Imagine that this forms a single case in a learner's testing set. We consider this case to be *unseen* for the learner if it does not appear in the corresponding training set. But what if the following case appears in the training set?

```
0.000 yes 3.444 left up 2 119 8.343 72 t 65.225 f
```

Although the two cases seem identical there is a small difference: the seventh value is 8.343 rather than 8.342. Technically, then, we can still treat the original case as unseen. But in doing so we may feel a little uncomfortable. An extremely close approximation of the unseen case exists in the training set. Any learner which sees all the cases in the training set has virtually seen the unseen case.

Where such 'virtual seen' cases exist within testing data, measures of generalisation performance may be misleading. Ideally, experimenters should eliminate virtual seens from any testing data before usage. This may involve ensuring that every test case has a sufficient level of *dissimilarity* with every training case.

Where

(2) Construct the testing set by randomly selecting (without replacement) the appropriate number of cases from the dataset.

(3) Form the training set out of the remaining cases.

(4)

The 1-NN algorithm used for this experiment used basic 'city-block' distance measure. The distance $D(A, B)$ between two cases A and B was defined to be

$$D(A, B) = \sum_{i=1}^{n} d(A_i, B_i)$$

where $d(A_i, B_i)$ was the normalised numeric difference between A_i and B_i if both values were numeric, and the number of explicit character differences expressed as a fraction of the length of the shortest string, if both values were strings (i.e., symbolic values). In the case of one of the values being missing, the difference was defined as 1/10 of the maximum difference.

The generalisation performance achieved by the 1-NN algorithm using 2/3-sized training sets (the size Holte used) is shown in Table 1. The performance of C4.5 is also shown.

Dataset	BC	CH	GL	G2	HD	HE	HO	HY
1R	68.7	67.6	53.8	72.9	73.4	76.3	81.0	97.2
C4.5	72.0	99.2	63.2	74.3	73.6	81.2	83.6	99.1
1-NN	69.7	90.1	70.1	80.6	78.1	79.3	78.5	96.9
Dataset	IR	LA	LY	MU	SE	SO	VO	V1
1R	93.5	71.5	70.7	98.4	95.0	81.0	95.2	86.8
C4.5	93.8	77.2	77.5	99.9	97.7	97.5	95.6	89.4
1-NN	94.6	85.8	76.8	100.0	87.9	100.0	93.1	88.1

The performance data for 1-NN and C4.5 are shown in Figure 1 in graph form. Interestingly, the 1-NN algorithm produced performance which was either comparable or superior to C4.5 in seven of the 16 cases. In the remaining nine cases the performance was on average no more than 3 percentage points worse than that of C4.5. In all cases the performance was superior to that of Holte's 1R algorithm.

The measured performance of 1-NN algorithm in this study appears to be broadly compatible with its performance (or the performance of a K-NN variant) as reported in similar studies such as [Aha and Kibler, 1989] and [Henery, 1994]. However, the performance obtained in this study is in general superior to that reported by Weiss and Kapouleas [1989]. They recorded a mean generalisation

This result is in agreement with Friedman's analysis [Friedman, 1994] which explains the surprising robustness of NN methods against the so-called 'curse of dimensionality' in terms of the redundant distributional properties of common datasets. It is also in agreement with the general implications of Holte's study. Holte showed that very simple learning processes can produce good performance on these problems. The present study has shown much the same thing. But of course 1-NN and 1R are 'simple' in different ways. 1R attempts to construct a rule based on observations on the *minimum* number of attributes. 1-NN on the other hand uses a rule which takes into account observations on *all* the available attributes. Thus the results of this study show that the Holte datasets are simple in at least two different senses.

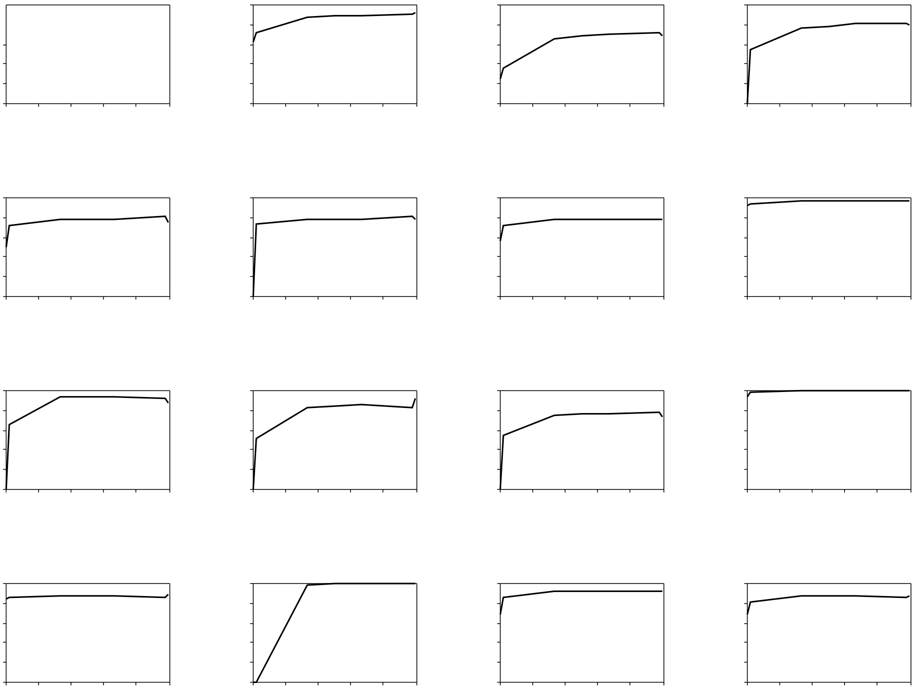
3 The effect of varying the training set proportion

To try to get a better idea about the reasons for the rather small difference between the performance of 1-NN, 1R and C4.5 on the Holte datasets, experiments were carried out to determine the average performance of the 1-NN algorithm on a *range* of training set sizes. The performance of the algorithm was in fact sampled on training sets built by randomly choosing 0.5%, 2%, 33% (1/3) and 66% (2/3), 98% and 99.5% of the original cases. The generalisation performance was then averaged over 50 runs at each training set size. The results of these experiments are displayed in Figure 2.

In general, one expects the performance of the NN algorithm to increase with the size of the training set. The performance should be very poor if the training set is nearly empty and very good if the training set contains nearly all the possible cases. Thus, given the training set proportions used in this survey, we expect generalisation curves to approximate an upwards sloping diagonal. In fact, *none* of the curves shown in Figure 2have this form. The curve for the GL dataset is perhaps the best approximation. But in general the curves are remarkably *flat*.

The implications of this are worth some consideration. In order for a dataset to have a high, flat generalisation curve, it is essential that the 1-NN algorithm performs well on nearly empty training sets, i.e., training sets which include only a small proportion of the dataset. But we should only expect this to occur if the data are highly organised, i.e., if the classes in the data are very cleanly separated. In this situation any example taken from a class can serve as an exemplar for the class and thus provide a 1-NN algorithm with an effective representation of that class. Thus a very few examples may well suffice to produce excellent performance from the 1-NN algorithm.

Of course, even with clean separation of classes, a 1-NN algorithm cannot produce good performance unless the training set contains at least one exemplar



4 Summary and Conclusion

In some cases, the instances that we present to a learner may be incomensurable and thus impossible to test for similarity. More frequently, there is an explicit or implicit distance metric over instances. In this situation, a given testing set may contain very close approximations of cases from the training set. The paper has described such cases as ‘virtual seens’ and noted that generalisation statistics derived in the presence of virtual seens may be misleading or ambiguous.

The performance of the 1-NN algorithm can be used to derive a generalisation baseline against which true or *relative* generalisation can be measured. This approach was demonstrated though an application involving Holte’s comparative study of the performance of 1R and C4.5 on 16 commonly used datasets from the UCI repository. The results of this experiment revealed that most of the datasets in the Holte selection contain data showing *extremely* clean separation between classes. For all the Holte benchmarks, the performance achievable through ‘lookup’ of virtual seen cases is extremely close to the performance level achieved by learning methods such as C4.5. We have to conclude therefore that these datasets do not pose a substantive tests of generalisation. If we equate learning ability with generalization ability then we have to conclude that these datasets do not effectively test anything that we can meaningfully call ‘learning’.

This conclusion is a little startling given the central role that the UCI datasets have played in the evolution of Machine Learning methods. However, the wider implications are hard to trace out. Certainly, we can dispense with the oft-stated assumption that ‘real-world’ problems are necessarily challenging for learning methods. All of the Holte datasets are derived from the ‘real-we608010Tdgc0.57(w)59(-324.24

[4] Bergadano, F., Kodratoff, Y. and Morik, K. (1992). Machine learning and knowledge acquisition: summary of reserach contributions presented at IJCAI'91. *AI Communications*, *5*, No. 1 (pp. 19-24).

[5] Holte, R. (1993). Very simple classification rules perform well on most commonly used datasets. *Machine learning, 3* (pp. 63-91).

[6] Quinlan, J. (1993). *C4.5: Programs for Machine Learning.* San Mateo, California: Morgan Kaufmann.

[7] Friedman, J. (1994). *Flexible Metric Nearest Neighbor Classification.* Unpublished MS.

[8] Henery, R. (1994). Review of previous empirical comparisons. In D. Michie, D. Speigelhalter and C. Taylor (Eds.), *Machine Learning, Neural and Statistical Classification.* Ellis Horwood.