	No. of the control of	April 1			Page No. Date	H
======================================		J+-4	25 Natta	d Lingling	,	
		771	cosing	1		
	Q1) a					
_;		Python	Thiony	Geometry	Worked	Research
-i	doci		0	0	0	0
	do c 2					
	doc3	0		Ю		
7:	doc5	0	0	0		0
, 	doc 6	0	0	0	0	0
	anc v		U	0	0	1]
, ` ,						
		We sonside	No. 1	1 2 1	1 1	_ 1
		rector.	3500C	as our	Text (1)	st)
		Now,				
		,				100
		Cosine som	slavity (4212	(1 \1	1 3
·			0 (141VJ	
·:						
::-/		LMeri	(Tu) =	2 Li ²		100
·			J	F-1		
":/ _{"—} —						
···: 		cosine (pyth	von Reserver	y = (10)	000 10 11 (0 17001
T				- 1100	010 0	1001
				\		10011
				=0		
;						
· .						
Scarce Co.	- 1					

Page No. Date
cosine (Theory, Research) = [0,1,1,0,0,0] (0,1,1,0,0,1) [0,1,1,0,0,1]
554i2 55i2 [M/[V]
$ \mu = \sqrt{1 + 1} = 1.4/4$
(1) = 53 = 1.73 CS (theory, perenon) = 2 - 08164 (1.71) (1.73)
(S(geometry, Search) = (010000] [011001] [01000]
$\frac{1}{\sqrt{1}\sqrt{3}} = \frac{0.5773}{1.73}$
(05/ne sim (prosked, research) = (011100) [011001]
$\frac{2}{5353} = \frac{2}{3} = 0.667$

_	Page No.
======================================	
	Now, we have found vojne simillity
	Setwisen the fist rector research and all
÷	other vectors and the word chosest
	to relarch is :-
7	
TT	Theory with vosine similarity
	0.8164
5	
	No, this is not a danification task as
	we have just represented our of words
M	us vectors and that dres, it in health
	The second of th
· 17	Also, when we find similarly I atoper
	Also, when we find similarly between 2 vectors that doesn't automatically closelys the them into entry ories.
	No them into entry ories
T-f	To use these retto embeddings for
	classification we ned to define what our dustes our and the factors which
	will decide where one vertex will
`-: `-:-:-	belong.
]-v.	
<u></u>	let us say we wish to closing each
	Let us say we wish to closing each word as positive the Or regative than we
[=	is more than a specific value is pailine
-]	or via veson.
T.	

	Page No. Date
	We am wer ever hise advanced dissipiation
	techniques out as SVM's, Naive Baye's,
	Newsal nativores the
	100000 1000000
	The methods will take one verte as am
	The most may the beat 1'1't in fring
	input and return the probability of being
	in one of the challes.
	1 + 1 1 1 1 1 1 1 1 Clic
	so we need to add a personicitistic
	town model or one property
	tron this into a clossifier
	11 ·
Q2)	Rurdting the CU me oftam: -
	() () () () () () () () () ()
	(dada structurges) (whis Ninjus) (Mingly bolleger
doci	0 0
chc2	
ch 3	
doc4	0
ches	
Goc 6	0
	(College Landon) (Struttoes Algorithms)
1	(College London) (Strikers 19808 19808
chel	Ø L
dice	Ö
doc3	0
doc4	0
doce	0
0.00	
	Scanned with CamScanner

By Choosing to grown he Ostrino (bonds of photosism (with) (birrity college Looled) - dec2 0 0 0 - dec3 0 0 0 - dec5 0 0 - dec6 1 (- dec5 0 0 - dec5 0 0 - dec5 0 0 - dec6 0	==	=	1.62 4.45	Page No. Date
- doc2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	== 	-	By choosing to storng me of	storin'
- doc2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0		-	(London plutrum anitar) (trinity college Landal
- doc 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	, ,		O O	
- doc 5 - doc 6 - doc 5 - doc 6 - doc 7 - doc 7 - doc 7 - doc 7 - doc 8 - doc 9 - doc 9 - doc 1 - doc 5 - doc 6 - doc 5 - doc 6 - doc		-	0	0
Clock Cl		_ doc3	0	0
Clark of suchers of sont ms (pleitrum quitary of on the ms) (pleitrum quitary of one	7	-doc4	0	0
- doc 1 - doc 5 - doc 6 - doc 2 - doc 2 - doc 3 - doc 6 - doc 7 - doc 7 - doc 8 - doc 8 - doc 9 - d		_	J	0
- doc 3 0 0 0 0 - doc 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	7	_cloc6		
- doc 3 0 0 0 0 - doc 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0			the same and the	The state of the s
- doc 3 0 0 0 0 - doc 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0			(data structures algorithms)	(sleitrum quitos as
- doc 3 - doc 4 - doc 5 - doc 6 - doc 2 - doc 5 - doc 5 - doc 5 - doc 6 - doc 5 - doc 6 - doc 7 - doc 7 - doc 7 - doc 8 - doc 9 - d		-	, ,	0
- doc 5 - doc 6 - doc 6 - doc 6 - doc 1 - doc 2 - doc 3 - doc 5 - doc 5 - doc 5 - doc 6 - the figure is a Shrall document in the north month and for most and for most and for most and for most most and for most and	· ·	. •		0
- doc 5 - doc 6 - doc 1 - doc 2 - doc 5 - doc 5 - doc 5 - doc 5 - doc 6 - hoc 6 The resume is a Shrall downent in the north monty words and in	 		0	0.0
- doc6 (where London, plutoum) - doc2 - doc3 - doc5 - doc6 The serve is a Shrall downest in the	; ; 			0
- doc 2 0 - doc 3 0 - doc 5 0 - doc 6 1 The resume is a Shraw around in the	<u></u>		0	Trends 1: 0
- doc 2 0 - doc 3 0 - doc 5 0 - doc 6 1 The reme is a Shraw downent with	, 	doc6	0	
- doc 2 0 - doc 3 0 - doc 5 0 - doc 6 1 The reme is a Shraw downent with				
- doc 2 0 - doc 3 0 - doc 5 0 - doc 6 1 The reme is a Shraw downent with			(Whege London, pletsum)	
doc 3 doc 5 doc 6 The resume is a Shraw downent in the not many words and is	<u>,</u>		0	5
doc5 doc6 The resume is a Shraw drument in the			0	
- doc6 The resume is a Shraw downent with	, ,		0	
doc6 The resume is a Shraw downent with		-	0	
The resume is a Shrau downent with			0	
- may have fortisted and	<u></u>	MOC 6		
- may have fortisted and			4	
- may have fortisted and			The Jesume Is a Shrall a	wument with
that our high in wrigam but hot			not many words and	
that our higher in wright but hot			h-grans like to -gram	one dose work
to gram the Java-or python or			that Dur highly in un	igram but hot
			in to grown like In	a - or puttion or
			•	0

	Page No. Date
	github.
	best week is delivered using unigram
	and bistam Joequency.
	He should place Unigram over bigram as data like Java and Python is lost in bigrams, which is returally very important.
	Home, we should prejes unigram.
	A STATE OF THE STA
	to the second se
	A philas
-	

```
hot_vector.py × 🔑 cosine_similarity.py × 🐉 autocomplete.py × 🎉 lemmatization.py ×
                                                                             👼 bigram.py 🐣
 import nltk
 import pandas
 import pickle
 from nltk.corpus import stopwords
 from collections import Counter
 stopwords_en = set(stopwords.words('english'))
 # loading the document
 resume = open('../assets/resume.txt', 'r')
 document = resume.read().lower()
 resume.close()
# tokenizing the document
tokenizer = nltk.RegexpTokenizer(r'\w+')
tokens = tokenizer.tokenize(document)
# removing stop words from tokens
tokens = [token for token in tokens if token not in stopwords_en and token.isalpha()]
# divide the document into n parts
documents = []
n = 6
k = len(tokens) // n
for i in range(n):
    documents.append(tokens[k * i: len(tokens) if i == n - 1 else (i + 1) * k])
def count_n_grams(data, n, start_token='<s>', end_token='<e>'):
    n_{grams} = \{\}
    for index, token in enumerate(data):
        n_gram = tuple(data[index: index + n])
        n_{grams}[n_{gram}] = n_{grams.get}(n_{gram}, 0) + 1
    return n_grams
bigrams = []
 trigrams = []
 for document in documents;
    bigrams.append(count_n_grams(tokens, 2))
    trigrams.append(count_n_grams(tokens, 3))
most_common_bi = set()
for bigram_freq in bigrams:
 for token in most_common_bi > for index, bigram_freq in enume...
                    E 6: TODO
     Python Console
```

R R Console

Python Console

₩ 6: TODO

```
one_hot_vector.py 🕅 🔝
                  cosine_similarity.py ×
                                         🚜 autocomplete.py 🗵 🚜 lemmatization.py 🗵
                                                                                   👼 bigram.py
     def count_n_grams(data, n, start_token='<s>', end_token='<e>'):
         n_grams = {}
         for index, token in enumerate(data):
              n_gram = tuple(data[index: index + n])
              n_grams[n_gram] = n_grams.get(n_gram, 0) + 1
          return n_grams
      bigrams = []
      trigrams = []
      for document in documents:
          bigrams.append(count_n_grams(tokens, 2))
           trigrams.append(count_n_grams(tokens, 3))
0
42
43
       most_common_bi = set()
       for bigram_freq in bigrams:
44
           for word, freq in Counter(bigram_freq).most_common(5):
45
46
                most_common_bi.add(word)
47
48
49
        most_common_tri = set()
        for bigram_freq in trigrams:
58
 51
            for word, freq in Counter(bigram_freq).most_common(5):
 52
                most_common_tri.add(word)
 53
 54
        print(most_common_bi)
         print(most_common_tri)
 55
  56
  57
  58
         vectors_bi = {}
         for token in most_common_bi:
  59
  60
             vector = [0] * 6
             for index, bigram_freq in enumerate(bigrams):
   61
                 vector[index] = int(token in bigram_freq)
   62
   63
              vectors_bi[token] = vector
   64
   65
          print(vectors_bi)
           for token in most_common_bi >> for index, bigram_freq in enume...
```

Ι

	Page No. Date
== 03)	I We can use wordnet to find all
	inglifors of a word that one horms
T	and Han su the different sy nonyms
	of a word and su these
	aginitions es, sample vode;
H	
	Cole for tim ding Syrrony mo (Python)
	. 0000
\	the ort next next imp of wordent as un.
	The state of the s
[word - dog!
	synsets = wn synsets (wood, por = 'n')
	print (synsets [o]. definition)
1	The cagnition on
W-12	A member of glows Canif ()
N-4-	
	for synet in synoth: point (synoth depints on())
	Point (Jynou- Wan) Dn()
<u> </u>	
h-J	authut
	A member or Comme (-)
	A member of Cremis (anis A dull unaltractive girl or woman Diformal Toom for Man Someone who is from ally
	Diformal Toom for Man
<u></u>	Someone who is from ally
	(He get vortoms (ynongens)
	ay horg mo)

	Page No.
I to λ	Lemes and use it in Lammahigant on process.
(49)	de la la lamination on process.
	Londo and use of in the
b)	Using wordnet me vill find the closest
	Syronys and also maider closest syronys.
	the will seeme all words into its
	file mill stance and wisher to and worsider
	lovest common by prongers and consider
	those work,
	De for dog and out we consider & Comivore.
	Se de mos is to
	is a matile and with the state.
	The matro humsey of the matro
	destease but beginner with the but his
	The matrix humber of tikens will include. destease but frequency will include it will board to more generalization but loss will board to more generalization but loss
	g in John ation.
	10 (28) 1 (38) 1
	•

Scanned with CamScanner