**NAME: Arjun Bakshi**

**Enrolment Number: BT18GEC134 Date: 09/10/2021**

**Section: D1**

**LAB – 7: Apply cosine similarity to measure the similarity between pair of documents.**

**Code:**

**X = open("Doc1.txt", "r")**

**Y = open("Doc2.txt", "r")**

**X\_ = X.read()**

**Y\_ = Y.read()**

**X\_list = word\_tokenize(X\_)**

**Y\_list = word\_tokenize(Y\_)**

**l1 = []**

**l2 = []**

**X\_set = {w for w in X\_list}**

**Y\_set = {w for w in Y\_list}**

**# form a set containing keywords of both strings**

**rvector = X\_set.union(Y\_set)**

**lis\_vector = list(rvector)**

**for w in rvector:**

**if w in X\_list:**

**l1.append(1) # create a vector**

**else:**

**l1.append(0)**

**if w in Y\_list:**

**l2.append(1)**

**else:**

**l2.append(0)**

**c = 0**

**# cosine formula**

**for i in range(len(rvector)):**

**c += l1[i]\*l2[i]**

**cosine = c / float((sum(l1)\*sum(l2))\*\*0.5)**

**t = X\_**

**u = Y\_**

**data = [t, u]**

**count\_vectorizer = CountVectorizer(stop\_words='english')**

**sparse\_matrix = count\_vectorizer.fit\_transform(data)**

**doc\_term\_matrix = sparse\_matrix.todense()**

**df = pd.DataFrame(doc\_term\_matrix, columns=count\_vectorizer.get\_feature\_names(**

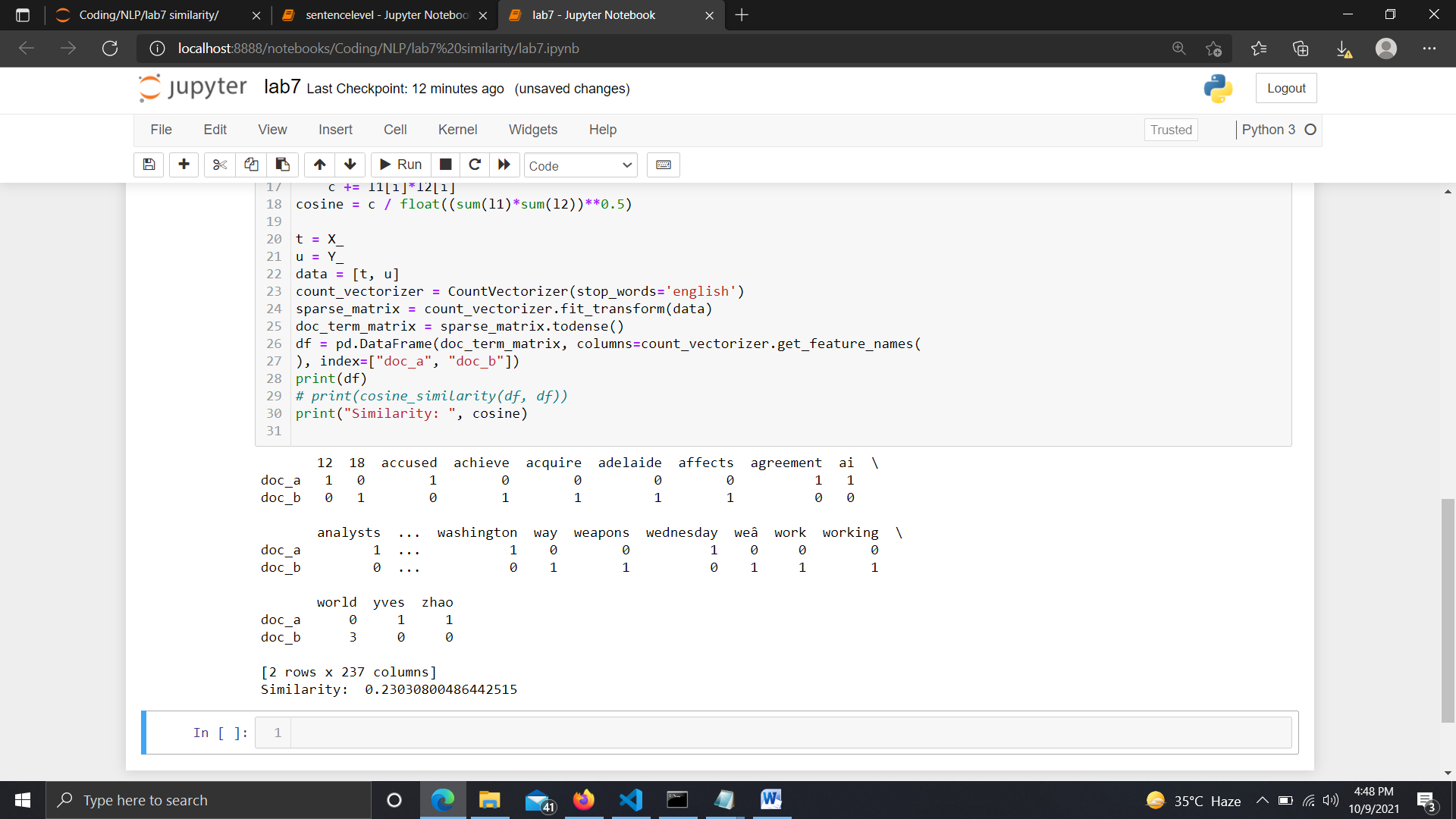
**), index=["doc\_a", "doc\_b"])**

**print(df)**

**# print(cosine\_similarity(df, df))**

**print("Similarity: ", cosine).**

**Output:**

****

**Part – 2: Find out the similarity measure, keeping six words in sequence as the maximum limit.**

**Code:**

X = open("Doc1.txt", "r")

Y = open("Doc2.txt", "r")

X\_ = X.read()

Y\_ = Y.read()

# tokenization

X\_list = word\_tokenize(X\_)

Y\_list = word\_tokenize(Y\_)

sw = stopwords.words('english')

# form a set containing keywords of both strings

for i in range(0, len(Y\_list), 6):

l1 = []

l2 = []

new\_x = X\_list[i:i+6]

new\_Y = Y\_list[i:i+6]

rvector = new\_x + new\_Y

lis\_vector = list(rvector)

print(new\_x, new\_Y)

for w in rvector:

if w in new\_x:

l1.append(1) # create a vector

else:

l1.append(0)

if w in new\_Y:

l2.append(1)

else:

l2.append(0)

c = 0

x = zip(l1, l2)

both = [i for i in x]

q = [(lis\_vector[i], both[i]) for i in range(len(l1))]

# cosine formula

for i in range(len(rvector)):

c += l1[i]\*l2[i]

try:

cosine = c / float((sum(l1)\*sum(l2))\*\*0.5)

except ZeroDivisionError:

cosine = 0

print("Similarity: ", cosine)

# W/O stopwords

print("\nW/O stopwords\n")

for i in range(0, len(Y\_list), 6):

l1 = []

l2 = []

new\_x = X\_list[i:i+6]

new\_Y = Y\_list[i:i+6]

X\_set = {w for w in new\_x if not w in sw}

Y\_set = {w for w in new\_Y if not w in sw}

rvector = X\_set.union(Y\_set)

lis\_vector = list(rvector)

print(X\_set, Y\_set)

for w in rvector:

if w in X\_set:

l1.append(1) # create a vector

else:

l1.append(0)

if w in Y\_set:

l2.append(1)

else:

l2.append(0)

c = 0

x = zip(l1, l2)

both = [i for i in x]

q = [(lis\_vector[i], both[i]) for i in range(len(l1))]

# cosine formula

for i in range(len(rvector)):

c += l1[i]\*l2[i]

try:

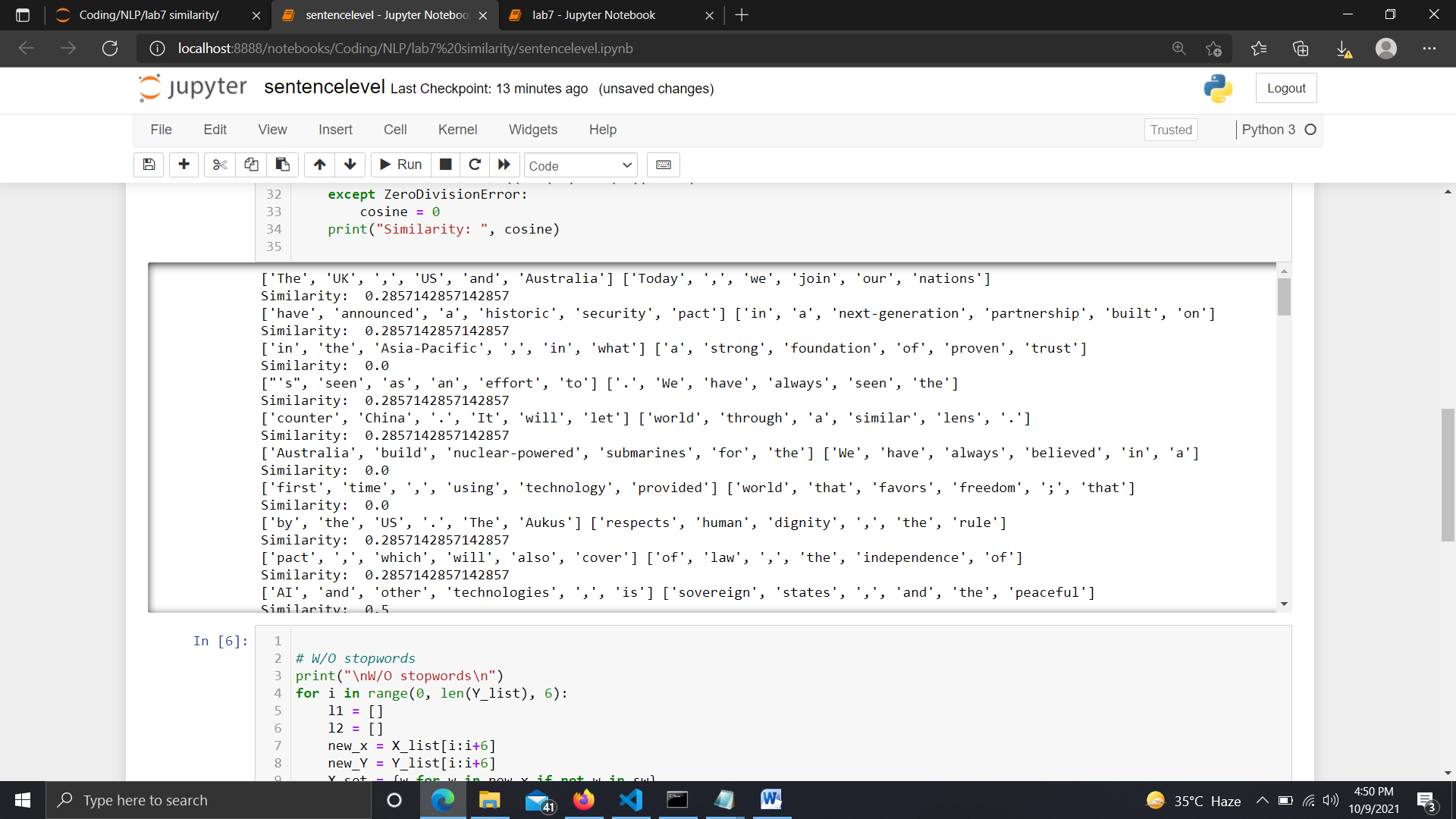
cosine = c / float((sum(l1)\*sum(l2))\*\*0.5)

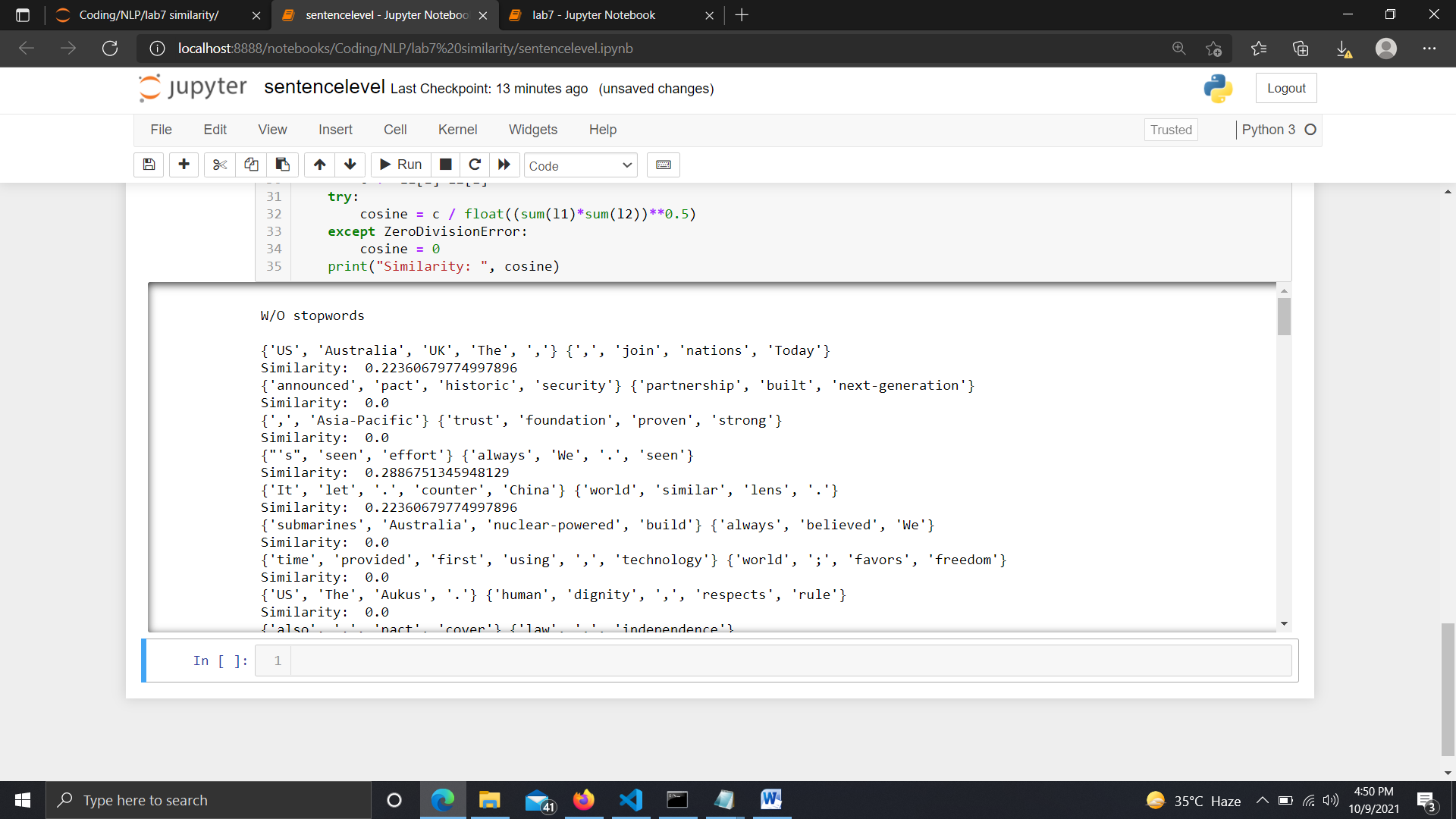
except ZeroDivisionError:

cosine = 0

print("Similarity: ", cosine).

**Output :**

****

****

**Analysis :**

1st Part, it is observed that by removing the Stopwords the similarity decreased, this is because by nature stopwords are common in any document in general and hence may not always give best measure of similarity

While in 2nd file in which we determined Sentence likeliness it was seen that most of the sentences had low or 0 comparability and by removing the stopwords the previous one additionally expanded.

Document Similarity was around 0.25.

While Sentence level Similarity created 0.0, 0.03 Similarity .