The problem of bias in training data in regression problems in medical decision support

B. Mac Namee*, P. Cunningham*, S. Byrne**, O.I. Corrigan**

*Department of Computer Science,
**Department of Pharmaceutics,
Trinity College, Dublin 2.

Abstract

This paper describes a bias problem encountered in a machine learning approach to outcome prediction in anti-coagulant drug therapy. The outcome to be predicted is a measure of the clotting time for the patient; this measure is continuous and so the prediction task is a regression problem. Artificial Neural Networks (ANN) are a powerful mechanism for learning to predict such outcomes from training data. However experiments have shown that an ANN is biased towards values more commonly occurring in the training data and is thus less likely to be correct in predicting extreme values. This issue of bias in training data in regression problems is similar to the associated problem with minority classes in classification. However this bias issue in classification is well documented and is an on-going area of research. In this paper we consider stratified sampling and boosting as solutions to this bias problem and evaluate them on this outcome prediction problem and on two other data sets. Both approaches produce some improvements with boosting showing the most promise.

1. Introduction

This paper describes a bias problem in research on using Artificial Neural Networks (ANNs) in decision support in anti-coagulant drug therapy (Byrne et a. 2000). A central objective of the research is to predict a patient’s INR reading given a particular dose of the anti-coagulant drug warfarin. The INR reading is an international standardised method of reporting a patient’s prothrombin time (the time it takes for the patient’s blood to clot). ANN models have been developed to perform this regression task with very good results – the prediction of the ANN is on average 30% more accurate than that of practising physicians (Byrne et al. 2000). It is clear from a detailed analysis of the results from the ANN model that its accuracy is not uniform over the range of possible INR values. This can be seen in Figure 1 where, high in the INR range, average errors are negative while errors low in the range errors are positive. This bias toward predicting commonly occurring outcomes arises from the dominance of these outcomes in the training data; the distribution of the training data is shown in Figure 2.

This bias is a fundamental characteristic of the neural network models; however there has been very little research on this issue in a regression context. There is however a considerable body of research on skewed distributions in classification problems[1]. This is also referred to as the minority class problem and a brief review of research on this can be found in (Kubat et al. 1998). Decision trees and nearest neighbour classifiers will perform badly predicting outcomes of the minority class if the majority class is dominant in the training data.

The first solution considered here to the problem of bias in regression data is one that is dominant in the classification literature. It is commonly termed stratified sampling and involves differentially sampling the training data so as to balance (flatten) the distribution shown in Figure 2. This can be achieved by replicating the training data from the tails of the

[1] In a classification problem the outcome to be predicted is a class or category whereas in regression the outcome is numeric.

distribution or by discarding some of the data from the centre. Clearly the risk with this is that performance on tails may be improved at the expense of the commonly occurring cases. This brings us to an issue discussed in detail in (Provost & Fawcett, 2000) (Provost, et al. 1998). The "best" classifier for a given problem cannot be determined without considering the costs of different misclassification possibilities (Pazzani et al. 1994). Which is worse, a missed positive diagnosis or a false positive diagnosis? If, for instance, false negatives are determined to be three times as serious as false positives then a "best" classifier that minimises the total cost of errors can be pursued. Provost et al. (1998) acknowledge that in practical situations quantifying these costs is impractical or impossible. Instead they advocate looking for classifiers that dominate according to the ROC criteria they present (ROC stands for Receiver Operating Characteristic, see section 2 for details).

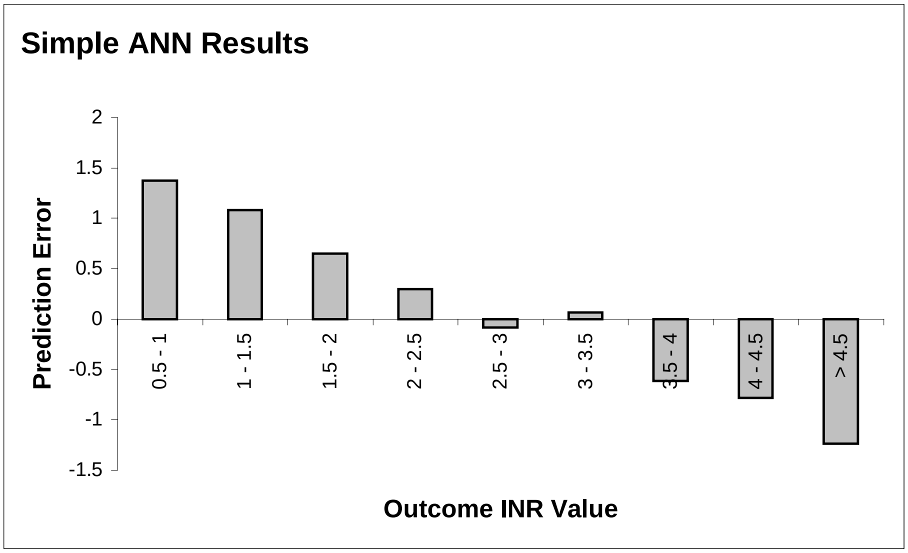

Figure 1. A graph of the prediction errors produced from a 6-fold cross validation using a simple ANN network trained with the warfarin data set. Average prediction errors are presented for different ranges of expected INR values.

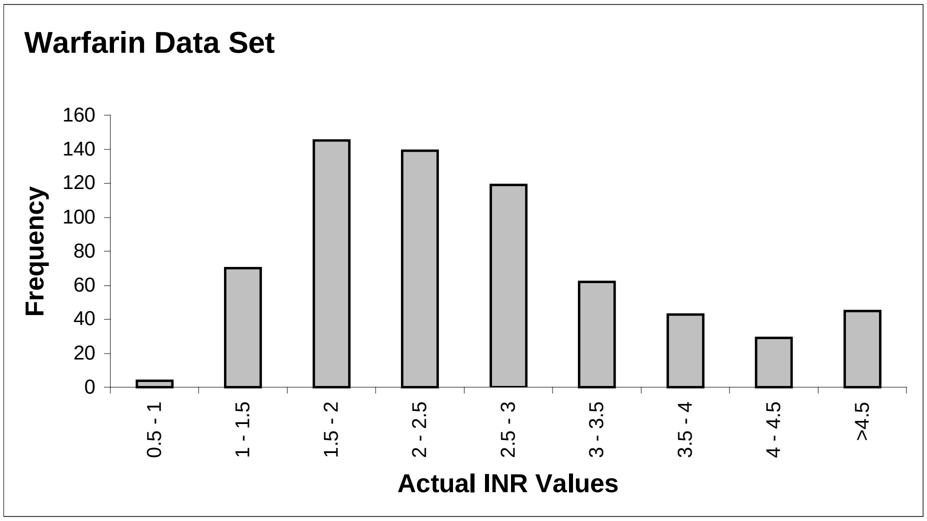

Figure 2. A histogram of the outcome INR values in the warfarin data set .

The problem in regression is essentially the same, it may be more important to predict accurately the rare events in the tails of the distribution that the commonly occurring outcomes. However, it is difficult to quantify this difference in importance. Because of this, the evaluation criteria used here are similar in spirit to that proposed by Provost et al. (1998). One model dominates another if it improves performance in the tails of the distribution *without* damaging performance in the middle (see Figure 12). We are 'indifferent' to models that improve performance on the tails while damaging performance in the middle of the distribution (see Figure 15). Clear dominance must be shown.

In the next section the problem of bias in training data is explored in detail. The emphasis is on classification rather that regression because most of the relevant research has addressed classification where the problem is more clearly visible. In section 3 the techniques we propose for tackling the problem are described. An evaluation of these techniques on the warfarin dataset and two other datasets is presented in section 4. The paper concludes in section 5 with an assessment of the applicability of the techniques.

2. Bias in Training Data

2.1. The Minority *Class Problem*

The minority class problem is an issue in domains where one class accounts for a significant majority of observations. (Breiman et al, 1984) (Lewis & Catlett, 1994) (Turney, 2000) (Provost et al. 1998) (Provost & Fawcett, 2000) (Chan & Stolfo, 1998). It is a common problem in medical domains for example the *in-vitro* fertilisation problem reported in (Cunningham et al. 2000) where unsuccessful outcomes dominate successful ones by a factor of 3:1. In medical diagnosis problems positive examples may account for less, sometimes much less, than 5% of the available cases. Skewed class distributions also occur in fraud detection tasks and in natural language processing (Cardie & Howe, 1997).

These skewed class distributions presents two immediate problems:

1. It is difficult to assess the quality of the classifier. Using simple accuracy is not adequate; consider a scenario where positive examples count for 5% of the data. A classifier that outputs the negative class 100% will be 95% accurate but 0% accurate on positive examples that may be the more important.
2. The classifier may be biased towards predicting the majority class resulting in poor accuracy on the minority class. This will happen if the classifier is a decision tree, a neural network or a nearest neighbour classifier for example.

Considering the quality assessment issue first, perhaps the most robust measure of the performance of a classifier is the area under the ROC curve (Swets 1988). The ROC curve is a plot of true positives against false positives for a family of classifiers. Many classifiers can be tuned to adjust the trade-off between false positives and false negatives. For example, this can be done in a neural network by adjusting the threshold in the output neuron. So the different values of this threshold represent a family of classifiers with different numbers of false positives and false negatives. The ROC curve represents this family of classifiers. In this view a classifier from this family has two components, the discrimination mechanism and the threshold mechanism. The discrimination mechanism is the same for the whole family and the specific threshold mechanism defines an individual classifier. Swets (1988) proposes that the area under the ROC curve is a good measure of the discriminating power and thus the performance of a classifier.

This ROC analysis still does not identify a best classifier however. We need to commit on the trade-off between false positives and false negatives to identify a specific classifier from the family represented by the ROC curve.

An alternative approach to performance measurement is to borrow the measure of geometric mean of *precision* and *recall* as used in information retrieval research. In binary classification this would be:

$$g = \sqrt{acc_{+} \times acc_{-}}$$

where acc_{+} is accuracy on positive examples and acc_{-} is accuracy on negative examples. The geometric mean has the advantage that it is non-linear and punishes big disparities between positive and negative accuracies. (Kubat et al. 1998).

This second issue of bias towards the majority class is essentially the same as the problem of bias in regression data that we are concerned with in this paper. Various solutions have been proposed such as stratified sampling (Buntine, 1989), case-specific feature weights (Cardie & Howe, 1997) and algorithms that are intrinsically insensitive to skewed class distributions (Kubat et al. 1998). For our regression problem we consider stratified sampling and boosting which might be viewed as an alternative sampling approach where the data distribution is adjusted dynamically to focus on samples that are difficult to learn.

Another important issue for consideration in minority class problems in the number of minority samples available. It may be that the prior probability distribution is skewed but data in abundant and there are several hundreds or thousands of examples of the minority class available. In which case it may be sufficient to reduce the size of the set of majority examples to that of minority examples and proceed as normal. The more difficult (and typical) case is where training data is scarce and it is not desirable to discard majority class data. In that case the number of minority class samples available may be small (50 in the oilspill detection problem described in Kubat et al. and 290 in the IVF problem described in Cunningham et al. 2000).

2.2. Bias in Regression Problems

In regression problems the effect of bias in the training data is similar to that in classification. This effect manifests itself as a bias in the network towards output values in the ranges for which there were a large number of training patterns. The net result of this is, when test patterns are presented to the system, prediction errors for those patterns with expected results in the well represented region are low while those with expected results outside this region are high.

It is worth illustrating this effect in detail using the warfarin data. The warfarin data set is made up of 656 samples each described by 22 features. These features describe a patient through attributes including age, sex, target INR value, etc. Associated with each sample is the actual resulting INR level. Using these outcome INR levels, a histogram of the data set was constructed. This is shown in figure 2.

As can be seen in figure 2, the distribution of data in the warfarin data set is skewed, with the majority of cases having an INR value of between 1.5 and 3. The effect that this has on the prediction accuracy of a simple ANN network trained with this data can be seen in figure 1, which shows the network's average prediction errors for different ranges of expected INR values. If the graph in figure 1 is examined in conjunction with the histogram shown in figure 2 it can be clearly seen that the network performs best when presented with test cases with expected INR values in the region of highest frequency, while the average error for low values is positive and for high values is negative. While this graph shows this bias it flatters the performance of the ANN in the centre of the distribution where the mixture of positive and negative errors cancel each other. A true picture of the magnitude of the errors is shown in Figure 3, which shows the average *absolute* error. The importance of Figure 1 is that it shows how the errors are biased towards the high frequency values.

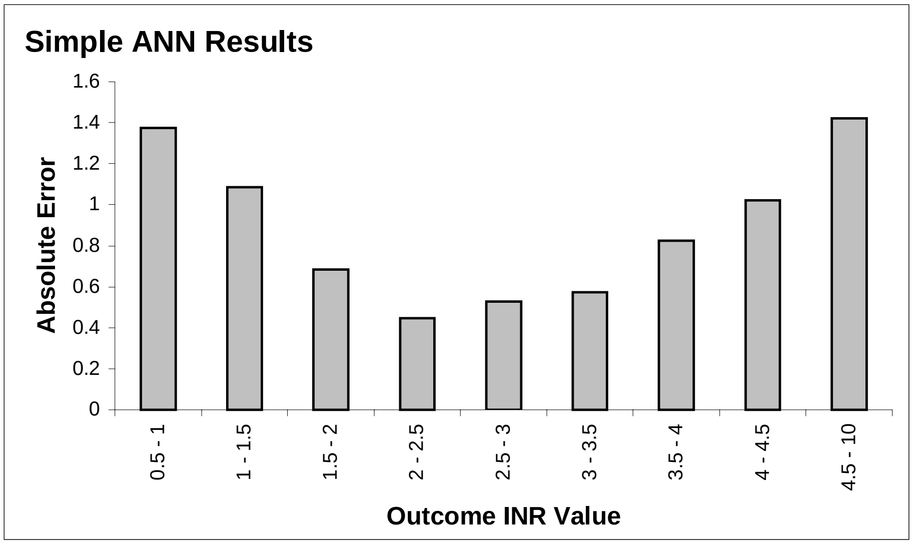

Figure 3. A graph of the prediction errors produced from a 6-fold cross validation using a simple ANN network trained with the warfarin data.

3. Possible Solutions

3.1. Stratified Sampling

The objective with stratified sampling is to balance the data set in order to remove the skew. Turney (2000) achieves this by building training and test data sets by sampling with replacement with the proviso that the data sets contain 50% of class 0 and 50% of class one.

The approach adopted in these experiments is to balance the data sets by dividing the data into intervals based on outcome and replicating the data in the minority intervals so that each interval has roughly the same number of data samples. Intuitively this has the effect of duplicating the data in the tails of the histogram in Figure 2 in order to bring them in line with the highest values.

3.2. Boosting

Boosting is a technique that attempts to improve the prediction accuracy of a system by constructing a committee of learning systems (normally classifiers). When applied to neural networks boosting involves constructing an ensemble of networks in which each network is iteratively trained using a training set determined by the performance of the previous network. Constructing a training set for a particular network involves including duplicate copies of certain samples and reducing the representation of others based on the performance of the previous network. Evidently it is the samples on which the network did not perform well that are duplicated. In this paper we refer to this raising the weight (representation) of a sample in the training as boosting. Properly speaking, boosting refers to raising the performance of a weak learner (a single network) to that of a strong network – the ensemble of networks (Schapire, 1999). From this description it is clear that boosting is a form of *adaptive* sampling and might be expected to perform better than stratified sampling precisely because it focuses on the problematic samples. In using it here we are expecting that the poorly represented samples in the tails of the distribution will benefit from the boosting.

It should be clear that this tampering with the distribution underlying the training data needs to be done with caution. The obvious risk is that noisy data will get boosted and will damage the performance of the ensemble. Indeed this is a well-known problem with boosting (Schapire, 1999) and arises with the third dataset analysed here.

Since the standard version of boosting is designed to work for classification, two modifications designed to work for regression were employed in this research. The first was an implementation of the *ADABoost.R2* algorithm presented in (Drucker, 1999). The second technique, the *Big Error Margin Technique* was based on work presented in (Avnimilach & Intrator, 1998). Each of these will now be described in some detail.

3.2.1. ADABoost.R2

ADABoost.R2 (Drucker, 1999) is a variation on the ADABoost (Freund & Schapire, 1996) algorithm specifically designed for regression problems. When applied to neural networks ADABoost.R2 proceeds by iteratively training networks using training sets determined by the performance of the previous network.

The key architectural features of the ADABoost.R2 algorithm are an initial data set and a sampling distribution. Each of the training data elements has a value in the sampling distribution and this value represents the probability of that element being included in the next training set. Initially, each value in the distribution is set to the same value giving each element in the initial data set an equal chance of being included in the first training set. An initial training set is populated from this distribution and the first ANN is trained. Each member of the initial data set is then presented in turn to the network and the prediction errors are recorded. Using these prediction errors, the distributions for each element of the initial training set are adjusted using the formulae presented in the algorithm in figure 4.

Once the distributions have been adjusted a new training set can be populated. This set is then used to train the next network, and the steps described above are repeated. New networks are iteratively trained until either the average prediction error for a network goes above 0.5 or a maximum number of networks have been trained.

The ADABoost.R2 algorithm has been used to good effect on other problems and has the elegant advantage that it does not require any parameter setting. However, as will be seen in the evaluation section of this paper it did not perform satisfactorily in the experiments described here. An explanation for this is proposed in the evaluation section of this paper.

3.2.2. Big Error Margins Technique

The Big Error Margin (BEM) boosting technique is quite similar to the ADABoost.R2 method. It is introduced in (Feely, 2000) and based on an approach proposed in (Avnimilach & Intrator, 1998). Again, boosting takes place by iteratively training a number of networks in turn. Also similar to the ADABoost.R2 method, the training set of a particular network is determined by the performance of the network preceding it. The real difference between the two techniques comes in the manner in which elements are selected for boosting and the amount by which each element is boosted.

AdaBoost.R2 Algorithm

```
Given: initialDataSet of size $m$: $(x_1, y_1),\ldots,(x_m, y_m)$.
1. Initialise $D_1 [i] = 1/m$ for all $i$; set $t = 1$
    2. Populate currentTrainingSet from initialDataSet using $D_t$
    3. Construct a new network $h_t$ and train it using currentTrainingSet
    4. Calculate the maximum loss, $L_{max}$, over the initialTrainingSet where:
           $L_{\max} = \sup|h_t(x_i) - y_i| \text{ over all } i$
    5. Calculate individual loss for each element in initialTrainingSet as follows:
           $L_i = \frac{|h_t(x_i) - y_i|}{L_{\max}}$
    6. Calculate weighted loss: $\bar{L} = \sum_{i=1}^{m} L_i D_t(i)$
    7. Set: $\beta_t = \bar{L}/(1 - \bar{L})$
    8. Calculate next distribution: $D_{t+1}(i) = \frac{D_t(i)\beta_t^{(1-L_i)}}{Z_t}$
           where $Z_t$ is a normalisation factor chosen so that $\sum_i D_{t+1}$ sums to 1.
9. $t=t+1$; Repeat steps 2–8 until $L<0.5$ or a pre-set number of networks are constructed
```

Figure 4. The ADABoost.R2 algorithm

In the BEM technique the representation of the "distribution" of a training set is also slightly different from that in the ADABoost.R2 system. In this case, the distribution represents how many times an element will be included in a training set rather than the probability of it appearing. Initially, the distribution values for each element in the initial training set are set to one. This results in the first training set generated including one copy of every element in the initial training set.

After a network has been trained using a particular training set, each element of the initial data set is presented to the network again, and the resulting prediction errors are noted. It is then determined which elements resulted in an accurate prediction by comparing the prediction errors generated against a preset value known as the *Big Error Margin* (BEM). If the prediction error generated for a particular element is greater than the BEM then the predicted value for that element is considered to be *incorrect*. Otherwise, the predicted value is considered *correct*.

The numbers of correct and incorrect predictions are then recorded. These values are used to calculate values known as the *UpFactor* and the *DownFactor* according to the following formulae:

$$UpFactor = \frac{|initialTrainingSet|}{errCount}$$

$$DownFactor = \frac{1}{UpFactor}$$

Using these values, the number of copies of each training element to be included in the next training set is calculated. If an element was correctly classified by the preceding network the number of copies of that element present in the next training set is given by the number of copies of that element in the current training set multiplied by the *DownFactor*. Similarly, if an element were misclassified in the previous iteration, the number of copies in the next training set is given by the number of copies of that element in the current training set multiplied by the *UpFactor*.

The algorithm for the Big Error Margin boosting technique is given in figure 5.

Big Error Margin Boosting Algorithm

```
Given initialTrainingSet, initialDistribution and BEM
$D_0[i]$ = initialDistribution[i] for all i
t = 0; errCount = 0
1. Populate currentTrainingSet using $D_t$
2. Construct network $h_t$ and train it using currentTrainingSet.
3. For each member of initialTrainingSet:
     Present that element to the trained network and record the prediction error.
     If the resulting prediction error is greater then BEM,
         mark that element as incorrectly predicted.
         errCount = errCount + 1
     Otherwise, mark that element as correctly predicted.
4. Calculate the UpFactor and DownFactor as follows:
     $UpFactor = \frac{|initialTrainingSet|}{errCount}$
     $DownFactor = \frac{1}{UpFactor}$
5. Cycle through each member of the originalTrainingSet and update each element's
   distribution in the next training set as follows:
       If the element was incorrectly predicted by the current network:
           $distribution_{t+1}[i] = distribution_t[i] \times UpFactor$
       If the element was correctly predicted by the current network:
           $distribution_{t+1}[i] = distribution_t[i] \times DownFactor$
6. Populate a new training set according to the distribution just calculated.
7. Repeat steps 1-6 until a given maximum number of networks has been created.
```

Figure 5. The Big Error Margin boosting technique

In contrast with the ADABoost.R2 algorithm, the Big Error Margin technique suffers from the fact that it requires fine-tuning of two parameters in order to achieve optimum results. These are the value of the BEM itself and the number of networks to include in the ensemble.

4. Evaluation

In this section the experiments undertaken to evaluate the proposed solutions illustrated above will be described and results will be presented. Our central objective is to improve the performance on the tails of the warfarin data without damaging performance in the centre of the distribution. Given that objective, the results on the warfarin data are the most important, however the techniques have been evaluated on two other data-sets to assess their generality.

We were unable to locate other regression datasets in a medical domain. So we selected the Abalone data-set from the UCI repository as one candidate and created an artificial dataset according to a cardiac risk assessment scenario as another. The experiments on these data sets will now be described.

4.1. Warfarin Data

The warfarin data used in these experiments is as described in the introduction. First we will describe a simple ANN predictor that shows the bias problem then we will show results with stratified sampling and the two boosting techniques described in section 3.

4.1.1. Simple ANN

The first experiment on the warfarin data was to construct a simple ANN to predict INR values. The purpose of this test was two fold, firstly to verify the results presented in (Byrne et al 2000), and secondly to establish a benchmark against which to compare the stratified sampling and boosting techniques.

The network was constructed following the standard multi-layer perceptron (MLP) network architecture and was then tweaked in an attempt to find the optimum values for the learning rate, momentum rate, number of hidden units and the number of epochs for which to train the network. Once these were established, the network was trained and its generalisation performance tested using 6-fold cross validation.

By comparing Figure 1 and Figure 2 it is clear that for frequently occurring INR values the prediction error is very low while for lower frequency values the error is higher. Moreover, the errors are biased in the direction of the mean. Overall the network achieved an absolute average prediction error of 0.74 INR units.

4.1.2. ANN with Stratified Sampling

The next step was to repeat this 6-fold testing on a network trained with data produced by stratified sampling as described in section 3.1. The results from this set of experiments were quite promising with an overall decrease in the absolute average prediction error across the testing set.

As can be seen from the results graph in Figure 6 the stratified sampling of the training set resulted in a large improvement in the prediction errors of the outlying elements. This did result, however, in some damage to the prediction errors in the high frequency regions. Although this did cause an overall improvement in absolute average training errors, an overall absolute prediction error of 0.69 against the value of 0.74 achieved using the simple ANN, it was not an altogether satisfactory result as the increase in error in the highly populated range is not acceptable. It was hoped that the use of boosting would overcome this problem.

4.1.3. Boosted MLP Ensemble Using ADABoost.R2

This set of experiments implemented a boosted ANN ensemble using the ADABoost.R2 boosting technique as described above. The resulting prediction errors recorded over a 6-fold cross validation are shown in figure 7. The overall absolute average prediction error arising from the use of the ADABoost.R2 technique was 0.77, a considerable increase over the value of 0.74 achieved using the simple ANN**.**

Clearly these are very poor results for the ADABoost.R2 algorithm. In fact, use of an ANN ensemble trained using the ADABoost.R2 technique caused an increase in the absolute average prediction error achieved by the simple ANN. This is disappointing given the promising results presented for the technique in (Drucker, 1999). An examination of the distributions produced in the boosting process showed significant boosting occurring and showed dramatic variation in the samples being boosted from one boosting step to the next. Boosting is concentrated on a few training elements with particularly large prediction errors due to the skew in the original data set. The effect of this is that those elements are dramatically boosted while others are neglected. This happens again and again with dramatic changes to the data set at each step in the boosting process. This leads to a series of training sets that are dramatically different to one and other and so it becomes difficult for any inherent order to emerge. The result of this is that the individual networks in the ensemble differ too greatly in their predictions, causing the large prediction errors generated by the ensemble.

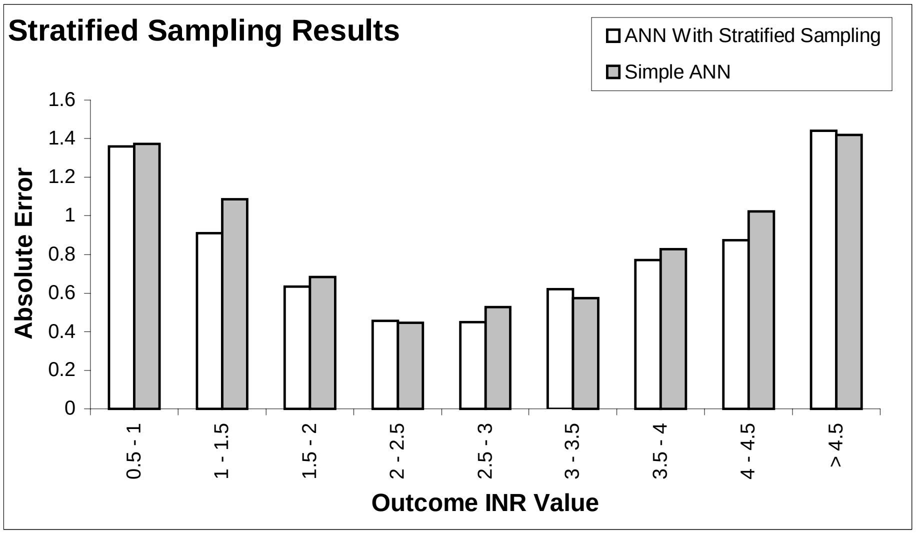

Figure 6. A graph of the prediction errors produced from a 6-fold cross validation using a simple ANN trained with the warfarin data set which had been prepared using stratified sampling.

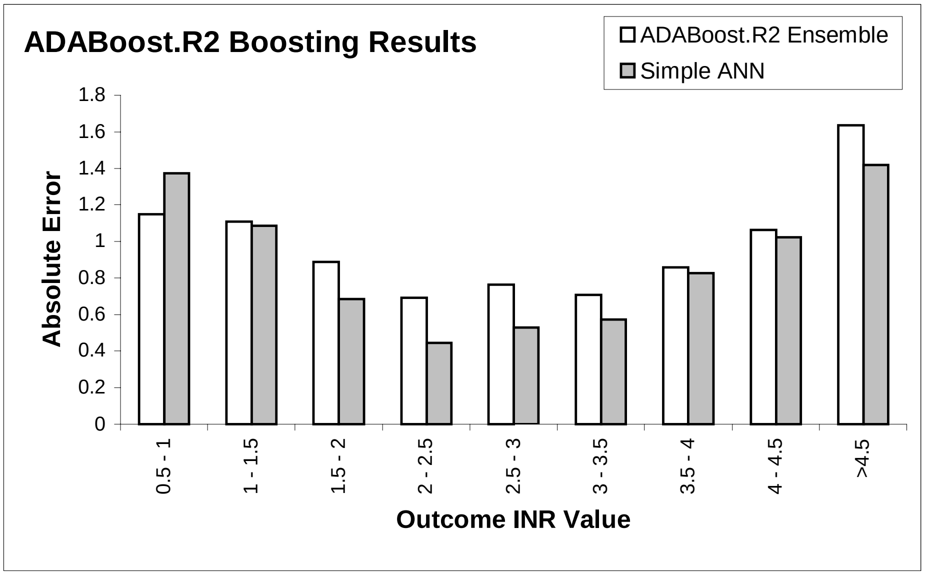

Figure 7. A graph of the prediction errors produced from a 6-fold cross validation using an ensemble of simple ANNs trained using the ADABoost.R2 algorithm.

Given this it can be argued that ADABoost.R2 is not a suitable boosting technique for problems in which there exists a bias in the training data. Due to this and the poor results produced using the ADABoost.R2 on the warfarin problem, along with poor preliminary results using the two other data sets, it was decided not to pursue the ADABoost.R2 boosting technique.

4.1.4. Boosted ANN Using Big Error Margins Technique

The biggest obstacle to overcome with an implementation of the Big Error Margin (BEM) boosting technique is determining the correct values for the BEM and the number of networks to use in the ensemble. This involved a calibration process in which a range of values was tested to find the BEM value that minimised overall error on the output. This is the main disadvantage that the BEM technique has over the ADABoost.R2 technique.

A 6-fold cross validation was performed using the BEM technique and the results were very promising. The effect of using boosting was to decrease the overall absolute average prediction error to 0.66 from the value of 0.74 achieved using a simple ANN. The really interesting result, however, is in the fact that this improvement comes about by reducing the prediction errors for those elements with expected INR values in the outlying ranges without damaging prediction errors for those elements with expected INR values in the high frequency ranges. The absolute average prediction errors for the different INR value ranges, produced using BEM boosting are shown in Figure 8.

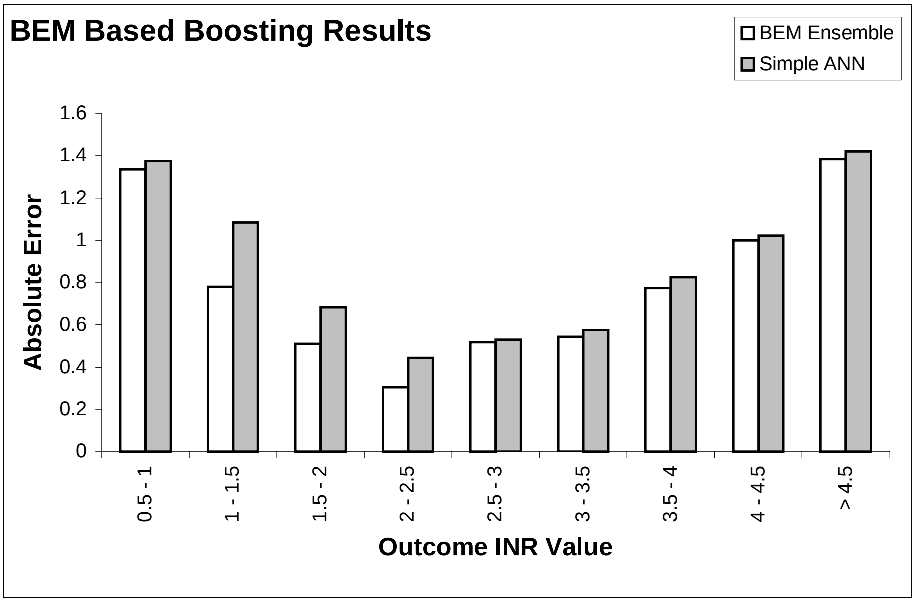

Figure 8. A graph of the prediction errors produced from a 6-fold cross validation using an ensemble of simple ANNs trained using the Big Error Margin boosting technique on the warfarin data set.

At this point it is worth reflecting on why the BEM technique gives better performance than the ADABoost.R2 technique. The amount by which individual data elements are boosted under the different techniques offers some explanation.

Using the ADABoost.R2 technique the amount to boost a particular element is determined by the prediction error arising from the presentation of that element to the network, i.e. those elements with high prediction errors are strongly boosted while those with smaller prediction errors are weakly boosted. Due to the biased nature of the data sets being used in this work, some elements will always generate extraordinarily high prediction errors. As a result, at each iteration of the boosting process, a small number of elements will be very strongly boosted. This results in erratic changes in the data sets produced over the course of the boosting process.

By contrast, when using the BEM technique boosting is not dependant on the size of the prediction error on a particular element. That is, once elements are selected for boosting (selection being based on the error exceeding the BEM) they are all boosted by the same amount. This results in much less dramatic boosting which, it would appear, results in a more stable system and produces better results.

Thus the BEM technique meets our objective of improving the prediction performance on the tails of the distribution without damaging the performance in the centre of the distribution. In the remainder of this section we examine the effect of stratified sampling and BEM boosting on two other datasets with a view to determining the generality of this solution to bias in training data.

4.2. Cardiac Risk Data

Due to a scarcity of medical regression data sets an artificial data set was created for the next set of experiments. This data set was created based on the Heart Attack Risk Index Metric available at www.medal.org. The metric calculates an individual's risk of suffering a first heart attack based on a series of parameters including age, sex and cholesterol levels. In order to create a data set, an initial population was generated by selecting values for each of the attributes required based on distributions calculated from statistics gathered about the specific attributes.

Once this initial population was generated, 650 data elements were randomly selected from this population to create the final data set. The final data set displayed skew similar to that in the warfarin data (see Figure 9). The metric has been normalised into the 0-1 range to suit the neural network processing.

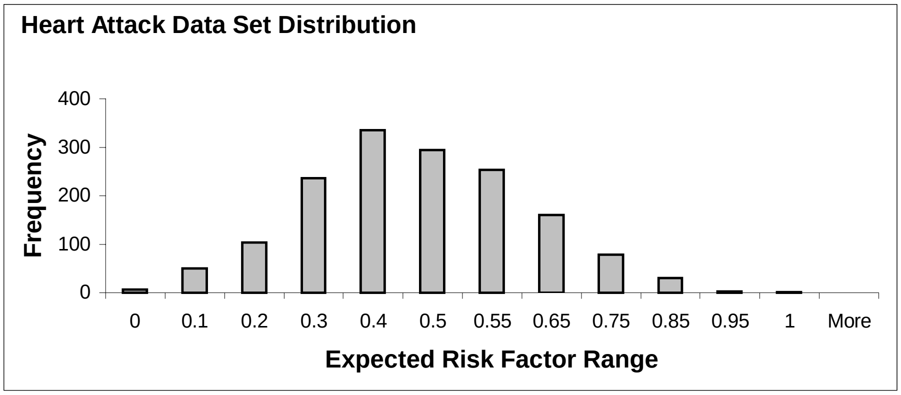

Figure 9. The distribution of expected heart risks in the Heart Attack Risk Index data set.

4.2.1. Simple ANN

Training a simple ANN to predict risk indices based on presented attributes proved quite successful. As with the warfarin data, the system was able to predict values quite well in the high frequency regions. However, the now familiar pattern, of poor performance to either side of this strong region did show up. A graph of prediction errors against the expected value ranges from which the test elements came can be seen in Figure 10. The absolute average prediction error for a simple ANN applied to the Heart Attack Risk Index problem was 0.049.

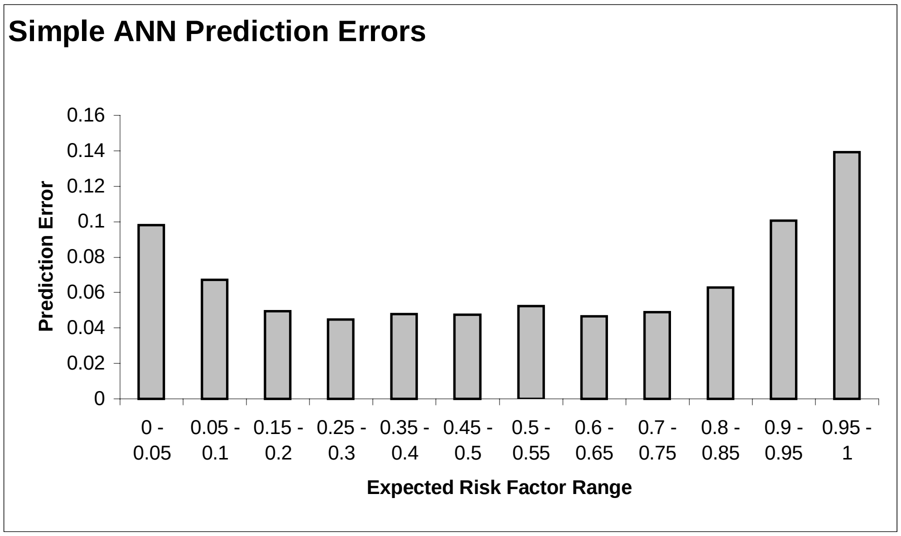

Figure 10. A graph of the prediction errors produced from a 4-fold cross validation using a simple ANN trained with the Heart Attack Risk Index data set.

4.2.2. ANN Network with Stratified Sampling

When applied to the heart attack risk problem, stratified sampling went some way towards improving the prediction accuracy in the outlying regions. However, this improvement was at too high a cost in the regions of high frequency resulting in a poor overall average prediction error. A graph of the prediction errors against the ranges of expected heart attack risk indices is shown in figure 11. To compare these results with those achieved using a simple ANN, the network using stratified sampling achieved an absolute average prediction error of 0.052, while the simple ANN, as stated earlier, achieved an overall absolute average prediction error of 0.049.

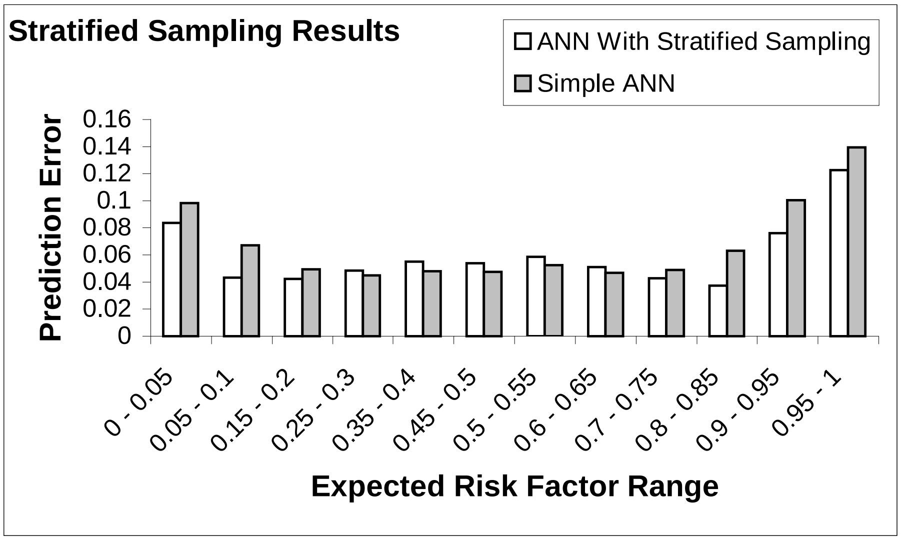

Figure 11. A graph of the prediction errors produced from a 4-fold cross validation using a simple ANN trained with the Heart Attack Risk Index data set prepared using stratified sampling.

4.2.3. Boosted ANN Network Using Big Error Technique

BEM boosting performed exactly as expected on this problem. Prediction errors in the outlying regions were greatly reduced and the prediction accuracy in the regions of high frequency was not adversely affected, and in fact, improved in some cases.

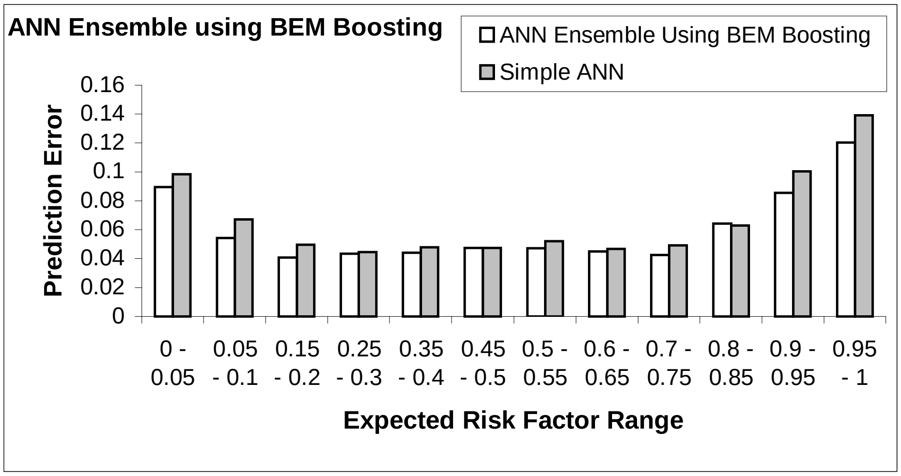

Figure 12. A graph of the prediction errors produced from a 4-fold cross validation using an ensemble of simple ANNs trained with the Heart Attack Risk Index data set using the Big Error Margin boosting technique.

Summarising the results for the heart data, the simple ANN had a average error of 0.049, the error for Stratified Sampling was higher than this at 0.052 due to increased errors in the mid region and the BEM boosting had the best figures at 0.045 with no increase in error in the mid range (see Figure 12). This reinforces the results on the warfarin data.

4.3. Abalone Data

The Abalone data set is available from the UCI Machine Learning Repository (www.ics.uci.edu/~mlearn/MLRepository.html). This data set is concerned with predicting the age of an Abalone specimen (a type of shellfish) based on physical measurements, e.g. length, diameter etc. The complete data set is made up of 4177 elements each of which is described by 9 attributes. Associated with each of these elements in an actual age value for the Abalone. Early testing showed that when used with the Abalone data set boosting techniques resulted in the creation of extremely large training sets. This resulted in unacceptably long training times when these techniques were used. For this reason, 1550 elements were randomly selected from the complete data set, to form a smaller, more manageable set. This set was divided into a training set of 550 (the size was chosen to make this a comparable set to the warfarin training set) and a testing set of 1000 elements. A histogram of the actual age values of the complete Abalone data set can be seen in figure 13.

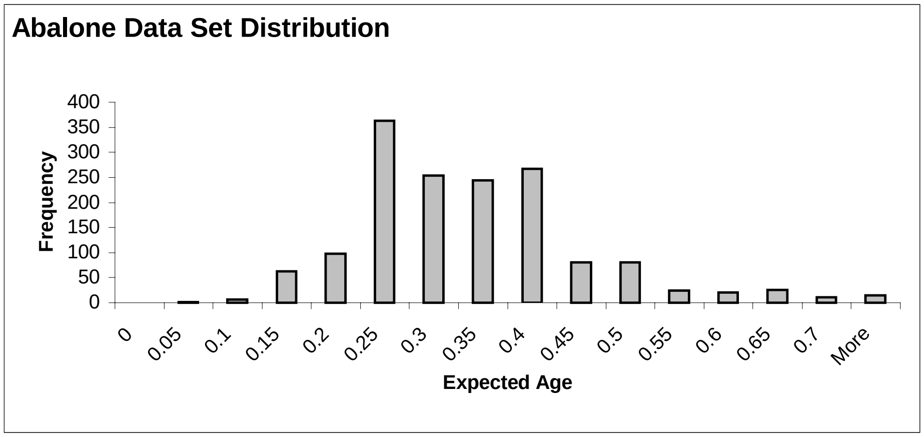

Figure 13. The distribution of expected Abalone ages in the Abalone data set.

4.3.1. Simple ANN

Again, the simple ANN was able to learn the data quite well and, because of the pronounced skew in the data, the variation in error across the distribution was quite marked. The absolute average prediction errors for different ranges of expected abalone sizes resulting from a 4-fold cross validation are shown in figure 14.

Once again viewing this graph in conjunction with the data histogram reveals a familiar pattern. The bias in the training set causes the prediction produced by the network to be skewed towards that area in which the expected abalone age values in the training set had the highest frequencies. The overall absolute average prediction error arising from the application of a simple ANN to the Abalone problem was 0.055.

4.3.2. ANN with Stratified Sampling

Applying an ANN using stratified sampling to the Abalone problem did not improve things. The absolute average prediction errors for different ranges of expected abalone sizes, resulting from a 4-fold cross validation, are shown in figure 15.

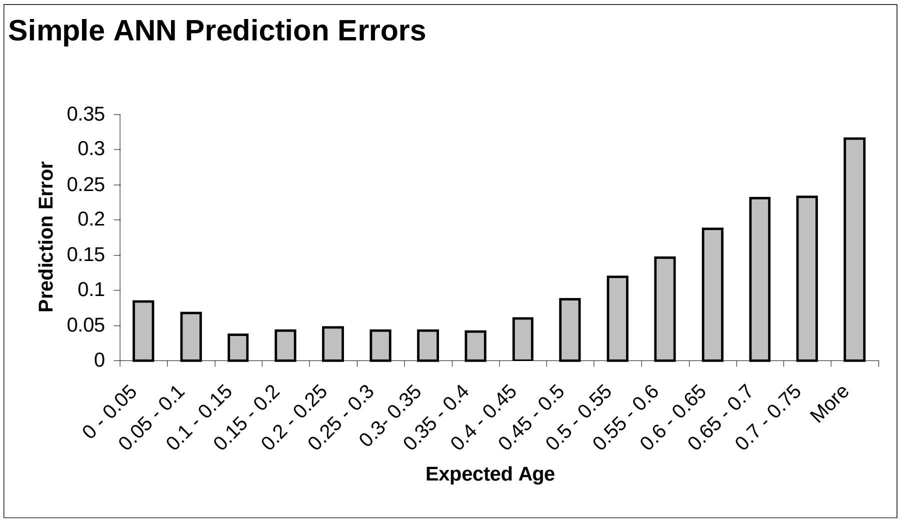

Figure 14. A graph of the prediction errors produced from a 4-fold cross validation using a simple ANN trained with the Abalone data set.

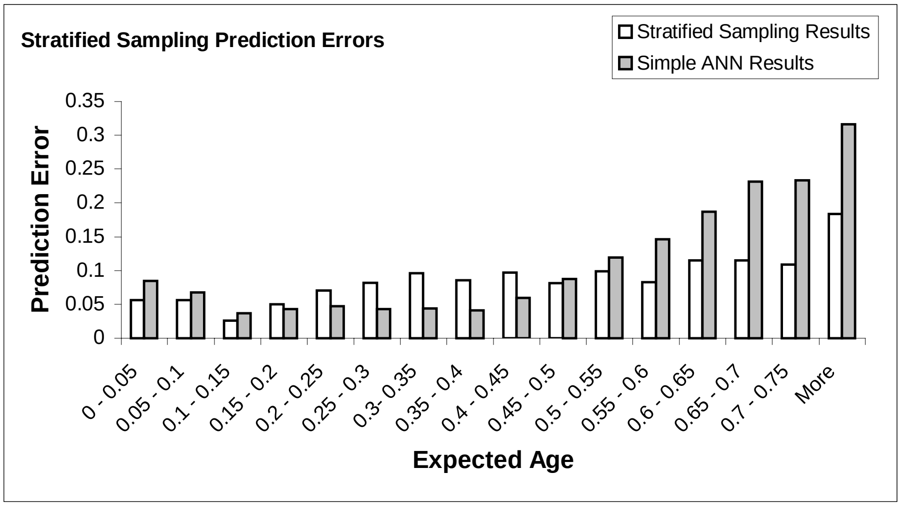

Figure 15. A graph of the prediction errors produced from a 4-fold cross validation using a simple ANN trained with the Abalone data set prepared using stratified sampling.

As can be seen stratified sampling greatly improved the prediction accuracy of those elements with expected age values in the outlying regions, but this is at far too high a cost to those elements with expected age values in the more densely populated regions. The overall absolute average prediction error achieved using a ANN in conjunction with stratified sampling was 0.08, an increase from the value of 0.055 achieved using a simple ANN.

4.3.3. Boosted ANN Using Big Error Technique

Training a boosted MLP ensemble using the Big Error Margin technique with the Abalone data set was not as successful as was hoped, but did give rise to some interesting results. The overall absolute average prediction error arising was 0.07, which is significantly higher than the value of 0.055 achieved using a simple ANN. It can be seen in Figure 16 that this is due to dis-improvements in the centre of the distribution. This was a very surprising result as it was not in line with the behaviour shown in the previous experiments. However, examinations of the training sets produced over the course of boosting suggests an explanation.

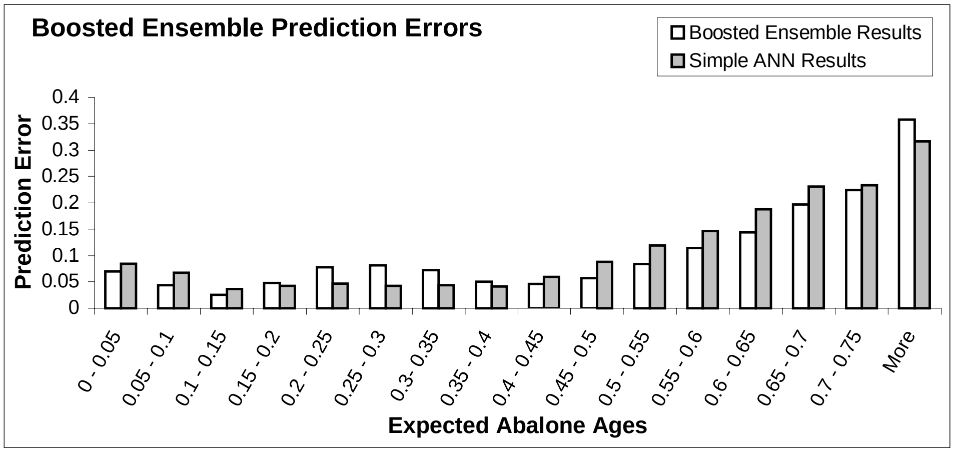

Figure 16 A graph of the prediction errors produced from a 4-fold cross validation using an ensemble of simple ANNs trained with the Abalone data set using the Big Error Margin boosting technique.

Figure 17 shows how the training sets varied over the course of boosting. The histograms show the amount of boosting happening in each interval. It can be seen that although boosting took place in the tails of the distribution as we expect. There is also significant boosting in regions that are well populated – particularly around 0.25. Detailed examination of the data sets produced through boosting has shown that a small number of data elements, the expected ages of which fall within the most densely populated ranges, are repeatedly boosted at every stage in the boosting process. This would imply that the networks are not capable of learning these elements in spite of the fact that they are so strongly boosted. This suggests that these elements are in fact noisy patterns in the data set. As stated in section 3.2 very noisy data will cause problems for boosting because the noisy elements will be difficult to learn and will get boosted.

The dis-improvement in the most heavily populated regions can be explained by the fact that repeated boosting of these noisy elements has the effect of reducing the numbers of other, noise free, elements included in the training set. This causes them to be poorly learned, resulting in the pattern displayed.

5. Conclusions

It is often the case that data on regression problem in medical decision support will have non-uniform distributions. ANNs (and other machine learning techniques) trained to solve these problems will be biased towards high frequency values. This issue arose in our research on predicting outcome in anti-coagulant drug therapy. Thus the objective of the research described here was to improve performance on the tails of the distribution of outcomes without damaging performance in the centre of the distribution.

In this paper we have emphasised that it is impossible to optimise performance on such a biased distribution without considering the cost associated with errors at different points in the distribution. In practice it is often not possible to quantify these costs so we use a measure of *dominance* in comparing solutions. One solution dominates another if it improves performance in one region without damaging performance in any other region.

Using this criterion we have found that stratified sampling can reduce errors at the tails of the distribution but not without some deterioration in the centre. We found that the BEM version of boosting produced dominating solutions for the warfarin data and for an artificial data set we created for these experiments. We also showed that the well-known problem with boosting of intolerance to significant levels of noise is an issue that needs to be considered.

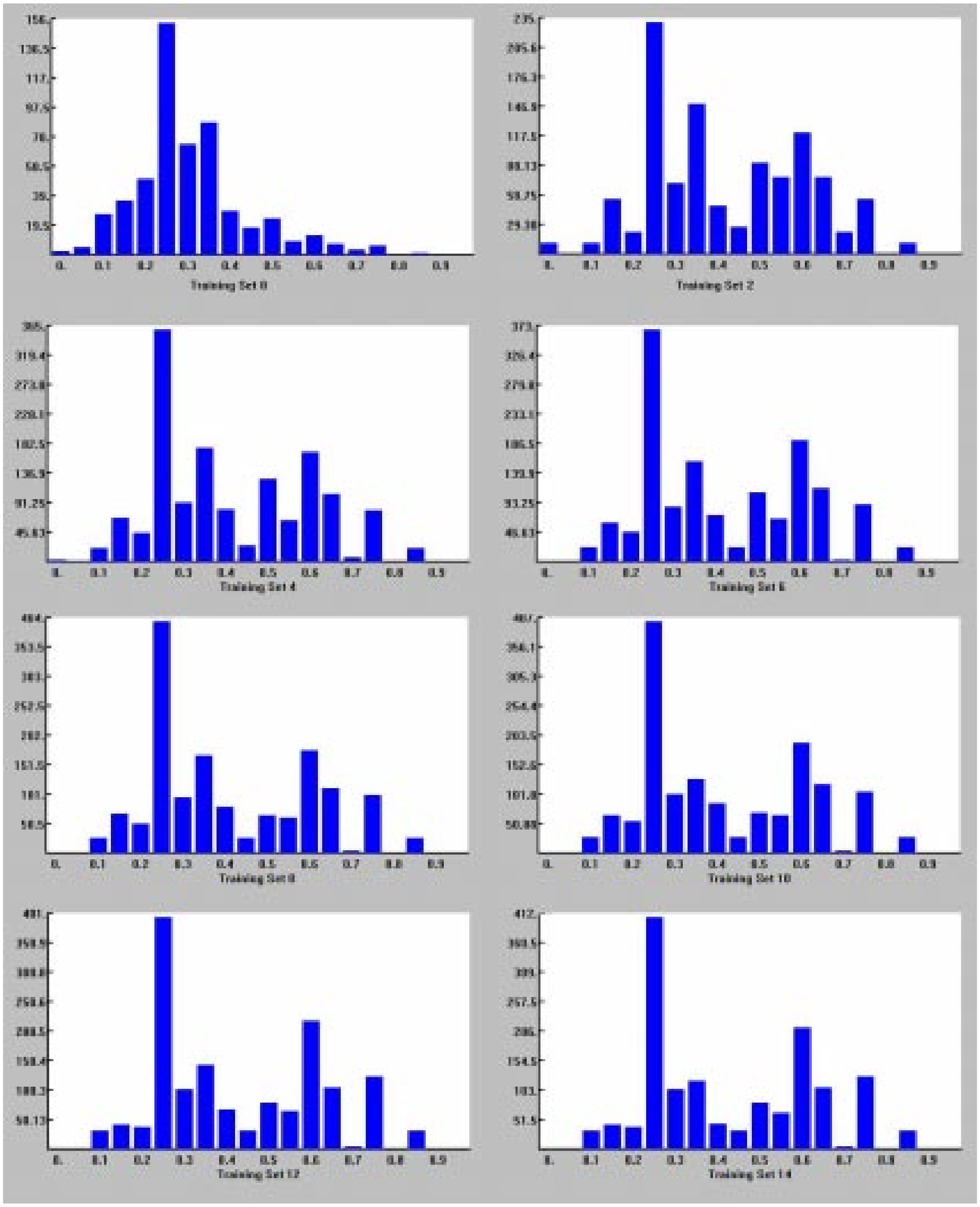

Figure 17. A selection of training sets generated over the course of boosting using the Big Error Margin technique on the Abalone data set. Notice how the histogram bar in the 0.25 position remains high for every iteration.

References

Avnimilach, R., Intrator, N., (1998) *Boosted Mixture of Experts: An Ensemble Learning Scheme*, Neural Computation 11: 483-497, 1999.

Breiman L., Friedman J.H., Olshen, R.A., Stone C.J., (1984) Classification and Regression Trees, Wadsworth, Belmont, CA, USA.

Buntine, W.L., (1989) *Stratifying samples to improve learning*. In Proceedings of the Knowledge Discovery in Databases Workshop, Detroit.

Byrne, S., Cunningham, P., Barry, A., Graham, I., Delaney T., Corrigan, O.I., (2000) Using Neural Nets for Decision Support in Prescription and Outcome Prediction in Anticoagulation Drug Therapy, N. Lavrac, S. Miksch (eds.): *The Fifth Workshop on Intelligent Data Analysis in Medicine and Pharmacology (IDAMAP-2000). Workshop Notes of the 14th European Conference on Artificial Intelligence (ECAI-2000),* also available as TCD-CS-2000-18 at www.cs.tcd.ie/publications/tech-reports/

Cardie, C., Howe, N., (1997) Improving Minority Class Prediction Using Case-Specific Feature Weights, *Proceedings of 14th International Conference on Machine Learning*, pp57-65.

Chan, P., Stolfo, S., (1998) Toward Scalable Learning with Non-uniform Class and Cost Distributions: A Case Study in Credit Card Fraud DetectionProc. *Fourth Intl. Conf. Knowledge Discovery and Data Mining*, p164-168, 1998.

Cunningham, P., Carney, J., Jacob, S., (2000) Stability Problems with Artificial Neural Networks and the Ensemble Solution**,** to appear in *AI in Medicine*, Vol. 20.

Dietterich, T. G. (1990). Machine learning. *Annual review of computer science*, 4, 255-306.

Drucker, H., (1999) Boosting Using Neural Networks, in *Combining Artificial Neural Nets: Ensemble and Modular Multi-Net Systems,* A.J.C. Sharkey (ed.), pp51–78, Springer, London.

Feely, R., (2000) *Predicting Stock Market Volatility Using Neural Networks*, B.A (Mod) Dissertation, Department of Computer Science, Trinity College Dublin.

Freund, Y., Schapire, R.E., Experiments with a new Boosting Algorithm. *in Proceedings of the 13th International Conference on Machine Learning*, pp148 – 156, 1996.

Kubat, M., Holte, R. C., Matwin, S. (1998). Machine learning for the detection of oil spills in satellite radar images. *Machine Learning*, 30 (2/3).

Lewis D.D., Catlett J., (1994) Heterogeneous Uncertainty Sampling for Supervised Learning, *Proceedings of 11th International Conference on Machine Learning*, pp.148-156, Morgan Kaufmann.

Pazzani, M., Merz, C., Murphy, P., Ali, K., Hume, T., & Brunk, C. (1994). Reducing Misclassification Costs. *Proceedings of the 11th International Conference on Machine Learning* (pp. 217--225), Morgan Kaufmann.

Provost, F., T. Fawcett, (2000) Robust Classification for Imprecise Environments, to appear in *Machine Learning,* 2000 (available at http://www.stern.nyu.edu/~fprovost/).

Provost, F.J., Fawcett, T., and Kohavi, R. (1998). The case against accuracy estimation for comparing induction algorithms. *Proceedings of the 15th International Conference on Machine Learning (ICML-98),* pp. 445-453, Morgan Kaufmann.

Schapire, R.E., (1999) *A Brief Introduction to Boosting*, In *Proceedings of the 16th International Joint Conference on Artificial Intelligence,* pp1401 – 1405, 1999.

Swets, J.A. (1988). Measuring the Accuracy of Diagnostic Systems. *Science*, 240, pp1285-1293.

Turney, P.D., (2000) Learning Algorithms for Keyphrase Extraction, *Information Retrieval,* 2 (4): 303-336.