Experiment No-8

Aim-Implement Text Similarity Recognizer for the chosen text documents.

Theory-

Document similarity, as the name suggests, determines how similar are the two given documents. By “documents”, we mean a collection of strings. For example, an essay or a .txt file. Many organizations use this principle of document similarity to check plagiarism. It is also used by many exams conducting institutions to check if a student cheated from the other. Therefore, it is very important as well as interesting to know how all of this works.Document similarity is calculated by calculating document distance. Document distance is a concept where words(documents) are treated as vectors and is calculated as the angle between two given document vectors. Document vectors are the frequency of occurrences of words in a given document

Example-

we are given two documents D1 and D2 as:
D1: “This is a geek”

D2: “This was a geek thing”

The similar words in both these documents then become:

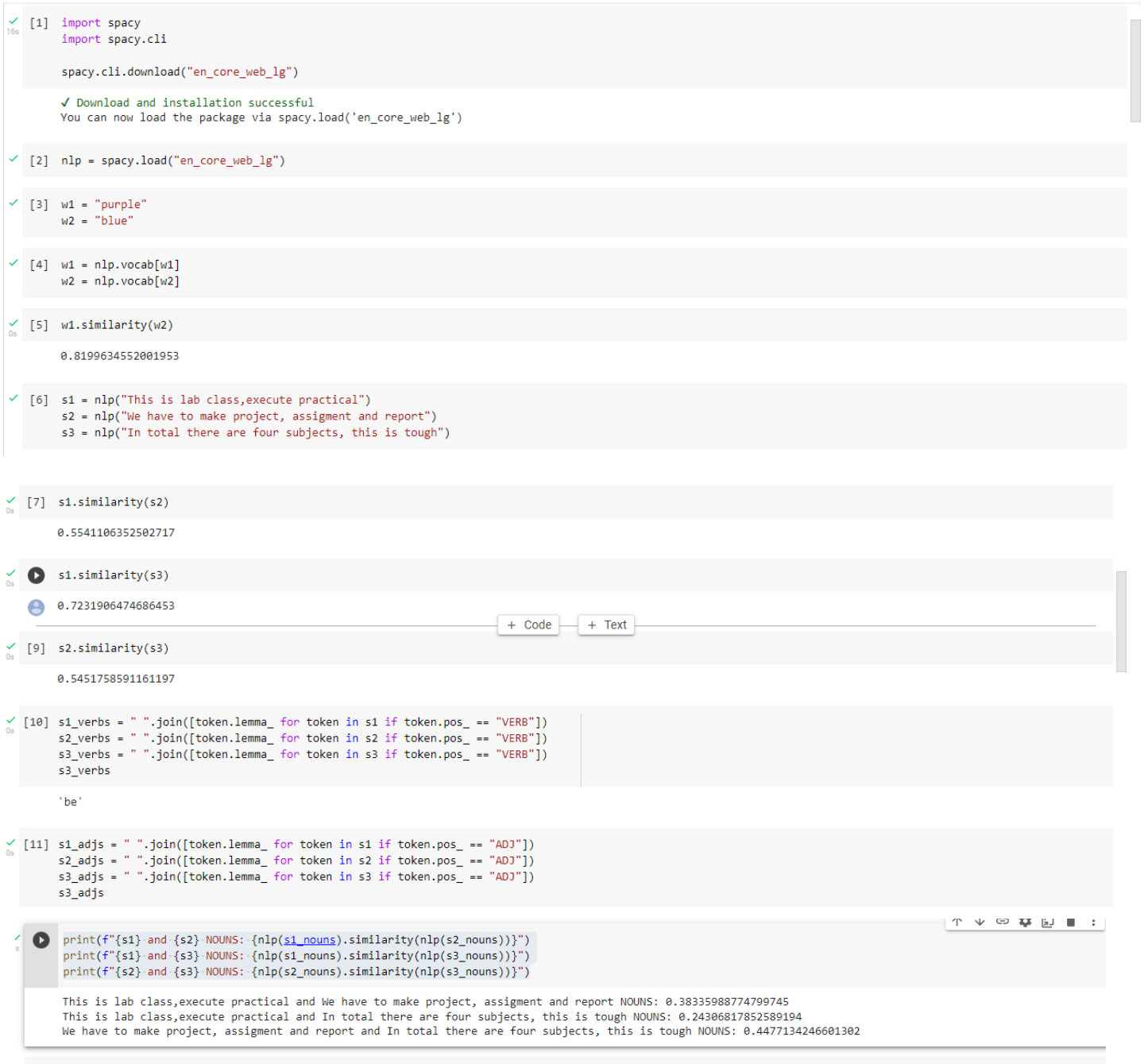
"This a geek"

Code -

```
import spacy
import spacy.cli
spacy.cli.download("en_core_web_lg")
nlp = spacy.load("en_core_web_lg")
w1 = "purple"
w2 = "blue"
w1 = nlp.vocab[w1]
w2 = nlp.vocab[w2]
w1.similarity(w2)
s1 = nlp("This is lab class,execute practical")
s2 = nlp("We have to make project, assigment and report")
s3 = nlp("In total there are four subjects, this is tough")
s1.similarity(s2)
s1.similarity(s3)
s2.similarity(s3)
s1_verbs = " ".join([token.lemma_ for token in s1 if token.pos_ == "VERB"])
s2_verbs = " ".join([token.lemma_ for token in s2 if token.pos_ == "VERB"])
s3_verbs = " ".join([token.lemma_ for token in s3 if token.pos_ == "VERB"])
s3_verbs
s1_adjs = " ".join([token.lemma_ for token in s1 if token.pos_ == "ADJ"])
s2_adjs = " ".join([token.lemma_ for token in s2 if token.pos_ == "ADJ"])
s3_adjs = " ".join([token.lemma_ for token in s3 if token.pos_ == "ADJ"])
s3_adjs
s1_nouns = " ".join([token.lemma_ for token in s1 if token.pos_ == "NOUN"])
s2_nouns = " ".join([token.lemma_ for token in s2 if token.pos_ == "NOUN"])
s3_nouns = " ".join([token.lemma_ for token in s3 if token.pos_ == "NOUN"])
s3_nouns
print(f"{s1} and {s2} VERBS: {nlp(s1_verbs).similarity(nlp(s2_verbs))}")
print(f"{s1} and {s3} VERBS: {nlp(s1_verbs).similarity(nlp(s3_verbs))}")
print(f"{s2} and {s3} VERBS: {nlp(s2_verbs).similarity(nlp(s3_verbs))}")
print(f"{s1} and {s2} ADJECTIVES: {nlp(s1_adjs).similarity(nlp(s2_adjs))}")
print(f"{s1} and {s3} ADJECTIVES: {nlp(s1_adjs).similarity(nlp(s3_adjs))}")
```

```
print(f"{s2} and {s3} ADJECTIVES: {nlp(s2_adjs).similarity(nlp(s3_adjs))}")
print(f"{s1} and {s2} NOUNS: {nlp(s1_nouns).similarity(nlp(s2_nouns))}")
print(f"{s1} and {s3} NOUNS: {nlp(s1_nouns).similarity(nlp(s3_nouns))}")
print(f"{s2} and {s3} NOUNS: {nlp(s2_nouns).similarity(nlp(s3_nouns))}")
```

Output-



```
[1]  import spacy
     import spacy.cli

     spacy.cli.download("en_core_web_lg")

     ✓ Download and installation successful
     You can now load the package via spacy.load('en_core_web_lg')

[2]  nlp = spacy.load("en_core_web_lg")

[3]  w1 = "purple"
     w2 = "blue"

[4]  w1 = nlp.vocab[w1]
     w2 = nlp.vocab[w2]

[5]  w1.similarity(w2)

     0.8199634552001953

[6]  s1 = nlp("This is lab class,execute practical")
     s2 = nlp("We have to make project, assigment and report")
     s3 = nlp("In total there are four subjects, this is tough")

[7]  s1.similarity(s2)

     0.5541106352502717

     s1.similarity(s3)

     0.7231906474686453

     + Code   + Text

[9]  s2.similarity(s3)

     0.5451758591161197

[10] s1_verbs = " ".join([token.lemma_ for token in s1 if token.pos_ == "VERB"])
     s2_verbs = " ".join([token.lemma_ for token in s2 if token.pos_ == "VERB"])
     s3_verbs = " ".join([token.lemma_ for token in s3 if token.pos_ == "VERB"])
     s3_verbs

     'be'

[11] s1_adjs = " ".join([token.lemma_ for token in s1 if token.pos_ == "ADJ"])
     s2_adjs = " ".join([token.lemma_ for token in s2 if token.pos_ == "ADJ"])
     s3_adjs = " ".join([token.lemma_ for token in s3 if token.pos_ == "ADJ"])
     s3_adjs

     print(f"{s1} and {s2} NOUNS: {nlp(s1_nouns).similarity(nlp(s2_nouns))}")
     print(f"{s1} and {s3} NOUNS: {nlp(s1_nouns).similarity(nlp(s3_nouns))}")
     print(f"{s2} and {s3} NOUNS: {nlp(s2_nouns).similarity(nlp(s3_nouns))}")

     This is lab class,execute practical and We have to make project, assigment and report NOUNS: 0.38335988774799745
     This is lab class,execute practical and In total there are four subjects, this is tough NOUNS: 0.24306817852589194
     We have to make project, assigment and report and In total there are four subjects, this is tough NOUNS: 0.4477134246601302
```