Danhua Yang
SID: 010769681

Rank: 3; Accuracy: 0.8448

Approach:

Read document data and preprocess:

The “train.dat” file has been read line by line using “.readlines()” function and splitten each line into a list “X_train”. Loop this list to get the first element which is “+1” or “-1” and save into a result list “y_train”. The “test.dat” file has been read line by line and splitten each line into a list “ X_test”.

Loop both X_train and X_test word by word to filter out the word whose length is smaller than 3.

Nltk library has been imported for word lemmatization. In this program the “WordNetLemmatizer” has been used to reduce the word inflectional forms.

Sklearn TfidfVectorizer library has been imported for document data to CSR matrix transformation. In the program, the TfidfVectorizer has been used. The parameters used to make the vectorizer is: lowercase = True, stop_words = english, use_idf = True, max_df = 0.01, min_df = 0.0005, norm = l2, ngram_range = (1,3), tokenizer = lemmatizer. After have the vectorizer ready, it has been used to fit the X_train to extract all the features. Then transform both X_train and X_test into two separate CSR matrix “X_train_idf_l2_dtm” and “X_test_idf_l2_dtm”

Similarity computation:

Sklearn linear_kernel library has been used to compute the pairwise Cosine similarities with train and test two separate CSR matrix “X_train_idf_l2_dtm” and “X_test_idf_l2_dtm”. The pairwise Cosine similarities has been saved into a size 18506 x 18506 matrix “cosine_sim_idf”.

K nearest neighbour selection and result determination:

The K value chosen for this solution is 900 which means 900 nearest neighbours will be used for the result determination.

Numpy library has been used to sort the top K similarity indices. The function used in the program is “.argpartition()” which will return the top K column indices for each row of the cosine_sim_idf matrix. The sorted indices has been saved into a size 18506 x K matrix.

For each record in test file, with the top K similarity indices, a result will be calculated with the sum(neighbour’s value x it’s similarity) / sum(neighbour’s similarity). Then if the result is equal or greater than the threshold (0.8 in this solution), “+1” will append into a list “y_test_dis”. Otherwise, “-1” will be appended.

Loop the y_test_dis list and write the result into a file.

Methodology for choosing the approach and associated parameters:

Sklearn train_test_split library has been used to split the train set into 80%(train) and 20%(test) to estimate the accuracy of the classifier model in order to modify the approach and parameters.

Minimal word filter length: 3 and 4 have been tested, and 3 is the final choice because 3 provides better result.

Tf-idf: with or without Tf-idf vectorizer have been tested. And by applying Tf-idf normalization, the result can have 2-3% increment.

Lemmatizer: when construct the vectorizer, after lemmatizer was introduced to the Tf-idf vectorizer, the result can have almost 2% increment.

Ngram-range: I tested without ngram, (1,2), (1,3) and (1,4). Ngram-range (1,3) gives the best result accuracy. With this Ngram range, the vectorizer will extract the features with length of 1, 2, 3 words.

L-2 Norm: l2 norm has been used to reduce the computation for Cosine similarity. This won't affect the result accuracy.

Knn: Majority method and distance weighted method have been tested. And the distance weighted method provides better accuracy.

Threshold and K selection: Using the binary search concept, I manually selected the numbers and tested to determine the threshold is 0.8 and K is 900.