Comp 543 Lab5

Kai-Po Lin (kl72)

【Q1 Code – Dictionary Implementation】

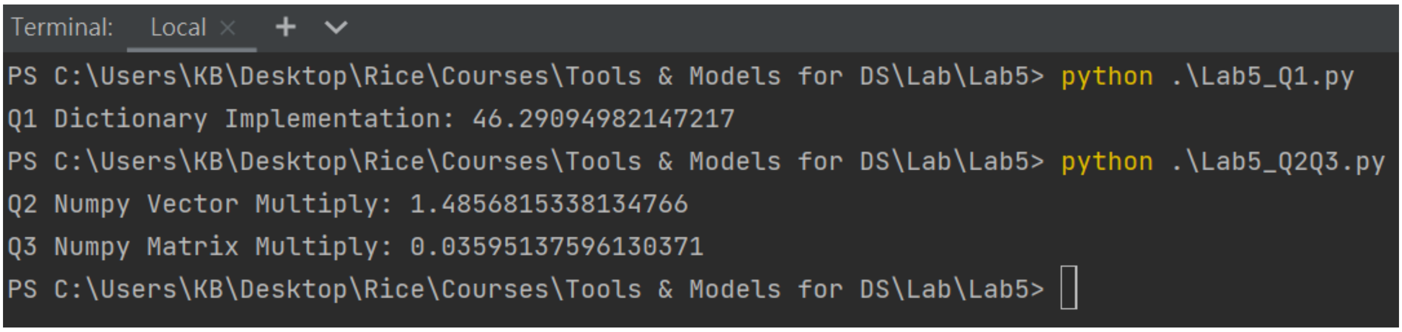
```
# For your reference, here is the dictionary-based LDA for use with the first sub-problem.
import numpy as np
import time



# This returns a number whose probability of occurrence is p
def sampleValue(p):
	return np.flatnonzero(np.random.multinomial(1, p, 1))[0]



# There are 2000 words in the corpus
alpha = np.full(2000, .1)
# There are 100 topics
beta = np.full(100, .1)

# This gets us the probability of each word happening in each of the 100 topics
wordsInTopic = np.random.dirichlet(alpha, 100)
# wordsInCorpus[i] will be a dictionary that gives us the number of each word in the document
wordsInCorpus = {}

# Generate each doc
for doc in range(0, 50):
	# No words in this doc yet
	wordsInDoc = {}

	# Get the topic probabilities for this doc
	topicsInDoc = np.random.dirichlet(beta)

	# Generate each of the 2000 words in this document
	for word in range(0, 2000):
		# Select the topic and the word
		whichTopic = sampleValue(topicsInDoc)
		whichWord = sampleValue(wordsInTopic[whichTopic])

		# And record the word
		wordsInDoc[whichWord] = wordsInDoc.get(whichWord, 0) + 1
	# Now, remember this document
	wordsInCorpus[doc] = wordsInDoc
```

```
# Q1 Answer
start = time.time()
# coOccurrences will be a map where the key is a
# (wordOne, wordTwo) pair, and the value is the number of times
# those two words co-occurred in a document, so this will be a
# value between zero and 50
coOccurrences = {}

# now, have a nested loop that piles up coOccurrences
# YOUR CODE HERE
for doc in wordsInCorpus:
    for wordOne in wordsInCorpus[doc]:
        for wordTwo in wordsInCorpus[doc]:
            if wordOne <= wordTwo:
                if (wordOne, wordTwo) not in coOccurrences:
                    coOccurrences[(wordOne, wordTwo)] = 1
                else:
                    coOccurrences[(wordOne, wordTwo)] += 1

end = time.time()
print("Q1 Dictionary Implementation:", end - start)
```

【Q2 Code – Numpy Vector Implementation & Q3 Code – Numpy Matrix Implementation】

```
# And here is the array-based LDA for use with the second two.
import numpy as np
import time

# There are 2000 words in the corpus
alpha = np.full(2000, .1)

# There are 100 topics
beta = np.full(100, .1)

# This gets us the probability of each word happening in each of the 100 topics
wordsInTopic = np.random.dirichlet(alpha, 100)

# wordsInCorpus[i] will give us the vector of words in document i
wordsInCorpus = np.zeros((50, 2000))

# Generate each doc
for doc in range(0, 50):
    # Get the topic probabilities for this doc
    topicsInDoc = np.random.dirichlet(beta)
    # Assign each of the 2000 words in this doc to a topic
    wordsToTopic = np.random.multinomial(2000, topicsInDoc)
    # And generate each of the 2000 words
    for topic in range(0, 100):
        wordsFromCurrentTopic = np.random.multinomial(wordsToTopic[topic], wordsInTopic[topic])
        wordsInCorpus[doc] = np.add(wordsInCorpus[doc], wordsFromCurrentTopic)


# Q2 Answer
start = time.time()
# coOccurrences[i, j] will give the count of the number of times that
# word i and word j appear in the same document in the corpus
coOccurrences = np.zeros((2000, 2000))

# Now, have a nested loop that piles up coOccurrences
# YOUR CODE HERE
for doc in range(len(wordsInCorpus)):
    wordsInCorpus[doc] = np.clip(wordsInCorpus[doc], 0, 1)
    coOccurrences += np.outer(wordsInCorpus[doc], wordsInCorpus[doc])

end = time.time()
print("Q2 Numpy Vector Multiply:", end - start)


# Q3 Answer
start = time.time()
coOccurrences = np.zeros((2000, 2000))

# Now, create coOccurrences via a matrix multiply
# YOUR CODE HERE
for doc in range(len(wordsInCorpus)):
    wordsInCorpus[doc] = np.clip(wordsInCorpus[doc], 0, 1)
wordsInCorpusTrans = np.transpose(wordsInCorpus)
coOccurrences = np.dot(wordsInCorpusTrans, wordsInCorpus)

end = time.time()
print("Q3 Numpy Matrix Multiply:", end - start)
```

【Result】



```
PS C:\Users\KB\Desktop\Rice\Courses\Tools & Models for DS\Lab\Lab5> python .\Lab5_Q1.py
Q1 Dictionary Implementation: 46.29094982147217
PS C:\Users\KB\Desktop\Rice\Courses\Tools & Models for DS\Lab\Lab5> python .\Lab5_Q2Q3.py
Q2 Numpy Vector Multiply: 1.4856815338134766
Q3 Numpy Matrix Multiply: 0.035951375961303710
PS C:\Users\KB\Desktop\Rice\Courses\Tools & Models for DS\Lab\Lab5>
```