Natural Language Processing
Assignment - 1

Question - 1 :

parse_data_line: In this method, the text line and the label are selected through indexing. The given labels are converted into lower-case and passed into *convert_label* method to normalise the different types of labels given into 'FAKE' or 'REAL'.

pre_process: In this method, a string is taken as an input. Punctuations are separated from the words by a space by using regular expressions. Then the string is split into tokens by using *split()* method where the separator is a space. This list of tokens is returned at the end of the function.

Question - 2 :

to_feature_vector: In this method, the input is taken as a list of word. These words are made in to the keys of the *feature_vector* dictionary, where the values are the number of occurrences of the word in the input list of words.

If the word is already present in the dictionary keys, its values if incremented by one, else the word becomes the new key in the *feature_vector* dictionary and assigned a value of one.

Question - 3 :

cross_validate: This is the method for performing k-fold cross validation on a model.

1. Initially, the fold size is calculated from the given data and the number of folds.
2. The for loop iterates over the range (0, length of the dataset), where the step size is the fold size.
3. We split the given data into training set(9 folds) and test set(1 fold). For every iteration, the part of the data which is taken as the test set moves by one unit of fold size.
4. For every iteration, we train and test the classifier with the given function *train_classifier.*
5. We calculate Precision, Recall, Score, Support and Accuracy for every iteration and store the values in their respective lists.
6. After we exit the loop we calculate the Precision, Recall, Score, Support and Accuracy of the model for the entire k-fold cross validation process and store them in a dictionary called cv_results.

Question - 4 :

After training and validating the model, a confusion matrix heat map has been plotted, we can get a view of *false_negatives* and *false_positives* of the 'FAKE' label.

I have made all the samples that have been labeled as *false_positives* and *false_negatives* in to lists of respective names. These lists have been written in CSV files.

After observing these lists and playing round with *pre_process* method,

- The punctuation marks are affecting the accuracy of classifying fakes.
- There are a few words that are occurring more often in fake texts, giving more weight to such words in feature vector can improve the performance of the classifiers for classifying fakes.

Question - 5 :

To improve the pre-processing, have introduced lemmatisation, removing stopwords, removing/separating punctuations, removing space at the start and at the end of text and converting the words to lower-case.

Also created a vocabulary of words from raw data where the words have a minimum frequency of 2.

After playing around with the code and trying different combinations of the preprocessing methods, the combination of separating punctuations, stripping spaces, not converting the words to lower-case and using the vocabulary created to select features yielded the best results (Precision: 0.559, Accuracy: 0.558)

Have also included the number of words in a sentence in the feature vector dictionary. Haven't seen any significant improvement in the performance.

Have also tried removing stopwords.

The combination of not removing stopwords and including the number of words per sentence has improved the performance.(Precision: 0.565, Accuracy: 0.568)

Question - 6 :

Have included the other features of the dataset in the feature vector dictionary(as evident in the code). In order to differentiate from the actual words, have attached 'text_' before the keyword in dictionary(*eg: subject* becomes *text_subject*).

After trying many combinations of the features, excluding the *speaker* and *state* yielded the best results (Precision: 0.604, Accuracy: 0.606).

Even-though this combination yielded the best results overall, the model is still confused while classifying the 'FAKE' texts.

This shows that even if the overall accuracy and precision are high, it might not be the same with individual labels.