Using Text Classification to Predict the Gene Knockout Behaviour of S. Cerevisiae

Patrick Caldon

School of Computer Science and Engineering
University of New South Wales,
Kensington, NSW, Australia,
Email: patc@cse.unsw.edu.au

Abstract

A naive Bayes classifier was used to analyze gene behavior based on text data and presented as an entry for the 2002 KDD Cup, a data mining exercise to predict the behavior of the yeast S. Cerevisiae. The solution presented was based on the multinomial event model for text classification(McCallum & Nigam 1998) with a feature selection mechanism added. Despite this simple model, performance close to that of the best entries in the competition could be obtained, which were using more sophisticated techniques. It appears that seemingly minor effort in using prior knowledge to conflate the gene classes, as well as the previously described effectiveness of the naive Bayes method contributed to this success.

1 Introduction

Biological data consists in many forms; a vast quantity of data is held in academic research papers. It is clear that there is a great deal of information in these kinds of resources, however it is difficult for automated systems to extract such data. Natural language methods are the obvious mechanism for extracting this kind of information.

The 2002 KDD Cup consisted of two tasks, both based on bioinformatics data mining. “Task 2” of this challenge was to analyses a set of knockout data of the yeast Saccharomyces Cerevisiae, and predict the behavior of the organism according to some hidden system when the relevant gene had been knocked out. A variety of data were given to the contestants, most notably a body of approximately 15,000 abstracts from MEDLINE. In the course of constructing the entry for this challenge, it became apparent that mining the text data in these abstracts was the most effective approach, and furthermore that a few simple modifications to the parsing process seemed to greatly enhance the capability of the data mining software.

Simple Bayesian classifiers are very popular, on account of the ease of implementation and effectiveness despite their simplicity(Friedman, Geiger & Goldszmidt 1997). The classifier assumes that the data are generated according to some probabilistic system, makes estimates of these probabilities, and then combines the probabilities using Bayes theorem. Classification is done by finding the class most likely to have generated the particular data.

Naive Bayes classifiers make a further assumption that the process generating the data do so in such a way that each attribute generated is independent of all others. The so-called *naive Bayes assumption* is that the probability of both attributes occurring together is the product of the probabilities of their occurring independently, i.e. if a and b are events or attributes, then $p(a \wedge b) = p(a) \cdot p(b)$. In practice this assumption works well, and while some theories provide partial accounts for why it might work, ultimately it is unclear why(Friedman et al. 1997, Mitchell 1997).

The *bag of words* approach has been utilized in many systems to construct feature vectors corresponding to particular data items. Typically the words in a document are *stemmed*, removing common endings such as `ing` or `ed` in English, and then counted (Porter 1980). The counts in particular documents form the attribute in the feature vector.

There are several approaches to extracting biological information form abracts which have been tried. Template based information extraction techniques have been used with some success(Thomas, J., Milward, D., Ouzounis, C., Pulman, S.& Carroll, M. 2000). Stapley et al. used this approach in analyzing S. Cerevisiae, attempting to construct a predictor for sub-cellular location based on available abstracts (Stapley, Kelley & Sternberg 2002). Craven and Kumlien used naive Bayes to extract features from MEDLINE (Craven, & Kumlien 1999). Fukuda et al. construct a sophisticated hand-crafted parser for identifying references to proteins in text, which was applied to MEDLINE data(Fukuda, Tamura, Tsunoda & Takagi 1998).

2 Dataset

The dataset used was directly from the 2002 KDD Cup competition. This consisted of approximately 15,000 potentially relevant abstracts, all truncated to 250 words. There were also a set of gene aliases, that is mappings form standard to systematic names for genes. The experimental data was derived from a microarray experiment on an unknown cellular system, where the cell was exposed to some (unknown) enviornment. The results were presented as a list of knocked-out gene and the corresponding class, where `nc` indicated that no change was observed in the functioning of the cell, `change` indicated that the functioning of the pathway under examination changed, and `control` indicated that some other unrelated pathway also changed (as well as possibly the `change` pathway). The task was to construct classifiers, one capable of classifying the `change` genes as opposed to the `nc` and `control` genes, and the other for classifying the `change` and `control` together against the `nc` set.

	change	control	nc	Total
Training Data	38	46	2934	3018
Test Data	19	24	1446	1489
Total	57	70	4380	4507

3 Methodology

For the purposes of this analysis we viewed a gene as a set of documents written about a gene, from which attributes could be derived. Each gene was then reduced into a feature vector describing that gene. Based on the training data-set feature selection was performed, which produced the set of features giving best accuracy over the data set. These features were then used as the basis for calculating the probability of each class in the training set.

3.1 Text Parsing

Most of the labour of this project was in parsing the text to extract the words from it. Abstracts were separated into sentences, and then into words. Various features of the text needed to be corrected, such as the removal or expansion of abbreviations, the normalization of numbers into a common syntax, and the deletion of references from the text.

An ad-hoc "gene-like-word" recognizer was implemented; in general it was impossible for this to work entirely correctly due to words like `loci` becoming confused with a gene-like word such as `LOCI`. (Indeed `LOC1` is a recognized part of the S. Cerevisiae genome). Protein suffixes were removed (e.g. `PPR1p` ↦ `PPR1`) and various prefixes (e.g. `h`, `HUM`, `Delta`) were also deleted. When known these standard names were then mapped to their equivalent systematic name, which was used as the standard name throughout the rest of the program. (e.g. `PPR1` ↦ `YLR014C`). This simple analysis seemed to function well, despite being far simpler than the approach outlined by Fukuda et al.(Fukuda et al. 1998).

These words were then counted and parsed into three classes; dictionary words, non-dictionary words, and gene names (stored internally as the systematic name of the gene, e.g. `YLR014C`). Early experiments showed the gene names had much greater predictive value than the dictionary words, non-dictionary words or other combinations, and so the gene names were concentrated on for the remainder of the exercise. A variety of stemming techniques were also attempted, in order to improve performance, essentially using the standard porter stemming algorithm (Porter 1980) with some additions to deal with common biological words (e.g. `mitocondria` and `golgi`).

3.2 Bag of Words Model

Having parsed the abstracts, a count of words corresponding to each gene was created. This was used in the calculation of the probability of a word occurring in a context relating to a particular class, i.e. $p(w|c)$. For each word (i.e. each standard name in the final model) a probability was estimated of that word occurring in a particular class. In the event that there was more than one abstract describing the gene all abstracts describing the gene were concatenated together to be considered as the abstract describing the gene.

Originally the following formula based on the Laplacian probability estimate was used(McCallum 1998):

$$p(w_i|c_j) = \frac{1 + N_{ij}}{N_j + |V|}$$

where N_{ij} is the number of times word i has appeared in the context of class j, and N_j is the total number of times any word has appeared in the context of class N_j, and $|V|$ is the total vocabulary size.

The performance using this was poor however, and it was necessary to ensure that minor changes of N_{ij} had a small effect in perturbing the prior odds. To this end the estimates were changed to the following formula, based on the m-probability estimate(Mitchell 1997):

$$p(w_i|c_j) = \frac{N_{ij} + \frac{m}{|V|}}{N_j + m}$$

where m is some constant parameter, which is clearly identical to the original when $|V| = m$. Empirical testing showed that this worked well, and values of $m = 100$ had the highest score in cross-fold validation and was used for the final testing; in retrospect this was probably a mistake since other tested values higher than this whilst not achieving the same peak value were more robust when the feature selection was not as well tuned.

3.3 Multinomial Model

The multinomial formula was used to estimate the odds of a gene given a particular class, according to the multinomial formula for probability estimation(McCallum 1998):

$$p(g_i|c_j) = p(D|c_j)|g_i|! \prod_k \frac{p(w_k|c_j)^{N_{kj}}}{N_{kj}!}$$

Here $p(g_i|c_j)$ is the probability of finding a specific gene i (i.e. vector of word counts) given a class j. The probability of finding a document of that size is given by $p(D|c_j)$; it was assumed that this was independent of the class and so ignored in the implementation. Note that the naive Bayes assumption is employed here.

Having calculated $p(g_i|c_j)$ it is possible to determine the most probable class given a particular gene $p(c_i|g_j)$ by employing Bayes rule, $p(c_i|g_j) = p(c_i) \cdot p(g_j|c_i)/p(g_j)$. Originally classification was performed using the standard approximation:

$$\text{class of } g_i = \underset{j}{\text{argmax}} p(c_i)p(g_j|c_i)$$

In order to produce rank orderings, it was noted that every gene did in fact have a class, i.e. for gene j $\sum_i p(c_i|g_j) = 1$. Given that $p(g_i)$ was constant, this gave the final formula used for estimation:

$$p(c_i|g_j) = \frac{p(c_i)p(g_j|c_i)}{\sum_k p(c_k)p(g_j|c_k)}$$

In the case that more than one class was being tested for, the sum of the $p(c_i)p(g_j|c_i)$ was used as the numerator.

3.4 Feature Selection

An entropy based feature selector was used in order to find the best words to base the classification on; these have been used in a variety of text mining settings with considerable success (McCallum 1998). The feature selector measures mutual information between the class and the feature; where this value is high, the feature will be a useful predictor, and when this is low the feature will be less useful.

The formula used for this estimation was besed on the mutual information calculation given by McCallum and Nigam(McCallum 1998).

$$I(C,W) = \sum_{i \in C} \sum_{t \in \{W\}} p(c_i, w_t) \log(\frac{p(c_i, w_t)}{p(c_i)p(w_t)})$$

In this equation, $p(c_i)$ is the number of occurances of a word in class c_i divided by the total number of word occurances, $p(w_t)$ is the total number of occurances of word w_t divided by the total number of word occurances, and $p(c_i, w_t)$ is the number of occurances of word w_t in class c_t.

The probability estimates used were those calculated for the multinomial distribution of the documents; these are apparently satisfactory for most purposes(McCallum 1998). Features in the feature vector were removed if (according to the counts) they had low information content and thus low predictive value.

A search of the parameter space was made in order to find optimal sizes for the feature selector. The optimal value for m was taken to be that of the optimal value with the corresponding feature size.

3.5 Use of ROC Curves

A Receiver Operating Characteristic (ROC) curve is commonly used in signal detection theory to examine the performance of a classification system in terms of the number of false alarms generated versus the number of true hits generated (Egan, 1975). The ROC curve is generated by considering the rate at which true positives accumulate versus the rate at which false positives accumulate. If a system produces a rank ordering of data, with the most likely data at the top of the curve and the least likely data at the bottom, the ROC curve will be a curve between $(0, 0)$ and $(1, 1)$ with each point on the x-axis corresponding to a data point, and the height on the y-axis corresponding to the number of true positives thus far identified. Thus an ideal system will commence by identifying all the positive examples and so the curve will rise to $(0, 1)$ immediately, having a zero rate of false positives, and then continue along to $(1, 1)$. The area under the ROC curve is used as the measure of classifier performance.

4 Experimental Results

Our initial submission to the KDD Cup competition was disappointing, on account of a methodological error in training the feature selector. Retraining this correctly resulted in much better performance however, comparable with the better entries in the cup.

Note that `narrow` refers to the predictor of the `change` data set only, and `broad` refers to the predictor for both the `change` and `control` data sets. "Simple gene parser" refers to the program doing no feature construction; this is compared with the more substantial feature construction used for the other examples.

It appears that most of the performance advantage came on account of the use of the feature selection algorithm, with some additional accuracy from the parser. More significantly, the parser reduced the sensitivity to the number of features used in the feature selection. This can be seen in the average of the 3-fold cross validation results.

The results in this table were either the quoted results from the KDD competition or the results gained from experiments on the test data having trained the learner on the training data.

It should be observed that with the feature selector at its optimum value the majority of the genes had no features applicable to them; of the 1489 genes in the test set, 793 had no features applicable to them. When this cutoff is examined in the context of the ROC curve, it seems that those genes for which there are no data perform approximately at chance, and the ROC curve inflects close to this cutoff point. Thus the number of genes with features in the final analysis gives a useful region of confidence about the predictions; the classifier has good predictive value for approximately the first third of the genes which it outputs, and relatively poor performance thereafter.

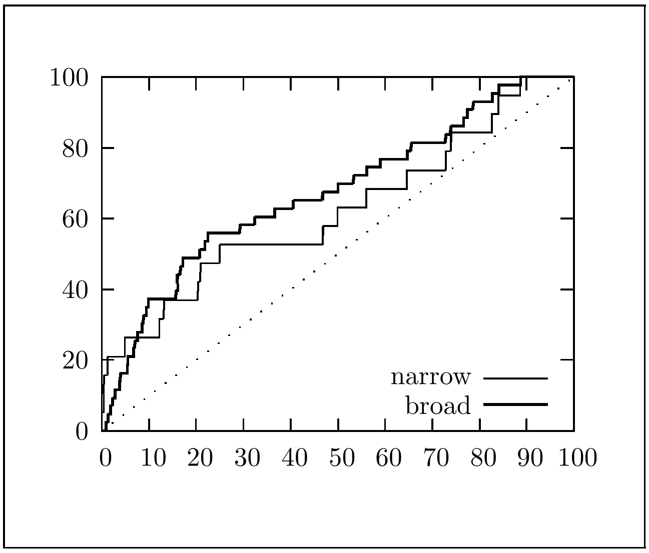

Figure 1: ROC Curve - 1250 features, feature construction

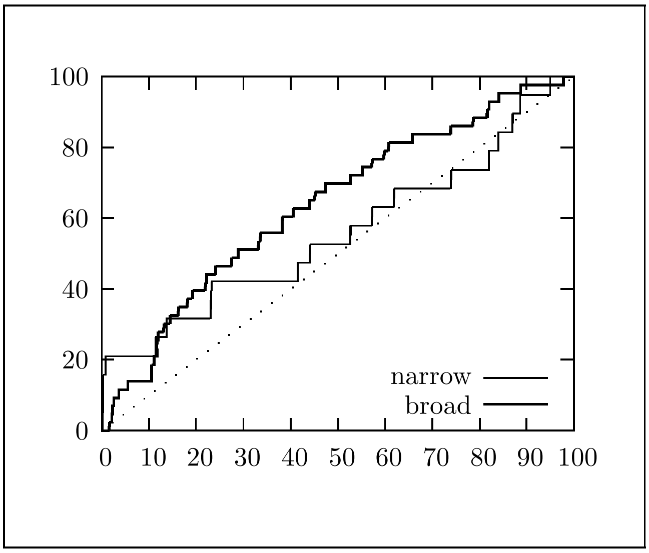

Figure 2: ROC Curve - unlimited features, feature construction

5 Conclusion

This paper describes the development of a text-mining system to determine the presence or absence of genes in some pathway of the yeast S. Cerevisiae.

	Narrow	Broad	Total
Full parser, 1250 features	0.623	0.679	1.303
Full parser, all features	0.565	0.660	1.223
Simple gene parser, 1250 features	0.575	0.675	1.249
Simple gene parser, all features	0.557	0.644	1.201
Best in KDD Cup 2002	0.684	0.678	1.322

Table 1: Results on test data

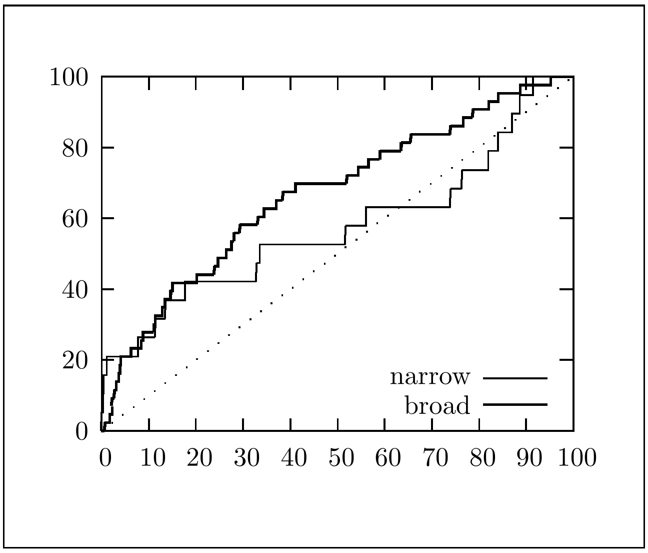

Figure 3: ROC Curve - 1250 features, no feature construction

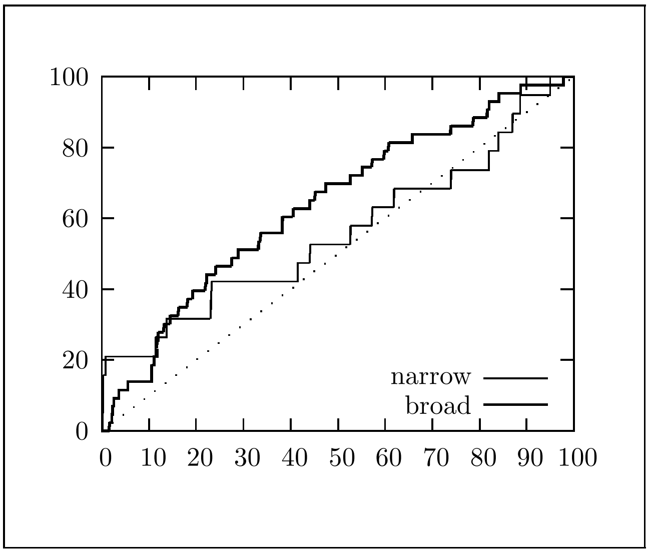

Figure 4: ROC Curve - unlimited features, no feature construction

Since the identity of the specific pathway was unknown to the program (and its designer) this methodology should be applicable to more generalized problems. One useful problem where this could be immediately applied is predicting subcelluar localization; work on this has already commenced.

Naive Bayes is a simple learning model; what this paper has shown is the real power of feature construction and feature selection prior to exercising the learning algorithm. Use of feature construction and feature selection resulted in performance close to that of more sophisticated techniques, such as Support Vector Machines. The model has releatively few parameters which need tuning, however there is no clear methodology for obtaining the precise number of features to use in a feature selection context.

Whilst this work used a relatively simple model for protein parsing, using more complex models such as that in (Fukuda et al. 1998) may give better results, and this would be a worthwhile work. Similarly suing this feature extraction and feature construction with more complex learning algorithms (such as Bayes nets or SVMs) would also be worthwhile. Application to a problem of more immediate interest to biologists, e.g. protein sub-cellular localization would also be of benefit; currently work has commenced to compare this work with the results in (Stapley et al. 2002).

Acknowledgements

Many thanks to Mike Bain, Bruno Gaeta and Rex Kwok for their work and assistance during this project.

References

McCallum, A. & Nigam, K. (1998), A comparison of event models for Naive Bayes text classification, *in* AAAI-98 Workshop on Learning for Text Categorization, 1998.

Automatic Extraction of Protein Interactions from Scientific Abstracts *in* Pacific Symposium on Biocomputing 5:538-549 (2000).

Friedman, N., Geiger, D. & Goldszmidt, M. (1997), Bayesian Network Classifiers, *Machine Learning*, 29(2-3), pp 131–663.

Mitchell, T. M. (1997), *Machine Learning*, McGraw-Hill, 1997.

Stapley, B.J., Kelley, L. A. & Sternberg M.J.E., (2002), Predicting the sub-cellular location of proteins from text using support vector machines *in* Pacific Symposium on Biocomputing, 2002.

Craven, M. & Kumlien, J., (1999), Constructing biological knowledge-bases by extracting information from text sources, *in* Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology, Germany, 1999, pp 77-86,

Porter, M., (1980), An algorithm for suffix stripping, *Program*, 14(3) pp 130-137.

Fukuda, K., Tamura, A., Tsunoda, T.,& Takagi, T., Toward information extraction: Identifying protein names from biological papers, *in* Proceedings of the Pacific Symposium on Biocomputing '98 (PSB'98), pp 707–718, 1998.