NLP Assignment 1 – Pavan Kumar

Question 1)

Load Data Function:

- The load_data function's goal is to read a file separated by tabs and append its contents to the raw_data global list variable, TSV file is given as input file which contains, ID, label and text.
- File Reading: The function iterates over the lines in the file, excluding the header line, using the csv.reader.
- Data processing: The text and label for each line are extracted using the parse_data_line function, and they are then appended to raw_data.

Parse Data Line Function:

- The parse_data_line function's goal is to extract the text and label from a given data line.
- The text appears in the third column (data_line[2]), and the label appears in the second (data_line[1]), which are returned after the function call.
- Len(raw_data) shows 33540 lines of text containing labels (Positive,Negative) and text

Question 2)

- The function `to_feature_vector} is defined in the code to produce a Bag of Words (BoW) feature vector.
- The binary values of 1 or 0 signify whether a word is present or absent in the given token list.
- The global counts for every feature across instances are kept in the global dictionary, `global_feature_dict`.
- n-grams as features are included in addition to individual words using the `ngrams` function from the {nltk} library.
- Using the binary representation, the feature vectorization procedure is made simpler for both individual words and n-grams.
- For every distinct feature that is detected throughout the token processing, the function updates the global dictionary.
- For situations where word presence is important, the code's emphasis on simplicity and binary feature values makes it appropriate.
- Additional preprocessing stages or different feature weighting algorithms could be added in future upgrades.

Question 3)

- A cross_validate function is defined in the code to assess a classifier on a given dataset by k-fold cross-validation.
- The input dataset's features (texts) and labels are separated, and StratifiedKFold is initialised with a predetermined number of folds.
- Evaluation metrics are recorded for every fold and kept in the cv_results dictionary. These metrics include precision, recall, accuracy, and F1 score.
- Each training set is used to train a classifier, and its performance is assessed on the matching validation set using the function.
- The evaluation metrics for each fold are printed during the process, and the model with the highest accuracy across all folds is chosen as the best model.
- 33540 Data points from raw data are split in 80% of train data(26832) and 20% of test data(6708)
- 68410 global feature vectors are created from the training dataset (global_feature_dict)
- The following metrics are obtained after 10 fold cross validation on the training dataset {'precision': 0.8408628179028772, 'recall': 0.8423902109267768, 'f1_score': 0.841173930204401, 'accuracy': 0.8423902109267768}

Question 4)

- The code defines the confusion_matrix_heatmap function, which uses the matplotlib package to plot a confusion matrix.
- The metrics are used to create the confusion matrix.confusion_matrix function from scikit-learn, which uses the given class labels as the basis for the predicted labels (preds) and true labels (y_test).
- Based on the anticipated and true labels, false positives and false negatives are distinguished and kept in different lists (false_positives and false_negatives).
- Using the confusion matrix and plt.matshow, a heatmap is produced. Matplotlib methods are then used to display the heatmap with further formatting.
- The class labels are represented by the ticks and labels (false_positives and false_negatives) on the x and y axes.
- To show the real values, text annotations are given to each matrix cell.
- The test_data is used to extract the test_texts, test_labels, and test_predictions.
- True positives : 3951 , True Negatives: 1726, False Positives: 432, False Negatives: 599
- We were able to get around 84 % accuracy with this simple SVC model which could be improved futher

False Positive: Tony Blair said God will judge me over Iraq Here s another biblical truism You we reap what you sow https t co EVpntoZd , (Predicted: positive, True: negative)

The phrase "God will judge" is linked to religious judgement. Such wording may be difficult for the model to comprehend nuancedly, leading it to incorrectly identify it as positive sentiment.

False Negative: Imagine if Ether had dropped in the social media age We may not have same respect we do for Jay z today , (Predicted: negative, True: positive) –

"Imagine" is used in the false negative, which discusses a speculative situation. It's possible that the model fails to adequately convey the context and the optimistic message conveyed through creativity.

- We need to improve the model to understand contextual nuances, political, religious data corpus. Fine tuning of the model with different pre processing techniques would help to reduce the errors, negations should be implemented on the model such as to handle (e.g., "not compulsory")

Question 5)

pre_process(text):

- Text was converted to lowercase using text.lower() , allowing consistency and removing case-related differences.
- Added pre-processing to remove URL links, special characters, digits, and emoticons for a cleaner text representation.
- Lemmatization was applied as a crucial step to improve model accuracy; this was especially helpful for text classification.

to_feature_vector():

- Bigrams and trigrams were added to the feature representation in order to give features a more complete context than unigrams within sentences.

train_classifier():

- Tested a range of SVC classifier hyperparameters, such as max_iter, C values, and weights. Stop words were attempted to be eliminated, but no discernible increase in model accuracy was found; as a result, this feature was not put into use. The model's accuracy significantly improved as a result of these code modifications, rising from 84 % to 86.7%.