PRE-PROCESSING :

Preprocessing steps:-
- removal of white spaces
- removing urls, usernames,html tags, digits
- lower case the words and remove puncutation marks
-expanded contraction

Note:- data is preprocessed only to get meaningful words needed to generate sentences, for model training, trained or test data is not preprocessed at all.

1. **Incorporate one of the smoothing algorithms:**
   Laplace smoothing is used

So for the smoothing part of this problem we have used laplace algorithm
To calculate the below probability for each bigram, bigram count, unigram count ,vocab size is calculated and stored

$$P(w_i \mid w_{i-1}) = \frac{count(w_{i-1}w_i) + 1}{count(w_{i-1}) + V}$$

2.) You're expected to propose a solution to include the sentiment component. For instance,
you may modify your probability equation as follow:

Prob(wi | wi-1) = (count(wi-1 wi) / count(wi-1)) + β

where β is the sentiment component. You can define β as you deem fit. Note that you
are free to incorporate β at any other place as well, i.e., at the sentence-level, unigram-level, or bigram-level, at the numerator or denominator, etc. Please write a
modular well-documented code for this section.

Mention in the report the method you tried and justify your solution.

1) A bigram and unigram count model for positive labelled data and negatively labelled data is formed which would be used later. Let these be called sentiment oriented models.(term is used later
2) The focus was to pick words that have higher chance of being positive or negative
3) To do this before generating a sentence, it is decided that what would be the sentiment of the sentence (it is done randomly)
4) The first word of the sentence is picked randomly from the list of positive start words, this is stored before hand by first segregating the data into two parts positive and negative and then storing the first word of each sentence in a list, by doing so we can ensure that from the beginning we generate a comparatively higher sentiment oriented(which we decided in the beginning) sentence compared to randomly picking any word.

5) After picking the first word(say prev word), for the next word, from our bigram model(list) , only the bigrams in which the first word is the prev word is selected and stored in a dictionary along with laplace score.

6) Laplace score would be the basis on which we will pick the highest 10 scored bigrams(to select the next word)[this step is done in best_word function]

7) Let these words be called candidate words

8) Among the 10 candidate words, to select the next word a value would be calculated on the basis of which finally two words are selected

9) Before finding the best word, on the basis of the sentiment to which the sentence should be oriented, the sentiment bigram model is retrieved. That is if sentiment is pos(positive) then ugram = unigram model of positive words and bgram = bigram model of positive words similarly for negative sentiment the negative unigram and bigram model are retrieved

10) A ratio is calculated and the essence lies in the fact that we are trying to increase the numerator for the words that are more positive or negative depending on the sentiment given

11) Each candidate key is a bigram= (prev ,next), where prev is the word that was last stored in the sentence and next is the word which is a candidate for the word added after prev.

12) **The ratio is calculated by taking the number of times the bigram (prev,next) was present in the corpus + how many times the bigram was present in positive or negative labelled corpus**

13) Justification :- This value would be high for words whose previous word in data corpus is prev which appeared more number of times in the given sentiment label data for example if my sentiment was ‘pos’(that is my sentence should be more positive sentiment oriented) then this method would look for words that are not just more likely to be adjacent to previous but also appears as a pair in positive labelled data.

14) This value is normalized by the unigram count of previous in original corpus + unigram count of previous in sentiment labelled data(term mentioned above) + Vocab size

---

3.) Use Vader sentiment score (using the vader library) to obtain labels (either 0 or 1) for the generated 500 samples.

Vader sentiment scores are generated using the vader library

Perpexility
https://en.wikipedia.org/wiki/Perplexity

The perplexity *PP* of a discrete probability distribution *p* is defined as

$$PP(p) := 2^{H(p)} = 2^{-\sum_x p(x)\log_2 p(x)} = \prod_x p(x)^{-p(x)}$$

Perplexity is calculated as the entropy raised to power of 2
Where entropy for each sentence is sum of log probability of each word normalized by the number of bigrams in a sentence

The log probability is obtained by negative of (laplace of the bigram)*(log base 2 of laplace of the bigram)

b. Report the Top-4 bigrams and their score after smoothing.

(('good', 'morning'), 0.004870327529526361)
(('last', 'night'), 0.003975649148962604)
(('will', 'not'), 0.003759869657851861)
(("can't", 'wait'), 0.002976190476190476)

c. Report the accuracy of the test set using dataset A for training.

0.8773291925465838 accuracy of test set A

Part B - For each solution that you try the followings needs to be recorded:

c. Report the average perplexity of the generated 500 sentences.

AVERAGE PERPLEXITY - **1.00111752970981**

d. Report 10 generated samples: 5 positives + 5 negatives

POSITIVE
1.)weeeell sure 'm going bed soon 'll see 'm sorry hear 're still
2.)luving sunshine little bit disappointed weather today 'm going back work tomorrow hopefully
3.)lmfaooo would n't think 'm sorry hear 're going sleep night sweet
4.)friday night friends 'm going back work tomorrow hopefully 'll take care much

5.)hayfever medication n't get ready work tomorrow hopefully 'll take care much
NEGATIVE
1.)busy day 'm sorry hear 're going bed soon 'll try go
2.)hates hates hates hates hates hates hates hates hates hates hates hates hates
3.)despite last day 'm sorry hear 're going bed soon 'll try go
4.)sushi 've missed tweets set today 'm going bed soon 'll see
5.)whedonverse bloody wales however may pdt ñ load updates seems working hard

e. Report the accuracy of the test set using dataset B for training.

0.8819875776397516 ACCURACY OF TEST SET B