Tanay Jhawar
B.E. 4 Roll No.22

EXPERIMENT 7

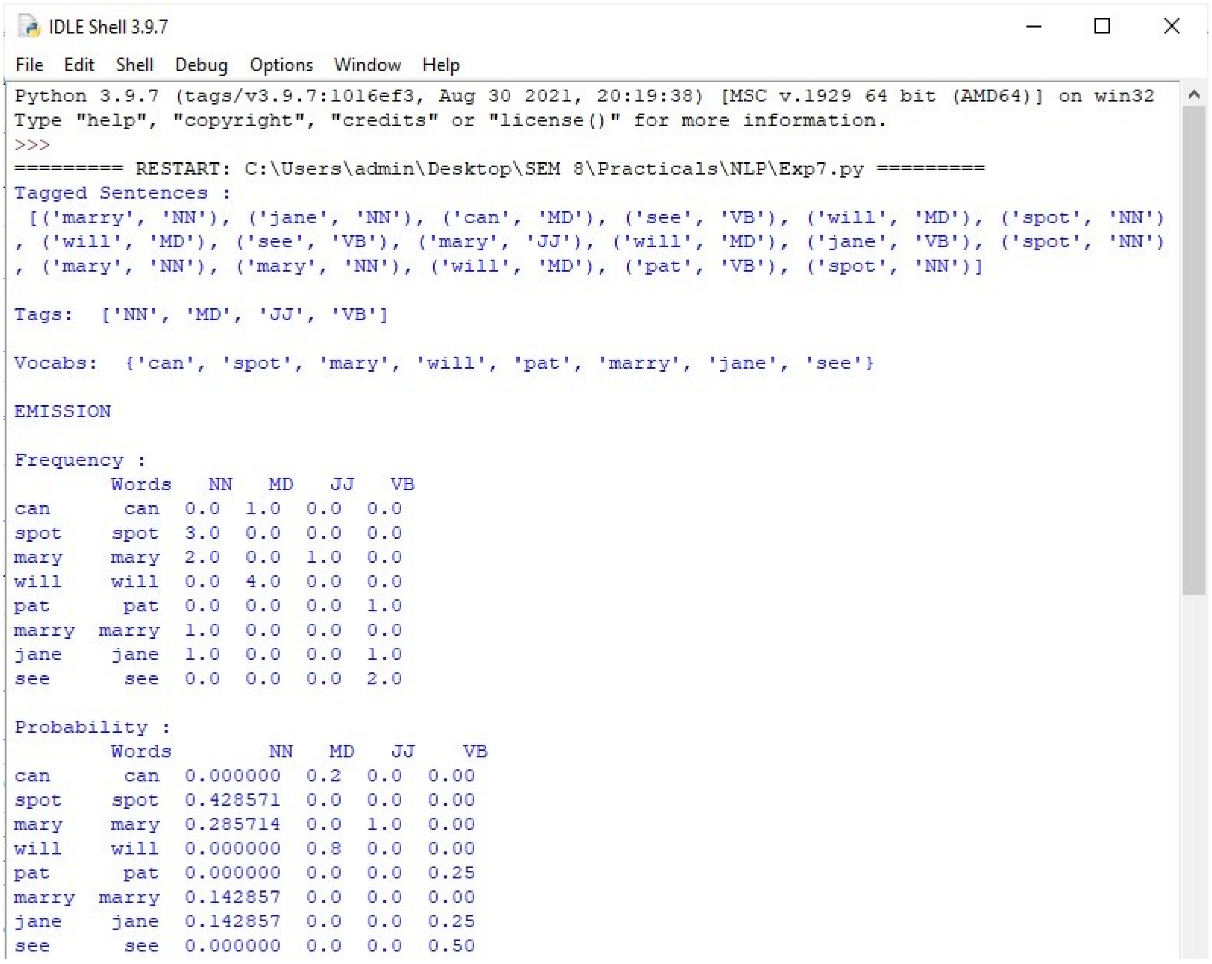
AIM: Implement POS Tagging using Hidden Markov Model

SOURCE CODE:

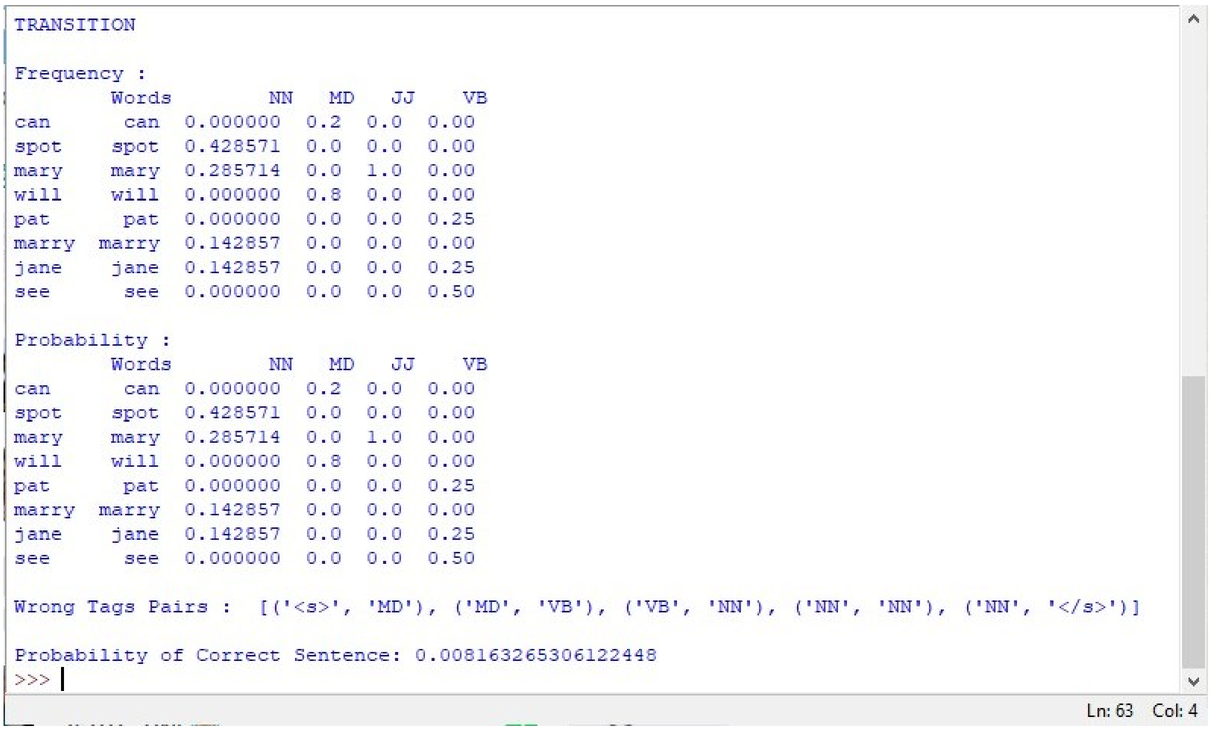
```
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import sent_tokenize, word_tokenize
from nltk.util import ngrams
from nltk.lm.preprocessing import pad_both_ends
from collections import Counter
import numpy as np
import pandas as pd
import string

text = '''Marry Jane can see Will.
Spot will see Mary.
Will Jane spot Mary?
Mary will pat Spot.'''
sent_text = nltk.sent_tokenize(text)
tagged_sents = []
tags_transitions = []
uni_tags = []
for sent in sent_text:
  tokenized_words = word_tokenize(sent)
  tokens = list(filter(lambda token: token not in string.punctuation, tokenized_words))
  tokens = list(x.lower() for x in tokens)
  tagger = nltk.pos_tag(tokens)
  tagged_sents.extend(tagger)
  tags_transition = [tup[1] for tup in tagger]
  uni_tags.extend(tags_transition + ['<s>', '</s>'])
  tags_transitions.extend(list(ngrams(pad_both_ends(tags_transition, n=2), n=2)))
print("Tagged Sentences :\n",tagged_sents)

tagged_words = [ tup for tup in tagged_sents ]
count_tagged_words = Counter(tagged_words)
tags = list({tag for word, tag in tagged_words})
vocabs = {word for word, tag in tagged_words}
print("\nTags: ",tags)
print("\nVocabs: ",vocabs)
```

```
em = pd.DataFrame({tag: [] for tag in ["Words"]+tags})

print("\nEMISSION")
for vocab in vocabs:
  em.loc[vocab] = [vocab] + [count_tagged_words[vocab, tag] for tag in tags]
em.set_index('Words')
print("\nFrequency :\n",em)
tag_freq_em = Counter(elem[1] for elem in tagged_sents)
for vocab in vocabs:
  for tag in tags:
    em.at[vocab, tag] /= tag_freq_em[tag]
print("\nProbability :\n",em)
print("\nTRANSITION")
tags_trans_freq = Counter(tags_transitions)
tr = pd.DataFrame({tag: [] for tag in ["Tags"]+tags+["</s>"]})
for tag_row in ["<s>"]+tags:
  tr.loc[tag_row] = [tag_row] + [tags_trans_freq[tag_row, tag_col] for tag_col in
tags+["</s>"]]
tr.set_index('Tags')
print("\nFrequency :\n",em)
tag_freq_tr = Counter(uni_tags)
for tag_row in ["<s>"]+tags:
  for tag_col in tags+["</s>"]:
    tr.at[tag_row, tag_col] /= max(1,tag_freq_tr[tag_row])
tr.set_index('Tags')
print("\nProbability :\n",em)

wrong_tag = [('will', 'MD'),
('can', 'VB'),
('spot', 'NN'),
('mary', 'NN')]
wrong_tags = [tup[1] for tup in wrong_tag]
wrong_tags_pairs = list(ngrams(pad_both_ends(wrong_tags, n=2), n=2))
print("\nWrong Tags Pairs : ", wrong_tags_pairs)

prob = 1
for pair in wrong_tags_pairs:
  prob *= tr.at[pair[0], pair[1]]
print("\nProbability of Correct Sentence:",prob)
```

OUTPUT:



```
Python 3.9.7 (tags/v3.9.7:1016ef3, Aug 30 2021, 20:19:38) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========= RESTART: C:\Users\admin\Desktop\SEM 8\Practicals\NLP\Exp7.py =========
Tagged Sentences :
 [('marry', 'NN'), ('jane', 'NN'), ('can', 'MD'), ('see', 'VB'), ('will', 'MD'), ('spot', 'NN')
, ('will', 'MD'), ('see', 'VB'), ('mary', 'JJ'), ('will', 'MD'), ('jane', 'VB'), ('spot', 'NN')
, ('mary', 'NN'), ('mary', 'NN'), ('will', 'MD'), ('pat', 'VB'), ('spot', 'NN')]

Tags:  ['NN', 'MD', 'JJ', 'VB']

Vocabs:  {'can', 'spot', 'mary', 'will', 'pat', 'marry', 'jane', 'see'}

EMISSION

Frequency :
        Words   NN   MD   JJ   VB
can       can  0.0  1.0  0.0  0.0
spot     spot  3.0  0.0  0.0  0.0
mary     mary  2.0  0.0  1.0  0.0
will     will  0.0  4.0  0.0  0.0
pat       pat  0.0  0.0  0.0  1.0
marry   marry  1.0  0.0  0.0  0.0
jane     jane  1.0  0.0  0.0  1.0
see       see  0.0  0.0  0.0  2.0

Probability :
        Words        NN   MD   JJ    VB
can       can  0.000000  0.2  0.0  0.00
spot     spot  0.428571  0.0  0.0  0.00
mary     mary  0.285714  0.0  1.0  0.00
will     will  0.000000  0.8  0.0  0.00
pat       pat  0.000000  0.0  0.0  0.25
marry   marry  0.142857  0.0  0.0  0.00
jane     jane  0.142857  0.0  0.0  0.25
see       see  0.000000  0.0  0.0  0.50
```



```
TRANSITION

Frequency :
        Words        NN   MD   JJ    VB
can       can  0.000000  0.2  0.0  0.00
spot     spot  0.428571  0.0  0.0  0.00
mary     mary  0.285714  0.0  1.0  0.00
will     will  0.000000  0.8  0.0  0.00
pat       pat  0.000000  0.0  0.0  0.25
marry   marry  0.142857  0.0  0.0  0.00
jane     jane  0.142857  0.0  0.0  0.25
see       see  0.000000  0.0  0.0  0.50

Probability :
        Words        NN   MD   JJ    VB
can       can  0.000000  0.2  0.0  0.00
spot     spot  0.428571  0.0  0.0  0.00
mary     mary  0.285714  0.0  1.0  0.00
will     will  0.000000  0.8  0.0  0.00
pat       pat  0.000000  0.0  0.0  0.25
marry   marry  0.142857  0.0  0.0  0.00
jane     jane  0.142857  0.0  0.0  0.25
see       see  0.000000  0.0  0.0  0.50

Wrong Tags Pairs :  [('<s>', 'MD'), ('MD', 'VB'), ('VB', 'NN'), ('NN', 'NN'), ('NN', '</s>')]

Probability of Correct Sentence: 0.008163265306122448
>>> 
```