Knowledge Discovery in Multi-Label Phenotype Data

Amanda Clare and Ross D. King

Department of Computer Science,
University of Wales Aberystwyth, SY23 3DB, UK
{ajc99,rdk}@aber.ac.uk

Abstract. The biological sciences are undergoing an explosion in the amount of available data. New data analysis methods are needed to deal with the data. We present work using KDD to analyse data from mutant phenotype growth experiments with the yeast *S. cerevisiae* to predict novel gene functions. The analysis of the data presented a number of challenges: multi-class labels, a large number of sparsely populated classes, the need to learn a set of accurate rules (not a complete classification), and a very large amount of missing values. We developed resampling strategies and modified the algorithm C4.5 to deal with these problems. Rules were learnt which are accurate and biologically meaningful. The rules predict function of 83 putative genes of currently unknown function at an estimated accuracy of $\geq 80\%$.

1 Introduction

The biological sciences are undergoing an unprecedented increase in the amount of available data. In the last few years the complete genomes of ~30 microbes have been sequenced, as well as that of "the worm" (*C. elegans*) and "the fly" (*D. melanogaster*). The last few months have seen the sequencing of the first plant genome Arabidopsis [10], and the greatest prize of all, the human genome [11, 33]. In addition to data from sequencing, new post genomic technologies are enabling the large-scale and parallel interrogation of cell states under different stages of development and under particular environmental conditions, generating very large databases. Such analyses may be carried out at the level of mRNA using micro-arrays (e.g. [6, 9]) (the transcriptome). Similar analyses may be carried out at the level of the protein to define the proteome (e.g. [2]), or at the level of small molecules, the metabolome (e.g. [27]). This data is replete with undiscovered biological knowledge which holds the promise of revolutionising biotechnology and medicine. KDD techniques are well suited to extracting this knowledge. Currently most KDD analysis of bioinformatic data has been based on using unsupervised methods e.g. [9, 17, 32], but some has been based on supervised methods [4, 7, 14]. New KDD methods are constantly required to meet the new challenges presented by new forms of bioinformatic data.

Perhaps the least analysed form of genomics data is that from phenotype experiments [25, 22, 18]. In these experiments specific genes are removed from the cells to form mutant strains, and these mutant strains are grown under different conditions with the aim of finding growth conditions where the mutant and the wild type (no mutation) differ ("a phenotype"). This approach is analogous to removing components from a car and then attempting to drive the car under different conditions to diagnose the role of the missing component.

In this paper we have developed KDD techniques to analyse phenotype experiment data. We wish to learn rules that given a particular set of phenotype experimental results predict the functional class of the gene mutated. This is an important biological problem because, even in yeast, one the best characterised organisms, the function of 30–40% of its genes are still currently unknown.

Phenotype experiment data presents a number of challenges to standard data analysis methods: the functional classes for genes exist in a hierarchy, a gene may have more than one functional class, and we wish to learn a set of accurate rules - not necessarily a complete classification. The recognition of functional class hierarchies has been one of the most important recent advances in bioinformatics [29, 1, 13]. For example in the Munich Information Center for Protein Sequences (MIPS) hierarchy (http://mips.gsf.de/proj/yeast/catalogues/funcat/) the top level of the hierarchy has classes such as: "Metabolism", "Energy", "Transcription" and "Protein Synthesis". Each of these classes is then subdivided into more specific classes, and these are in turn subdivided, and then again subdivided, so the hierarchy is up to 4 levels deep. An example of a subclass of "Metabolism" is "amino-acid metabolism", and an example of a subclass of this is "amino-acid biosynthesis". An example of a gene in this subclass is YPR145w (gene name ASN1, product "asparagine synthetase"). In neither machine learning or statistics has much work has been done on classification problems where there is a class hierarchy. However, such problems are relatively common in the real world, particularly in text classification [16, 24, 21]. We deal with the class hierarchy by learning separate classifiers for each level. This simple approach has the unfortunate side-effect of fragmenting the class structure and producing many classes with few members - e.g. there are 99 potential classes represented in the data for level 2 in the hierarchy. We have therefore developed a resampling method to deal with the problem of learning rules from sparse data and few examples per class.

Perhaps an even greater difficulty with the data is that genes may have more than one functional class. This is reflected in the MIPS classification scheme (where a single gene can belong to up to 10 different functional classes). This means that the classification problem is a *multi-label* one (as opposed to *multi-class* which usually refers to simply having more than two possible disjoint classes for the classifier to learn). There is only a limited literature on such problems, for example [12, 20, 30]. The UCI repository [3] currently contains just one dataset ("University") that can be considered a multi-label problem. (This dataset shows the academic emphasis of individual universities, which can be multi-valued, for example, business-education, engineering, accounting and fine-arts). The simplest approach to the multi-label problem is to learn separate classifiers for each class (with all genes not belonging to a specific class used as negative examples for that class). However this is clearly cumbersome and time-consuming when there are many classes - as is the case in the functional hierarchy for yeast. Also, in sparsely populated classes there would be very few positive examples of a class and overwhelmingly many negative examples. We have therefore developed a new algorithm based on the successful decision tree algorithm C4.5 [26].

A third challenge in prediction of gene function from phenotype data is that we wish to learn a set of rules which accurately predict functional class. This differs from the standard statistical and machine learning supervised learning task of maximising the prediction accuracy on the test set. The problem resembles in some respects association rule learning in data mining.

In summary our aim is to discover new biological knowledge about:

- the biological functions of genes whose functions are currently unknown
- the different discriminatory power of the various growth conditions under which the phenotype experiments are carried out

For this we have developed a specific machine learning method which handles the problems provided by this data:

- many classes
- multiple class labels per gene
- the need to know accuracies of individual rules rather than the ruleset as a whole

2 Experimental method

2.1 Data

We used three separate sources of phenotypic data: TRIPLES [18], EUROFAN [25] and MIPS [22].

- The TRIPLES (TRansposon-Insertion Phenotypes, Localization and Expression in *Saccharomyces*) data was generated by randomly inserting transposons into the yeast genome.
  URLs: http://ygac.med.yale.edu/triples/triples.htm, (raw data)
  http://bioinfo.mbb.yale.edu/genome/phenotypes/ (processed data)
- EUROFAN (European functional analysis network) is a large European network of research which has created a library of deletion mutants by using PCR-mediated gene replacement (replacing specific genes with a marker gene (kanMX)). We used data from EUROFAN 1.
  URL: http://mips.gsf.de/proj/eurofan/
- The MIPS (Munich Information Center for Protein Sequences) database contains a catalogue of yeast phenotype data.
  URL: http://mips.gsf.de/proj/yeast/

The data from the three sources were concatenated together to form a unified dataset, which can be seen at http://users.aber.ac.uk/ajc99/phenotype/. The phenotype data has the form of attribute-value vectors: with the attributes being the growth media, the values of the attributes being the observed sensitivity or resistance of the mutant compared with the wildtype, and the class the functional class of the gene. Notice that this data will not be available for all genes due to some mutants being inviable or untested, and not all growth media were tested/recorded for every gene, so there were *very many missing values* in the data.

The values that the attributes could take were the following:

n	no data
w	wild-type (no phenotypic effect)
s	sensitive (less growth than for the wild-type)
r	resistance (better growth than for the wild-type)

There were 69 attributes, 68 of which were the various growth media (e.g. calcofluor_white, caffeine, sorbitol, benomyl), and one which was a discretised count of how many of the media this mutant had shown a reaction to (i.e. for how many of the attributes this mutant had a value of "s" or "r").

2.2 Algorithm

The machine learning algorithm we chose to adapt for the analysis of phenotype data was the well known decision tree algorithm C4.5 [26]. C4.5 is known to be robust, and efficient [23]. The output of C4.5 is a decision tree, or equivalently a set of symbolic rules. The use of symbolic rules allows the output to be interpreted and compared with existing biological knowledge - this is not generally the case with other machine learning methods, such as neural networks, or support vector machines.

In C4.5 the tree is constructed top down. For each node the attribute is chosen which best classifies the remaining training examples. This is decided by considering the information gain, the difference between the entropy of the whole set of remaining training examples and the weighted sum of the entropy of the subsets caused by partitioning on the values of that attribute.

$$information\_gain(S, A) = entropy(S) - \sum_{v \in A} \frac{|S_v|}{|S|} * entropy(S_v)$$

where A is the attribute being considered, S is the set of training examples being considered, and S_v is the subset of S with value v for attribute A. The algorithms behind C4.5 are well documented and the code is open source, so this allowed the algorithm to be extended.

Multiple labels are a problem for C4.5, and almost all other learning methods, as they expect each example to be labeled as belonging to just one class. For yeast this isn't the case, as a gene may belong to several different classes. In the case of a single class label for each example the entropy for a set of examples is just

$$entropy(S) = -\sum_{i=1}^{N} p(c_i)\log p(c_i)$$

where $p(c_i)$ is the probability (relative frequency) of class c_i in this set.

We need to modify this formula for multiple classes. Entropy is a measure of the amount of uncertainty in the dataset. It can be thought of as follows: Given an item of the dataset, how much information is needed to describe that item? This is equivalent to asking how many bits are needed to describe all the classes it belongs to.

To estimate this we sum the number of bits needed to describe membership or non-membership of each class (see appendix for intuition). In the general case where there are N classes and membership of each class c_i has probability $p(c_i)$ the total number of bits needed to describe an example is given by

$$-\sum_{i=1}^{N} ((p(c_i)\log p(c_i)) + (q(c_i)\log q(c_i)))$$

where
$p(c_i) =$ probability (relative frequency) of class c_i
$q(c_i) = 1 - p(c_i) =$ probability of not being member of class c_i

Now the new information after a partition according to some attribute, can be calculated as a weighted sum of the entropy for each subset (calculated as above), where this time, weighted sum means if an item appears twice in a subset because it belongs to two classes then we count it twice.

In allowing multiple labels per example we have to allow leaves of the tree to potentially be a set of class labels, i.e. the outcome of a classification of an example can be a set of classes. When we label the decision tree this needs to be taken into account, and also when we prune the tree. When we come to generate rules from the decision tree, this can be done in the usual way, except when it is the case that a leaf is a set of classes, a separate rule will be generated for each class, prior to the rule-pruning part of the C4.5rules algorithm. We could have generated rules which simply output a set of classes - it was an arbitrary choice to generate separate rules, chosen for comprehensibility of the results.

2.3 Resampling

The large number of classes meant that many classes have quite small numbers of examples. We were also required only to learn a set of accurate rules, not a complete classification. This unusual feature of the data made it necessary for us to develop a complicated resampling approach to estimating rule accuracy based on the bootstrap.

All accuracy measurements were made using the m-estimate [5] which is a generalisation of the Laplace estimate, taking into account the *a priori* probability of the class. The m-estimate for rule r (M(r)) is:

$$M(r) = \frac{p + m\frac{P}{P+N}}{p + n + m}$$

where
P = total number of positive examples,
N = total number of negative examples.
p = number of positive examples covered by rule r,
n = number of negative examples covered by rule r
m = parameter to be altered

Using this formula, the accuracy for rules with zero coverage will be the *a priori* probability of the class. m is a parameter which can be altered to weight the *a priori* probability. We used m=1.

The data set in this case is relatively small. We have 2452 genes with some recorded phenotypes, of which 991 are classified by MIPS as "Unclassified" or "Classification not yet clear-cut". These genes of unknown classification cannot be used in supervised learning (though we can later make predictions for them). This leaves just 1461, each with many missing values. At the top level of the classification hierarchy (the most general classes), there are many examples for each class, but as we move to lower, more specific levels, the classes become more sparsely populated, and machine learning becomes difficult.

We aimed to learn rules for predicting functional classes which could be interpreted biologically. To this end we split the data set into 3 parts: training data, validation data to select the best rules from (rules were chosen that had an accuracy of at least 50% and correctly covered at least 2 examples), and test data. We used the validation data to avoid overfitting rules to the data. However, splitting the dataset into 3 parts means that the amount of data available for training will be even less. Similarly only a small amount will be available for testing. Initial experiments showed that the split of the data substantially affected the rulesets produced, sometimes producing many good rules, and sometimes none. The two standard methods for estimating accuracy under the circumstance of a small data set are 10-fold cross-validation and the bootstrap method [15, 8]. Because we are interested in the rules themselves, and not just the accuracy, we opted for the bootstrap method, because a 10-fold cross validation would make just 10 rulesets, whereas bootstrap sampling can be used to create hundreds of samples of the data and hence hundreds of rulesets. We can then examine these and see which rules occur regularly and are stable, not just artifacts of the split of the data.

The bootstrap is a method where data is repeatedly sampled with replacement to make hundreds of training sets. A classifier is constructed for each sample, and the accuracies of all the classifiers can be averaged to give a final measure of accuracy. First a bootstrap sample was taken from the original data. Items of the original data not used in the sample made up the test set. Then a new sample was taken with replacement *from the sample*. This second sample was used as training data, and items that were in the first sample but not in the second made up the validation set. All three data sets are non-overlapping.

We measured accuracy on the held-out test set. We are aware that this will give a *pessimistic* measure of accuracy (i.e. the true accuracy on the whole data set will be higher), but this is acceptable.

3 Results

We attempted to learn rules for all classes in the MIPS functional hierarchy http://mips.gsf.de/proj/yeast/catalogues/funcat/, using the catalogue as it was on 27 September 1999. 500 bootstrap samples were made, and so C4.5 was run 500 times and 500 rulesets were generated and tested. To discover which rules were stable and reliable we counted how many times each rule appeared across the 500 rulesets. Accurate stable rules were produced for many of the classes at levels 1 and 2 in the hierarchy. At levels 3 and 4 (the most specific levels with the least populated classes) no useful rules were found. That is, at the lower levels, few rules were produced and these were not especially general or accurate.

The good rules are generally very simple, with just one or two conditions necessary to discriminate the classes. This was expected, especially since most mutants were only sensitive/resistant to a few media. Some classes were far easier to recognise than others, for example, many good rules predicted class "CELLULAR BIOGENESIS" and its subclass "biogenesis of cell wall (cell envelope)".

Some examples of the rules and their accuracies follow. The full set of rules can be seen at http://users.aber.ac.uk/ajc99/phenotype/ along with the data sets used.

The 4 most frequently appearing rules at level 1 (the most general level in the functional catalogue) are all predictors for the class "CELLULAR BIOGENESIS". These rules suggest that sensitivity to zymolase or papulacandin_b, or any reaction (sensitivity or resistance) to calcofluor_white is a general property of mutants whose deleted genes belong to the CELLULAR BIOGENESIS class. All correct genes matching these rules in fact also belong to the subclass "biogenesis of cell wall (cell envelope)". The rules are far more accurate than the prior probability of that class would suggest should occur by chance.

These are two of the rules regarding sensitivity/resistance to Calcofluor White.

```
if  the gene is sensitive to calcofluor white and
    the gene is sensitive to zymolyase
    then its class is "biogenesis of cell wall (cell envelope)"
Mean accuracy:       0.909
Prior prob of class: 0.095
Std dev accuracy:    0.018
Mean no. matching genes: 9.3

if   the gene is resistant to calcofluor white
     then its class is "biogenesis of cell wall (cell envelope)"
Mean accuracy:       0.438
```

```
Prior prob of class: 0.095
Std dev accuracy:    0.144
Mean no. matching genes: 6.7
```

These rules confirm that Calcofluor White is useful for detecting cell wall mutations [28, 19]. Calcofluor White is a negatively charged fluorescent dye that does not enter the cell wall. Its main mode of action is believed to be through binding to chitin and prevention of microfibril formation and so weakening the cell wall. The explanation for disruption mutations in the cell wall having increased sensitivity to Calcofluor White is believed to be that if the cell wall is weak, then the cell may not be able to withstand further disturbance. The explanation for resistance is less clear, but the disruption mutations may cause the dye to bind less well to the cell wall. Zymolase is also known to interfere with cell wall formation [19]. Neither rule predicts the function of any gene of currently unassigned function. This is not surprising given the previous large scale analysis of Calcofluor White on mutants.

One rule that does predict a number of genes of unknown function is:

```
if   the gene is sensitive to hydroxyurea
     then its class is "nuclear organization"
Mean accuracy:       0.402
Prior prob of class: 0.215
Std dev accuracy:    0.066
Mean no. matching genes: 33.4
```

This rule predicts 27 genes of unassigned function. The rule is not of high accuracy but it is statistically highly significant. Hydoxyurea is known to inhibit DNA replication [31], so the rule makes biological sense.

Table 1 shows the number of genes of unassigned function predicted by the learnt rules at levels 1 and 2 in the functional hierarchy. These are plotted as a function of the estimated accuracy of the predictions and the significance (how many standard deviations the estimated accuracy is from the prior probability of the class). These figures record genes predicted by rules that have appeared at least 5 times during the bootstrap process.

Level 1

estimated accuracy	std. deviations from prior 2	3	4
$\geq$ 80%	83	72	35
$\geq$ 70%	209	150	65
$\geq$ 50%	211	150	65

Level 2

estimated accuracy	std. deviations from prior 2	3	4
$\geq$ 80%	63	63	63
$\geq$ 70%	77	77	77
$\geq$ 50%	133	126	126

Table 1. Number of genes of unknown function predicted

It can be seen that analysis of the phenotype growth data allows the prediction of the functional class of many of the genes of currently unassigned function.

Table 2 shows the number of rules found for the classes at level 1. We did not expect to be able to learn rules for every class, as some classes may not be distinguishable given the growth media that were used.

number of rules	class no	class name
17	1/0/0/0	METABOLISM
32	3/0/0/0	CELL GROWTH, CELL DIVISION AND DNA SYNTHESIS
3	4/0/0/0	TRANSCRIPTION
1	5/0/0/0	PROTEIN SYNTHESIS
2	6/0/0/0	PROTEIN DESTINATION
1	7/0/0/0	TRANSPORT FACILITATION
21	9/0/0/0	CELLULAR BIOGENESIS (proteins are not localized to the corresponding organelle)
5	11/0/0/0	CELL RESCUE, DEFENSE, CELL DEATH AND AGEING
77	30/0/0/0	CELLULAR ORGANIZATION (proteins are localized to the corresponding organelle)

Table 2. Number of rules that appeared more than 5 times at level 1, broken down by class. Classes not shown had no rules (2/0/0/0, 8/0/0/0, 10/0/0/0, 13/0/0/0 and 90/0/0/0).

Table 3 shows some general statistics for the rulesets. Due to the nature of the bootstrap method of collecting rules, only *average* accuracy and coverage can be computed (rather than *total*), as the test data set changes with each bootstrap sample.

	no. rules	no. classes represented	av rule accuracy	average rule coverage (genes)
level 1	159	9	62%	20
level 2	74	12	49%	11
level 3	9	2	25%	18
level 4	37	1	71%	28

Table 3. General statistics for rules that appeared more than 5 times. Surprisingly high accuracy at level 4 is due to very few level 4 classes, with one dominating class.

4 Discussion and conclusion

Working with the phenotypic growth data highlighted several learning issues which are interesting:

- We had to extend C4.5 to handle the problem of genes having more than one function, the multi-label problem.
- We needed to select rules for biological interest rather than predicting all examples, this required us to use an unusual rule selection procedure, and this together with the small size of data set led to our choice of the bootstrap to give a clearer picture of the rules themselves.

Biologically important rules were learnt which allow the accurate prediction of functional class for ~200 genes. We are in the process of experimentally testing these predictions. The prediction rules can be easily comprehended and compared with existing biological knowledge. The rules are also useful as they show future experimenters *which media provide the most discrimination between functional classes*. Many types of growth media are shown to be highly informative for identifying the functional class of disruption mutants (e.g. Calcofluor White), others are of little value (e.g. sodium chloride). The nature of the C4.5 algorithm is always to choose attributes which split the data in the most informative way. This knowledge can be used in the next round of phenotypic experiments.

Our work illustrates the value of cross-disciplinary work. Functional genomics is enriched by a technique for improved prediction of the functional class of genes: and KDD is enriched by provision of new data analysis challenges.

Acknowledgments

We would like to thank Ugis Sarkans for initial collection of the data and Stephen Oliver and Douglas Kell for useful discussions.

Appendix: Reasoning behind multi-class entropy formula

This appendix gives an intuition into the reason for the multi-class entropy formula.

How many bits are needed to describe all the classes an item belongs to? For a simple description, we could use a bitstring, 1 bit per class, to represent each example. With 4 classes {**a**,**b**,**c**,**d**}, an example belonging to classes **b** and **d** could be represented as 0101. But this will usually be more bits than we actually need. Suppose every example was a member of class **b**. In this case we would not need the second bit at all, as class **b** membership is assumed. Or suppose 75% of the examples were members of class **b**. Then we know in advance an example is more likely to belong to class **b** than not to belong. The expected amount of information gained by actually knowing whether it belongs or not will be:

p(belongs) * gain(belongs) + p(doesn't belong) * gain(doesn't belong)
= 0.75 * (log 1 - log 0.75) + 0.25 * (log 1 - log 0.25)
= - (0.75 * log 0.75) - (0.25 * log 0.25)
= 0.81

where gain(x) = information gained by knowing x

That is, we actually only need 0.81 of a bit to represent the extra information we need to know membership or not of class **b**. Generalising, we can say that instead of one bit per class, what we actually need is the total of the extra information needed to describe membership or non-membership of each class. This sum will be

$$-\sum_{i=1}^{N} ((p(c_i)\log p(c_i)) + (q(c_i)\log q(c_i)))$$

where $p(c_i)$ is probability of membership of class c_i and $q(c_i)$ is probability of non-membership of class c_i.

References

1. M. Andrade, C. Ouzounis, C. Sander, J. Tamames, and A. Valencia. Functional classes in the three domains of life. *Journal of Molecular Evolution*, 49:551–557, 1999.
2. W. P. Blackstock and M. P. Weir. Proteomics: quantitative and physical mapping of cellular proteins. *Tibtech*, 17:121–127, 1999.
3. C.L. Blake and C.J. Merz. UCI repository of machine learning databases, 1998.
4. M. Brown, W. Nobel Grundy, D. Lin, N. Cristianini, C. Walsh Sugnet, T. Furey, M. Ares Jr., and D. Haussler. Knowledge-based analysis of microarray gene expression data by using support vector machines. *Proc. Nat. Acad. Sci. USA*, 97(1):262–267, Jan 2000.
5. B. Cestnik. Estimating probabilities: A crucial task in machine learning. In *Proceedings of the Ninth European Conference on Artificial Intelligence (ECAI90)*, pages 147–149, 1990.
6. J. DeRisi, V. Iyer, and P. Brown. Exploring the metabolic and genetic control of gene expression on a genomic scale. *Science*, 278:680–686, October 1997.
7. M. des Jardins, P. Karp, M. Krummenacker, T. Lee, and C. Ouzounis. Prediction of enzyme classification from protein sequence without the use of sequence similarity. In *ISMB '97*, 1997.
8. B. Efron and R. Tibshirani. *An introduction to the bootstrap*. Chapman and Hall, 1993.
9. M. Eisen, P. Spellman, P. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proc. Nat. Acad. Sci. USA*, 95:14863–14868, Dec 1998.
10. The Arabidopsis genome initiative. Analysis of the genome sequence of the flowering plant arabidopsis thaliana. *Nature*, 408:796–815, 2000.
11. International human genome sequencing consortium. Initial sequencing and analysis of the human genome. *Nature*, 409:860–921, 2001.
12. Aram Karalic and Vlado Pirnat. Significance level based classification with multiple trees. *Informatica*, 15(5), 1991.
13. D. Kell and R. King. On the optimization of classes for the assignment of unidentified reading frames in functional genomics programmes: the need for machine learning. *Trends Biotechnol.*, 18:93–98, March 2000.

14. R. King, A. Karwath, A. Clare, and L. Dehaspe. Genome scale prediction of protein functional class from sequence using data mining. In *KDD 2000*, 2000.
15. R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *IJCAI 1995*, 1995.
16. D. Koller and M. Sahami. Hierarchically classifying documents using very few words. In *ICML 97*, 1997.
17. E. Koonin, R. Tatusov, M. Galperin, and M. Rozanov. Genome analysis using clusters of orthologous groups (COGS). In *RECOMB 98*, pages 135–139, 1998.
18. A. Kumar, K.-H. Cheung, P. Ross-Macdonald, P.S.R. Coelho, P. Miller, and M. Snyder. TRIPLES: a database of gene function in S. cerevisiae. *Nucleic Acids Res.*, 28:81–84, 2000.
19. M. Lussier, A. White, J. Sheraton, T. di Paolo, J. Treadwell, S. Southard, C. Horenstein, J. Chen-Weiner, A. Ram, J. Kapteyn, T. Roemer, D. Vo, D. Bondoc, J. Hall, W. Zhong, A. Sdicu, J. Davies, F. Klis, P. Robbins, and H. Bussey. Large scale identification of genes involved in cell surface biosynthesis and architecture in *Saccharomyces cerevisiae*. *Genetics*, 147:435–450, Oct 1997.
20. A. McCallum. Multi-label text classification with a mixture model trained by EM. In *AAAI 99 Workshop on Text Learning*, 1999.
21. A. McCallum, R. Rosenfeld, T. Mitchell, and A. Ng. Improving text classification by shrinkage in a hierarchy of classes. In *ICML 98*, 1998.
22. H.W. Mewes, K. Heumann, A. Kaps, K. Mayer, F. Pfeiffer, S. Stocker, and D. Frishman. MIPS: a database for protein sequences and complete genomes. *Nucleic Acids Research*, 27:44–48, 1999.
23. D. Michie, D. J. Spiegelhalter, and C. C. Taylor, editors. *Machine Learning, Neural and Statistical Classification*. Ellis Horwood, London, 1994. Out of print but available at http://www.amsta.leeds.ac.uk/~charles/statlog/.
24. D. Mladenic and M. Grobelnik. Learning document classification from large text hierarchy. In *AAAI 98*, 1998.
25. S. Oliver. A network approach to the systematic analysis of yeast gene function. *Trends in Genetics*, 12(7):241–242, 1996.
26. J. R. Quinlan. *C4.5: programs for Machine Learning*. Morgan Kaufmann, San Mateo, California, 1993.
27. L. M. Raamsdonk, B. Teusink, D. Broadhurst, N. Zhang, A. Hayes, M. C. Walsh, J. A. Berden, K. M. Brindle, D. B. Kell, J. J. Rowland, H. V. Westerhoff, K. van Dam, and S. G. Oliver. A functional genomics strategy that uses metabolome data to reveal the phenotype of silent mutations. *Nature Biotech*, pages 45–50, 2001.
28. A. Ram, A. Wolters, R. Ten Hoopen, and F. Klis. A new approach for isolating cell wall mutants in *Saccharomyces cerevisiae* by screening for hypersensitivity to calcofluor white. *Yeast*, 10:1019–1030, 1994.
29. M. Riley. Systems for categorizing functions of gene products. *Current Opinion in Structural Biology*, 8:388–392, 1998.
30. R. Schapire and Y. Singer. BoosTexter: A boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168, 2000.
31. K. Sugimoto, Y. Sakamoto, O. Takahashi, and K. Matsumoto. HYS2, an essential gene required for DNA replication in Saccharomyces cerevisiae. *Nucleic Acids Res*, 23(17):3493–500, Sep 1995.
32. P. Törönen, M. Kolehmainen, G. Wong, and E. Castrén. Analysis of gene expression data using self-organizing maps. *FEBS Lett.*, 451(2):142–6, May 1999.
33. J. C. Venter et al. The sequence of the human genome. *Science*, 291:1304–1351, 2001.