Assignment 2

Andréa Davis

February 24, 2011

1 Introduction

Using a support vector machine, SVM^{light}, I tested the efficacy of a number of different features for accurate classification of sentiment of movie reviews.

1.1 Method

1.2 Baseline

A simple baseline model was constructed that simply counted the number of occurrences of positive words vs. the number of negative words. The set of positive words and the set of negative words were created by simply listing 20 words of positive connotation and 20 words of negative connotation, that I imagined might be in a positive vs. negative review. In the event that there were an equal number of these types of words, positive and negative, the model classified the review as a tie. Percent correct counts ties as incorrect classifications. It is very unlikely that a set of 20 positive words and a set of 20 negative words exhausts the number of words that would predict whether a review was positive or negative; further, this model is limited by the type of features that it can find relevant (only unigrams).

Accordingly, it is expected that it will do rather poorly, and that SVM^{light} will outperform the baseline model for all the training sets it receives.

Words in positive and negative categories were:

Positive words good, great, fabulous, fantastic, brilliantly, rocked, beauty, sensational, faultless, masterpiece, grace, best, loved, excellent, greatest, exciting, original, perfect, perfection, fun

Negative words bad, awful, horrible, sucks, stupid, ridiculous, dumb, empty, worst, hated, waste, boring, predictable, moronic, idiotic, disgusted, detested, repelled, crappiest, lousy

1.3 SVM^{light}

SVM^{light} was compared on the following feature sets:

Unigrams Unigram words of frequency greater than 3, 4 or 5.

Bigrams Bigram words of top N frequency, where N is the number of unigrams of frequency greater than 3, 4, or 5.

Part of speech tags

Unigrams with part of speech tags

Adjectives

Top unigrams Unigram words of top M frequency, where M is the number of adjectives of frequency greater than 4.

These feature sets were compared with the baseline model, as well as with each other. Again, since SVM^{light} learns from training data, unlike the baseline model, and since it has more feature types available to it, it is expected to do better than the baseline model. Further, it is expected that some feature types will do better than others, and even that a combination of feature types may do best of all.

1.4 Relevant Files

Files are named:

posneg.py This is the baseline classifier

featurefinder.py This program takes in a folder of reviews, extracts the features listed above, and makes training files for each set of features extracted. It also makes validation files for each feature extracted, for testing. Training, validation, and testing files must be made together, or else feature codes will not match up.

In all cases, the input/output files will be at the top of the program, so as to easily change what will be fed into them. Training, validation, and testing files are of the format:

$< target >< feature >:< value >< feature >:< value >$

In making the training, validation, and testing files, features are coded as unique numbers, and are arranged in ascending order. This is to make them readable to SVM^{light}.

2 Training

2.1

Results for the validation sets are given in Table 1.

	Unigrams	Bigrams	Unigrams with POS Tags	Adjectives	Top Unigrams
Percent correct	85.00%	84.50%	82.00%	76.00%	82.00%
Number correct	170	169	164	152	164
Number incorrect	30	31	36	48	36
Precision	87.23%	87.10%	82.65%	76.53%	84.04%
Recall	82.00%	81.00%	81.0%	75.00%	79.00%

Table 1: Results of training on different features

2.2 Optimization

The best results were obtained with the unigram and bigram features. I adjusted the number of unigram and bigram features used, to see if this would consistently improve the score. Table 2 gives the results; Unigrams4 means that in the training data, only unigrams of frequency 4 or more were included, Unigrams5 means that only unigrams of frequency 5 or more were included, etc.

	Unigrams4	Bigrams4	Unigrams5	Bigrams5	Unigrams3	Bigrams3
Percent correct	85.00%	84.50%	83.5%	83.5%	85.00%	85.5%
Number correct	170	169	167	167	170	169
Number incorrect	30	31	33	33	30	29
Precision	87.23%	87.10%	86.02%	85.26%	87.23%	87.37%
Recall	82.00%	81.00%	80.0%	81.0%	82.00%	83.00%

Table 2: Results of training on ”optimized” features

2.3 Number of features

The number of distinct features for each feature set is given in Table 3:

Model	Number of distinct features
Unigrams4	16234
Bigrams4	16234
Unigrams with POS tags	57249
Adjectives	3112
Top Unigrams	3112
Unigrams5	14017
Bigrams5	14017
Unigrams3	19630
Bigrams3	19630

Table 3: Number of distinct features for each feature set

3 Testing

The optimal model during the training was the model that used the top N bigrams, where N is the number of unigrams of frequency 3 or higher (designated as Bigrams3), achieving 85.5 percent correct.

Model	Percent Correct	Number of Ties
Baseline	62.00%	31
Unigrams4	**89.50%**	NA
Bigrams4	84.00%	NA
Unigrams with POS tags	**89.50%**	NA
Adjectives	82.50%	NA
Top Unigrams	87.00%	NA
Unigrams5	**89.50%**	NA
Bigrams5	84.50%	NA
Unigrams3	**89.50%**	NA
Bigrams3	84.50%	NA

Table 4: Results of each model on the testing data, best performances in boldface

4 Discussion

4.1 Comparison with Pang et al's results

Interestingly, every feature set I tested did better than what was reported by Pang, et al. The exception was the baseline, which only performed at 62% correct, rather than the higher 64% reported by "Human 2." However, it did fall in the range of the two humans, 58% - 64% .

It could be the case that it is the training and testing sets themselves that account for the difference, rather than anything about the way the features were implemented. However, given that the baseline was about the same, and given that all feature sets did better, it seems at least as likely that trimming the number of distinct features down to only those features which occur with some frequency accounts for the difference. From the validation testing, it was apparent that cutting down the number of features too much results in lower accuracy. However, in the actual testing, when the model was trained on a larger training set (training + validation sets), the models that trimmed the features to only more frequently occurring ones did better. The difference between validation and testing can be accounted for by remembering that items which are of frequency 3 or higher are likely to be of higher frequency (ie, higher than 3) in a larger data set, particularly when the smaller data set is a subset of the larger one.

4.2 Additional features

Firstly, choosing values for N (unigrams of frequency N or more) and M (adjectives of frequency M or more) depends on the size of the training set. The larger the training set, the higher N and M should be. This is because unigrams and adjectives that are of a particular frequency in a smaller training set are likely to be even more frequent in a larger training set. Including too many features lowers the performance of the model - hence why

Unigrams5 did so much better when training on both the training set and the validation sets, prior to testing on the test set.

Another type of feature might also help with misclassified reviews: tendency to use the conditional perfect (eg, ”would have been”) in negative reviews. The frequency of the conditional perfect could be a good indication of a negative review.

More broadly, verb tense could be useful in classifying reviews.

Also, there is a tendency in negative reviews to have fairly positive words at the beginning, setting up the author’s expectations, but then to explain only in the final part of the review how the film was a disappointment. Features could be taken only from the last part of the review. In both positive and negative reviews, in fact, the author’s main point is often most strongly made in the final paragraph or even last one or two sentences. This may be more relevant for classifying a review than any other part of it; thus, position of features (unigrams or bigrams, etc.) could be taken into account.

One positive review that was misclassified was likely misclassified due to the content of the film, which was a villain-hero scenario. The author, in explaining the plot, talked about the villain quite a bit - the ”bad” guy - which added a lot of negative words to the review. One could potentially take the subject of the film into account, or include it as a feature in addition to other features. This would require a lot of training data, however, for the classifier to learn to expect certain kinds of words for certain genres of film, and would run into the problem of sparse data.