Practical no 5

	Name: Shivam Tawari.
	Roll ma: A-58
	Subject: NLP
	Aim: write a python program for distribution
	metrice to find computational similarity
	from given dataset.
	Theory:
to-	Distributional Matrix Semantics Model:
->	DSM are the computational model that
	build contextual sermantic representation
	from corpus data.
>	DSH are model for semantic
	representation. The semantic
	context is represented by a
	rodor.
->	Distributed semantics is the
	study of Statistical patterns.
1	Distribution are rectors in
-	
	multidimentasional semantic space
No or prochagancy decreey, work or o	that is object with magnifule

and direction. The semantic space has dimension which correspond to possible context as gathered gram a given coupus. Constructing word spaces: 1 Pick the word you are interested in torget. Defines a context window number of words surrording target ward. The context can be general be defined in terms of documents, paragraph. count number of times the (3) torget words (0- occurs with context words co-occurance matrix. @ Build reckers out of function these co-occurance counts.

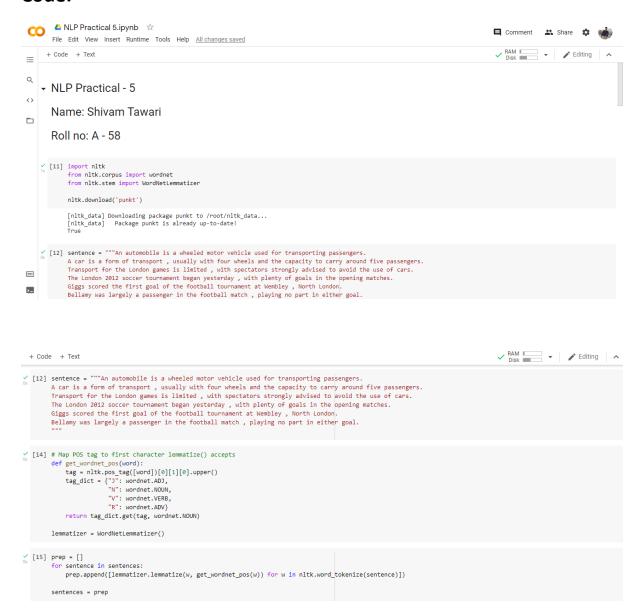
have Hence, we Conclusion: implemented the Successfully to find the brogram computational distributional matrix dataset. from given similarity

Practical - 5

Name: Shivam Tawari

Roll no: A – 58

Code:



→ Target Words

```
[16] ## example: automobile, car, soccer, football
target_words = map(str, input('Target Words: ').split(','))
target_words = [t_wrds.lower().replace(' ', '') for t_wrds in target_words]
                 Target Words: soccer, football, car, automobile ['soccer', 'football', 'car', 'automobile']
 Vocab
[17] ## example: wheel, transport, passenger, tournament, London, goal, match
term_vocab = map(str, input('Term Vocabulary: ').split(','))
term_vocab = [t_voc.lower().replace(' ', '') for t_voc in term_vocab]
                  Term Vocabulary: wheel, transport, passenger, tournament, london, goal, match ['wheel', 'transport', 'passenger', 'tournament', 'london', 'goal', 'match']
                                                                                                                                                                                                                                                                                                         ✓ RAM Disk Fediting ∧
   + Code + Text
[17] ## example: wheel, transport, passenger, tournament, London, goal, match
term_vocab = map(str, input('Term Vocabulary: ').split(','))
term_vocab = [t_voc.lower().replace(' ', '') for t_voc in term_vocab]
                  Term Vocabulary: wheel, transport, passenger, tournament, london, goal, match ['wheel', 'transport', 'passenger', 'tournament', 'london', 'goal', 'match']
      [20] matrix = {}
    for k in target_words:
        matrix[k] = {}
                         for v in term_vocab:
                                  matrix[k][v] = 0
                 for k, v in matrix.items():
                      print(k, v)
                  soccer {'wheel': 0, 'transport': 0, 'passenger': 0, 'tournament': 1, 'london': 1, 'goal': 1, 'match': 1}
football ('wheel': 0, 'transport': 0, 'passenger': 1, 'tournament': 1, 'london': 1, 'goal': 2, 'match': 1}
car {'wheel': 1, 'transport': 2, 'passenger': 1, 'tournament': 0, 'london': 1, 'goal': 0, 'match': 0}
automobile {'wheel': 1, 'transport': 1, 'passenger': 1, 'tournament': 0, 'london': 0, 'goal': 0, 'match': 0}
                                                                                                                                                                                                                                                                                                        ✓ RAM Disk Editing ∧
    + Code + Text
                  for K in target_words:
                     matrix[k] = {}
for v in term_vocab:
                                    matrix[k][v] = 0
                  for k, v in matrix.items():
        print(k, v)
                  soccer {'wheel': 0, 'transport': 0, 'passenger': 0, 'tournament': 1, 'london': 1, 'goal': 1, 'match': 1} football ('wheel': 0, 'transport': 0, 'passenger': 1, 'tournament': 1, 'london': 1, 'goal': 2, 'match': 1} car {'wheel': 1, 'transport': 2, 'passenger': 1, 'tournament': 0, 'london': 1, 'goal': 0, 'match': 0} automobile {'wheel': 1, 'transport': 1, 'passenger': 1, 'tournament': 0, 'london': 0, 'goal': 0, 'match': 0}
 / [22] final = []
                  Tinal = []
for tgt_wrd1, trm_vocab1 in matrix.items():
    final.append(tgt_wrd1)
    for tgt_wrd2, trm_vocab2 in matrix.items():
        if tgt_wrd2 in final:
                                  similarity = sum([a*b for a,b in zip(list(trm_vocab1.values()), list(trm_vocab2.values()))])
print(f'{tgt_wrd1} . {tgt_wrd2} = {similarity}')
                  soccer . football = 5
soccer . car = 1
soccer . automobile = 0
football . car = 2
football . automobile = 1
car . automobile = 4
```