Monsoon 2020 - NLP
Assignment 2

Extracting sentences from file and analysing the lengths of the sentences.

Splitting into sentences

```
file_sentences = filedata.split("\n")
length_sentences = []
for i in range(0,len(file_sentences)):
    length_sentences.append(len(file_sentences[i]))
max_length_of_sentence = max(length_sentences)
min_length_of_sentence = min(length_sentences)
average_length_of_sentence = int(sum(length_sentences)/len(length_sentences))
print('The number of sentences is {}'.format(len(file_sentences)))
print('maximum length of sentence is {}'.format(max_length_of_sentence))
print('minimum length of sentence is {}'.format(min_length_of_sentence))
print('avearge length of sentence is {}'.format(average_length_of_sentence))
```

```
The number of sentences is 55146
maximum length of sentence is 1960
minimum length of sentence is 0
avearge length of sentence is 178
```

Splitting sentences into words and tags pairs

```
print('length of words_and_tags is : {}'.format(len(words_and_tags)))
print('length of words is          : {}'.format(len(words)))
print('length of tags is           : {}'.format(len(tags)))
print('length of wat is            : {}'.format(len(wat)))
```

```
length of words_and_tags is : 1106057
length of words is          : 1106057
length of tags is           : 1106057
length of wat is            : 55146
```

Getting unique words and unique tags

```
# vocabulary : Unique words
Vocab = set(words)
print('Number of unique words in corpus is : {}'.format(len(Vocab)))
```

```
Number of unique words in corpus is : 56051
```

```
Tag_set = set(tags)
print('Number of unique tags in corpus is : {}'.format(len(tags)))
```

```
Number of unique tags in corpus is : 1106057
```

Getting Tag-count i.e. Number of words for a particular tag. Just pasting the sample output.

For tag FW-QL, the count of words is 1
For tag QL-NC, the count of words is 2
For tag PPL, the count of words is 1233
For tag WPS+BEZ-NC, the count of words is 2
For tag WDT+BEZ-NC, the count of words is 2
For tag HV, the count of words is 3928
For tag NNS-TL-HL, the count of words is 14

Calculating Emission Probabilities:-

Pasting the sample one

```
for key in EmissionProbs['The']:
    print('{}:{}'.format(key,EmissionProbs['The'][key]))
```

```
FW-BEZ:1.784089489928815e-05
FW-IN+AT-TL:1.784089489928815e-05
UH-TL:1.784089489928815e-05
MD*-HL:1.784089489928815e-05
FW-AT+NP-TL:1.784089489928815e-05
FW-QL:1.784089489928815e-05
CD-NC:1.784089489928815e-05
NPS$:1.784089489928815e-05
PN$:1.784089489928815e-05
NP+BEZ-NC:1.784089489928815e-05
```

Transition Probabilities

```
print(TransitionProbs['.']['NN'])
```

```
0.0006407657804919512
```

Viterbi Algo:Sample Output

```
print(Viterbi(['The'],tagged_words))
```

```
['AT-TL']
```

3 fold Cross Validation for Bigrams

Fold-1:-
Time taken for predicting tags for fold 1

```
start_time = time.time()
validation_predicted_1 = Viterbi(validation_words_1,tagged_words)
end_time = time.time()
difference = end_time - start_time
print('The time taken for predicting tags on validation set 1 is : {}'.format(difference))
```

```
The time taken for predicting tags on validation set 1 is : 116.97707605361938
```

Statistics of fold 1

Number of words for fold 1 is 409618
Number of actual tags for fold 1 is 409618
Number of predicted tags for fold 1 is 409618

Some Wrongly predicted tags are:-
The Wrongly predicted words in fold 1 are:-
['The', 'AT', 'AT-TL']
['Grand', 'JJ-TL', 'FW-JJ-TL']
['an', 'AT', 'AT-HL']
['primary', 'NN', 'JJ']
['no', 'AT', 'RB-NC']
['evidence', 'NN', 'VB']
['that', 'CS', 'WPS-HL']
['place', 'NN', 'NN-NC']

The time taken for computing confusion matrix for fold 1 is : 5.0981035232543945

```
print(confusion_matrix_1)
```

```
[[ 0.  0.  0. ...  0.  0.  0.]
 [ 0.  1.  0. ...  0.  0.  4.]
 [ 0.  0.  1. ...  0.  0.  7.]
 ...
 [ 0.  0.  0. ...  0.  0.  0.]
 [ 0.  0.  0. ...  0.  0.  0.]
 [26.  1.  1. ...  0.  0.  0.]]
```

correct predictions are : 348417
The time taken for computing accuracy for fold 1 is : 0.8437738418579102
THE ACCURACY FOR FOLD 1 is : 79.3893334765562

Precision for fold-1 is : 0.5746009794741124
Recall for fold-1 is : 0.38479725363084805
F1-score for fold-1 is : 0.39580662229413827

FW-NR -> {'precision': 0, 'recall': 0, 'f1_score': 0}
CC-TL-HL -> {'precision': 0.5, 'recall': 0.007692307692307693, 'f1_score': 0.015151515151515154}
PPSS+BEZ -> {'precision': 0, 'recall': 0, 'f1_score': 0}
WPS -> {'precision': 0.9201850780798149, 'recall': 0.6990333919156415, 'f1_score': 0.7945068664169788}
VBG-NC -> {'precision': 0, 'recall': 0.0, 'f1_score': 0}
HV-NC -> {'precision': 0, 'recall': 0.0, 'f1_score': 0}

Fold-2

Time taken to predict tags for fold-2

```
start_time = time.time()
validation_predicted_2 = Viterbi(validation_words_2,tagged_words)
end_time = time.time()
difference = end_time - start_time
print('The time taken for predicting tags on validation set 2 is : {}'.format(difference))
```

```
The time taken for predicting tags on validation set 2 is : 115.0758125782013
```

Statistics for fold-2

Number of words for fold 2 is 440834
Number of actual tags for fold 2 is 440834

Number of predicted tags for fold 2 is 440834

Some wrongly predicted tags in fold2 are
The Wrongly predicted words in fold 2 are:-
['too', 'RB', 'QL']
[',', ',', ',-HL']
['that', 'CS', 'WPS-HL']
['among', 'IN', 'NIL']
['the', 'AT', 'NIL']
['of', 'IN', 'IN-TL']
['to', 'TO', 'TO-NC']
['trust', 'VB', 'VB-NC']

The time taken for computing confusion matrix for fold 2 is : 5.717228651046753

```
print(confusion_matrix_2)
```

```
[[ 2.  0.  0. ...  0.  0.  3.]
 [ 0.  3.  0. ...  0.  0. 12.]
 [ 0.  0.  0. ...  0.  0.  7.]
 ...
 [ 0.  0.  0. ...  0.  0.  0.]
 [ 0.  0.  0. ...  0.  0.  0.]
 [20.  5.  1. ...  0.  0.  0.]]
```

correct predictions are : 351767
The time taken for computing accuracy for fold 2 is : 0.4713904857635498
THE ACCURACY FOR FOLD 2 is : 79.7998793196532

Precision for fold-2 is : 0.6830511442017804
Recall for fold-2 is   : 0.4430239388819362
F1-score for fold-2 is : 0.4588588103555425

FW-NR -> {'precision': 1.0, 'recall': 1.0, 'f1_score': 1.0}
CC-TL-HL -> {'precision': 0, 'recall': 0.0, 'f1_score': 0}
PPSS+BEZ -> {'precision': 0, 'recall': 0, 'f1_score': 0}

Fold-3

```
start_time = time.time()
validation_predicted_3 = Viterbi(validation_words_3,tagged_words)
end_time = time.time()
difference = end_time - start_time
print('The time taken for predicting tags on validation set 3 is : {}'.format(difference))
```

```
The time taken for predicting tags on validation set 3 is : 63.14765930175781
```

Statistics for fold-3

Number of words for fold 3 is 255605
Number of actual tags for fold 3 is 255605
Number of predicted tags for fold 3 is 255605

Some wrongly predicted tags in fold3
The Wrongly predicted words in fold 3 are:-
['calls', 'NNS', 'VBZ']
['A', 'AT', 'AT-TL']
["we're", 'PPSS+BER', 'PPSS+BER-NC']
['promising', 'VBG', 'JJ']
['We', 'PPSS', 'PPSS-TL']
['can', 'MD', 'MD-NC']
['more', 'AP', 'RBR']
['of', 'IN', 'IN-TL']

The time taken for computing confusion matrix for fold 3 is : 3.600268602371216

```
print(confusion_matrix_3)
```

```
[[ 0.  0.  0. ...  0.  0.  1.]
 [ 0.  0.  0. ...  0.  0.  2.]
 [ 0.  0.  0. ...  0.  0.  1.]
 ...
 [ 0.  0.  0. ...  0.  0.  2.]
 [ 0.  0.  0. ...  0.  0.  0.]
 [11.  0.  0. ...  0.  0.  0.]]
```

correct predictions are : 203368
The time taken for computing accuracy for fold 3 is : 0.3460569381713867
THE ACCURACY FOR FOLD 3 is : 79.58334148393028

Precision for fold-3 is : 0.4707791341806501
Recall for fold-3 is : 0.3967421356076746
F1-score for fold-3 is : 0.394240415623694

FW-NR -> {'precision': 0, 'recall': 0, 'f1_score': 0}
CC-TL-HL -> {'precision': 0, 'recall': 0.0, 'f1_score': 0}
PPSS+BEZ -> {'precision': 1.0, 'recall': 1.0, 'f1_score': 1.0}

Trigrams :-

```
start_time = time.time()
validation_predicted_trigrams = Viterbi_Trigrams(validation_words_trigrams,tagged_words)
end_time = time.time()
difference = end_time - start_time
print('The time taken for predicting tags on validation set Trigrams is : {} secs'.format(difference))
```

```
The time taken for predicting tags on validation set Trigrams is : 47.37774157524109 secs
```

Statistics for Trigram validation

Number of words for validation trigram is 154768
Number of actual tags for validation trigram is 154768
Number of predicted tags for validation trigram is 154768

Some wrongly predicted words in Trigram validation is
The Wrongly predicted words in Trigrams are:-
['her', 'PP$', 'PP$-NC']
['first', 'OD', 'OD-NC']
['she', 'PPS', 'PPS-NC']
['her', 'PP$', 'PP$-NC']
['enjoyed', 'VBD', 'VBN']
['usual', 'JJ', 'RB']
['not', '', '*-NC']*

The time taken for computing confusion matrix for Validation Trigrams is : 2.572845220565796 secs

```
print(confusion_matrix_trigrams)
```

```
[[  1.   0.   0. ...   0.   0.   1.]
 [  0.   1.   0. ...   0.   0.   1.]
 [  0.   0. 140. ...   0.   0. 201.]
 ...
 [  0.   0.   0. ...   0.   0.   0.]
 [  0.   0.   0. ...   0.   1.   0.]
 [  1.   3. 195. ...  38.   0.   0.]]
```

correct predictions are : 84711
The time taken for computing accuracy for validation trigrams is : 0.16094088554382324
THE ACCURACY FOR TRIGRAMS VALIDATION is : 79.58334148393028

PPSS+BEZ -> {'precision': 1.0, 'recall': 1.0, 'f1_score': 1.0}*
NN+HVZ-TL -> {'precision': 1.0, 'recall': 0.3333333333333333, 'f1_score': 0.5}
RBR -> {'precision': 0.6965174129353234, 'recall': 0.717948717948718, 'f1_score': 0.7070707070707071}
FW-IN+NN-TL -> {'precision': 0, 'recall': 0, 'f1_score': 0}
RP-HL -> {'precision': 0, 'recall': 0.0, 'f1_score': 0}

- There is no much difference between the accuracies of bigrams and trigrams in this case.