Improving Progressive Sampling via Meta-learning on Learning Curves

Rui Leite and Pavel Brazdil

LIACC/FEP, University of Porto
Rua do Campo Alegre, 823, 4150-180 Porto
{rleite,pbrazdil}@liacc.up.pt

Abstract. This paper describes a method that can be seen as an improvement of the standard progressive sampling. The standard method uses samples of data of increasing size until accuracy of the learned concept cannot be further improved. The issue we have addressed here is how to avoid using some of the samples in this progression. The paper presents a method for predicting the stopping point using a meta-learning approach. The method requires just four iterations of the progressive sampling. The information gathered is used to identify the nearest learning curves, for which the sampling procedure was carried out fully. This in turn permits to generate the prediction regards the stopping point. Experimental evaluation shows that the method can lead to significant savings of time without significant losses of accuracy.

1 Introduction

The existence of large datasets creates problems for many data mining algorithms that are readily available. Memory requirements and processing times are often rather excessive. Besides, using all the data does not always lead to marked improvements. The models generated on the basis of a part of the data (sample) are often precise enough for the given aim, while the computational cost involved is incomparably lower.

These problems have motivated research in different data reduction methods. In this paper we are concerned with one particular data reduction method, which is oriented towards reducing the number of examples to be used, and is often referred to as sampling.

The aim of the sampling methods is, in general, to determine which proportion of the data should be used to generate the given model type (e.g. a decision tree). At the same time, we want the model to be comparable to the model that would be generated using all the available data. The existing methods can be divided into two groups: Static sampling methods and dynamic sampling methods [3]. As for the first group, the aim is generate a sample by examining the data, but without considering the particular machine learning algorithm to be used afterwards. Some researchers refer to this method as a *filter approach*.

In contrast to this the *dynamic sampling methods* take the machine learning algorithm into account. The final sample is determined by searching though the space of alternatives. The system explores the alternatives in a systematic

J.-F. Boulicaut et al. (Eds.): ECML 2004, LNAI 3201, pp. 250–261, 2004.

manner and the performance of the machine learning algorithm is used to guide the future search. Some researchers refer to this method as a *wrapper approach*. It was shown that the dynamic (wrapper) methods obtain in general better results than the static (filter) methods, although they tend to be slower [3].

One particular dynamic method that can be used in conjunction with large datasets is called *efficient progressive sampling* [2]. The method starts with a small data sample and in each subsequent step uses progressively larger sample to generate a model and to check its performance. This continues until no significant increase in accuracy is observed. One important characteristic is the size of the samples used in each subsequent step. The sizes follow a geometric progression. Another important aspect is how convergence is detected. The authors use a method referred to as LRLS (linear regression with local sampling). This method works as follows. Supposing the algorithm is examining sample n_i, LRLS uses 10 samples of similar size to generate models and estimate their accuracies. These estimates are supplied to linear regression algorithm and the inclination of the resulting line is examined. If it is about horizontal (i.e. the inclination is sufficiently near to zero), the process of sampling is terminated. As it was shown by the authors, this method worked well with the large datasets considered. However, a question arises when exactly this method is useful.

We have re-implemented a similar method and used it on a conjunction of both large and medium size datasets. We have verified that in many medium size datasets the method required more time than a simple scheme that would learn from all the data. This is easy to explain. The method constructs a succession of models using progressively increasing samples. However, in many cases the accuracy will simply keep increasing and hence the stopping condition will not be satisfied. This means that the algorithm will process all the data, but with an additional overhead of using a succession of increasing samples beforehand.

Our aim was to improve the method so that it could be applied to any dataset, no matter what its size is. The basis strategy relies on eliminating some samples from consideration. We use previous knowledge about the algorithm itself, that is, meta-learning on past results. This is justified by quite good previous results with this technique [6].

The rest of the paper is organized as follows. Section 2 describes the proposed method in detail. Section 3 describes the evaluation method and experimental results obtained. Finally, we present the conclusions.

2 Predicting the Stopping Point in Sampling

Dynamic sampling methods use a succession of models generated by a given learning algorithm on the basis of a sequence of progressively increasing samples. The aim is to determine the point in which the accuracy does not increase any more. We call this point a stopping point. Fig. 1 shows a typical learning curve and the stopping point is represented by p^*. Our aim is to predict the point p^* using an initial segment consisting of #p points. Let us examine again the learning curve represented in Fig. 1. Suppose the points p_1, p_2, p_3 and p_4

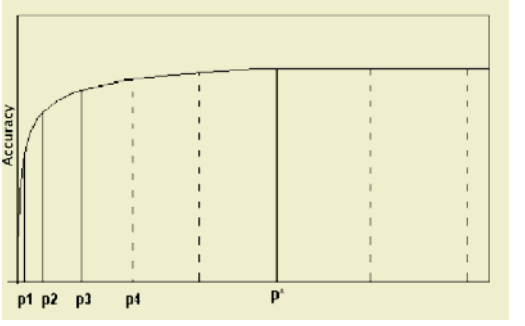

Fig. 1. Learning Curve

constitute the initial segment. So, our aim is to estimate the stopping point using these four points, without considering the points further on.

The prediction of p^* is done on the basis of previous knowledge about the algorithm in question. The knowledge used is in the form of learning curves obtained before on other (similar) datasets. The aim is to use these curves to predict the stopping point on a curve that is only partly known (we have information about the initial segment only).

The details of this method are described in the following. First, we will discuss how the learning curves are represented. Then, we will show how certain learning curves are identified on the basis of existing information for the purpose of prediction. Finally, we show how the prediction of the stopping point is generated. The reader can consult Fig. 2. for an overview of the method.

2.1 Representation of Learning Curves and Identifying the Stopping Point

Suppose we have datasets $\{D_1, D_2, ..., D_n\}$ and for each one we have a learning curve available (later we will discuss a variant of this basic method which uses N learning curves per dataset). Each learning curve is represented by a vector

Algorithm
Input:
    i (dataset in question), A (database of n learning curves)
Parameters:
    #p (size of the initial segment)
    k (number of neighbors), ϵ (tolerance)
Run given algorithm on dataset i
    while varying samples from m=1 to #p.
Calculate accuracies $A_{i,m}$ (partial learning curve)
Analyze the learning curves stored and
    calculate distances $d_{i,j} (j = 1..n)$
Identify k curves with the smallest distance and the stopping points $p^*_{n_1} ... p^*_{n_k}$
Combine predictions of stopping points
    to obtain $\hat{p}^*_i$ and return this value

Fig. 2. The basic algorithm for predicting stopping points

$< A_{i,1}, A_{i,2}, .., A_{i,z} >$, where $A_{i,m}$ represents the accuracy of the given algorithm on dataset D_i on m-th sample in the sequence. Following Provost et al. [2] the sizes follow a geometric progression. The sequence spans across the whole dataset.

The particular stopping point p_i^* for dataset i can be readily identified. This is done as follows. First, we identify the global maximum using $A_{i,pmax} = max(A_{i,j})$. Then, given a tolerance ϵ, we identify the earliest point in the sequence whose accuracy is within the tolerance limit of the global maximum. This can be formulated as follows:

$$p_i^* = min\{n : |A_{i,pmax} - A_{i,n}| < \epsilon\} \tag{1}$$

2.2 Identification of Appropriate Learning Curves for the Purpose of Prediction

Suppose we are interested in dataset D and we have information about the initial segment of the learning curve (e.g. the first #p=4 points). We employ a nearest neighbor algorithm (k-NN) to identify similar datasets (as in [6]) and retrieve the appropriate learning curves. Here the k-NN algorithm represents a meta-learner that helps us to resolve the issue of predicting the stopping point. As k-NN uses a distance measure to identify k similar cases, we need to adapt the method to our problem. Here we just use the information concerning the initial segment. The distance function between datasets D_i and D_j is defined by

$$d(i,j) = \sum_{m=1}^{\#p} (A_{i,m} - A_{j,m})^2 \tag{2}$$

where m spans across the initial segment.

2.3 Generating the Prediction Concerning the Stopping Point

Once k learning curves have been identified, we can generate the prediction regards the stopping point on a new curve. This is done by retrieving the stopping points associated with k learning curves and generating a prediction using this information. Let us see how this is done in detail.

Let the associated indeces of the k most similar learning curves be n_1, $n_2, ..., n_k$. Then let the stopping points of each curve be $p_{n_1}^*, p_{n_2}^*, ..., p_{n_k}^*$. In general, the values can differ. One obvious way to estimate the stopping point p_i on the basis of this information is by using the *median* (or the *mean*) value[1].

2.4 Using Aggregated Learning Curves

It is a well known fact that the performance of many algorithms may vary substantially, as data is drawn from a given source. This phenomenon is usually

[1] In all experiments reported here we have used the *median*, as it is less sensitive to outliers.

referred to as variance [4]. The problem is even more apparent if we use small samples. As a consequence, the learning curves do not always look like the one shown in Fig. 1 which is monotonically increasing. The curves obtained from real data often include points that appear to jump up and down. This has an adverse effect on the method described earlier.

To minimize this problem we have decided to generate a smoothed-out curve on the basis of N learning curves per dataset. Each individual learning curve is obtained using a different portion of the data, using a method similar to N cross-validation. Each point $A_{i,m}$, the m-th point of smoothed curve for dataset i, represents the mean of the corresponding points of the individual learning curves.

In the following the method described in this section is referred to shortly as MPS (meta-learning + progressive sampling).

3 Empirical Evaluation

To evaluate the method MPS proposed above we have used the leave-one-out evaluation strategy. We identify a dataset, say D_i, and the aim is to predict the stopping point for this dataset. All other datasets except D_i (and with the associated initial segments) are used to generate the prediction $\hat{p}_i^*$, in the way described earlier. The predicted stopping point is compared to the true stopping point (retrieved from our database). Besides, we also compare the errors associated with the two stopping points and the times used to obtain each solution.

We have used 60 datasets in the evaluation. Some come from UCI [1], others were used within project METAL [5]. All datasets used are shown in Table 2 in the Appendix.

The samples are generated using a geometric progression as follows. The size of m_i-th sample is set to the rounded value of $2^{6+0.5\times m_i}$. Thus the size of the first sample is $2^{6.5}$, giving 91 after rounding, and the second sample is 2^7, giving 128 etc. Table 1 shows the relationship between the sample number and the actual sample size.

Table 1. Relationship between the sample number and the actual sample size

m	1	2	3	4	5 ...	10	15	20	25
size	91	128	181	256	362 ...	2048	11585	65536	370728

We have used C5.0 [8] as the base algorithm. That is, our aim was to predict the stopping point of C5.0 on the basis of the initial segment. In the experiments reported here the initial segment included 4 points (#p=4). The tolerance limit ϵ was set to 0.001. In the experiments presented here we also used the dataset size as a predictive attribute. For each dataset we have retrieved a smoothed-out curve.

Regards the meta-learning method, we have used k-NN. In the experiments reported here k was set to 3[2].

3.1 Results Concerning Savings and Losses

The results obtained are shown in Fig. 3. As we can see, there is on the whole quite good agreement between the predicted stopping point and the true value. Here we use the re-scaled values shown in Table 1. The points can be divided into three groups. The first one includes perfect predictions ($\hat{p}_i = p_i^*$). The second group includes all cases for which $\hat{p}_i < p_i^*$. That is, if we followed the prediction, the sampling process would terminate somewhat prematurely. In general, one would expect that this would affect the error of the base algorithm (in general the error will be a bit larger than it would be, if it terminated at the right point). The third group includes all cases for which $\hat{p}_i > p_i^*$. In general, this will not affect the error, unless of course, the base algorithm suffers from overfitting.

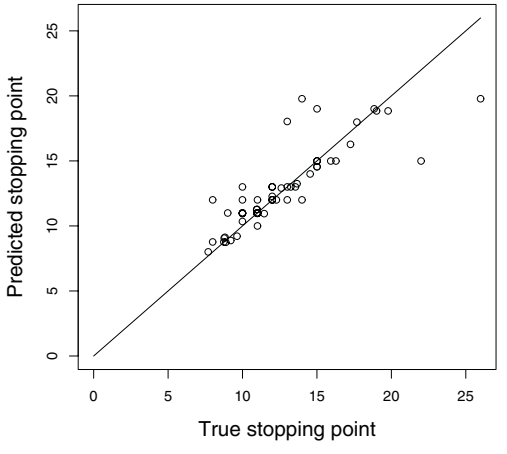

Fig. 3. Comparison between predicted and true stopping points

We can analyse the situation in Fig. 3 more closely and examine the differences between the predicted and the true value and calculate The Mean Absolute Error (MAE). This calculation gives the value 1.04. In other words, our predictions are about 1.04 steps off the ideal value.

Let us now see what would happen if we used a fixed prediction throughout. The best fixed prediction is the mean of the true stopping points (12.59). If we used this, the Mean Absolute Error (MAE) would be 2.67. This value is substantially larger than the value obtained using the method MPS.

We can analyse the computational savings achieved. We compare two situations. One involves using the traditional progressive sampling method while

[2] This value lead to the best results. Later on we discuss this issue further.

trying to identify the true stopping point. In general we need to run through at least p_i^* points.

The second situation involves our method, that is training the base algorithm on #p=4 points to be able to obtain the predicted stopping point. In addition, we need to train the base algorithm on the corresponding sample. So we can compare how many points we effectively skip and this gives an indication of the computational savings. If we carry out the calculations, we see that on average the savings is 7.6 points (varying between 2 and 20). That is, our method avoids constructing and evaluating at least 7x10 classifiers on average when compared to the progressive sampling method.

3.2 Results Concerning Actual Times and Accuracies

The analysis presented so far was oriented towards comparing the predicted stopping point with the actual one. In this section we will provide figures concerning actual times, and also, analyse the impact of being off target on accuracy of the base algorithm.

In Fig. 4 we compare the times of two approaches. The first one is our method (MPS), which requires training $\#p+1$ classifiers (vertical axis). The second one is the baseline method representing a simplified version of the progressive sampling method [2] (horizontal axis). As can be seen in practically all datasets the method leads to time significant savings. Our method is 12.25 faster on average.

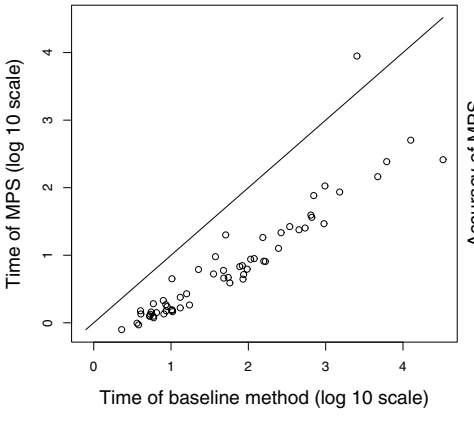

Fig. 4. Comparison of total training times

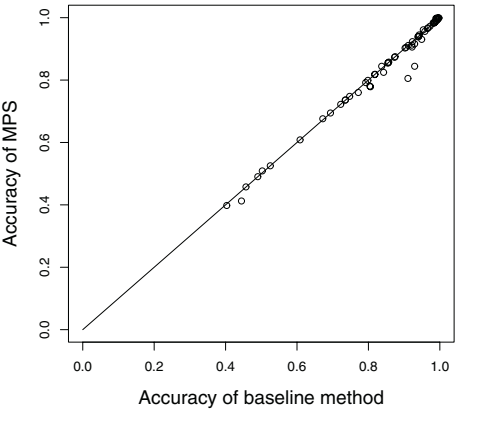

Fig. 5. Comparison of accuracies

The comparison of accuracies of the two methods for various datasets is shown in Fig. 5. The differences in accuracies for the two methods are relatively small. However, as could be expected, the accuracy of our method is a bit lower than the accuracy of the baseline method. On average the difference is 0.51%. This could be considered as the price to pay for the speed-up.

Closer analysis shows that for 32 datasets the difference is zero. On the other hand, in few datasets there is a noticeable difference. This is due to the fact that the method has identified a stopping point which is in fact premature.

4 Parameters of Our Method: The Values Used and Future Work

The method described involves various parameters. Our aim here is to briefly review the set of parameters involved and justify why certain choices were made and discuss futher work.

As has been pointed out earlier, the method requires that experiments be conducted on different datasets. The aim of these is to obtain learning curves that are stored for future use together with the true stopping point. As each learning curve is represented by a sequence of points, we need to determine how the learning curves are represented, how many learning curves are constructed per dataset and how the true stopping point is identified.

Given a new dataset, the method uses a k-NN to identify the most similar cases. In this step we need to determine the size of the initial segment of the learning curve in the matching procedure and which characteristics of the dataset should be taken into account. Finally we need to set the value of k in the k-NN procedure.

All these parameters can be varied and we could study what the effects of these variations on the overall result. We have done some studies to this effect, but obviously an exhaustive study is not practicable. In the following we give a short overview of our position on these issues and point out to further work that could be carried out.

Choice of Datasets: The study carried out by Provost et al [2] was limited to relatively few large datasets. We have used many more datasets (60) here, but we did not follow any particular strategy when selecting these. Further work could be carried out to see what the results would be if we focused the study on certain datasets only (e.g. datasets above certain size or satisfying some other criterion).

Representation of the Learning Curve: Each learning curve is represented by a sequence of points. The sample sizes follow a geometric progression. Both the initial size (91 cases) and the increment represent parameters of the method are considered fixed. Other settings could be tried in future, although we do not think the results could be improved dramatically this way. Besides, instead of saving point-to-point information about learning curves, one could take a model-based approach. In principle it would be possible to fit a predefined type of curve through the points and save the curve parameters. The distance measure could then be redefined accordingly. As the curve fitting is subject to errors, it remains to be seen, whether this approach would lead to better results.

Number of Curves Constructed per Dataset: We have used both a single curve and N=10 curves per dataset. As has been pointed out earlier, the N curves were compacted into a single aggregated smoothed-out curve. The results with this curve (representing 10 individual curves) were much better than the results with a single curve. The number of curves (10 in our case) is a parameter of the method. Further work could be done to determine the advantages / disadvantages of using other values. In one earlier study [7] we have used also 10 curves per dataset, without generating a smoothed-out curve. The k-NN matching procedure was more complex. The initial segment obtained on a new dataset was then matched against all the individual curves and average distance calculated. The overall results were comparable to the results presented here. The advantage of using smoothed-out curves is that the matching procedure is much simpler. A more comprehensive comparison could be carried out in future. Besides, we could try to establish whether one method is significantly better than another.

Detection of the True Stopping Point: As has been described earlier the detection of the true stopping point involves constant ϵ, which was set to 0.001 (the differences of accuracy less than this value are considered insignificant). The choice of this value affects both the position of the stopping point on a curve and the overall precision of the method. If a larger value were chosen (e.g. 0.01), the stopping point would, in general, appear earlier. More work could be done to characterize the effect of these choices quantitatively.

Size of the Initial Segment of the Learning Curve Used in Matching: As has been pointed out earlier, the size of the initial segment was set to 4 points. We have experimented with other different values. Reducing the number (e.g. to 1,2 or 3) led, on the whole, to comparable or inferior performance. Increasing the number did not seem to bring further benefits and this is why we have settled for the value of 4.

Using Data Characteristics: In this work we have used not only the learning curves, but in addition, used one particular characteristic of the dataset, which is dataset size (i.e. number of cases). Dropping this attribute led to marked decrease of performance (MAD would rise from 1.04 to 1.89).In future we intend to investigate whether some other characteristics could be useful (e.g. number or entropy of classes etc.).

Weights of the Initial Segment and the Dataset Characteristics: As the k-NN matching procedure uses both the initial segment consisting of N points and one characteristic - the dataset size - it is important to determine the weights that should be attributed to each of these items in the k-NN matching procedure.

We have begun by using equal weights for all parts, but then found that this was not the best setting. To our surprise the best results were obtained when relatively large weight was attributed to dataset size (94%), and relatively little weight to the initial segment (6%). Despite its rather small weight, the initial

segment was important. If it were dropped all together (corresponding to giving it weight 0), the overall MAD value would rise from 1.04 to 1.2.

An interesting question arises why the dataset size appears to be so important. We have carried out a study to clarify this. The results are shown in Fig. 6, showing where the stopping points lie for different datasets. Each dataset is represented by a point positioned at a particular coordinate X,Y. The X coordinate (horizontal axis) corresponds to the dataset size and Y coordinate (the vertical axis) to the stopping point. Both values are rescaled values, expressed in terms of the points in the geometrical progression adopted (see Table 1 for details on re-scaling). As can be seen many points lie on the diagonal. These are the datasets for which the stopping point lies exactly at the end. In all these cases the best thing to do is to use all the data. The finding above suggests that

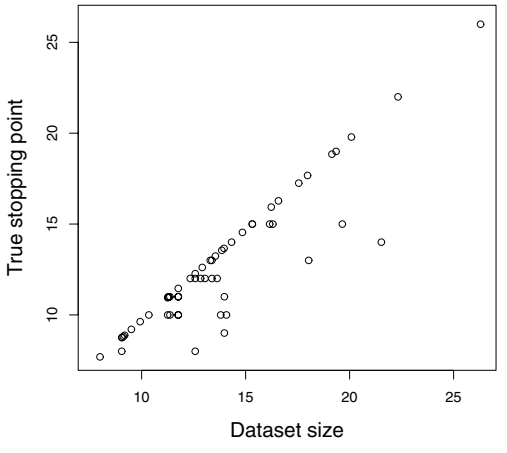

Fig. 6. Dataset sizes vs stopping points (both expressed in terms of the number of the sample)

we could use a simplified method probably without deterioration of performance. If we can confidently classify a case using a k-NN on the basis of the dataset size only, we could skip construction of the initial segment. Future work could be carried out, to evaluate how this would work in practice. Another interesting issue is how well the method would work if we focussed the attention on large datasets only, where presumably the stopping point does not coincide with the full data. This will be investigated in future.

Value of k in the k-NN Procedure: In our experiments we have used the value k=3. We have experimented with other different values (both lower and higher than 3), but the results were on the whole comparable or inferior to the ones obtained with the setting used. These results could be validated further by conducting further experiments.

5 Conclusions

We have described a method that can be seen as an improvement of the standard progressive sampling. We have been concerned with the issue of how to avoid using some of the samples in this progression. We have employed a meta-learning approach that enables us to predict the position of the stopping point. The method requires just four iterations of the progressive sampling and the information gathered is used to identify the nearest learning curves. This in turn permits to generate the prediction regards the stopping point.

We have carried out experimental evaluation of the method using 60 datasets. We have shown that the method can lead to significant savings of time. The experimental results indicate that it is possible to skip 7 or 8 samples on average, leading to significant savings of time. On average our method is 12.25 faster than the standard method. The accuracy of the method presented is a bit lower, but this is an acceptable price to pay for the speed-up.

The work carried out led to some unexpected surprises, however. We have found that some dataset characteristics, such as dataset size, are quite informative and help to improve the results. We have carried out a study that helps to explain why the dataset size appears to be important. An interesting issue arises whether there are other characteristics that could be used, which would work even better, this should be investigated in future.

Acknowledgments

The authors wish to thank anonymous referees for their useful comments. The authors gratefully acknowledge the financial support from FCT within multi-annual funding (Financiamento Pluriannual) attributed to the Portuguese R&D Units.

References

1. Blake C.L. and Merz C.J. UCI repository of machine learning databases, 1998.
2. Provost Foster J., Jensen David, and Oates Tim. Efficient progressive sampling. In *Knowledge Discovery and Data Mining*, pages 23–32, 1999.
3. John George H. and Langley Pat. Static versus dynamic sampling for data mining. In Evangelos Simoudis, Jiawei Han, and Usama M. Fayyad, editors, *Proc. 2nd Int. Conf. Knowledge Discovery and Data Mining, KDD*, pages 367–370. AAAI Press, 2–4 1996.
4. Breiman L. Bias, variance, and arcing classifiers. Technical Report 460, Statistics Department, University of California, 1996.
5. Metal project site. http://www.metal-kdd.org/.
6. Brazdil P., Soares C., and Costa J. Ranking learning algorithms: Using IBL and meta-learning on accuracy and time results. *Machine Learning*, 50:251–277, 2003.
7. Leite R. and Brazdil P. Improving progressive sampling via meta-learning. In *Progress in Artificial Intelligence, 11th Portuguese AI Conference (EPIA03)*. Springer-Verlag, 2003.
8. Quinlan R. C5.0 "an informal tutorial". RuleQuest, http://www.rulequest.com/see5-info.html, 1998.

Appendix

Table 2. Datasets used

dataset	n cases	dataset	n cases	dataset	n cases
acetylation	1511	connect.4	67557	Adult	32560
covtype	581012	Byzantine	17750	dis	3772
contraceptive	1473	heart.disease.clev.	1541	dna.splice	3186
hypothyroid	3163	ibm.stock.val	8087	isolet	7797
injury.severity	7636	krkopt	28056	internetad	3279
kr.vs.kp	3196	led24	3200	letter.recognition	20000
led7	3200	mfeat	2000	mushrooms	8124
musk.clean2	6598	mushrooms.exp	8416	nettalk	146934
musk	6598	nursery	12960	parity	1024
optdigits	5620	quisclas	5891	page.blocks	5473
recljan2jun97	33170	pendigits	10992	task1	111077
pyrimidines	6996	taska.part.hhold	17267	quadrupeds	5000
taska.part.related	18254	sat	6435	taskb.hhold	12934
segmentation	2310	ad	3279	shuttle	58000
adult	48842	sick	3772	agaricus.lepiota	8124
sick.euthyroid	3163	allbp	3772	spambase	4601
allhyper	3772	splice	3190	allhypo	3772
thyroid0387	9172	allrep	3772	triazines	52264
ann	7200	waveform21	5000	car	1728
waveform40	5000	cmc	1473	yeast	1484