XAVIER FALCO
RAFI WITTEN
ROBIN ZHOU

REPORT

SENTIMENT AND OBJECTIVITY CLASSIFICATION

CS 224 N
FINAL PROJECT

TASK

CLASSIFICATION

GOAL

The goal for this assignment was to develop a technique for text classification that is robust enough to be applied across a variety of different classification challenges. The task that we planned to train on was sentiment classification (determining if a review is positive or negative), but we were interested in the question of if the same techniques that worked on sentiment classification could work on objectivity classification, or determining if a sentence describes an objective or subjective claim. The hope is that we could determine which techniques make specific use of particular properties of the languages and which are more general and could be used across arbitrary classification tasks. The ultimate goal was to develop techniques that can train for any sentiment classification task when given a suitable dataset.

TECHNIQUES PROPOSED

Given the goal of trying to create a generic classification algorithm, the question comes in what techniques to implement. Ultimately, the techniques we settled upon were Naïve Bayes, K-nearest neighbors classification and a generic machine learning technique, SVM. The plan was to try to each of them on the first task, sentiment classification and see if after optimizing them for that task, they performed satisfactorily on the second task, objectivity classification.

USE OF CORPUS

The corpuses that we used were from source [1]. We used a combination of polarity_dataset2.0 and sentence polarity dataset v1.0 in sentiment classification. The polarity dataset is made exclusively from movie reviews that have been hand-labeled. The two datasets differs because one is still organized by review while the other is a sort of slop of sentiment sentences extracted from reviews. For objectivity classification we used subjectivity dataset 1.0. This second dataset was rather small, only 5000 sentences of each type, but it proved sufficient for our purposes. Again, the dataset has no clear unifying theme and more seems to be linked by whatever the author deemed interesting.

NAÏVE BAYES

MAXIMUM LIKELIHOOD

TRIGRAM WITH KATZ-BACKING OFF AND ABSOLUTE DISCOUNTING

ALGORITHM

Naïve Bayes is the method in which we train two different language models, one on each of the two tasks at hand. Then, to classify a text, the text's likelihood is determined under the two language models. The text is then classified into the corpus that gave it the larger probability. The technique is rather straightforward, but it theoretically produces the optimal result assuming that the a priori probability of the two options (or theoretically n options) is equal. To see this:

P(Text && Classification n) = P(Text | Classification n) P(Classification n)

If all of our original P(Classification 1)'s are constant, then the highest probability for the text will be given by choosing the highest P(Text | Classification n) or by choosing the classification that gives us the highest probability of the text. In theory, this framework of the noisy channel model is the best possible, but in practice it is limited by our ability to create accurate language models. Instead, for text classification it is common to switch to a less natural model that is more powerful and is this able to make better use of the data. Language models, even trigram models, are deeply flawed by their reliance on only immediate preceding text and these flaws are what opens up the possibility of more generic machine learning techniques to surpass them. Regardless, Naïve Bayes is a very natural way of handling the problem and is a good baseline for a classification model.

MODEL

The model that was used was Katz Backing-off mixed with absolute discounting, built on top of a trigram model. Its direct relevance in our discussion of Naïve Bayes is rather minute, except that the performance of Naïve Bayes has a lot to do with how much smoothing is done.

$$P_{bo}(w_i|w_{i-n+1}\cdots w_{i-1}) = \begin{cases} d_{w_{i-n+1}\cdots w_{i-1}} \frac{C(w_{i-n+1}\ldots w_i)}{C(w_{i-n+1}\cdots w_{i-1})} & \text{if } C(w_{i-n+1}\cdots w_i) > k \\ \alpha_{w_{i-n+1}\cdots w_{i-1}} P_{bo}(w_i|w_{i-n+2}\cdots w_{i-1}) & \text{otherwise} \end{cases}$$

$$\beta_{w_{i-n+1}\cdots w_{i-1}} = 1 - \sum_{\{w_i:C(w_{i-n+1}\cdots w_n)>0\}} d_{w_{i-n+1}\cdots w_{i-1}} \frac{C(w_{i-n+1}\ldots w_i)}{C(w_{i-n+1}\cdots w_{i-1})}$$

$$\alpha_{w_{i-n+1}\cdots w_{i-1}} = \frac{\beta(w_{i-1}\cdots w_{i-n+1})}{\sum_{\{w_i:C(w_{i-n+1}\cdots w_n)=0\}} P(w_i|w_{i-n+2}\cdots w_{i-1})}$$

These equations are pulled from Wikipedia and completely define Katz-backing off. Because our model of Katz-backing off was based on absolute discounting, the constant d can be thought of as being the subtraction of probability mass from events based on their count. The constant d was optimized at great lengths, but ultimately it must be considered to be of little consequence. It was interesting that on the test data set for sentiment classification d was chosen to be equivalent to subtracting .15 from events that occurred once and .3 from events that occurred more than once, far less than the standard constants of .55 and .75 used for creating language models.

RESULTS

The data for sentiment analysis was broken into 3 chunks. The bulk was left to serve as the corpus, but 40 reviews of each type were excluded. Of these, 20 of each type were used for the numerical optimization previously mentioned and 20 of each type were used to get a final estimate. Ultimately 28/40 were correctly classified on the optimization data set while 35/40 where correctly classified on the validate data set. However, this does not exactly fill us with confidence when we examine the data – the log probabilities for the optimized data are presented below (which could be considered as the validate set).

	"Positive Sentiment" Score	"Negative Sentiment" Score	Correctly Classified
Pos1	-1700	-1726	1
Pos2	-2081	-2166	1
Pos3	-2045	-2103	1
Pos4	-1556	-1628	1
Pos5	-1311	-1298	0
Pos6	-1665	-1724	1
Pos7	-1500	-1511	1
Pos8	-1437	-1442	1
Pos9	-1315	-1354	1
Pos10	-1653	-1626	0
Pos11	-1738	-1737	0
Pos12	-1293	-1279	0
Pos13	-1442	-1437	0
Pos14	-1363	-1413	1
Pos15	-1605	-1675	1
Pos16	-1158	-1207	1
Pos17	-1257	-1301	1
Pos18	-1889	-1993	1
Pos19	-1773	-1813	1
Pos20	-1859	-1774	0

Neg1	-2300	-2297	1
Neg2	-1650	-1661	0
Neg3	-2188	-2073	1
Neg4	-2151	-2074	1
Neg5	-1787	-1784	1
Neg6	-2197	-2084	1
Neg7	-1789	-1751	1
Neg8	-2151	-2052	1
Neg9	-2161	-2127	1
Neg10	-1736	-1680	1
Neg11	-2543	-2418	1
Neg12	-2405	-2358	1
Neg13	-1884	-2013	0
Neg14	-2477	-2549	0
Neg15	-2529	-2533	0
Neg16	-1761	-1736	1
Neg17	-1884	-1932	0
Neg18	-2125	-2150	0
Neg19	-2071	-1986	1
Neg20	-2296	-2283	1
			28

Although we get 28 correct out of 40, there are many close calls (consider Neg5) that we got right, that these results seem like noise. It is worth noting that although the results for Neg5 only differ by 3, that means that our estimated probabilities differ by a factor of 10^3.

Its not clear what to make of the fact that we did worse on the validate data set than we did on the test data. However, to do a Bayesian analysis of our models, we see that if we combine the results for the two we get:

((80!)/(63!*17!))/(2^80) = 8.38*10^-8

Therefore, to get more than 28+35=63 right just by chance would be less than 8.37*10^-8 *17<10^-6. This number only goes up if we just consider the validate data set, to under 10^-7. Therefore, we have very good statistical evidence that this technique, at p = .05 or really any reasonable p-value, has predictive ability. Our estimate for its accuracy is between 63/80 and 35/40 or .7875 and .875; it is difficult situation because of validation data results was inferior to our test results.

Therefore, we have rather strong predictive ability at the task of classifying movie reviews based on sentiment. How do we do at the second task? For it we get 152/200 sentences tested correct (using the same constants as in the previous task). The 152/200 statistic is over a 75% correct rate and is similarly statistically significant.

ERROR ANALYSIS

Lets examine some sentences that we got wrong from the objectively dataset:

obj20: in the end , debby , beth , and virginia find , if not the relationships of their dreams , peace with each other and within themselves .

subj72: the film isn't especially dynamic , but it brims with insightful , poignant memories from survivors .

Here are some interesting ones that we got right:

obj51: he tries to ignor it , but he later finds out on the news that the daughter of a senator has been kidnapped and is being help ransom for $15m .

subj50: minority report is exactly what the title indicates , a report .

First, note that the dataset is low quality. There is a typo ("ignor") in objective 51 along with in many other sentences from the corpus. Consider the differences between obj20 and subj50. Its not immediately clear what makes one objective and the other subjective. Why is the narrator saying that the protagonists have certain different feelings any different from saying that a movie is a "report"?

It raises the question of whether the sentence "minority report is exactly what the title indicates , a report . " would be listed as subjective if it was changed to "A *SPECIAL REPORT TO THE PRESIDENT* is exactly what the title indicates, a report ." This new claim is unambiguously true, but not in a way that the language model can capture. The classification task of subjective vs. objective is inherently ambiguous; since the task is vague to a human, it seems over-confident to think that it can be solved perfectly with machine-learning, although Naïve Bayes does rather well.

It is disappointing that subj72 was incorrectly classified. The words 'dynamic', 'insightful' and 'poignant' all seem to indicate subjectivity. I would suspect that this error is linked to our rather sparse training data set rather than anything structural about our technique. No doubt the words insightful and poignant correlate very highly with subjectivity – it is difficult to imagine that a sentence using those adjectives could ever by objective. However, subjectivity objectivity datasets need to be hand-classified so they tend to be rather small.

The model is also plagued by sparsity because of the widespread use of names. "Debby, Beth and Virginia" serves just to confuse the language model. To deal with this, one technique would be to create a script that goes through, takes unambiguously male and female names and replaces them by he or she accordingly (this would not necessarily work on Virginia which can be used as either a state or a girl's name).

FUTURE TECHNIQUES

Moreover, although our approach to calculating log probabilities across the space of possible movie reviews seems intuitive, further analysis suggests that we did not quite have a probability distribution. Our language model creates a probability distribution across sentences, but consider the two possible reviews: "sentence a" and "sentence a|sentence b" so the second has two sentences in

the corpus. If we sum across the probability of all of these (for all a), we see that the method to get the probability of a series of sentences is inaccurate. If the sum across all a is one, the sum across all b and a must be greater. Its not clear how to fix this without incorporating something similar to the END token for sentences. However, I think that this bug may not lead to real problems because likely the importance of this factor is the same across both models, so it the effects cancel out when comparing the probability of a text under two language models. Regardless, further research into a END token for corpuses would be interesting and would allow us to capture the idea of corpuses being of different probability.

The problem with Naïve Bayes is that language models are not that good. Because of their short length they are useless on capturing sense in longer clauses. To make Naïve Bayes become a very strong method would require developing new ways of thinking about language models – improvements in our technique of Naïve Bayes are not enough to improve the technique by too much. Perhaps the language model could take into account greater context (all the words that had come before it) and slightly reweight probabilities accordingly. Each previous word could take a certain amount and give a certain amount of probability mass to and from each continuation, making sure to take as much as it gave total. That would preserve the vital feature that the probability across the space sums to 1. However, since it becomes more difficult to preserve a probability distribution as the language models become more complicated, we instead turn to more generic machine learning techniques that are more powerful, but are less natural because they less directly take advantage of the structure of sentences.

K-NEAREST NEIGHBOR

SUPERVISED MACHINE LEARNING

K-NN ALGORITHM WITH DISTINCT DISTANCE FUNCTIONS

The algorithm implemented consisted of a k-nearest neighbor supervised learning algorithm. The algorithm works by assigning some metric which evaluates the degree of similarity of two language models. For each review, we can then assign and train a language model. When trying to evaluate the sentiment of an arbitrary review, the algorithm utilizes the metric to identify the k most similar reviews. Assuming k is an odd number, it then tabulates the majority sentiment, and assigns that sentiment to the arbitrary review.

A pseudo code for the algorithm is as follows:

```
ArrayList<LanguageModel> models = new ArrayList<LanguageModel>
For(m: m is a review in the training corpus) {
    Create a language model L
    Train L on m
    Models.add(L)
}
ConfigureMetric()
For(m: m is a review in the testing corpus) {
    ArrayList<int> nearestNeighbors = new ArrayList<int>
    For(t = 0; t < models.size(); t++) {
        If(nearestNeighbors.size() < k) nearestNeighbors.add(t)
        Else {
                int index = findFarthestNeighborInArray(nearestNeighbors)
            if(metric(models.get(t), m) < metric(models.get(nearestNeighbors.get(index))), m)
                nearestNeighbors.add(index, t)
                }
    }
    Store(m, majoritySentiment(nearestNeighbors))
}
```

The language models used were simple unigram models. The smoothing technique added was pretending that one unknown word had been seen, so that the word-probability returned was count(word) / (total + 1.0). Furthermore, the language models were modified so that they contained a field indicating the sentiment of the review the model was trained on, if known – either "positive" or "negative".

IMPROVEMENTS MADE

The initial metric used for calculating the "distance" between two models in evaluating closest neighbor consisted of summing the absolute difference between the word probabilities returned for all words across either corpus, i.e.

```
private int returnDistance(UnigramModel model1, UnigramModel model2) {
```

```
        double total = 0;
        for(String word: combinedVocabulary(model1, model2)) {
            total += Math.abs(model1.getWordProbability(word) -
model2.getWordProbability(word));
            }
    Return total;
}
```

The second implemented metric attempted to balance out the difference in word probabilities with difference in corpus size, so that the new metric only calculated the absolute difference between the word probabilities of all the words found in the first model passed in (presumably the model in the "testing corpus" and not the "training corpus"). This helped removed some of the problems associated in the initial implementation which unnecessarily handicapped pairings of small models against larger models

```
        private int returnDistance(UnigramModel model1, UnigramModel model2) {
        double total = 0;
        for(String word: vocabulary(model1)) {
            total += Math.abs(model1.getWordProbability(word) -
model2.getWordProbability(word));
            }
    Return total;
}
```

The third implementation attempted to refine the attempts of the second attempt to remediate size problems by introducing a list of "stopwords" – common English word that were common in all language models and therefore could not help deducing whether two models were "nearer". These words included 'the,' 'is,' 'a,' 'an,' and other words, and was taken from http://www.dcs.gla.ac.uk/idom/ir_resources/linguistic_utils/stop_words.

```
private int returnDistance(UnigramModel model1, UnigramModel model2) {
        double total = 0;
        for(String word: vocabulary(model1)) {
            if(!stopwords.contains(word))
```

```
                total += Math.abs(model1.getWordProbability(word) -
model2.getWordProbability(word));
                }
        Return total;
}
```

Where stopwords is an array containing all the stop words.

The most useful implementation, however, consisted of tabulating the 100 most 'polarized' words and summing the absolute difference in word probabilities between the two models on these words. 'Polarized' words in this context denote words that have the highest absolute difference between the sum of the probability given by all positive models and the sum of the probability given by all negative models. This new metric is thus of two folds, a train function which computes the 100 most polarized words and a metric function.

```
Private void train() {
    ArrayList<String> tempwords = new ArrayList<String>();
    ArrayList<double> positive = new ArrayList<double>();
    ArrayList<double> negative = new ArrayList<double>();
    For(String word: totalVocabulary) {
        Tempwords.add(word);
        Positive.add(0.0);
        Negative.add(0.0);
        Int I = totalVocabular.indexOf(word);
        For(UnigramModel model: models) {
            If(model.getRating() == "positive) positive.add(I, positive.get(i) +
model.getWordProbability(word));
            Else negative.add(I, negative.get(i) + model.getWordProbability(word));
        }
    }
    Arrange(tempwords, positive, negative);
    }
Private void arrange(ArrayList<String> words, ArrayList<double> pos, ArrayList<double>
neg) {
    int index = words.indexOf(word);
```

```
        For(String word: words) {
            Double difference = Math.abs(pos.get(index) – neg.get(index));
            If(top100.size() < 100) {
                Top100.add(words);
                Diff.add(difference);
            } else {
                Int least = leastEntry(diff);
                If(diff.get(index) < difference) {
                    Top100.add(least, word);
                    Diff.add(least, difference);
                }
            }
        }
    }
private int returnDistance(UnigramModel model1, UnigramModel model2) {
            double total = 0;
            for(String word: top100) {
                if(!stopwords.contains(word))
                total += Math.abs(model1.getWordProbability(word) -
model2.getWordProbability(word));
                }
        Return total;
}
```

TESTING AND RESULTS

The testing involved running it on one twentieth of the data set. Two different data sources were used. The first consisted of movie reviews, namely 1000 positive movie reviews and a 1000 negative reviews, and the second consisted of 5000 objective sentences and 5000 subjective sentences. The following results were observed:

Metric used	Positive correct	Negative correct	Objective correct	Subjective correct
Simple absolute difference (1)	52 %	50 %	48 %	54 %
Restricted	52 %	52 %	52 %	54 %

absolute difference (2)				
Stop words (3)	60 %	62 %	56 %	52 %
Most polarized words (4)	74 %	68 %	54 %	48 %

The results indicate that the first two methods implemented did little better than blind guessing and were blatantly inefficient. The third method implemented, that of stop words, was successful in a statistically significant way. Indeed, it allowed for an average of 60 percent correctness on the positive /negative set and 54 percent on the objective / subjective set.

The explanation for the success of stop words reposes mainly on the fact that ignoring high-count words such as the, is, and an allows the metric to stop favoring pairings between reviews of similar sizes. This is because the length of reviews is in a range which leaves a significant difference in the probability assigned to such words as the, is, and an between short and lengthy reviews (lengthy reviews tend to have smaller probabilities assigned to such common English words because one-sentence reviews will most likely be guaranteed to contain the, is, or an). Thus, the metric is allowed to evaluate similarity based on more characteristic words i.e. non-super common words.

Finally, the last metric used enjoyed mixed success. It was very effective in evaluating positive and negative reviews, but of little help in evaluating objective/subjective sentences. An analysis into the top most polarized words computed for both sets helps explain why:

Positive Negative:

…

word: creativeness positive 747.9047506493644negative 660.8296539519479

word: beautiful' positive 750.9065028659206 negative 663.4791328676945

word: fishy positive 254.07137237411825 negative 287.04032738845586

word: discord positive 159.96397993867217 negative 180.41225498919016

word: shunned positive 162.6134588544207 negative 183.4140072057479

word: first-rate positive 159.39032077212275 negative 141.4036476836945

word: seriousness positive 41.88485732627808 negative 46.69379409311886

word: dangerous positive 43.21794947876256 negative 48.19769356904113

word: financially-strapped positive 28.620456554857082 negative 31.675495892832842

word: polite positive 29.945196012731056 negative 33.17637200111157

word: dynamic positive 31.270699412545444 negative 34.67799717306064

…

Objective Subjective:

…

word: home objective 9990.65314631208 subjective 10506.12381035932

word: copious objective 10525.551608476244 subjective 10009.115239190372

word: live objective 9289.093616936932 subjective 9767.86748191623

word: foreign objective 9399.866174206692 subjective 9884.43427061777

word: pursuit objective 7610.832273668757 subjective 7239.443439035072

word: an objective 7258.017573350542 subjective 7630.305618344435

word: popular objective 1158.1296113176563 subjective 1214.6604322438995

word: hasn't objective 1472.6809595165093 subjective 1544.9663335650705

word: three objective 380.8340637443656 subjective 402.4024873840851

…

The words most polarized for the positive negative are strong indicators of positiveness or negativeness and are thus helpful in determining the category of a particular review.

On the other hand, the shortness of the training corpus for each language model in the objective subjective category yields an arbitrarily-random set of words which do not seem correlated to objective and subjective. We conclude that the fourth metric, given the size of the training for each language model, was thus unhelpful in determining the similarity of two reviews.

IMPROVEMENTS

The most influential part of the algorithm in determining success rate is the metric used. Choosing an appropriate metric involves making a decision as to the trade-off of training corpus size and reliability of the data. Thus, a language model trained on a single review is not very reliable, but this process allows for a large number of language models – 'neighbors' – with which to work. This was the rationale throughout implementation and testing – deficiencies in the language models were ultimately outweighed by the sheer number of neighbors to work with. It might perhaps improve prediction, however, if some of the reviews under the same category are combined to form a single language model. This would remedy some situations in which a review to be tested falls solidly under one category, but is only fractionally similar to some models under that category – whereas it might be closer by word choice to a few reviews of the opposite category.

Similarly, the language model utilized was a unigram model which assigned to each unknown word a single count. This returned probability count(x) / (total + 1.0) for the probability of all words

x it had seen and probability 1.0 / (total + 1.0) for all unknown words. Better smoothing, or different language models techniques, could theoretically influence the results of the prediction. This option was not pursued because drastic changes in the success of prediction is highly improbable given the size of the corpus each language model is trained on, but it is notionally possible that different language models have some effect in the success of the prediction. Certainly, smoothing techniques are important here given the influence of unknown words (most pairs of word probabilities contain at least one unknown word probability). Given this, a different smoothing technique seems like a reasonable improvement to the algorithm.

A final improvement consists of implementing some time-saving algorithms. Currently, it takes about four hours for the data sets to be processed and evaluated on the last metric implemented. While it is obvious that this metric will take significantly longer to run, some improvements may be implemented to render faster speed.

LITERATURE

K-nearest-neighbor (kNN) classification is one of the most fundamental and simple classification methods and should be one of the first choices for a classification study when there is little or no prior knowledge about the distribution of the data. K-nearest-neighbor classification was developed from the need to perform discriminant analysis when reliable parametric estimates of probability densities are unknown or difficult to determine. Indeed, the following studies were used as background in implementing the k-nearest neighbor algorithm:

A k-nearest neighbor classification rule based on Dempster-Shafer theory
Denoeux, T.;
Systems, Man and Cybernetics, IEEE Transactions on
Volume 25, Issue 5, May 1995 Page(s):804 - 813

The problem of classifying an unseen pattern on the basis of its nearest neighbors in a recorded data set is addressed from the point of view of Dempster-Shafer theory. Each neighbor of a sample to be classified is considered as an item of evidence that supports certain hypotheses regarding the class membership of that pattern. The degree of support is defined as a function of the distance between the two vectors. The evidence of the k nearest neighbors is then pooled by means of Dempster's rule of combination. This approach provides a global treatment of such issues as ambiguity and distance rejection, and imperfect knowledge regarding the class membership of training patterns. The

effectiveness of this classification scheme as compared to the voting and distance-weighted k-NN procedures is demonstrated using several sets of simulated and real-world data

Nearest neighbor pattern classification
Cover, T.; Hart, P.;
Information Theory, IEEE Transactions on
Volume 13, Issue 1, Jan 1967 Page(s):21 - 27
Abstract:

The nearest neighbor decision rule assigns to an unclassified sample point the classification of the nearest of a set of previously classified points. This rule is independent of the underlying joint distribution on the sample points and their classifications, and hence the probability of errorRof such a rule must be at least as great as the Bayes probability of errorR^{ast}--the minimum probability of error over all decision rules taking underlying probability structure into account. However, in a large sample analysis, we will show in theM-category case thatR^{ast} leq R leq R^{ast}(2 -- MR^{ast}/(M-1)), where these bounds are the tightest possible, for all suitably smooth underlying distributions. Thus for any number of categories, the probability of error of the nearest neighbor rule is bounded above by twice the Bayes probability of error. In this sense, it may be said that half the classification information in an infinite sample set is contained in the nearest neighbor.

SUPPORT VECTOR MACHINE

MAXIMUM MARGIN CLASSIFIER

ALGORITHM

Briefly, the SVM approach takes a set $\mathbf{D}$ of training instances $(\mathbf{x_i}, c_i)$, where $\mathbf{x_i}$ is a real-valued feature vector of length p and c_i is the true classification for that instance. We want to find a pair of parallel hyperplanes $\{\mathbf{w} \cdot \mathbf{x} - b = 1, \mathbf{w} \cdot \mathbf{x} - b = -1\}$ that maximally separates positive and negative-classified instances, where $\mathbf{w}$ is a vector normal to the hyperplane and $\mathbf{x}$ is the set of points comprising the hyperplane.

As SVM is a common and well-specified technique, we refrain from restating its various forms here and refer interested readers to the Wikipedia article:
http://en.wikipedia.org/wiki/Support_vector_machine

IMPLEMENTATION

We selected an open-source machine learning library, the Java Machine Learning Library at java-ml.sourceforge.net.

Since Pang et al. 2002 (http://www.cs.cornell.edu/home/llee/papers/sentiment.pdf) reported good SVM performance when creating feature vectors using unigram features, we adapted elements of their best-performing strategies in designing classifiers for the sentiment and subjectivity tasks. In the general spirit of their method, for sentiment analysis, we counted the number of unique unigrams present in the dataset, and represented each review as a sparsely encoded feature vector, where a unigram feature had a value of 1 if it was present in the review, and 0 otherwise. A similar procedure was followed for subjectivity analysis – the number of unique unigrams was counted, and each sentence in the dataset was represented as a sparsely encoded feature vector of unigrams. A sparse encoding was the obvious choice for these problems, given the number of total features and the relatively few features occurring in each data example.

RESULTS

Number of unique unigrams in polarity_dataset2.0: 50921.
Number of unique unigrams in subjectivity dataset 1.0: 23919

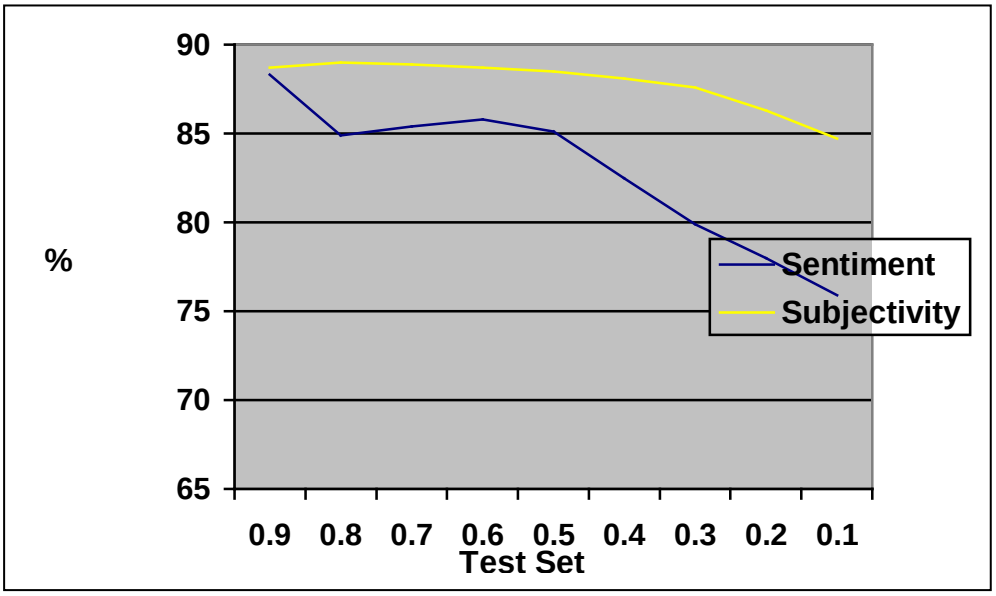
We selected an open-source machine learning library, the Java Machine Learning Library at java-ml.sourceforge.net.

Table of results for the sentiment task:

n = 2000 (1000+, 1000-)				
Train	Test	Correct	Incorrect	% Correctly Classified
0.9	0.1	175	23	88.3
0.8	0.2	338	60	84.9
0.7	0.3	511	87	85.4
0.6	0.4	685	113	85.8
0.5	0.5	850	148	85.1
0.4	0.6	989	209	82.5
0.3	0.7	1117	281	79.9
0.2	0.8	1247	351	78.0
0.1	0.9	1365	433	75.9

Table of results for the subjectivity task:

n = 10000 (5000 objective, 5000 subjective)				
Train	Test	Correct	Incorrect	% Correctly Classified
0.9	0.1	886	112	88.7
0.8	0.2	1779	219	89.0
0.7	0.3	2666	332	88.9
0.6	0.4	3549	449	88.7
0.5	0.5	4427	571	88.5
0.4	0.6	5286	712	88.1
0.3	0.7	6133	865	87.6
0.2	0.8	6905	1093	86.3
0.1	0.9	7628	1370	84.7



The results compare very favorably to the results obtained by Pang et al. 2002. Their best SVM result on the sentiment task was 82.9% - ours was 88.3%. The SVM classifier maintains robust performance until the training set outnumbers the test set, at which point the sparsity of training examples makes the constructed classifier unable to effectively classify test examples with many features unseen in training. Similar performance vs. set size trends were observed with the subjectivity task, providing confirmation of the effectiveness of our approach on wholly different data. Due to the increased number of training examples, one does not observe a large dropoff in performance, lending evidence to the notion that it is the presence of a sufficiently representative training set, not the ratio of training to test set sizes, that is the more important factor in performance. If 30% of the dataset is sufficient to construct sufficiently discriminatory hyperplanes, classifier performance will be good despite the relatively unorthodox size of the test set.

Here we recount, for each task, a number of examples that the classifier classified incorrectly. Each of these examples was misclassified with the classifier trained on 90% of the dataset.

The sentiment task:

True classification was **POSITIVE**, misclassified as **NEGATIVE**:

cv911_20260.txt:

usually when a blockbuster comes out , it's loaded with effects , stars , bad scripts , and plenty of action . mystery men may contain an all-star cast , and efects , but the clever script and characters are what really works , which is rare to see this year .
things go a little haywire , when the sinister casanova frankenstein (geoffrey rush) is released into the city , where he captures captain amazing , and plans to wreak havoc upon champion city .
well , the trio decide to take matters in their own hands , by saving the city , but first they need some assitance . this is where the film takes a turn for the better .
... the whole premise is rather ridiculous , but packs a few punches to keep interest . ...
for one , the film is considerably clever . it literally pokes fun at super hero films , like batman and robin , superman etc . in fact , many scenes are similar to batman and robin , including the opening sequence , only altered in a humorous and superior way .
a part of the cleverness comes from the cast .
sometimes a film with such talent is overblown , but the acting is what keeps it alive here .
while azaria and macy were enteratining , 2 characters really stood out . one was paul reuben .
no matter how disgusting or revolting " mr . splein " may be , you still can't help but laugh .
it' so incredibly moronic , it's just a riot watching reuben relieve himself of bodily functions .
janeane garafalo also was an interesting character .
she seemed to be the most outgoing and convincing character on screem , due to her enthusiasm , that kept the film flowing . men is worth seeing alone , for those 2 troubled heroes .
on the downside , a few of the heroes and especially the villain never really lift off .
kel mitchell and geoffrey rush , were both utterly useless . their parts were so limited , they'd be lucky at all to be on screen for more than 20 minutes . ben stiller too was wasted , mostly because of his unlikeable power and dialogue . none of these characters get a rise out of anybody , but happily they are lost in the charming flow of the film . as far as the budget goes , it was wisely spent on the cast , not the effects . while the set designs and action all look nice , i'm glad there was a seperate aspect , that the film focused on , and for that i applaud . slow at times , and rather pointless , mystery men still delivers . it forgets about money making , because it's not likely to make a bundle like it's proceeders , and that's what works . stupid ? maybe , but for once i'm not disappointed . no one expected an intelligent film , but you get a film thats wit captures your attention and makes you forget this miserable year .

Comment:
As a quick perusal of this review will show, the tone adopted by this reviewer produces a set of unigrams that are conductive to a negative rating. The overall message seems to be one of grudging approval, leavened with plenty of approbation. Words such as “disgusting,” “revolting,” “bad,” “disappointed” and “miserable” are tossed about but are negated fairly effectively, a rhetorical habit a human reader would catch, but our unigram-feature SVM is fooled.

True classification was **NEGATIVE**, misclassified as **POSITIVE**:

cv975_11920.txt:

however , he still smokes his lucky strikes , detests all forms of authority , and kills at a whim .
beyond that , the updated film retains little or no resemblance to the original pulpy page-turner by spillane , probably the most infamous and often reviled of all mystery writers .
the movie starts off with a bang : a howler of an opening credits sequence that is a cheap steal from the james bond series , complete with cheesy graphics and an overbearing jazz score by bill conti (" rocky ") .
after that , the movie and the book begin the same , with the murder of jack williams (frederick downs) , a one-armed detective and hammer's **best** friend .
hammer declares that he will seek vengeance for jack's death , and with the help of his devoted secretary , the blond and shapely velda (laurene landon) , and the alternately friendly / antagonistic police chief pat chambers (paul sorvino) , he is immediately on the killer's trail .
here the movie splits completely from the book , and dives into a convoluted and improbable tale of government conspiracy and mind control tactics involving the mafia , the cia , one of hammer's vietnam vet buddies , and a kinky sex clinic .
many of the same characters from the book appear in the movie , but they take on slightly different roles .
for instance , charles kalecki (alan king) , a numbers runner and narcotics dealer in the book , turns into a suave mob boss .
and , more importantly , hammer's suspicious love interest , charlotte bennett (barbara carrera) , morphs from a run-of-the-mill psychiatrist into the coordinator and founder of the sex clinic .
" i , the jury " is one of several cinematic renditions of spillane's books (including a 1953 version which was made in 3-d) , but this film differs from those earlier versions in one major way : it includes all of the sex and violence spillane wrote about that could never be given screen treatment due to hollywood's production code .
although this takes the 1982 version of " i , the jury " closer to the core of the original subject matter , it is in this aspect that the film received the most criticism , because it took this new license to extremes that many argued surpassed what was in the book .
rest assured , the movie not only includes a great deal of nudity , but it is thoroughly violent , especially toward women .
it features one woman having her neck slashed , a set of twins forced to strip before being stabbed to death by a psychotic sexual deviant programmed by the cia (judson scott) , and another woman shot point-blank in the belly by hammer himself .
no one would deny that spillane's writing has a definite misogynistic nature , but the movie seems to take it a step further by giving it such **glorious** screen treatment ; its constant equation of sex and violence , much of which is played with the intention of being erotic , is quite unsettling .
it's no surprise that the movie , like the book , fades to black with a dead woman on the floor .
" i , the jury " had a troubled production and was not well-supported by the studio that made it , which is one explanation why it didn't do well in theaters and many people have forgotten that it was ever made .
the script was written by larry cohen , who is best known for his creatively cheesy but nonetheless **effective** monster movies , like " it's alive " (1974) and its two sequels , " q " (1981) , and " the stuff " (1985) .
cohen wrote the script thinking he was going to helm the project as well , but he was yanked from the director's chair after only a week's worth of shooting because he was already $100 , 000 over budget .
heffron was obviously brought in not for his **talent** , but because he could make the movie **rapidly and efficiently .**
cohen had personal interest in the updated version of hammer , but heffron has none .

he shot the movie quickly and clumsily , and although some scenes ring **true** , most of them are flat , trite , and invariably dull .
the movie features numerous car chases , shoot-outs , and stunts , but heffron's background in television is the **dominant** tone ; despite the graphic violence and full-frontal nudity , " i , the jury , " takes on the air of a made-for-tv quickie , with no real **punch or depth** .
(review continues on and has been truncated here)

Comment: The length of this review (more than half of the text was redacted in the version seen in this paper) is difficult for our SVM to deal with, as a longer text means a fuller feature vector. Rather than remain short and sweet with a few words strongly indicative of sentiment, this review, by sheer force of linguistic/expressive necessity, includes many examples of both positive and negative words that are negated, emphasized etc. to form semantic-level structures. Along with the largely featureless ocean of plot summary, this makes the review extremely difficult to classify with only unigrams. Highlighted are the unigrams which may have nudged this review into a positive classification – many of them are negated.

The subjectivity task:

A number of "misclassified" examples, we felt, could conceivably be placed in either category by a reasonable human observer. We include them (without further analysis) to demonstrate that the dataset is not completely "clean." This may not have too much of an effect on classifier performance if the creators of the dataset were evenhanded in their treatment of ambiguous examples, and there aren't too many of them. Suppose an ambiguous example was classified **T** 50% of the time and **F** 50% of the time by the classifier, and a set of human classifiers produces the same 50/50 judgment call. The expected performance of our classifier over a group of these ambiguous examples is 50%, so the presence of ambiguous examples serves as an effective upper bound on the correctness % of a classifier, no matter how good.

True classification **SUBJECTIVE**, misclassified as **OBJECTIVE**:

Clearly misclassified examples:

- this is a story that zings all the way through with originality , humour and pathos .

Comment: "zings," "humour" and "pathos" are not common features in a dataset of predominantly American usage featuring everyday language, which may contribute to the difficulty of prediction. A human unaware of the definition of "zings" may also struggle.

- iwai creates yuichi's world as much through disembodied moments of sight and sound as through action , building to a surprising stab of melancholy .

Comment: "surprising" is possibly the only subjective word in the sentence, and as a result, the sentence may simply have been treated by the classifier as closer to the objective data points.

- about as big a crowdpleaser as they possibly come .

Comment: "crowdpleaser" is a rare portmanteau, and "as they possibly come," while conveying subjectivity information, is not sufficiently well captured by our unigram features to contribute sufficient predictive power.

- despite hoffman's best efforts , wilson remains a silent , lumpish cipher ; his encounters reveal nothing about who he is or who he was before .
Comment: "lumpish cipher" is rare, but "despite" and "nothing" potentially convey useful information. This one had us scratching our heads.

- what's next ? the porky's revenge : ultimate edition ?
Comment: Higher semantic structures such as rhetorical questions (in this case conveying sarcasm, a fact which is never made explicit by any particular word) are not captured by unigram features, and are in fact poorly modeled by n-grams of any practical length.

- behind the glitz , hollywood is sordid and disgusting . quelle surprise !
Comment: Foreign phrase, may be rare in the dataset. However, "sordid" and "disgusting" would seem to offer predictive power, except that they may appear prominently in objectively-labeled plot summaries.

Ambiguous examples:
- the story the movie tells is of brian de palma's addiction to the junk-calorie suspense tropes that have all but ruined his career .
- a period story about a catholic boy who tries to help a jewish friend get into heaven by sending the audience straight to hell .

True classification **OBJECTIVE,** misclassified as **SUBJECTIVE**:

Clearly misclassified examples:
- he makes all efforts to bring ghisu back but does not succeed .
Comment:
This is a plot summary, and classification may have been confounded by the word "succeed."

- beautifully shot sequences episodically shift back and forth from the past to the present .
Comment:
The example is a bit ambiguous, as "beautifully shot" could be construed as a subjective statement, but the researchers chose to label the entire sentence as objective/factual.

- john quincy archibald's son michael collapses while playing baseball as a result of a heart failure .
Comment:
The words "failure" and "collapse" may feature in subjective reviews, enough to place this sentence closer to the subjective set of data points.

- an inexplicable crack in the pyrenees mountains provokes excitement and scientific curiosity .
Comment:
"Inexplicable," "excitement," "provokes" and "curiosity" are plausible anchor words for subjective sentences, although in this case the author is merely attempting to write an entertaining summary using active language.

Ambiguous examples:
- calvin's barbershop is filled with an eclectic and hilarious cast of characters that share their stories , jokes , trials and tribulations .
- unlike most reunion stories that climax with a cliche happy ending , daughter from danang is a real-life drama .
- disturbing for its unabashed honesty , our cast of characters both love and despise each other , their very actions acknowledging the pressures inherent in a tightly bonded peer group .
- what resulted is a stirring and emotional reflection of that day and his experience .
- not all the fights , just enough to keep the audience , and the money coming back .

FUTURE TECHNIQUES

Our preceding error analysis lends insight into the direction of future work to improve performance. Wikipedia helpfully alerted us to the "soft margin" SVM modification of Cortes and Vapnik, meant to mitigate the effects of mislabeled examples. This seems to be a promising strategy for handling problems such as the two we examined; even a human classifier will find that in terms of positive/negative sentiment and subjectivity, some statements will not skew strongly one way or the other. If the human labelers make what can clearly be termed a mistake, meaning they assign a label a conclusion that the majority of reasonable human observers will not make, the strength of the classifier will not be impacted as strongly.

It is reasonable to suspect that pre-chunking features may be of benefit. As an example, the phrase "heart failure" would, if chunked correctly, be treated as a neutral token rather than as a potentially subjective pair of features. Care must be taken in any chunking scheme to maintain the

information conveyed by any particular phrase. If chunking is too aggressive, the list of features shrinks dramatically and the SVM loses predictive power, since we are still not accounting for n-gram history in our model (though incorporating history and more aggressive chunking could be an interesting combination).

As an aside, one must note that for these tasks, where there is no well-defined "correct" classification for each data example except the human-classified label it is given, the more ambiguous data examples can potentially be classified in either category, and thus a "misclassification" cannot universally be considered a "mistake" by the classifier when researcher subjective judgment plays a nontrivial role in labeling the dataset. Happily, this only starts to become a significant factor affecting performance when the vast majority of "easy" examples are correctly labeled by a classifier, which was the case with our SVMs.

CROSSOVER ANALYSIS

DATA

The goal was to compare all of the results on the task of sentiment analysis. Here is each of the model's results:

	Naives Bayes	K Nearest Neighbors	SVM
Percent Correct	75%	71%	88%

Each of these is an estimate – the results differed for slightly different training techniques, and so on. Moreover, K Nearest Neighbors was handicapped because it was the most time intensive and was not able to make full use of the data set, so its accuracy was heavily dependent on how many cycles it was run for.

ANALYSIS

The result is that SVM is the strongest of the three techniques. An 88% accuracy rate on sentence classification is very strong. Moreover, SVM did the best on the movie review classification. These two facts in conjunction seem to support the theory that SVM is the best, most general technique. It should not surprise us that SVM is stronger than Naïve Bayes, because at worst it could simply model the trigrams themselves. However, because it was able to take advantage of more subtle features, SVM is a runaway winner. Comparing SVM to SNN is more difficult because they are such vastly different approaches to the problem. However, based on these calculations

along with the observation that SVM is a more time-efficient (although memory usage inefficient) technique, we suggest SVM as the best classification technique.

FUTURE WORK

Anecdotally, the three models did not seem have complete overlap in predictive ability. It seems reasonable to conclude that the models are capturing different properties of the sentences. KNN is determining which words are important; Naïve Bayes is interpreting properties of language itself (difference in writing structures in the two types of sentences), while SVM is finding a different way to use make use of the unigrams, similarly to KNN except without a stoplist.

These differences open up the possibility of some sort of lambda smoothing that incorporates all three techniques. This lambda smoothing could be based on lambda smoothing in language models which does the same task – combining different probabilities for events based on a priori understanding of the accuracy of the estimates. Since each capture different features of the data, there is reasonable hope that combining all of them could improve the analysis. The biggest concern is that SVM is so much more accurate and powerful than the other two models that it would render them irrelevant. Finding a technique to combine all three models seems like it could capture the best elements of each of the models.

GROUP PROJECT

XAVIER FALCO – RAFI WITTEN – ROBIN ZHOU

STATEMENT

RAFI WITTEN

I did the Naïve Bayes model, found the corpuses and handled all of the crossover analysis.

XAVIER FALCO

I completed the KNN portion of the assignment and assisted in the testing.

ROBIN ZHOU

I worked on the SVM portion, implementation, testing, and writeup, and compiled the final report and submission.

Sources:

[1] http://www.cs.cornell.edu/people/pabo/movie%2Dreview%2Ddata/

[2] Thumbs up? Sentiment Classification using Machine Learning Techniques, Bo Pang, Lillian Lee, Shivakumar Vaithyanathan, http://www.cs.cornell.edu/home/llee/papers/sentiment.pdf, *Proceedings of EMNLP 2002*