Good. Great that you attempted all the extra questions. I think you could motivate more about why you used the other statistical tests (chi-square, ANOVA, etc), and some of the figures have layout issues. Otherwise nice structure, good selection of tables and figures.

Learning from Data - Assignment 3 - Support Vector Machines and K-Means

Remko Boschker
master student of information science at the Rijks Universiteit Groningen
s1282603, r.boschker@student.rug.nl

Abstract

This study does unsupervised clustering of product reviews with topic and sentiment labels and texts by different authors using the k-means algorithm. It tries to steer binary clustering towards either sentiment or topic labels to no avail. Six-way clustering of the reviews achieves a Rand Index of 0.0742 and V-Measure of 0.0724 for topics; binary clustering a RI of 0.4904 for topics and a RI of 0.0205 for sentiment. The author clustering achieves a RI of 0.6725 and a VM of 0.5687 using word uni- and bigrams only. This study also does a supervised binary classification of the product review sentiments using a support vector machine. It tunes the algorithm on the choice of linear or radial kernel and of the feature set. It achieves the highest f1-score of 0.8531 using a linear kernel with a 6 character n-gram and the output from the k-means clustering as features.

You can probably round to two decimal points for these results.

1 Introduction

In this study I use the k-means and support vector machine algorithms for a number of classification tasks. I investigate the role of some of the algorithms' parameters and try different sets of features. The first tasks involve the unsupervised clustering of product reviews into six clusters hopefully corresponding to the six topics the reviews are about and into two clusters corresponding either to sentiment labels or topic labels. The last clustering task involves clustering 2500 texts into 50 clusters corresponding to the 50 authors of the texts. Then for supervised classification using SVM I do a binary classification on sentiment labels. I try a linear and a radial kernel with different word and character n-grams in addition to part-of-speech tags and the labeling from the k-means algorithm.

2 Data

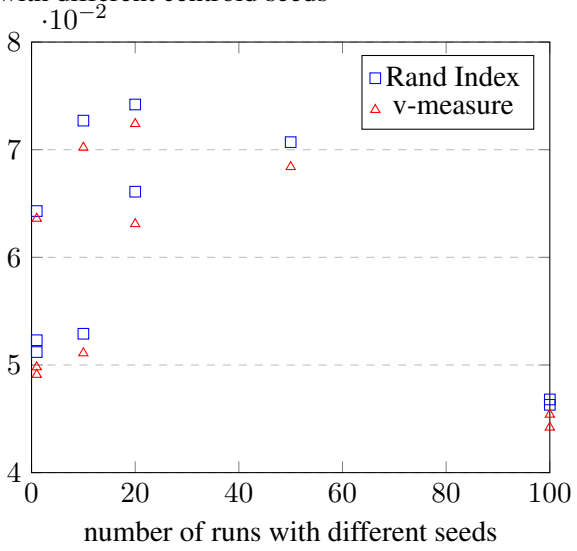
The study uses a corpus of six thousand product reviews. Each review consists of a label indicating whether the review is positive or negative and a label indicating to which of the followings six topics it belongs: books, camera, dvd, health, music or software. The review also contains a file reference and the actual text of the review. The topic and sentiment labels are distributed almost equally across the corpus. About half of the reviews about a particular topic are labeled positive. Table 1 shows the counts for the labels in the corpus.

Table 1: counts of topic and sentiment labels

topic	cnt	%	pos	%	neg	%
books	993	16.5	471	47	522	53
music	1027	17.1	531	52	496	48
dvd	1012	16.9	490	48	522	52
health	986	16.4	470	48	516	52
software	994	16.6	502	51	492	49
camera	988	16.5	504	51	484	49
total	6000	100.0	2968	48	3132	52

I also use a subset of the Reuters Corpus Volume 1. It was created by Zhi Liu (National Engineering Research Center for E-Learning, Hubei Wuhan, China). The top 50 authors (with respect to total size of articles) were selected, and for each of them 50 articles were picked that have at least one subtopic of the class CCAT(corporate/industrial). This is a way to minimise the topic factor in distinguishing among the texts. The training corpus consists of 2,500 texts (50 per author); the test corpus also includes 2,500 texts (50 per author), non-overlapping with the training texts.

Figure 1: plot of Rand Index and V-Measure for different numbers of times the algorithm was run with different centroid seeds



3 K-Means

K-Means is an unsupervised machine learning algorithm to cluster similar samples together in a vector space. I run the algorithm on the review data with the *k*-value equal to 6 to evaluate performance in classifying the reviews by topic. I experiment with changing the number of times the algorithm is run with different seed centroids. I use no preprocessor and the tf-idf vectoriser. Next I run the algorithm on a subset of the review data containing only two topics. I try to steer the outcome of the clustering towards either grouping the topics or the sentiment labels together by using feature selection based on the variance threshold or on selecting the best features with regards to *f*, chi^2 or mutual information measures. Thirdly I run the clustering algorithm on the author data to try and cluster the texts from the same author together. I try different combinations of word and character n-gram vector representations as well as part of speech tags. Performance of the clustering will be evaluated on Rand Index and V-Measure scores using four-fold cross-validation. For the author clustering task I evaluate the configuration with the highest score on the unseen test data.

Figure 1 plots the Rand Indices and V-Measures for different numbers of runs of the k-means algorithm on the review data with *k* equals 6 and evaluation against the topic labels. Each run the algorithm is initialised with different seed centroids that the samples cluster around. The run with the lowest resulting inertia, inter-cluster sum of squares, is selected as the best run. I expected a higher number of runs to result in a clustering with higher scores. For different runs of the experiment with identical settings the resulting scores vary to a similar extend as between experiments run with a different number of seedings. Therefore I can only conclude that the clustering is unstable. In general both the Rand Index and V-Measure show that most reviews are in the wrong cluster and that clusters contain many reviews that have a label different from the one assigned to the cluster. Comparing the golden labels of the reviews assigned to a particular cluster confirms this as well.

Table 2: scores for feature selection in clustering topic or sentiment

selector	RI topic	RI sentiment
none	0.0965	0.0006
variance 0	0.0288	0.0003
variance 1e-6	0.0231	0.0002
10 best F	0.4904	0.0205
10 best MI	0.0561	0.0002
10 best Chi2	0.0104	0.0000
100 best F	0.0106	0.0017
100 best MI	0.0141	0.0006
100 best Chi2	0.0369	0.0005

Next I take a subset of the review data that contains only reviews of cameras and of music. With the number of clusters set to two I try to steer the clustering towards either clustering the reviews with a positive or negative sentiment together or clustering the reviews about cameras or about music together. My assumption is that sentiment is conveyed by less specific words than topic. Selecting features based on various tests related to variance could influence the clustering. I try setting a variance threshold of *1e-5* and *1e-6* and I try selecting the k-best features for *k* equal to *5, 10, 20, 50, 100, 250* based on *ANOVA F-value*, chi^2 *statistics* and on *Mutual Information*. The scores for evaluating on the topic labels continue to be much higher than on the sentiment labels. Some of the resulting scores can be found in table 2. Selecting features tends to increase the difference in scoring. This somewhat supports the assumption. Because the clustering is already greatly in favour of the topic labels it does not help me in steering the clustering towards the sentiment labels.

The last set of experiments with k-means clustering uses the author corpus described in the data section of this report. There are fifty authors and therefore the number of clusters is set to fifty. I

You could explain this in more detail — what you mean by 'less specific' and how this relates to variance, and why you chose the various statistics tests that you used.

Normally written chi-squared or x2

Adjectives vs Nouns is another way to approach this — assuming that sentiment info is often in adjectives and category info in nouns. It also doesn't perform that well though.

Table 3: scores for author clustering for different features

features	RI	V-measure
word {n=1,2}	0.6633	0.5755
word {n=1,2}	0.6725	0.5687
word {n=1,2} + char {n=5}	0.6511	0.5685
word {n=1,2} + POS	0.5703	0.5325
char {n=5}	0.6554	0.5691
word {n=1,2,3} lemma	0.6614	0.5712
word {n=1,2,3} stemmed	0.6435	0.5663
word {n=3}	0.6520	0.5685

vary the features used by the algorithm. I use tf-idf vectors with word n-grams, character n-grams and part-of-speech labels. I list the Rand Index and V-Measure scores for cross-validation on the development data in table 3. The differences between the various n-gram features are small and including the part-of-speech tags has a negative impact. Also here the clustering is not stable and it is hard to draw any conclusions from this data. The word uni+bigram is fastest and it seems to score slightly higher. I evaluate it on the test data to find a Rand Index of **0.7011** and a V-Measure of **0.5854**.

4 Support Vector Machines

I use a SVM to perform binary classification of the review data into reviews with a positive or negative sentiment. I evaluate the influence of the soft-margin constant C and use both a linear and a radial basis kernel function. For the radial basis kernel function I evaluate different values for γ, the inverse of how far the influence of a sample selected as a support vector reaches. I next try to find the best performance by evaluating different combinations of features including clustering based features. Performance is evaluated on the f1-score of the classification using four-fold cross-validation.

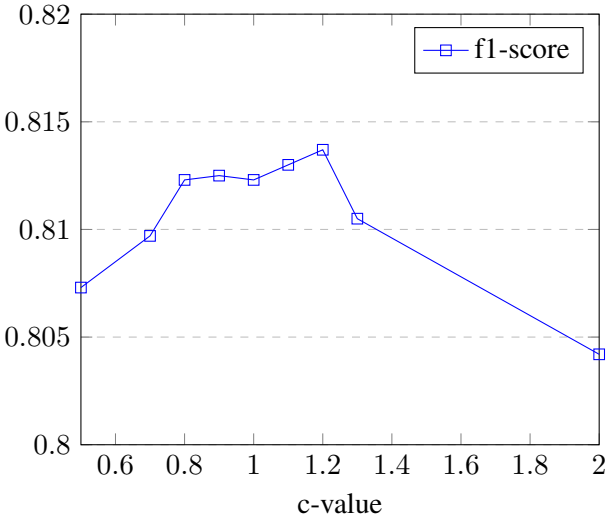
A large C also allows the classifier to misclassify training examples, so it doesn't only affect new data.

The c-value, the soft margin constant, determines the error penalty for samples that are close to the separating hyperplane. If the c-value is higher the SVM optimisation will choose a smaller margin to classify samples correctly. But if the margin is smaller the chance that new data is misclassified is greater. However I do not observe a accuracy-recall trade-off. Figure 2 plots the f1-score for c-values around the default value of 1.0. A c-value of **1.2** scores the highest.

Normally a *precision*-recall tradeoff

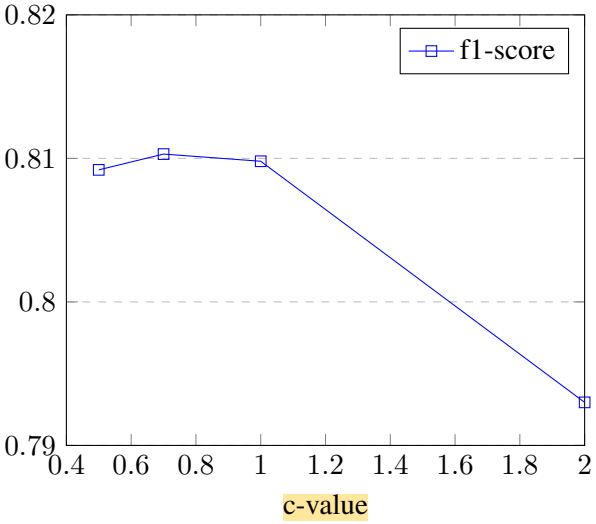
Linear kernels are supposed to work best for text classification tasks. However when I run the SVM algorithm with a radial basis function kernel it performs as well as with the linear kernel. I in-

Figure 2: f1-scores for differenct c-values



There's no reason for these plots to take up the whole page.

Figure 3: f1-scores for γ values for a rbf kernel and a c-value of 1.0



Is this gamma or C? The caption says gamma.

vestigate the influence of the kernel coefficient γ. This coefficient determines the sensitivity of the optimisation to differences in input vectors. The precise impact is dependent on dimensionality and normalisation. Figure 3 shows the impact of tuning the γ-value with the c-value set to 1.0. Tuning the c-value together with the γ-value leads to a maximum f1-score of **0.8153** at a c-value of **1.4** and a γ-value of **0.9**.

To find the best performing SVM classifier I experiment with the features. A count vector does not perform well and I use tf-idf vectors. Lemmatisation or stemming does not improve performance and neither does adding the part-of-speech tags. I try different word and character n-grams and find that character n-grams work best at 6 characters across word boundaries. Moving back to a linear kernel with a c-value of 0.9 improves the f1-score even further. Lastly I include the labels from a k-means clustering using uni- and bigrams to reach a score of **0.8531**. Table 4 shows the result of some of the features I try.

Table 4: scores for sentiment clustering using SVM for different features

kernel	features	f1-score
rbf	word {n=1,2}	0.8188
rbf	word {n=2}	0.7643
rbf	word {n=1,2,3}	0.8077
rbf	word {n=3}	0.6703
rbf	char {n=3}	0.8178
rbf	char {n=3,4,5}	0.8410
rbf	char {n=5}	0.8482
rbf	char {n=6}	0.8513
linear	char {n=6}	0.8525
linear	char_wb {n=6}	0.8270
linear	char {n=6} + POS	0.8525
linear	char {n=6} + Clustering	0.8531

It's nice to put the best result in a table in Bold.

5 Discussion/Conclusion

I investigate a supervised and an unsupervised method for classifying text. K-Means, the unsupervised method, does not perform well. I can make two observations. The author classification task that uses a lot more text performs better than the review classification task. Yet, and seemingly in contradiction to the first observation, performance peaks for the binary review classification when using only the ten best features based on an ANOVA measure. The clustering is unstable and it is hard to draw any conclusions about the factors determining performance for these tasks.

The support vector machine performs much better. I find two results striking. The first is that while linear separation is considered to be more appropriate for text classification due to the high number of features, the radial basis function does just as well. Secondly, although I cannot imagine what language feature it represents, the large, 6, character n-gram works very well. On the other hand, as in previous machine learning experiments I did, the differences are very small, less then half a percent, and significance is an issue. In general I do not find that more features makes for better classification.

Character based language models are often state of the art these days. It seems a bit strange, but remember that with 6-character bigrams you have most of the information from a word based model, but also can gain knowledge from punctuation, word length, number of spaces, etc that is lost in a word based model.