Ken Hubbard

IST 664

Final Project

For my final project, I opted to use the Kaggle movie review dataset in order to predict the sentiment of randomly generated phrases. My task was to find different features that would train a classifier to produce the most accurate results when looking at if a phrase were either negative, somewhat negative, neutral, somewhat positive, or positive.

When setting up the data to be modified, I made sure to use 10,000 random phrases out of the 156,060 phrases presented in the dataset, as using all phrases timed out the program and rendered it unusable. I did not filter any of the tokens out, as it didn’t really matter in the grand scheme of the phrase, where the emotion words were the only thing being measured anyways within the finite string.

My first feature was a unigram baseline, so that words present in the document could be matched and rendered between the 5 sentiments. I then created a train and test set that looked at 5000 phrases for each one and tested the accuracy. Despite the single accuracy being 0.53, I cross-validated the accuracy to get the average of 10 folds, which recorded as 0.5375. Looking at the most informative features, “ugly”, “awful”, and “tedious” had a strong affinity to the label of negative, but I was surprised to see words like “hours” and “years” having a correlation to being positive instead of neutral. I then created a confusion matrix to see how accurate the classifier was for the individual emotions of negative, somewhat negative, neutral, somewhat positive, and positive to learn that the classifier was strong in determining neutral sentiments (41.4%), and poorer with the stronger emotions (1% and 0.8%). After creating an evaluation measures function, I was able to determine that for neutral, its highest efficacy by far, this classifier resulted in 0.812 for precision (percentage of predicted yes answers that are right), 0.629 for recall (percentage of actual yes answers that are right), and 0.709 in F1 (balance of both precision and recall).

Next, I wanted to create a classifier that compared the words in the movie review phrases to a word-sentiment subjectivity file “subjclueslen1”. Here, I created a readSubjectivity function to create a dictionary for the words in this new file as well as the 4 different classes for it. To create this new featureset, I added weights to the words to show more strength in if they were negative, somewhat negative, somewhat positive, or positive. When testing the classifier, it received a 0.5332 accuracy. When cross-validating the classifier with 10 folds, it received a 0.5493 accuracy. Here, the most informative features were fairly similar to the unigram words. For the subjectivity classifier, the confusion matrix became SLIGHTLY more consistent than the unigram classifier, with 37.9% for neutral, and 1.4% and 0.9% for positive and negative, respectfully. This classifiers precision decreased to 0.744, however, but increased for recall at 0.670 and the F1 decreased to 0.705, yet all metrics increased slightly for positive and negative in this classifier.

My next featureset examined negation words. For this, I analyzed the phrases for negation words like “not”, “isn’t” or “hardly” and grouped them with the following word in one token to reverse the sentiment for that word. By doing this, it would better classify a phrase by it’s true sentiment. While the accuracy was 0.513, the cross-validation over 10 folds was 0.5216. The most informative feature words included the new token “NOTreally” with a negative sentiment. Despite a lower average accuracy than the other classifiers, the conversion matrix shows a continued trend towards a slightly more even disbursement for the different emotional tags, with neutral decreasing to 38.5% and positive and negative increasing to 1.5% and 1.3%. The other evaluation measures follow suit with neutral precision decreasing to 0.757, recall increasing (compared to unigram classifier) to 0.646, and F1 further decreasing to 0.697.

My last classifier dealt with part-of-speech tagging. This function helps further classify the words to make the sentiment more blatant. The features function essentially tags each word with either a N, V, J, or R for nouns verbs, adjectives, or adverbs. The accuracy for this classifier only resulted in a 0.5252 score, and a 0.5331 accuracy after cross-validating with 10 folds. The conversion matrix for the featureset starts to look more like the one for the unigram featureset, with neutral at 40.8%, positive at 1.2%, and negative at 1.0%. The precision, recall, and F1 scores all start to look more similar as well, with 0.802, 0.640, and 0.712 respectively for neutral.

The last step in the project is to use the best classifier as part of the processKaggle function. For this, I chose the subjectivity featureset, as it posed the highest accuracy, despite being more catered towards neutral matches than the other emotions. To test this feature, I took to the Anaconda PowerShell Prompt, where I entered “classifyKaggle.py corpus/ 3” in order to retrieve data from the Kaggle dataset with the classify, connect to the corpus, and then choose 3 random phrases to test their sentiment.

In conclusion, many different featuresets were tested for this dataset. In order to find the greatest accuracy in a classifier, it took the subjectivity document provided to us in lab. The unigram featureset was a good baseline, but the subjectivity featureset took it to the next level with accurate word/emotion analysis. While some of the other featuresets provided better precision, recall, and F1 for certain matches, it all comes down to overall accuracy for total matches. This was proven to be the case with the subjectivity featureset. Accuracy is also not something to be taken too literally when only tested once. That’s why cross-validation testing with a decent number of folds is important to calculate the average of many tests. This provides a more accurate…accuracy. When comparing the featuresets accuracy, it is important to look at context. For the negation featureset, the accuracy was less than the subjectivity. This could be because the negation string of tokens could have not been the most precise when determining sentiment for a phrase. Perhaps adding more negation words or taking out some from the first list would have changed the accuracy. For the POS featureset, it makes sense as to why the featureset wouldn’t be more accurate than the others. Adding tags for parts of speech doesn’t really contribute much to a dataset when looking at sentiment analysis, regardless if it is for a movie review dataset or others. When it comes down to this project, it is a no-brainer that the subjectivity featureset best contributed to the accuracy of a classifier to figure out the sentiment of different phrases.