```
225229124
```

Exercise 1

In [1]:

```
import nltk
from nltk.tokenize import sent_tokenize, word_tokenize
from nltk.tag import pos_tag

text = word_tokenize('And now for something completely different')
nltk.pos_tag(text,tagset='universal')
```

Out[1]:

```
[('And', 'CONJ'),
 ('now', 'ADV'),
 ('for', 'ADP'),
 ('something', 'NOUN'),
 ('completely', 'ADV'),
 ('different', 'ADJ')]
```

In [2]:

```
import nltk
nltk.download('brown')
```

```
[nltk_data] Downloading package brown to
[nltk_data]     C:\Users\8mpra\AppData\Roaming\nltk_data...
[nltk_data]   Package brown is already up-to-date!
```

Out[2]:

```
True
```

Exercise2

In [3]:

```
from nltk.corpus import brown
tagsen = brown.tagged_sents()
```

In [4]:

```
br_train = tagsen[0:50000]
br_test  = tagsen[50000:]
br_test[0]
```

Out[4]:

```
[('I', 'PPSS'),
 ('was', 'BEDZ'),
 ('loaded', 'VBN'),
 ('with', 'IN'),
 ('suds', 'NNS'),
 ('when', 'WRB'),
 ('I', 'PPSS'),
 ('ran', 'VBD'),
 ('away', 'RB'),
 (',', ','),
 ('and', 'CC'),
 ('I', 'PPSS'),
 ("haven't", 'HV*'),
 ('had', 'HVN'),
 ('a', 'AT'),
 ('chance', 'NN'),
 ('to', 'TO'),
 ('wash', 'VB'),
 ('it', 'PPO'),
 ('off', 'RP'),
 ('.', '.')]
```

step2

In [5]:

```
from nltk import DefaultTagger

t0 = nltk.DefaultTagger('NN')
t1 = nltk.UnigramTagger(br_train,backoff = t0)
t2 = nltk.BigramTagger(br_train,backoff = t1)
```

step3

In [6]:

```
t2.accuracy(br_test)
```

Out[6]:

```
0.9111006662708622
```

step4

In [7]:

```
to_train = [len(l) for l in br_train]
sum(to_train)
```

Out[7]:

```
1039920
```

In [8]:

```
to_test = [len(l) for l in br_test]
sum(to_test)
```

Out[8]:

```
121272
```

In [9]:

```
t1.accuracy(br_test)
```

Out[9]:

```
0.8897849462365591
```

In [10]:

```
t2.accuracy(br_test)
```

Out[10]:

```
0.9111006662708622
```

In [11]:

```
print(br_train[0])
```

```
[('The', 'AT'), ('Fulton', 'NP-TL'), ('County', 'NN-TL'), ('Grand', 'JJ-TL'), ('J
ury', 'NN-TL'), ('said', 'VBD'), ('Friday', 'NR'), ('an', 'AT'), ('investigatio
n', 'NN'), ('of', 'IN'), ("Atlanta's", 'NP$'), ('recent', 'JJ'), ('primary', 'N
N'), ('election', 'NN'), ('produced', 'VBD'), ('``', '``'), ('no', 'AT'), ('evide
nce', 'NN'), ("''", "''"), ('that', 'CS'), ('any', 'DTI'), ('irregularities', 'NN
S'), ('took', 'VBD'), ('place', 'NN'), ('.', '.')]
```

In [12]:

```
print(br_train[1277])
```

```
[('``', '``'), ('I', 'PPSS'), ('told', 'VBD'), ('him', 'PPO'), ('who', 'WPS'),
('I', 'PPSS'), ('was', 'BEDZ'), ('and', 'CC'), ('he', 'PPS'), ('was', 'BEDZ'),
('quite', 'QL'), ('cold', 'JJ'), ('.', '.')]
```

In [13]:

```
print(br_train[1277][11])
```

```
('cold', 'JJ')
```

In [14]:

```
br_train_flat = [(word,tag) for sent in br_train for (word,tag) in sent]
```

In [15]:

```
print(br_train_flat[:40])
```

```
[('The', 'AT'), ('Fulton', 'NP-TL'), ('County', 'NN-TL'), ('Grand', 'JJ-TL'), ('J
ury', 'NN-TL'), ('said', 'VBD'), ('Friday', 'NR'), ('an', 'AT'), ('investigatio
n', 'NN'), ('of', 'IN'), ("Atlanta's", 'NP$'), ('recent', 'JJ'), ('primary', 'N
N'), ('election', 'NN'), ('produced', 'VBD'), ('``', '``'), ('no', 'AT'), ('evide
nce', 'NN'), ("''", "''"), ('that', 'CS'), ('any', 'DTI'), ('irregularities', 'NN
S'), ('took', 'VBD'), ('place', 'NN'), ('.', '.'), ('The', 'AT'), ('jury', 'NN'),
('further', 'RBR'), ('said', 'VBD'), ('in', 'IN'), ('term-end', 'NN'), ('presentm
ents', 'NNS'), ('that', 'CS'), ('the', 'AT'), ('City', 'NN-TL'), ('Executive', 'J
J-TL'), ('Committee', 'NN-TL'), (',', ','), ('which', 'WDT'), ('had', 'HVD')]
```

In [16]:

```
print(br_train_flat[13])
```

```
('election', 'NN')
```

In [17]:

```
nltk.ConditionalFreqDist(br_train_flat)['cold']
```

Out[17]:

```
FreqDist({'JJ': 110, 'NN': 8, 'RB': 2})
```

In [18]:

```
br_train_2grams = list(nltk.ngrams(br_train_flat, 2))
br_train_cold = [a[1] for (a,b) in br_train_2grams if b[0] == 'cold']
fdist = nltk.FreqDist(br_train_cold)
[tag for (tag, _) in fdist.most_common()]
```

Out[18]:

```
['AT',
 'IN',
 'CC',
 'QL',
 'BEDZ',
 'JJ',
 ',',
 'DT',
 'PP$',
 'RP',
 '``',
 'NN',
 'VBN',
 'VBD',
 'CS',
 'BEZ',
 'DOZ',
 'RB',
 'PPSS',
 'BE',
 'VB',
 'VBZ',
 'NP$',
 'BEDZ*',
 '--',
 'DTI',
 'WRB',
 'BED']
```

In [19]:

```
[(w2+"/"+t2, t1) for ((w1,t1),(w2,t2)) in br_train_2grams]
```

Out[19]:

```
[('Fulton/NP-TL', 'AT'),
 ('County/NN-TL', 'NP-TL'),
 ('Grand/JJ-TL', 'NN-TL'),
 ('Jury/NN-TL', 'JJ-TL'),
 ('said/VBD', 'NN-TL'),
 ('Friday/NR', 'VBD'),
 ('an/AT', 'NR'),
 ('investigation/NN', 'AT'),
 ('of/IN', 'NN'),
 ("Atlanta's/NP$", 'IN'),
 ('recent/JJ', 'NP$'),
 ('primary/NN', 'JJ'),
 ('election/NN', 'NN'),
 ('produced/VBD', 'NN'),
 ('``/``', 'VBD'),
 ('no/AT', '``'),
 ('evidence/NN', 'AT'),
 ("''/''", 'NN'),
```

In [20]:

```
br_pre_cfd = nltk.ConditionalFreqDist([(w2+"/"+t2, t1) for ((w1,t1),(w2,t2)) in br_train_2grams
br_pre_cfd['cold/NN'].most_common()
```

Out[20]:

```
[('AT', 4), ('JJ', 2), (',', 1), ('DT', 1)]
```

In [21]:

```
br_pre_cfd['cold/JJ'].most_common()
```

Out[21]:

```
[('AT', 38),
 ('IN', 14),
 ('CC', 8),
 ('QL', 7),
 ('BEDZ', 7),
 ('JJ', 4),
 ('DT', 3),
 (',', 3),
 ('PP$', 3),
 ('``', 2),
 ('NN', 2),
 ('VBN', 2),
 ('VBD', 2),
 ('CS', 1),
 ('BEZ', 1),
 ('DOZ', 1),
 ('RB', 1),
 ('PPSS', 1),
 ('BE', 1),
 ('VB', 1),
 ('VBZ', 1),
 ('NP$', 1),
 ('BEDZ*', 1),
 ('--', 1),
 ('RP', 1),
 ('DTI', 1),
 ('WRB', 1),
 ('BED', 1)]
```

In [22]:

```
bi_tag = nltk.BigramTagger(br_train)
bi_tag.tag(word_tokenize('I was very cold'))
```

Out[22]:

```
[('I', 'PPSS'), ('was', 'BEDZ'), ('very', 'QL'), ('cold', 'JJ')]
```

In [23]:

```
bi_tag.tag(word_tokenize('I had a cold'))
```

Out[23]:

```
[('I', 'PPSS'), ('had', 'HVD'), ('a', 'AT'), ('cold', 'JJ')]
```

In [24]:

```
bi_tag.tag(word_tokenize('I had a severe cold'))
```

Out[24]:

```
[('I', 'PPSS'), ('had', 'HVD'), ('a', 'AT'), ('severe', 'JJ'), ('cold', 'JJ')]
```

In [25]:

```
bi_tag.tag(word_tokenize('January was a cold month'))
```

Out[25]:

```
[('January', None),
 ('was', None),
 ('a', None),
 ('cold', None),
 ('month', None)]
```

In [34]:

```
bi_tag.accuracy(br_train)
```

Out[34]:

```
0.7991528194476498
```

In [27]:

```
bi_tag.tag(word_tokenize('I faild to do so'))
```

Out[27]:

```
[('I', 'PPSS'), ('faild', None), ('to', None), ('do', None), ('so', None)]
```

In [28]:

```
bi_tag.tag(word_tokenize('I was happy, but so was my enemy'))
```

Out[28]:

```
[('I', 'PPSS'),
 ('was', 'BEDZ'),
 ('happy', 'JJ'),
 (',', ','),
 ('but', 'CC'),
 ('so', 'RB'),
 ('was', 'BEDZ'),
 ('my', 'PP$'),
 ('enemy', 'NN')]
```

In [29]:

```
bi_tag.tag(word_tokenize('So, how was the exam'))
```

Out[29]:

```
[('So', 'RB'),
 (',', ','),
 ('how', 'WRB'),
 ('was', 'BEDZ'),
 ('the', 'AT'),
 ('exam', None)]
```

In [30]:

```
bi_tag.tag(word_tokenize('The students came in early so they can get good seats'))
```

Out[30]:

```
[('The', 'AT'),
 ('students', 'NNS'),
 ('came', 'VBD'),
 ('in', 'IN'),
 ('early', 'JJ'),
 ('so', 'CS'),
 ('they', 'PPSS'),
 ('can', 'MD'),
 ('get', 'VB'),
 ('good', 'JJ'),
 ('seats', 'NNS')]
```

In [31]:

```
bi_tag.tag(word_tokenize('She failed the exam, so she must take it again'))
```

Out[31]:

```
[('She', 'PPS'),
 ('failed', 'VBD'),
 ('the', 'AT'),
 ('exam', None),
 (',', None),
 ('so', None),
 ('she', None),
 ('must', None),
 ('take', None),
 ('it', None),
 ('again', None)]
```

In [32]:

```
bi_tag.tag(word_tokenize('That was so incredible'))
```

Out[32]:

```
[('That', 'DT'), ('was', 'BEDZ'), ('so', 'QL'), ('incredible', 'JJ')]
```

In [33]:

```
bi_tag.tag(word_tokenize('Wow, so incredible'))
```

Out[33]:

```
[('Wow', None), (',', None), ('so', None), ('incredible', None)]
```

In [ ]:

```

```