]:
```
import spacy
nlp = spacy.load('en_core_web_sm')
doc = nlp("data science has ai has good career scope ahead")
```

In [11]:
```
for token in doc:
    print(token.text)
```

```
data
science
has
ai
has
good
career
scope
ahead
```

In [12]:
```
for token in doc:
    print(token.pos_)
```

```
NOUN
NOUN
AUX
AUX
VERB
ADJ
NOUN
NOUN
ADV
```

In [13]:
```
for token in doc:
    print(token.text, token.pos_)
```

```
data NOUN
science NOUN
has AUX
ai AUX
has VERB
good ADJ
career NOUN
scope NOUN
ahead ADV
```

In [23]:
```
text = """There are broadly two types of extractive summarization tasks depending on wha

An example of a summarization problem is document summarization, which attempts to autom

Image collection summarization is another application example of automatic summarization
```

In [24]:
```
text
```

Out[24]:
```
'There are broadly two types of extractive summarization tasks depending on what the sum
marization program focuses on. The first is generic summarization, which focuses on obta
ining a generic summary or abstract of the collection (whether documents, or sets of ima
ges, or videos, news stories etc.). The second is query relevant summarization, sometime
s called query-based summarization, which summarizes objects specific to a query. Summar
ization systems are able to create both query relevant text summaries and generic machin
e-generated summaries depending on what the user needs.\n\nAn example of a summarization
problem is document summarization, which attempts to automatically produce an abstract f
rom a given document. Sometimes one might be interested in generating a summary from a s
ingle source document, while others can use multiple source documents (for example, a cl
uster of articles on the same topic). This problem is called multi-document summarizatio
n. A related application is summarizing news articles. Imagine a system, which automatic
ally pulls together news articles on a given topic (from the web), and concisely represe
nts the latest news as a summary.\n\nImage collection summarization is another applicati
on example of automatic summarization. It consists in selecting a representative set of
images from a larger set of images.[13] A summary in this context is useful to show the
most representative images of results in an image collection exploration system. Video s
ummarization is a related domain, where the system automatically creates a trailer of a
long video. This also has applications in consumer or personal videos, where one might w
ant to skip the boring or repetitive actions. Similarly, in surveillance videos, one wou
ld want to extract important and suspicious activity, while ignoring all the boring and
redundant frames captured.'
```

In [25]:
```
import spacy
from spacy.lang.en.stop_words import STOP_WORDS
from string import punctuation
```

In [26]:
```
stopwords = list(STOP_WORDS)
stopwords
```

Out[26]:
```
['many',
 'regarding',
 'my',
 'noone',
 'as',
 'either',
 'against',
 'ca',
 'i',
 'those',
 'within',
 'than',
 'they',
 'enough',
 '’re',
 'becomes',
 'therefore',
 'whereby',
 'been',
 'are',
 'us',
 'often',
 'down',
 'three',
 'each',
 'am',
 'full',
 'latter',
 'together',
 'a',
 'around',
 '‘s',
 'our',
 'bottom',
 'other',
 'we',
 'eleven',
 'from',
 'me',
 'beside',
 'several',
 'in',
 'mine',
 'six',
 'eight',
 'before',
 'whom',
 'cannot',
 'an',
 'were',
 'five',
 'might',
 'it',
 'empty',
 'something',
 'amongst',
 'yourself',
 'seemed',
 'thereby',
 'once',
 'wherever',
 'these',
 'would',
 "'m",
 'among',
 'this',
 'all',
 'could',
 'name',
 'thru',
 'doing',
 'ten',
 'third',
 'again',
 'whither',
 'under',
 'the',
 'no',
 'himself',
 'below',
 'whereafter',
 '‘d',
 'thereafter',
 'its',
 'neither',
 'both',
 'though',
 "'re",
 '’s',
 'whereas',
 'hereafter',
 'sometime',
 'after',
 'made',
 'wherein',
 'except',
 'amount',
 'indeed',
 'used',
 'where',
'is',
'onto',
'over',
'next',
"'s",
'already',
'fifteen',
'fifty',
'some',
'latterly',
'anyone',
'n’t',
'’m',
'when',
'serious',
'hence',
'nobody',
'somewhere',
'through',
'please',
'else',
"'ve",
'he',
'‘re',
'done',
'last',
'whole',
'thence',
'had',
'do',
'should',
'by',
'never',
'yourselves',
'that',
'becoming',
'has',
'may',
'off',
'until',
'twenty',
'she',
'anything',
'part',
'themselves',
'herein',
'even',
'to',
'see',
'besides',
'if',
'whose',
'due',
'itself',
'elsewhere',
'nine',
'for',
'also',
'less',
'still',
'own',
'just',
'‘m',
'throughout',
'one',
'at',
'there',
'moreover',
'top',
'then',
'few',
'rather',
'almost',
're',
'using',
'hereby',
'will',
'put',
'whether',
'but',
'well',
'behind',
'thus',
'seem',
'seems',
'somehow',
'become',
'n‘t',
'above',
'and',
'get',
'towards',
'while',
'you',
'hundred',
'can',
'others',
'beyond',
'across',
'ourselves',
'side',
'about',
'more',
'myself',
'former',
'of',
'via',
'always',
'your',
'really',
'most',
"'ll",
'became',
'sixty',
'whence',
'every',
'‘ve',
'up',
'have',
'nowhere',
'was',
'yours',
'hers',
'first',
'perhaps',
'him',
'nevertheless',
'with',
'does',
'out',
'give',
'meanwhile',
'say',
'formerly',
'whoever',
'therein',
"'d",
'between',
'everything',
'anyway',
'per',
'along',
'how',
'’ve',
'afterwards',
'show',
'back',
'such',
'forty',
'keep',
'hereupon',
'much',
'unless',
'because',
'whatever',
'although',
'what',
'otherwise',
'now',
'same',
'twelve',
'yet',
'quite',
'into',
'must',
'herself',
'here',
'however',
'‘ll',
'thereupon',
'on',
'since',
'so',
'call',
'everywhere',
'’ll',
'make',
'another',
'upon',
'four',
'none',
"n't",
'anyhow',
'go',
'seeming',
'further',
'who',
'any',
'be',
'beforehand',
'why',
'sometimes',
'her',
'too',
'them',
'least',
'only',
'being',
'their',
'toward',
'or',
'namely',
'take',
'his',
'alone',
'mostly',
'ever',
'anywhere',
'nor',
'nothing',
'move',
''d',
'various',
'did',
'not',
'someone',
'two',
'which',
'during',
'whenever',
'ours',
'very',
'without',
'whereupon',
'front',
'everyone']
```

In [27]:
```
len(stopwords)
```

Out[27]:
```
326
```

In [28]:
```
nlp = spacy.load('en_core_web_sm')
```

In [29]:
```
text
```

Out[29]:
```
'There are broadly two types of extractive summarization tasks depending on what the sum
marization program focuses on. The first is generic summarization, which focuses on obta
ining a generic summary or abstract of the collection (whether documents, or sets of ima
ges, or videos, news stories etc.). The second is query relevant summarization, sometime
s called query-based summarization, which summarizes objects specific to a query. Summar
ization systems are able to create both query relevant text summaries and generic machin
e-generated summaries depending on what the user needs.\n\nAn example of a summarization
problem is document summarization, which attempts to automatically produce an abstract f
rom a given document. Sometimes one might be interested in generating a summary from a s
ingle source document, while others can use multiple source documents (for example, a cl
uster of articles on the same topic). This problem is called multi-document summarizatio
n. A related application is summarizing news articles. Imagine a system, which automatic
ally pulls together news articles on a given topic (from the web), and concisely represe
nts the latest news as a summary.\n\nImage collection summarization is another applicati
on example of automatic summarization. It consists in selecting a representative set of
images from a larger set of images.[13] A summary in this context is useful to show the
most representative images of results in an image collection exploration system. Video s
ummarization is a related domain, where the system automatically creates a trailer of a
long video. This also has applications in consumer or personal videos, where one might w
ant to skip the boring or repetitive actions. Similarly, in surveillance videos, one wou
ld want to extract important and suspicious activity, while ignoring all the boring and
redundant frames captured.'
```

In [30]:
```
doc = nlp(text)
doc
```

Out[30]:
```
There are broadly two types of extractive summarization tasks depending on what the summ
arization program focuses on. The first is generic summarization, which focuses on obtai
```

```
ning a generic summary or abstract of the collection (whether documents, or sets of imag
es, or videos, news stories etc.). The second is query relevant summarization, sometimes
called query-based summarization, which summarizes objects specific to a query. Summariz
ation systems are able to create both query relevant text summaries and generic machine-
generated summaries depending on what the user needs.

An example of a summarization problem is document summarization, which attempts to autom
atically produce an abstract from a given document. Sometimes one might be interested in
generating a summary from a single source document, while others can use multiple source
documents (for example, a cluster of articles on the same topic). This problem is called
multi-document summarization. A related application is summarizing news articles. Imagin
e a system, which automatically pulls together news articles on a given topic (from the
web), and concisely represents the latest news as a summary.

Image collection summarization is another application example of automatic summarizatio
n. It consists in selecting a representative set of images from a larger set of images.
[13] A summary in this context is useful to show the most representative images of resul
ts in an image collection exploration system. Video summarization is a related domain, w
here the system automatically creates a trailer of a long video. This also has applicati
ons in consumer or personal videos, where one might want to skip the boring or repetitiv
e actions. Similarly, in surveillance videos, one would want to extract important and su
spicious activity, while ignoring all the boring and redundant frames captured.
```

In [33]:
```python
tokens = [token.text for token in doc]
print(tokens)
```

```
['There', 'are', 'broadly', 'two', 'types', 'of', 'extractive', 'summarization', 'task
s', 'depending', 'on', 'what', 'the', 'summarization', 'program', 'focuses', 'on', '.',
'The', 'first', 'is', 'generic', 'summarization', ',', 'which', 'focuses', 'on', 'obtain
ing', 'a', 'generic', 'summary', 'or', 'abstract', 'of', 'the', 'collection', '(', 'whet
her', 'documents', ',', 'or', 'sets', 'of', 'images', ',', 'or', 'videos', ',', 'news',
'stories', 'etc', '.', ')', '.', 'The', 'second', 'is', 'query', 'relevant', 'summarizat
ion', ',', 'sometimes', 'called', 'query', '-', 'based', 'summarization', ',', 'which',
'summarizes', 'objects', 'specific', 'to', 'a', 'query', '.', 'Summarization', 'system
s', 'are', 'able', 'to', 'create', 'both', 'query', 'relevant', 'text', 'summaries', 'an
d', 'generic', 'machine', '-', 'generated', 'summaries', 'depending', 'on', 'what', 'th
e', 'user', 'needs', '.', '\n\n', 'An', 'example', 'of', 'a', 'summarization', 'proble
m', 'is', 'document', 'summarization', ',', 'which', 'attempts', 'to', 'automatically',
'produce', 'an', 'abstract', 'from', 'a', 'given', 'document', '.', 'Sometimes', 'one',
'might', 'be', 'interested', 'in', 'generating', 'a', 'summary', 'from', 'a', 'single',
'source', 'document', ',', 'while', 'others', 'can', 'use', 'multiple', 'source', 'docum
ents', '(', 'for', 'example', ',', 'a', 'cluster', 'of', 'articles', 'on', 'the', 'sam
e', 'topic', ')', '.', 'This', 'problem', 'is', 'called', 'multi', '-', 'document', 'sum
marization', '.', 'A', 'related', 'application', 'is', 'summarizing', 'news', 'article
s', '.', 'Imagine', 'a', 'system', ',', 'which', 'automatically', 'pulls', 'together',
'news', 'articles', 'on', 'a', 'given', 'topic', '(', 'from', 'the', 'web', ')', ',', 'a
nd', 'concisely', 'represents', 'the', 'latest', 'news', 'as', 'a', 'summary', '.', '\n
\n', 'Image', 'collection', 'summarization', 'is', 'another', 'application', 'example',
'of', 'automatic', 'summarization', '.', 'It', 'consists', 'in', 'selecting', 'a', 'repr
esentative', 'set', 'of', 'images', 'from', 'a', 'larger', 'set', 'of', 'images.[13',
']', 'A', 'summary', 'in', 'this', 'context', 'is', 'useful', 'to', 'show', 'the', 'mos
t', 'representative', 'images', 'of', 'results', 'in', 'an', 'image', 'collection', 'exp
loration', 'system', '.', 'Video', 'summarization', 'is', 'a', 'related', 'domain', ',',
'where', 'the', 'system', 'automatically', 'creates', 'a', 'trailer', 'of', 'a', 'long',
'video', '.', 'This', 'also', 'has', 'applications', 'in', 'consumer', 'or', 'personal',
'videos', ',', 'where', 'one', 'might', 'want', 'to', 'skip', 'the', 'boring', 'or', 're
petitive', 'actions', '.', 'Similarly', ',', 'in', 'surveillance', 'videos', ',', 'one',
'would', 'want', 'to', 'extract', 'important', 'and', 'suspicious', 'activity', ',', 'wh
ile', 'ignoring', 'all', 'the', 'boring', 'and', 'redundant', 'frames', 'captured', '.']
```

In [34]:
```python
len(tokens)
```

Out[34]:
```
323
```

In [49]:
```python
word_frequencies = {}
```

```
for word in doc:
    if word.text.lower() not in stopwords:
        if word.text.lower() not in punctuation:
            if word.text not in word_frequencies.keys():
                word_frequencies[word.text]=1
            else:
                word_frequencies [word.text] += 1
```

```
In [50]: word_frequencies
```

```
Out[50]: {'broadly': 1,
 'types': 1,
 'extractive': 1,
 'summarization': 11,
 'tasks': 1,
 'depending': 2,
 'program': 1,
 'focuses': 2,
 'generic': 3,
 'obtaining': 1,
 'summary': 4,
 'abstract': 2,
 'collection': 3,
 'documents': 2,
 'sets': 1,
 'images': 3,
 'videos': 3,
 'news': 4,
 'stories': 1,
 'etc': 1,
 'second': 1,
 'query': 4,
 'relevant': 2,
 'called': 2,
 'based': 1,
 'summarizes': 1,
 'objects': 1,
 'specific': 1,
 'Summarization': 1,
 'systems': 1,
 'able': 1,
 'create': 1,
 'text': 1,
 'summaries': 2,
 'machine': 1,
 'generated': 1,
 'user': 1,
 'needs': 1,
 '\n\n': 2,
 'example': 3,
 'problem': 2,
 'document': 4,
 'attempts': 1,
 'automatically': 3,
 'produce': 1,
 'given': 2,
 'interested': 1,
 'generating': 1,
 'single': 1,
 'source': 2,
 'use': 1,
 'multiple': 1,
 'cluster': 1,
 'articles': 3,
 'topic': 2,
 'multi': 1,
 'related': 2,
 'application': 2,
 'summarizing': 1,
 'Imagine': 1,
 'system': 3,
 'pulls': 1,
 'web': 1,
 'concisely': 1,
 'represents': 1,
 'latest': 1,
 'Image': 1,
 'automatic': 1,
 'consists': 1,
 'selecting': 1,
 'representative': 2,
 'set': 2,
 'larger': 1,
 'images.[13': 1,
 'context': 1,
 'useful': 1,
 'results': 1,
 'image': 1,
 'exploration': 1,
 'Video': 1,
 'domain': 1,
 'creates': 1,
 'trailer': 1,
 'long': 1,
 'video': 1,
 'applications': 1,
 'consumer': 1,
 'personal': 1,
 'want': 2,
 'skip': 1,
 'boring': 2,
 'repetitive': 1,
 'actions': 1,
 'Similarly': 1,
 'surveillance': 1,
 'extract': 1,
 'important': 1,
 'suspicious': 1,
 'activity': 1,
 'ignoring': 1,
 'redundant': 1,
 'frames': 1,
 'captured': 1}
```

In [51]:
```
len(word_frequencies)
```

Out[51]:
```
103
```

In [52]:
```
max_frequencies = max(word_frequencies.values())
max_frequencies
```

Out[52]:
```
11
```

In [53]:
```
for word in word_frequencies.keys():
    word_frequencies[word] =  word_frequencies[word]/max_frequencies
```

In [54]:
```
word_frequencies
```

```
Out[54]: {'broadly': 0.09090909090909091,
          'types': 0.09090909090909091,
          'extractive': 0.09090909090909091,
          'summarization': 1.0,
          'tasks': 0.09090909090909091,
          'depending': 0.18181818181818182,
          'program': 0.09090909090909091,
          'focuses': 0.18181818181818182,
          'generic': 0.2727272727272727,
          'obtaining': 0.09090909090909091,
          'summary': 0.36363636363636365,
          'abstract': 0.18181818181818182,
          'collection': 0.2727272727272727,
          'documents': 0.18181818181818182,
          'sets': 0.09090909090909091,
          'images': 0.2727272727272727,
          'videos': 0.2727272727272727,
          'news': 0.36363636363636365,
          'stories': 0.09090909090909091,
          'etc': 0.09090909090909091,
          'second': 0.09090909090909091,
          'query': 0.36363636363636365,
          'relevant': 0.18181818181818182,
          'called': 0.18181818181818182,
          'based': 0.09090909090909091,
          'summarizes': 0.09090909090909091,
          'objects': 0.09090909090909091,
          'specific': 0.09090909090909091,
          'Summarization': 0.09090909090909091,
          'systems': 0.09090909090909091,
          'able': 0.09090909090909091,
          'create': 0.09090909090909091,
          'text': 0.09090909090909091,
          'summaries': 0.18181818181818182,
          'machine': 0.09090909090909091,
          'generated': 0.09090909090909091,
          'user': 0.09090909090909091,
          'needs': 0.09090909090909091,
          '\n\n': 0.18181818181818182,
          'example': 0.2727272727272727,
          'problem': 0.18181818181818182,
          'document': 0.36363636363636365,
          'attempts': 0.09090909090909091,
          'automatically': 0.2727272727272727,
          'produce': 0.09090909090909091,
          'given': 0.18181818181818182,
          'interested': 0.09090909090909091,
          'generating': 0.09090909090909091,
          'single': 0.09090909090909091,
          'source': 0.18181818181818182,
          'use': 0.09090909090909091,
          'multiple': 0.09090909090909091,
          'cluster': 0.09090909090909091,
          'articles': 0.2727272727272727,
          'topic': 0.18181818181818182,
          'multi': 0.09090909090909091,
          'related': 0.18181818181818182,
          'application': 0.18181818181818182,
          'summarizing': 0.09090909090909091,
          'Imagine': 0.09090909090909091,
          'system': 0.2727272727272727,
          'pulls': 0.09090909090909091,
          'web': 0.09090909090909091,
          'concisely': 0.09090909090909091,
          'represents': 0.09090909090909091,
          'latest': 0.09090909090909091,
          'Image': 0.09090909090909091,
          'automatic': 0.09090909090909091,
          'consists': 0.09090909090909091,
          'selecting': 0.09090909090909091,
          'representative': 0.18181818181818182,
          'set': 0.18181818181818182,
          'larger': 0.09090909090909091,
          'images.[13': 0.09090909090909091,
          'context': 0.09090909090909091,
          'useful': 0.09090909090909091,
          'results': 0.09090909090909091,
          'image': 0.09090909090909091,
          'exploration': 0.09090909090909091,
          'Video': 0.09090909090909091,
          'domain': 0.09090909090909091,
          'creates': 0.09090909090909091,
          'trailer': 0.09090909090909091,
          'long': 0.09090909090909091,
          'video': 0.09090909090909091,
          'applications': 0.09090909090909091,
          'consumer': 0.09090909090909091,
          'personal': 0.09090909090909091,
          'want': 0.18181818181818182,
          'skip': 0.09090909090909091,
          'boring': 0.18181818181818182,
          'repetitive': 0.09090909090909091,
          'actions': 0.09090909090909091,
          'Similarly': 0.09090909090909091,
          'surveillance': 0.09090909090909091,
          'extract': 0.09090909090909091,
          'important': 0.09090909090909091,
          'suspicious': 0.09090909090909091,
          'activity': 0.09090909090909091,
          'ignoring': 0.09090909090909091,
          'redundant': 0.09090909090909091,
          'frames': 0.09090909090909091,
          'captured': 0.09090909090909091}
```

In [57]:
```python
sentence_tokens = [sent for sent in doc.sents]
sentence_tokens
```

Out[57]:
```
[There are broadly two types of extractive summarization tasks depending on what the sum
marization program focuses on.,
 The first is generic summarization, which focuses on obtaining a generic summary or abs
tract of the collection (whether documents, or sets of images, or videos, news stories e
tc.).,
 The second is query relevant summarization, sometimes called query-based summarization,
which summarizes objects specific to a query.,
 Summarization systems are able to create both query relevant text summaries and generic
machine-generated summaries depending on what the user needs.
 ,
 An example of a summarization problem is document summarization, which attempts to auto
matically produce an abstract from a given document.,
 Sometimes one might be interested in generating a summary from a single source documen
t, while others can use multiple source documents (for example, a cluster of articles on
the same topic).,
 This problem is called multi-document summarization.,
 A related application is summarizing news articles.,
 Imagine a system, which automatically pulls together news articles on a given topic (fr
om the web), and concisely represents the latest news as a summary.
 ,
 Image collection summarization is another application example of automatic summarizatio
n.,
 It consists in selecting a representative set of images from a larger set of images.[1
3] A summary in this context is useful to show the most representative images of results
in an image collection exploration system.,
 Video summarization is a related domain, where the system automatically creates a trail
er of a long video.,
 This also has applications in consumer or personal videos, where one might want to skip
the boring or repetitive actions.,
 Similarly, in surveillance videos, one would want to extract important and suspicious a
ctivity, while ignoring all the boring and redundant frames captured.]
```

In [58]:
```
len(sentence_tokens)
```

Out[58]:
```
14
```

In [61]:
```
sentence_scores = {}

for sent in sentence_tokens:
    for word in sent:
        if word.text.lower() in word_frequencies.keys():
            if sent not in sentence_scores.keys():
                sentence_scores[sent] = word_frequencies [word.text.lower()]
            else:
                sentence_scores [sent] += word_frequencies [word.text.lower()]
```

In [62]:
```
sentence_scores
```

Out[62]:
```
{There are broadly two types of extractive summarization tasks depending on what the sum
marization program focuses on.: 2.8181818181818183,
 The first is generic summarization, which focuses on obtaining a generic summary or abs
tract of the collection (whether documents, or sets of images, or videos, news stories e
tc.).: 3.9999999999999987,
 The second is query relevant summarization, sometimes called query-based summarization,
which summarizes objects specific to a query.: 3.909090909090909,
 Summarization systems are able to create both query relevant text summaries and generic
machine-generated summaries depending on what the user needs.
 : 3.2727272727272716,
 An example of a summarization problem is document summarization, which attempts to auto
matically produce an abstract from a given document.: 3.9999999999999996,
 Sometimes one might be interested in generating a summary from a single source documen
t, while others can use multiple source documents (for example, a cluster of articles on
the same topic).: 2.545454545454545,
 This problem is called multi-document summarization.: 1.8181818181818183,
 A related application is summarizing news articles.: 1.0909090909090908,
 Imagine a system, which automatically pulls together news articles on a given topic (fr
om the web), and concisely represents the latest news as a summary.
 : 2.9090909090909087,
 Image collection summarization is another application example of automatic summarizatio
n.: 2.909090909090909,
 It consists in selecting a representative set of images from a larger set of images.[1
3] A summary in this context is useful to show the most representative images of results
in an image collection exploration system.: 2.999999999999999,
 Video summarization is a related domain, where the system automatically creates a trail
er of a long video.: 2.2727272727272725,
 This also has applications in consumer or personal videos, where one might want to skip
the boring or repetitive actions.: 1.1818181818181817,
 Similarly, in surveillance videos, one would want to extract important and suspicious a
ctivity, while ignoring all the boring and redundant frames captured.: 1.454545454545454
4}
```

In [64]:
```
from heapq import nlargest
```

In [65]:
```
select_length = int(len(sentence_tokens)*0.3)
select_length
```

Out[65]:
```
4
```

In [66]:
```
summary = nlargest(select_length,sentence_scores, key = sentence_scores.get)
```

```
In [67]: summary
```

```
Out[67]: [An example of a summarization problem is document summarization, which attempts to auto
matically produce an abstract from a given document.,
 The first is generic summarization, which focuses on obtaining a generic summary or abs
tract of the collection (whether documents, or sets of images, or videos, news stories e
tc.).,
 The second is query relevant summarization, sometimes called query-based summarization,
which summarizes objects specific to a query.,
 Summarization systems are able to create both query relevant text summaries and generic
machine-generated summaries depending on what the user needs.
 ]
```

```
In [68]: sentence_scores
```

```
Out[68]: {There are broadly two types of extractive summarization tasks depending on what the sum
marization program focuses on.: 2.8181818181818,
 The first is generic summarization, which focuses on obtaining a generic summary or abs
tract of the collection (whether documents, or sets of images, or videos, news stories e
tc.).: 3.9999999999999987,
 The second is query relevant summarization, sometimes called query-based summarization,
which summarizes objects specific to a query.: 3.909090909090909,
 Summarization systems are able to create both query relevant text summaries and generic
machine-generated summaries depending on what the user needs.
 : 3.2727272727272716,
 An example of a summarization problem is document summarization, which attempts to auto
matically produce an abstract from a given document.: 3.9999999999999996,
 Sometimes one might be interested in generating a summary from a single source documen
t, while others can use multiple source documents (for example, a cluster of articles on
the same topic).: 2.545454545454545,
 This problem is called multi-document summarization.: 1.8181818181818183,
 A related application is summarizing news articles.: 1.0909090909090908,
 Imagine a system, which automatically pulls together news articles on a given topic (fr
om the web), and concisely represents the latest news as a summary.
 : 2.9090909090909087,
 Image collection summarization is another application example of automatic summarizatio
n.: 2.909090909090909,
 It consists in selecting a representative set of images from a larger set of images.[1
3] A summary in this context is useful to show the most representative images of results
in an image collection exploration system.: 2.999999999999999,
 Video summarization is a related domain, where the system automatically creates a trail
er of a long video.: 2.2727272727272725,
 This also has applications in consumer or personal videos, where one might want to skip
the boring or repetitive actions.: 1.1818181818181817,
 Similarly, in surveillance videos, one would want to extract important and suspicious a
ctivity, while ignoring all the boring and redundant frames captured.: 1.454545454545454
4}
```

```
In [70]: final_summary = [word.text for word in summary]
         final_summary
```

```
Out[70]: ['An example of a summarization problem is document summarization, which attempts to aut
omatically produce an abstract from a given document.',
 'The first is generic summarization, which focuses on obtaining a generic summary or ab
stract of the collection (whether documents, or sets of images, or videos, news stories
etc.).',
 'The second is query relevant summarization, sometimes called query-based summarizatio
n, which summarizes objects specific to a query.',
 'Summarization systems are able to create both query relevant text summaries and generi
c machine-generated summaries depending on what the user needs.\n\n']
```

```
In [71]: print(summary)
```

```
[An example of a summarization problem is document summarization, which attempts to auto
matically produce an abstract from a given document., The first is generic summarizatio
n, which focuses on obtaining a generic summary or abstract of the collection (whether d
ocuments, or sets of images, or videos, news stories etc.)., The second is query relevan
t summarization, sometimes called query-based summarization, which summarizes objects sp
ecific to a query., Summarization systems are able to create both query relevant text su
mmaries and generic machine-generated summaries depending on what the user needs.

]
```

```
In [ ]:
```