Homework 1 - CS584

Hoang-Dung Bui

Team Name: bui1720

Mason ID: G01301478

Rank: 320

Public Score: 0.63

1 Introduction

In homework 1, I built a KNN program to classify the Amazon text review. Besides the main function, I developed four functions which helped the program in well organization. The four functions are: *remove_words*, *create_label_and_wordlist*, *create_wordlist*, and *similarity_function*.

The *remove_words* function removes some words which bring neutral attitude to simplify the strings.

The *create_label_and_wordlist* function: processes the train data file, which separates labels, splits the review string into words and save it to a list. This list and label list are then processed in the main program.

The *create_wordlist* function processes the test data file. It splits the test review strings into words and saves them into a list. This list will be processed in the main program. The *similarity_function* function: calculate the distance or similarity between two reviews from train data and test data, respectively. The similarity is calculated by *cosine* approach.

In the main program, it reads the train and test data files, deploys function 2 and 3 to splits the reviews into words lists, builds a set of highest similarity values between two reviews (from train data and test data). At the end, the similarity values set will be used to evaluate the test review by majority vote.

There is a parameter which needs to change, that is the k_nn. It should be set to 1, 3, 5, or 7 to run the program.

The program run by python3, and requires some packages to run such as re, math and collections/Counter. The detail of the programs are discussed in detail in the following section.

2 Program's Detail

1. *remove_words* function: removes some words which provides neutral attitudes to simplify the strings. Its inputs are a string and a list of words which will be removed. The function *replace* of regular expression module is used to perform the work.

2. *create_label_and_wordlist* function processes the train data file. It runs a *for loop* to reach each line/string (corresponding to a review) in the file. The first step is to get the label of that review, and save it into *label_set* set. After that, it invokes the *remove_words* function to move a set of neutral words from the string. The string is then split into a list of words by function *split* from regular expression. The word list is then saved into *review_word_set* list. This list and the label one are then returned to the main program.

3. *create_wordlist* function processes the test data file. It operates in similar way to function 2, except

that there is no label set in this test data.

4. *similarity_function* function: calculate the distance/similarity between two reviews word lists from train data and test data, respectively. The word lists are converted into dictionaries, then sets to speed up the processing time. The dictionaries determine the number of repeated words in the reviews and how many time the words are repeated. To find the common words between two strings, the *intersection* operation of is used. From the common words, the indices of the words in the dictionaries can be tracked back and are used to calculate the dot product. The magnitude each dictionary (feature vectors) can be calculated by math functions such as *power* and *sqrt*. Finally, the similarity is calculated by *cosine* approach. For empty review, the function will return 0.

5. *main* program, it reads the train and test data files, deploys function 2 and 3 to splits the reviews into words lists. There are two loops which calculates the similarity values between two reviews (from train data and test data). If the value lies in top k_nn similarity value set, the train review will substitute the review with lowest similarity value in the set. At the end, the top k_nn similarity values will be used to evaluate the attitude of the test review by majority vote. The voting result will be saved on the *out_save.dat* file.

3 Experiment Results

Classification's parameters:

- $k_{nn} = 3$
- majority vote
- similarity: cosine approach

The highest accuracy is only 0.63, which needs to improve. However, due to the computer configuration, it needs 10 hours to run all the train and test data.

4 Conclusion

The KNN program ran well and provide a positive result. To improve the efficiency, we need to understand deeply the code and processing time of each operation.