CSE 5525 HW1 Report
Kun Peng

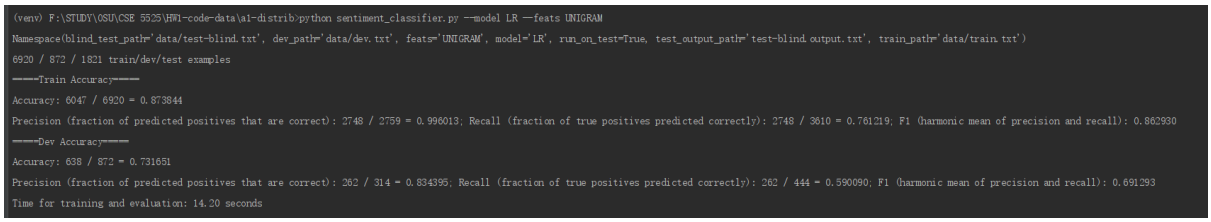
Q1. I generated the vocabulary in *train_model*. I discarded stop words (stopwords from nltk.corpus, which should be downloaded before running the project) and a set of some usual punctuations for the vocabulary, and I also lowercased all the words. The feature values referred to the occurrence of a certain feature in the sentence.

Q2. I generated the vocabulary in train_model and lowercased all the words. I discarded stop words if both two words of the bigram were in the set of stop words, and I discarded the bigrams that contained at least one kind of punctuation. The feature values referred to the occurrence of a certain feature in the sentence.

Note: time for training and evaluation varied for several seconds each time I ran the project with the same epochs, and this might depend on my computer performance.

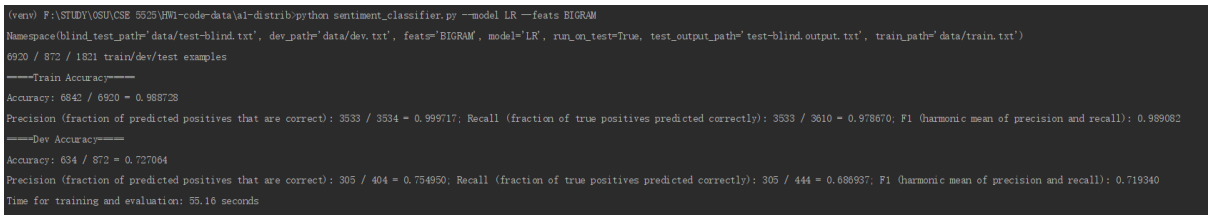
Q3.

Namespace(blind_test_path='data/test-blind.txt', dev_path='data/dev.txt', feats='UNIGRAM', model='LR', run_on_test=True, test_output_path='test-blind.output.txt', train_path='data/train.txt')
6920 / 872 / 1821 train/dev/test examples
=====Train Accuracy=====
Accuracy: 6047 / 6920 = 0.873844
Precision (fraction of predicted positives that are correct): 2748 / 2759 = 0.996013; Recall (fraction of true positives predicted correctly): 2748 / 3610 = 0.761219; F1 (harmonic mean of precision and recall): 0.862930
=====Dev Accuracy=====
Accuracy: 638 / 872 = 0.731651
Precision (fraction of predicted positives that are correct): 262 / 314 = 0.834395; Recall (fraction of true positives predicted correctly): 262 / 444 = 0.590090; F1 (harmonic mean of precision and recall): 0.691293
Time for training and evaluation: 14.20 seconds



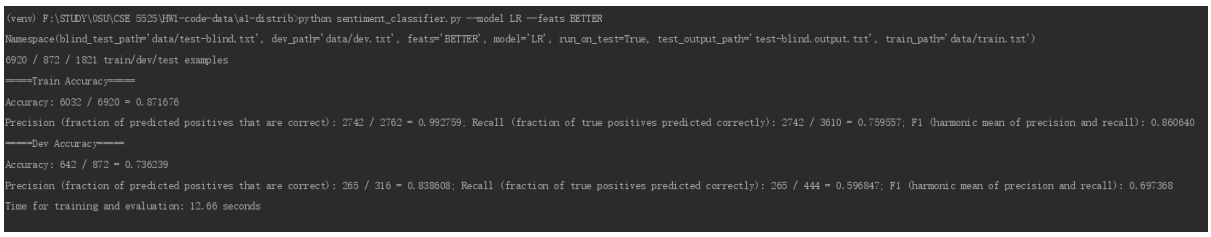
Q4.
Namespace(blind_test_path='data/test-blind.txt', dev_path='data/dev.txt', feats='BIGRAM', model='LR', run_on_test=True, test_output_path='test-blind.output.txt', train_path='data/train.txt')
6920 / 872 / 1821 train/dev/test examples
=====Train Accuracy=====
Accuracy: 6842 / 6920 = 0.988728

Precision (fraction of predicted positives that are correct): 3533 / 3534 = 0.999717; Recall (fraction of true positives predicted correctly): 3533 / 3610 = 0.978670; F1 (harmonic mean of precision and recall): 0.989082
=====Dev Accuracy=====
Accuracy: 634 / 872 = 0.727064
Precision (fraction of predicted positives that are correct): 305 / 404 = 0.754950; Recall (fraction of true positives predicted correctly): 305 / 444 = 0.686937; F1 (harmonic mean of precision and recall): 0.719340
Time for training and evaluation: 55.16 seconds

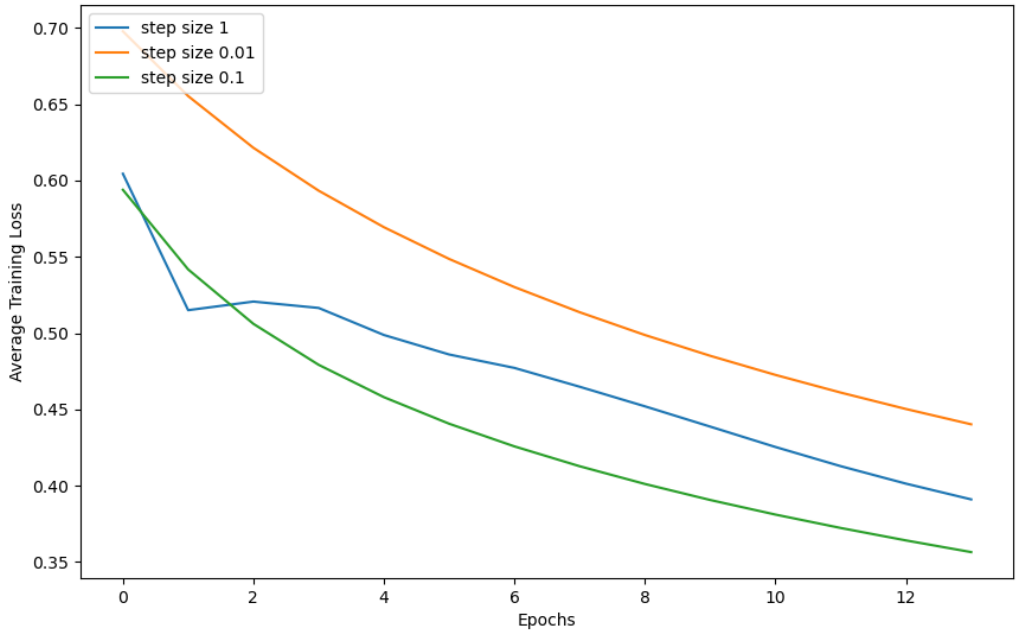


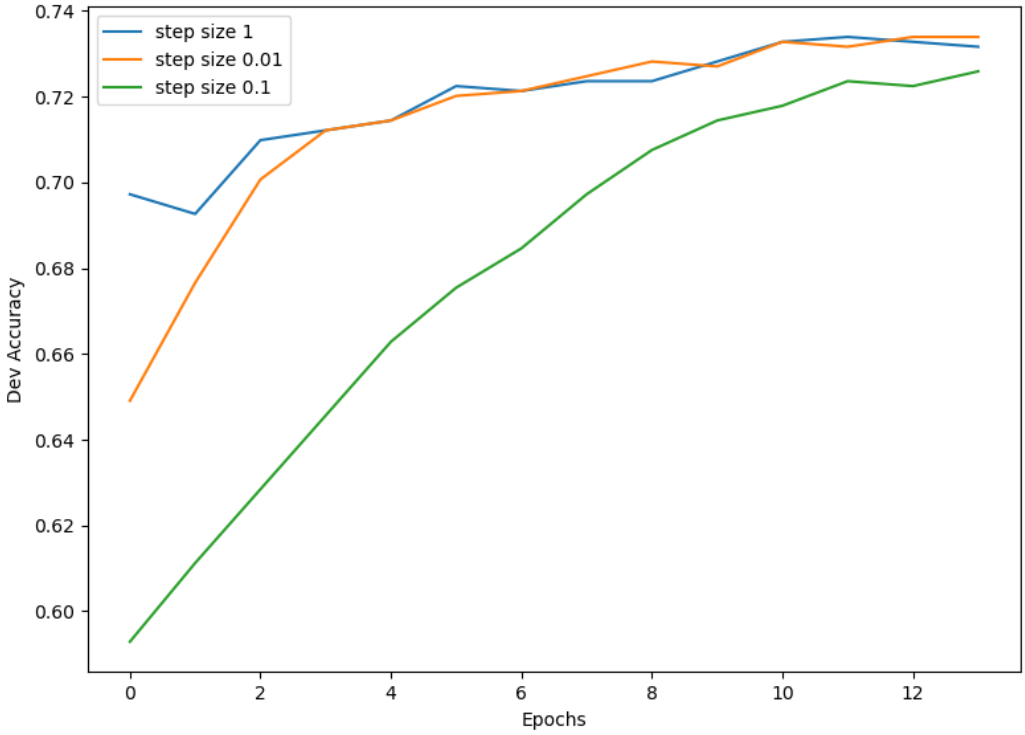
Q5. I implemented BetterFeatureExtractor based on UnigramFeatureExtractor. For this method I tried to discard rare words by only generating 75% of the words which were more common than the rest.

Namespace(blind_test_path='data/test-blind.txt', dev_path='data/dev.txt', feats='BETTER', model='LR', run_on_test=True, test_output_path='test-blind.output.txt', train_path='data/train.txt')
6920 / 872 / 1821 train/dev/test examples
=====Train Accuracy=====
Accuracy: 6032 / 6920 = 0.871676
Precision (fraction of predicted positives that are correct): 2742 / 2762 = 0.992759; Recall (fraction of true positives predicted correctly): 2742 / 3610 = 0.759557; F1 (harmonic mean of precision and recall): 0.860640
=====Dev Accuracy=====
Accuracy: 642 / 872 = 0.736239
Precision (fraction of predicted positives that are correct): 265 / 316 = 0.838608; Recall (fraction of true positives predicted correctly): 265 / 444 = 0.596847; F1 (harmonic mean of precision and recall): 0.697368
Time for training and evaluation: 12.66 seconds



Q6. BetterFeatureExtractor performs the best.





From the graphs it can be concluded that with smaller learning rates, (average) training loss and epochs are generally negatively correlated. With learning rate = 1, training loss and epochs are also negatively correlated after the second epoch. Among these three learning rates, 0.1 performs the best.

On the other hand, with each learning rate, the development accuracy and epochs seem to be positively correlated, although the accuracy is not always increasing. Learning rate 1 and 0.01 have similar performance

on development accuracy, but the development accuracy with learning rate 0.1 tend to be close to the accuracies with other two learning rates while epochs increase.