p7

April 6, 2024

[1]:
```
# importing libs
!pip install nltk
!pip install bs4
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages
(3.8.1)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages
(from nltk) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages
(from nltk) (1.3.2)
Requirement already satisfied: regex>=2021.8.3 in
/usr/local/lib/python3.10/dist-packages (from nltk) (2023.12.25)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages
(from nltk) (4.66.2)
Collecting bs4
  Downloading bs4-0.0.2-py2.py3-none-any.whl (1.2 kB)
Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.10/dist-
packages (from bs4) (4.12.3)
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.10/dist-
packages (from beautifulsoup4->bs4) (2.5)
Installing collected packages: bs4
Successfully installed bs4-0.0.2
```

[2]:
```
import nltk
nltk.download('stopwords')
nltk.download('punkt')
nltk.download('wordnet')
nltk.download('averaged_perceptron_tagger')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data…
[nltk_data]   Unzipping corpora/stopwords.zip.
[nltk_data] Downloading package punkt to /root/nltk_data…
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package wordnet to /root/nltk_data…
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data…
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger.zip.
```

```
[2]: True
```

```
[4]: para = "Rajgad (literal meaning Ruling Fort) is a hill fort situated in the␣
     ↪Pune district of Maharashtra, India. Formerly known as Murumdev, the fort␣
     ↪was the capital of the Maratha Empire under the rule of Chatrapati Shivaji␣
     ↪Maharaj for almost 26 years, after which the capital was moved to the Raigad␣
     ↪Fort.[1] Treasures discovered from an adjacent fort called Torna were used␣
     ↪to completely build and fortify the Rajgad Fort. "
     print(para)
```

```
Rajgad (literal meaning Ruling Fort) is a hill fort situated in the Pune
district of Maharashtra, India. Formerly known as Murumdev, the fort was the
capital of the Maratha Empire under the rule of Chatrapati Shivaji Maharaj for
almost 26 years, after which the capital was moved to the Raigad Fort.[1]
Treasures discovered from an adjacent fort called Torna were used to completely
build and fortify the Rajgad Fort.
```

```
[6]: para.split()
```

```
[6]: ['Rajgad',
      '(literal',
      'meaning',
      'Ruling',
      'Fort)',
      'is',
      'a',
      'hill',
      'fort',
      'situated',
      'in',
      'the',
      'Pune',
      'district',
      'of',
      'Maharashtra,',
      'India.',
      'Formerly',
      'known',
      'as',
      'Murumdev,',
      'the',
      'fort',
      'was',
      'the',
      'capital',
      'of',
      'the',
 'Maratha',
 'Empire',
 'under',
 'the',
 'rule',
 'of',
 'Chatrapati',
 'Shivaji',
 'Maharaj',
 'for',
 'almost',
 '26',
 'years,',
 'after',
 'which',
 'the',
 'capital',
 'was',
 'moved',
 'to',
 'the',
 'Raigad',
 'Fort.[1]',
 'Treasures',
 'discovered',
 'from',
 'an',
 'adjacent',
 'fort',
 'called',
 'Torna',
 'were',
 'used',
 'to',
 'completely',
 'build',
 'and',
 'fortify',
 'the',
 'Rajgad',
 'Fort.']
```

```
[8]: from nltk.corpus import stopwords
     swords=stopwords.words("english")
     print(swords)
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're",
"you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he',
'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it', "it's",
'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what',
'which', 'who', 'whom', 'this', 'that', "that'll", 'these', 'those', 'am', 'is',
'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having',
'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or',
'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about',
'against', 'between', 'into', 'through', 'during', 'before', 'after', 'above',
'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under',
'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why',
'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some',
'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very',
's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", 'now',
'd', 'll', 'm', 'o', 're', 've', 'y', 'ain', 'aren', "aren't", 'couldn',
"couldn't", 'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't", 'hasn',
"hasn't", 'haven', "haven't", 'isn', "isn't", 'ma', 'mightn', "mightn't",
'mustn', "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn',
"shouldn't", 'wasn', "wasn't", 'weren', "weren't", 'won', "won't", 'wouldn',
"wouldn't"]
```

[11]:
```
from nltk.tokenize import sent_tokenize
from nltk.tokenize import word_tokenize
sent=sent_tokenize(para)
print(sent[2])
```

```
[1] Treasures discovered from an adjacent fort called Torna were used to
completely build and fortify the Rajgad Fort.
```

[12]:
```
words=word_tokenize(para)
print(words)
```

```
['Rajgad', '(', 'literal', 'meaning', 'Ruling', 'Fort', ')', 'is', 'a', 'hill',
'fort', 'situated', 'in', 'the', 'Pune', 'district', 'of', 'Maharashtra', ',',
'India', '.', 'Formerly', 'known', 'as', 'Murumdev', ',', 'the', 'fort', 'was',
'the', 'capital', 'of', 'the', 'Maratha', 'Empire', 'under', 'the', 'rule',
'of', 'Chatrapati', 'Shivaji', 'Maharaj', 'for', 'almost', '26', 'years', ',',
'after', 'which', 'the', 'capital', 'was', 'moved', 'to', 'the', 'Raigad',
'Fort', '.', '[', '1', ']', 'Treasures', 'discovered', 'from', 'an', 'adjacent',
'fort', 'called', 'Torna', 'were', 'used', 'to', 'completely', 'build', 'and',
'fortify', 'the', 'Rajgad', 'Fort', '.']
```

[13]:
```
x=[word for word in words if word not in swords]
print(x)
```

```
['Rajgad', '(', 'literal', 'meaning', 'Ruling', 'Fort', ')', 'hill', 'fort',
'situated', 'Pune', 'district', 'Maharashtra', ',', 'India', '.', 'Formerly',
'known', 'Murumdev', ',', 'fort', 'capital', 'Maratha', 'Empire', 'rule',
'Chatrapati', 'Shivaji', 'Maharaj', 'almost', '26', 'years', ',', 'capital',
'moved', 'Raigad', 'Fort', '.', '[', '1', ']', 'Treasures', 'discovered',
'adjacent', 'fort', 'called', 'Torna', 'used', 'completely', 'build', 'fortify',
'Rajgad', 'Fort', '.']
```

[14]:
```
x=[word for word in words if word.lower() not in swords]
print(x)
```

```
['Rajgad', '(', 'literal', 'meaning', 'Ruling', 'Fort', ')', 'hill', 'fort',
'situated', 'Pune', 'district', 'Maharashtra', ',', 'India', '.', 'Formerly',
'known', 'Murumdev', ',', 'fort', 'capital', 'Maratha', 'Empire', 'rule',
'Chatrapati', 'Shivaji', 'Maharaj', 'almost', '26', 'years', ',', 'capital',
'moved', 'Raigad', 'Fort', '.', '[', '1', ']', 'Treasures', 'discovered',
'adjacent', 'fort', 'called', 'Torna', 'used', 'completely', 'build', 'fortify',
'Rajgad', 'Fort', '.']
```

[15]:
```
from nltk.stem import PorterStemmer
ps=PorterStemmer()
ps.stem('working')
y=[ps.stem(word) for word in x]
print(y)
```

```
['rajgad', '(', 'liter', 'mean', 'rule', 'fort', ')', 'hill', 'fort', 'situat',
'pune', 'district', 'maharashtra', ',', 'india', '.', 'formerli', 'known',
'murumdev', ',', 'fort', 'capit', 'maratha', 'empir', 'rule', 'chatrapati',
'shivaji', 'maharaj', 'almost', '26', 'year', ',', 'capit', 'move', 'raigad',
'fort', '.', '[', '1', ']', 'treasur', 'discov', 'adjac', 'fort', 'call',
'torna', 'use', 'complet', 'build', 'fortifi', 'rajgad', 'fort', '.']
```

[16]:
```
from nltk.stem import WordNetLemmatizer
wnl=WordNetLemmatizer()
wnl. lemmatize('working',pos='v')
```

[16]:
```
'work'
```

[18]:
```
print(ps.stem('went'))
print (wnl.lemmatize('went',pos='v'))
```

```
went
go
```

[19]:
```
z=[wnl.lemmatize(word,pos='v') for word in x]
print(z)
```

```
['Rajgad', '(', 'literal', 'mean', 'Ruling', 'Fort', ')', 'hill', 'fort',
'situate', 'Pune', 'district', 'Maharashtra', ',', 'India', '.', 'Formerly',
'know', 'Murumdev', ',', 'fort', 'capital', 'Maratha', 'Empire', 'rule',
'Chatrapati', 'Shivaji', 'Maharaj', 'almost', '26', 'years', ',', 'capital',
'move', 'Raigad', 'Fort', '.', '[', '1', ']', 'Treasures', 'discover',
'adjacent', 'fort', 'call', 'Torna', 'use', 'completely', 'build', 'fortify',
'Rajgad', 'Fort', '.']
```

[20]:
```
import string
string.punctuation
```

[20]:
```
'!"#$%&\'()*+,-./:;<=>?@[\\]^_`{|}~'
```

[21]:
```
t=[word for word in words if word not in string.punctuation]
print(t)
```

```
['Rajgad', 'literal', 'meaning', 'Ruling', 'Fort', 'is', 'a', 'hill', 'fort',
'situated', 'in', 'the', 'Pune', 'district', 'of', 'Maharashtra', 'India',
'Formerly', 'known', 'as', 'Murumdev', 'the', 'fort', 'was', 'the', 'capital',
'of', 'the', 'Maratha', 'Empire', 'under', 'the', 'rule', 'of', 'Chatrapati',
'Shivaji', 'Maharaj', 'for', 'almost', '26', 'years', 'after', 'which', 'the',
'capital', 'was', 'moved', 'to', 'the', 'Raigad', 'Fort', '1', 'Treasures',
'discovered', 'from', 'an', 'adjacent', 'fort', 'called', 'Torna', 'were',
'used', 'to', 'completely', 'build', 'and', 'fortify', 'the', 'Rajgad', 'Fort']
```

[22]:
```
from nltk import pos_tag
print(pos_tag(t))
```

```
[('Rajgad', 'NNP'), ('literal', 'JJ'), ('meaning', 'NN'), ('Ruling', 'NNP'),
('Fort', 'NNP'), ('is', 'VBZ'), ('a', 'DT'), ('hill', 'NN'), ('fort', 'NN'),
('situated', 'VBN'), ('in', 'IN'), ('the', 'DT'), ('Pune', 'NNP'), ('district',
'NN'), ('of', 'IN'), ('Maharashtra', 'NNP'), ('India', 'NNP'), ('Formerly',
'RB'), ('known', 'VBN'), ('as', 'IN'), ('Murumdev', 'NNP'), ('the', 'DT'),
('fort', 'NN'), ('was', 'VBD'), ('the', 'DT'), ('capital', 'NN'), ('of', 'IN'),
('the', 'DT'), ('Maratha', 'NNP'), ('Empire', 'NNP'), ('under', 'IN'), ('the',
'DT'), ('rule', 'NN'), ('of', 'IN'), ('Chatrapati', 'NNP'), ('Shivaji', 'NNP'),
('Maharaj', 'NNP'), ('for', 'IN'), ('almost', 'RB'), ('26', 'CD'), ('years',
'NNS'), ('after', 'IN'), ('which', 'WDT'), ('the', 'DT'), ('capital', 'NN'),
('was', 'VBD'), ('moved', 'VBN'), ('to', 'TO'), ('the', 'DT'), ('Raigad',
'NNP'), ('Fort', 'NNP'), ('1', 'CD'), ('Treasures', 'NNS'), ('discovered',
'VBN'), ('from', 'IN'), ('an', 'DT'), ('adjacent', 'JJ'), ('fort', 'NN'),
('called', 'VBN'), ('Torna', 'NNP'), ('were', 'VBD'), ('used', 'VBN'), ('to',
'TO'), ('completely', 'RB'), ('build', 'VB'), ('and', 'CC'), ('fortify', 'VB'),
('the', 'DT'), ('Rajgad', 'NNP'), ('Fort', 'NNP')]
```

[23]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf = TfidfVectorizer()
v=tfidf.fit_transform(t)
print(v.shape)
```

```
(70, 50)
```

[24]:
```
import pandas as pd
pd.DataFrame(v)
```

[24]:
```
                  0
0     (0, 35)\t1.0
1     (0, 25)\t1.0
2     (0, 29)\t1.0
3     (0, 37)\t1.0
4     (0, 17)\t1.0
..              …
65     (0, 5)\t1.0
66    (0, 18)\t1.0
67    (0, 40)\t1.0
68    (0, 35)\t1.0
69    (0, 17)\t1.0

[70 rows x 1 columns]
```