HW4

Yasaman Emami Deep Learning Nov23,2022

1. The text preprocessing steps are not in the same order as HW Q1 and code
   After each step of preprocessing the new format of data in stored as a new column in the pandas dataframe in which includes the original data as “review” and “sentiment column.
   The order in implementation is as follows:
   a) Convert to lower case
   b) Remove HTML & punctuations
   c) Tokenization (from nltk package)
   d) Removing stop words (filtered based on nltk english stopword list)
   e) Stemming and lemmatizing (PorterStemmer() & WordNetLemmatizer() from nltk package)

   Then the clean text is detokenized in new column as well in case its needed for future use.
2.
   a) TFIDF is applied on detokenized data column and TfidfVectorizer() function was used from sklearn package, vectors are stored in new column called ‘review_vectorized’
      The sentiments are **onehot encoded** to convert string type data to numerical an new column as sentiment_encode stores encoded sentiments
      Then svm with RBF kernel is applied. RBF is selected due to how much the RBF Kernel resembles the K-Nearest Neighborhood Algorithm, it is widely used. As RBF Kernel Support Vector Machines only need to store the support vectors during training and not the complete dataset, it has the benefits of K-NN and solves the space complexity issue. and the model generated based on test data from ‘review_vectorized’ column and the model was evaluated based on the test data and predictions with respect to the accuracy score. Accuracy of this model is 95 %.

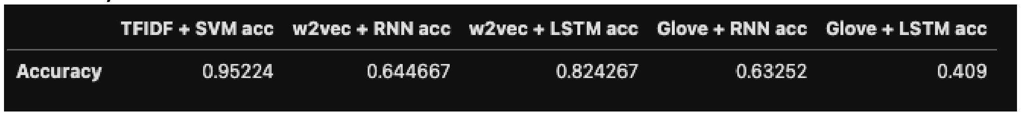
   b) Word2Vec (CBoW) + RNN
      Word2vec from genism package as a word embedding representor is applied on reviews embedding size is set to 100 then a keras embedding layer with w2v weights are generated after that a flatten layer is added to the model followed by 2 hidden layers with relu as activation and dropout with .5 and .2 values is applied to prevent overfitting then the last layer activation is softmax with number of 2 outputs . The model is compiled using adam optimizer and accuracy as the metric. Final accuracy calculated for this model is 64%. running time for this model was faster in compare to LSTM

   c) Word2Vec (CBoW) + LSTM
      For this part the keras embedding layer with w2v weights were genereated and added to the model then bidirectional LSTM layer was added which keeps the

sequence information in both directions(FWD and BWD) so training with this architecture takes longer time in compare to RNN . the accuracy for this model was calculated as 82%.

d) Glove + RNN
RNN architecture is as part b of this question but Glove was applied to create vectors from the text and accuracy is 63%.

e) Glove + LSTM
LSTM architecture is same as part c with Glove word vectorization method the accuracy is 40%



	TFIDF + SVM acc	w2vec + RNN acc	w2vec + LSTM acc	Glove + RNN acc	Glove + LSTM acc
Accuracy	0.95224	0.644667	0.824267	0.63252	0.409

The above table illustrates TFIDF + SVM showing the best outcome followed by word2vec + LSTM. W2v showing better result in compare with Glove. LSTM takes very long time to train in compare with RNN.

3. In this question after applying keras embedding layer and getting the vectors for the words a dataframe is generated to store each row as a vector and a dictionary is mapping the words to vectors. For this model accuracy is 50%. Then cosine similarity between vector of word 'good' and other rows are calculated the top 5 ones are represented as:

   a. jack [0.99675786]
   b. to [0.9965871]
   c. clairvoy [0.99651676]
   d. outstay [0.99675536]
   e. mceveeti [0.9965618]