European Journal of Operational Research 210 (2011) 368–378

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Innovative Applications of O.R.

Multiple classifier architectures and their application to credit risk assessment

Steven Finlay *

Department of Management Science, Lancaster University, LA1 4YX, UK

ARTICLE INFO

Article history:
Received 4 June 2008
Accepted 22 September 2010
Available online 29 September 2010

Keywords:
OR in banking
Data mining
Classifier combination
Classifier ensembles
Credit scoring

ABSTRACT

Multiple classifier systems combine several individual classifiers to deliver a final classification decision. In this paper the performance of several multiple classifier systems are evaluated in terms of their ability to correctly classify consumers as good or bad credit risks. Empirical results suggest that some multiple classifier systems deliver significantly better performance than the single best classifier, but many do not. Overall, bagging and boosting outperform other multi-classifier systems, and a new boosting algorithm, Error Trimmed Boosting, outperforms bagging and AdaBoost by a significant margin.

1. Introduction

Since the early 1990s the majority of consumer lending decisions in the US have been made using automated credit scoring systems (Rosenberg and Gleit, 1994). A similar situation exists in the UK and other countries. Credit scoring systems are not perfect and every year a significant proportion of consumer debt is not repaid because of the failure of such systems to identify individuals who subsequently default on their loans. The value of outstanding consumer credit (excluding residential mortgage lending) in the US and UK at the end of 2009 was $2.5 trillion and £171 billion respectively (The Federal Reserve Board, 2010a; Bank of England, 2010a). For the same period, annualized write-off rates for credit cards and personal loans in the US and UK were 5.4% (The Federal Reserve Board, 2010b) and 1.5% (Bank of England, 2010b) respectively. Consequently, there is considerable interest in improving the ability of credit scoring systems to discriminate between customers on the basis of their future repayment behaviour, because even small improvements in classification performance can yield significant financial benefits (Hand and Henley, 1997).

Credit scoring is traditionally viewed from a binary classification perspective. Classification or regression methods are used to create a classifier that generates a numerical output (a score) representing the likelihood of an individual being a 'good' or 'bad' credit risk over a given forecast horizon. 'Good' credit risks are those that repay their debt to the terms of the agreement. 'Bad' credit risks are those that default or display otherwise undesirable behaviour. Some methods applied in credit scoring produce probabilistic estimates of class membership, but some do not, and what is of primary importance is the relative ranking of model scores (Thomas et al., 2001a). Lending decisions are made on the basis of the properties of the ranked score distribution, with a cut-off defined that meets the user's business objectives. As long as an observation is correctly classified, the margin by which an observation passes or fails the cut-off is largely irrelevant (Hand, 2005). In some cases the cut-off may be based on a likelihood measure, such as the point in the score distribution where the good:bad odds ratio exceeds 10:1. In other cases the cut-off is chosen to yield a fixed proportion of accepted applications. For example, a finance house may provide retail credit to customers of a furniture retailer, but there is a contractual obligation to accept at least 90% of all credit applications. In practice, measures of group separation, such as the GINI coefficient and the KS statistic, are also popular, but such measures are only of interest in two situations. First, when the cut-off decision(s) are unknown at the time when the model is developed. This is the situation facing the developers of generic scoring models, such as FICO and Experian, whose models are used by many different lenders. Second, where multiple cut-offs are required so that different terms can be applied on the basis of individual risk – a practice referred to as "Pricing for risk".

Credit scoring was one of the first and most successful applications of data mining. What differentiates credit scoring from other data mining applications are the objectives of the financial institutions that employ it, the nature of the data sets employed and the business, social, ethical and legal issues that constrain its use within decision making processes. The most popular methodology applied to credit scoring is logistic regression (Crook et al., 2007). Other methods such linear regression/discriminant analysis (Durand, 1941; Myers and Forgy, 1963) and decision trees

* Tel.: +44 01772 798673/0777 2074 742.
E-mail address: steve.finlay@btinternet.com

0377-2217/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.ejor.2010.09.029

(Boyle et al., 1992) are also popular. This is because they are easy to develop, yield acceptable levels of performance and the structures of the resulting classifiers are easily explicable in support of legislative and operational requirements. In recent years there has been increasing use of neural networks (Desai et al., 1996; West, 2000), but these tend to be restricted to fraud detection and other back-end credit scoring processes where a lower level of explicability is required (Hand, 2001). Within the academic community there has been a growing body of literature into the application of many other methodologies. Some are statistical approaches such as survival analysis (Narain, 1992; Stepanova and Thomas, 2001), graphical models (Sewart, 1997), K-Nearest Neighbour (Henley and Hand, 1996), Markov chains (Thomas et al., 2001b) and quantile regression (Whittaker et al., 2005). Others are from the machine learning/data mining domain. Examples include: genetic algorithms (Desai et al., 1997; Finlay, 2009) and support vector machines (Baesens et al., 2003a; Huysmans et al., 2005). Empirical evidence across several studies has shown little support for any one classification or regression methodology consistently outperforming any other in terms of misclassification rates/cost, or measures of group separation (Baesens et al., 2003a; Boyle et al., 1992; Desai et al., 1996; Henley, 1995; West, 2000). Looking at the cumulative findings from across these studies there is some evidence that nonlinear approaches such as neural networks and support vector machines do, on average, outperform other methods (Crook et al., 2007), but only by a very small margin. There is no consensus as to which methodology a model developer should adopt for a given problem. Given this uncertainly, it is not unusual for a practitioner to construct several classifiers using different techniques, and then choose the one that yields the best solution for their problem. However, when comparing classifiers, it does not necessarily follow that the best classifier overall, dominates all others over all regions of the problem domain. Consequently, error rates can often be reduced by combining the output of several classifiers (Kittler, 1998). As noted by Zhu et al. (2001), the research into classifier combination (also referred to as multiple classifier systems, classifier fusion or classifier ensembles) is rich, and much of the relevant literature is covered in Kuncheva (2004). However, within the credit scoring community the number of papers relating to classifier combination is sparse, and few report empirical findings on data sets that can be said to be representative of those found in industry. Instead, most are based on "Toy" data sets, that are only a fraction of the size and/or dimensionality of those employed by the world's leading financial institutions (Finlay, 2006). In addition, previous studies have had a narrow focus. Standard practice has been to compare a single benchmark classifier with a narrow set of multiple classifier systems, often employing just one method of classifier construction. There has been little effort to compare and contrast the wide range of multiple classifier systems available in conjunction with the different mechanisms by which classifiers can be constructed.

The remainder of the paper is in two parts. First, I describe the types of multiple classifier architectures that can be employed, followed by a review of the literature of their application to credit scoring. In the second part of the paper a variety of multiple classifier systems are evaluated in terms of their ability to correctly classify good and bad credit risks. For this purpose two data sets are employed that have been supplied by industry sources. It would be impossible to cover all possible variants of multiple classifier systems for all types of classifier in a single paper. Therefore, the emphasis has been to report on a representative sample of the best known multi-classifier systems, for which empirical evidence exists that they can yield improved performance over a single classifier. In addition, a new multi-classifier algorithm – Error Trimmed Boosting (ET Boost for short) is presented.

2. Multiple classifier system architectures

Multiple classifier system can be classified into three architectural types. Static parallel (SP) systems are where classifiers are developed independently in parallel (Zhu et al., 2001). The outputs from each $C_t(t = 1 \ldots T)$ classifier are combined to deliver a final classification decision $w = \phi(C_1 \ldots C_{t=T})$, where w is selected from the set of possible class labels. Many combination functions are available. These include simple majority vote, weighted majority vote, the product or sum of model outputs, the min rule, the max rule and Bayesian methods. However, in practice most combination strategies are reported to yield very similar levels of performance (Kittler, 1998). Consequently, simple majority vote or weighted majority vote are often favoured due to their simplicity. Another option is for the outputs from each classifier to be used as inputs to a second stage classifier that delivers the final classification decision (Kuncheva et al., 2001).

Some SP approaches use identical representations of the data, but apply different methods of classifier construction. Classifier outputs are then combined to produce the final classification decision. Other SP approaches utilize a single method for classifier construction, but use different representations of the data. For example, with cross-validation data is segmented into k folds. k models are constructed with a different fold excluded for each model. A final classification decision is made using a simple majority vote of the decisions made using each of the k classifiers. Another method is bagging (bootstrap aggregating) (Breiman, 1996). Classifiers are constructed from different sub-sets of the data, randomly sampled with replacement. More recent variants of bagging include Random Forests (Breiman, 2001) and subagging (Paleologo et al., 2010).

The second type of architectures are multi-stage (MS) ones. Classifiers are constructed iteratively, with the parameter estimation process dependent upon the classification properties of the classifier(s) from previous stages. Some MS approaches generate classifiers that are applied in parallel using the same type of combination rules used for SP methods. For example, most forms of boosting (Schapire, 1990) generate a set of weak classifiers that are combined to create a stronger one. This is achieved by employing a re-sampling/reweighting strategy at each stage, based on a probability distribution or heuristic rule, derived from the misclassification properties of the classifier(s) from the previous stage(s). As the boosting algorithm progresses, increasing focus is placed on harder to classify/more borderline cases. The boosting algorithm terminates after a given number of iterations have occurred or some other stopping criteria is met. Classifier output from each stage is combined to make the final classification decision. The most well known boosting algorithm is arguably AdaBoost (Freund and Schapire, 1997) and many variants have subsequently appeared such as Arc-x4 (Breiman, 1998), BrownBoost (Freund, 2001) and LogitBoost (Friedman et al., 2000).

Other MS methods produce a separate classifier at each stage and then classify a proportion of observations at that stage. The next stage classifier is developed and applied to the remaining, unclassified, population. A variation on this theme is to make the classification decision for all observations using a single classifier from one stage, with all other classifiers discarded (Myers and Forgy, 1963). Improved classification can result because "outliers" at the tail ends of the score distribution are excluded. This allows the decision surface to obtain a better fit to more marginal cases while still correctly classifying the excluded cases – precision is sacrificed for improved accuracy. This approach also has a practical advantage in that it results in a single classifier. This is attractive in operational environments because it is simpler to implement and monitor a single classifier, than multiple classifiers with serial dependencies.

The third type of multiple classifier architecture is Dynamic Classifier Selection (DCS). Separate classifiers are developed or applied to different regions of the problem domain. One classifier may outperform all others based on global measures of performance, but weaker competitors will sometimes perform better across some regions (Kittler, 1997). DCS problems are normally approached from a global (DCS-GA) or local accuracy (DCS-LA) perspective (Kuncheva, 2002). With a DCS-GA approach classifiers are constructed using all observations within the development sample. Classifier performance is assessed for each region and the best one chosen. One limitation of DCS-GA approaches is that different classifiers may perform better within some regions than others, but there is no guarantee that any given classifier is optimal within any given sub-region. Better classification might result if a separate classifier is developed for that region. With a DCS-LA approach, regions of interest are determined first. Classifiers are then developed for each region. An issue in this case is sample size. It is relatively easy to construct a classifier and confirm it is robust and not over-fitting, but if the development sample is small then the locally constructed classifier may be inefficient, and not perform as well as a classifier developed on a larger and more general population. Consequently, there is a trade-off between the robustness of classifiers developed using large samples and the accuracy of local classifiers. This logically leads to the conclusion that for DCS-LA approaches one benchmark against which performance should be measured are the results obtained from applying a DCS-GA approach and vice versa.

With DCS-GA and DCS-LA approaches an important issue is identifying the best regions to consider. Sometimes regions are defined using purely mechanical means. A common method, described by Kuncheva (2002), is to apply a clustering algorithm, but many other segmentation algorithms exist. For example, Liu and Yuan (2001) applied a two-stage algorithm containing both global and local components. First they constructed classifiers across the entire problem domain. Then, they applied K-nearest neighbour to generate two sets of clusters containing observations that had been correctly or incorrectly classified by each classifier. Second stage classifiers were constructed for each cluster and used to make the final classification decision.

In many applied situations segmentation decisions are overwhelmingly driven by expert knowledge, business considerations, legal or other environmental constraints. For example, Toygar and Acan (2004) successfully applied a DCS-LA approach to face recognition. They segmented images of faces into regions by dividing them into equally sized horizontal strips, so that specific features, such as nose, eyes, mouth etc. fell into each strip. The rationale for choosing this segmentation was primarily expert opinion about the regions which are important in face recognition problems, and trial and error. The approaches exemplified by Toygar and Acan (2004), Liu and Yuan (2001) represent two ends of a spectrum. At one end segmentation is entirely determined by mechanical segmentation rules, and at the other, segmentation decisions are made subjectively using expert opinion. Credit scoring is a field that falls somewhere in the middle. As noted by Thomas et al. (2002). Mechanical means of segmenting a problem domain may be employed, such as applying the splitting rules associated with the construction of decision trees. However, segmentation is often driven by political decisions, the availability of data, product segmentation (e.g. cards and loans) or other business rules.

3. Applications of multiple classifier system for consumer credit risk assessment

A common strategy in credit scoring is to develop sub-population models for different regions of the problem domain (Banasik et al., 1996). Sub-population modelling is an example of a DCS-LA architecture, and most classifier combination strategies reported in the credit scoring literature adopt a DCS-LA approach, even if they are not described as such. One reason DCS-LA strategies are attractive is each model is developed and applied independently to a different region. Therefore, each credit application is scored by a single model. From an implementation perspective this simplifies performance monitoring, facilitates the redevelopment of models on a piecemeal basis and generates models that are more readily explicable to the general public in support of legislative requirements. This is regardless of the method used to generate the segments upon which the sub-population models are constructed, and therefore, offers a methodology whereby techniques that do not generate a simple (linear) parameter structure, such as clustering, neural networks or support vector machines, can be combined with methods that do. For example, the output from a complex (difficult to interpret) neural network could be used to define segments upon which simple linear scoring models are then constructed. If probabilistic estimates of class membership are required, then as long as the final models generate probabilistic estimates, then it is irrelevant whether the segmentation strategy does so or not. Not all of these properties can be said to be universally true of static parallel or multi-stage classifier systems.

Empirical analysis of credit data lends some, but not overwhelming, support for DCS-LA methodologies providing better classification performance than a single global classifier. Chandler and Ewert (1976) compared a single model, using gender as a dummy variable, with separate models for male and female credit applicants respectively. One of their findings was that the separate models used in combination led to a lower rate of rejected female applicants than the single model. Banasik et al. (1996) explored this further, comparing a single credit scoring model with models constructed on twelve different segmentations such as married/not married, retired/not retired and so on. They concluded that sub-population models sometimes led to better solutions, but not always. Another strategy along these lines was investigated by Hand et al. (2005) who considered binary segmentation using an "optimal partition" strategy based on maximising the product of two likelihood functions calculated from observations in each of the two potential segments. They observed small but significant improvements in classification performance. In another study, a preliminary forecasting model was constructed to estimate usage on a revolving credit product. The population was then segmented into two, on the basis of whether estimated usage was high or low (Banasik et al., 2001). The conclusion was that the performance of the sub-population models was superior to that of a single model constructed across the entire population, but that the improvement was marginal.

The limited research undertaken into the application of other types of classifier combination to credit scoring problems has arguably yielded better results. Myers and Forgy (1963) adopted a multi-stage approach in which they utilised a two stage discriminant analysis model. The second stage model was constructed using the lowest scoring 86% of the development sample used in the first stage. They reported that the second stage model identified 70% more bad cases than the first stage model. This idea has some parallels with boosting strategies that apply weight trimming rules to exclude observations that have insignificant weights because they are classified correctly with a high degree of confidence (Friedman et al., 2000).

Lin (2002) reported up to 3% improvement in GINI coefficient when applying a logistic model, followed by a neural network. Zhu et al. (2001) reported improvements of between 0.5% and 1.3% when using a logistic function to combine two scores, each constructed using different sets of features but with the same target variable. Zhu et al. (2002) applied a SP approach to combine

classifiers constructed using discriminant analysis, logistic regression and neural networks using a Bayesian combination rule. The results showed increasing performance as the number of classifiers increased, yielding similar, but not superior, performance to the single best classifier, which they surmised may have been a result of over fitting. West et al. (2005) compared ensembles of neural networks constructed using cross-validation, bagging and boosting (AdaBoost) against the single best network for three different data sets. They found bagging and cross validation yielded statistically significant reductions in classification error of 2.5% and 1.9% respectively, averaged across the three data sets. However, boosting yielded significantly worse error rates than a single classifier. This was attributed to outliers, noise and mislabelled learning examples.

Although these studies have utilized different data sets and different methodologies, two interesting features stand out. The first is that the greatest uplift in performance is reported by Myers and Forgy (1963) with their simple 2-stage MS strategy, utilizing the second stage model to make the final classification decision. They effectively used the first stage model to identify "outliers" at the tail end of the score distribution, allowing the classification model to be constructed on marginal cases. This is a somewhat similar in principle to the idea underpinning support vector machines. The second is that there has been relatively little research effort to compare and contrast different multiple classifier systems in conjunction with different classification methodologies within the credit scoring arena. Only in the study by West et al. (2005) was more than a single combination strategy given consideration, and in this case only one type of classifier (neural networks) was used.

4. Experimental design

Five base methods of classifier construction were considered: logistic regression (LR), linear discriminant analysis (LDA), classification and regression trees (CART), artificial neural networks (NN) and K-Nearest Neighbour (KNN). LR, LDA, CART, NN and KNN were chosen for a number of reasons. First, each utilizes a different form of parameter estimation/learning. Second, between them they generate four different model forms in concordance with Kuncheva's taxonomy of base classifiers (Kuncheva, 2004). Third, all are practically applicable within real world consumer lending environments, with known examples of their application within the financial services industry.

To begin, single classifiers were constructed using each method. These were used to provide benchmarks against which multiple classifier systems were assessed. For CART models binary splitting rules were employed based on maximum entropy. The part of the data set allocated to model development was segmented 80/20 train/test. The tree was grown using the 80 percent allocated for training, with a minimum leaf size set at 5 observations. Pruning was applied based on minimized sum of squared error, calculated using the 20 percent test sample, as advocated by Quinlan (1992).

For neural network models a MLP architecture was adopted with a single hidden layer and one output node. The logistic activation function was applied to all neurons in the hidden layer and the output layer. A relatively fast training algorithm was necessary to allow experiments to be completed in realistic time, and for this reason network training utilized the quasi-Newton algorithm with a maximum of 200 training epochs.

For each data set the network structure was determined a priori based on the results of preliminary experiments (the resulting network structure was then used for all experiments involving that data set; i.e. all networks had the same number of units in the hidden layer). Preliminary experiments were performed using a randomly selected 50 percent of observations split 80/20 train/test (50 percent was used to correspond with the size of the folds used in model development – as described below). N-1 models were created using $2,3,\ldots,N$ hidden units, where N was equal to the number of inputs. The performance of each model was then evaluated on the test set. A 5-fold strategy was applied, with a different fold acting as the test set each time. An average of the number of hidden units in the best model for each of the five folds was taken and rounded up to the nearest whole number, and this number of hidden units used for all subsequent experiments. Given the size of the data sets employed and the number of independent variables it was considered unlikely that over-fitting would be an issue for any reasonable number of hidden units. As for CART, the development data was segmented 80/20 train/test.

To reduce the chance of network training finding local minima, for each network 25 preliminary training runs were performed using random sub-sets of the training data, with the initial weights for each run selected at random from a normal distribution. The best set of weights found using the preliminary runs were then used to initialize the weights for the main training run. To ensure consistent results, the same random number initialization was used for all network training for all experiments.

For KNN a similar approach to finding K was adopted as that for choosing the number of hidden units for neural networks. A set of preliminary experiments were performed, using a randomly selected 50 percent of observations, with values of K of ranging from between 2 and 2,500 examined. Values above 2,500 were not considered as the run times became prohibitive above this value.

Five types of static parallel multiple classifier system were explored. The first static parallel system (SP1) combined the output of the five baseline classifiers to produce a final classification decision using a simple majority vote. The second static parallel system, SP2, applied stepwise logistic regression to generate a weighted function of the five baseline classifier scores. SP3 used a neural network with a single hidden layer to produce a final score. The five baseline scores acted as inputs, and five units were included in the hidden layer. SP4-LR applied cross validation. Stratified random sampling was applied to allocate observations to 50 equally sized folds. 50 classifiers were then constructed, with a different fold excluded for each model. A final classification decision was made using simple majority vote. In the case of tied votes the final classification decision was chosen at random. SP4-LDA, SP4-CART, SP4-NN and SP4-KNN applied cross validation again, but this time for LDA, CART, NN and KNN respectively. SP5-LR applied bagging with logistic regression, using up to 50 independently drawn samples with replacement. The final classification decision was made using simple majority vote. SP5-LDA, SP5-CART, SP5-NN and SP5-KNN applied an identical bagging algorithm using LDA, CART, NN and KNN respectively.

Three multi-stage classifier systems were investigated. The first multi-stage classifier system, MS1-LR, is the Discrete AdaBoost algorithm as described in Fig. 1, applied to logistic regression.

In Fig. 1, The training set M_0 is defined as containing n_0 observations $(x_1,y_1)\ldots(x_{n0},y_{n0})$ with class labels $y \in (-1,+1)$. At each iteration of AdaBoost a classifier is constructed using the weighted population (Step 4). A sum of errors measure (E_t in Step 5) is then calculated and applied to adjust the weighting assigned to each observation (Step 8). The result is that the weight assigned to correctly classified cases is diminished, while the weight for incorrectly classified cases is increased. Z_t in step 8 is a normalizing constant, chosen such that the sum of weights at each iteration are equal to one. The final classification decision is made using weighted majority vote (Step 11). The choice of T is problem specific, but values ranging from 25 to 100 are widely quoted in the literature. For the purpose of this study T was set to a maximum of 50, following preliminary testing that indicated no performance improvements results from having more iterations than this. The

1. Initialise t=0
2. Initialise a probability distribution $D_0(i) = 1/n_0$ for all $i \in (1 \ldots n_0)$
3. DO While t<T
4. Construct classifier C_t from M_0 using weights D_t
5. Calculate weighted error as: $E_t = \sum_{i=1}^{n^0} D_t(i) \left| C_t(x_i) - y_i \right|$
6. IF $E_t \leq 0$ OR $E_t \geq 0.5$ then STOP
7. Calculate $\alpha_t = 0.5 \ln\left(\frac{1-E_t}{E_t}\right)$
8. Create distribution $D_{t+1}(i) = \frac{D_t(i)\exp(-\alpha_t y_i C_t(x_i))}{Z_t}$
9. t=t+1
10. END While loop
11. Assign final classification decision as $C^* = \sum_{t=0}^{T-1} (\alpha_t C_t(x))$

Fig. 1. The Discrete AdaBoost Algorithm.

second multi-stage classifier system, MS2-LR, is a new boosting algorithm designated Error Trimmed Boosting (or ET Boost for short) and can be considered to be a boosted variant of the trimmed least squares regression approach proposed by Ruppert and Carroll (1980). As described in Fig. 2, ET Boost applies a trimming strategy that successively removes well classified cases from the training set.

With ET Boost, at the tth iteration the error, e_{ti}, is calculated for all n_0 observations using classifier C_t (Step 4). Then, $INT(n_t^* \lambda)$, $(0 < \lambda < 1)$ observations with the largest error are selected for construction of the $t+1$th classifier (Steps 5–7). λ is a control parameter that determines the rate at which the size of the development sample decreases between subsequent iterations. Note that a key feature of ET Boost is that at each iteration all observations are reconsidered for inclusion in classifier construction. There is no weighting of observations. Thus observations excluded from classifier construction in one iteration may be included in the next iteration if the relative error of an observation increases between subsequent iterations. ET Boost is a very simple, yet flexible, boosting algorithm because it can cater for many different error functions and many different combination functions (step 10). It can also be applied to any type of classifier/regression methodology, regardless of whether or not they generate likelihood estimates, and can therefore be applied to problems with

1. Initialize t=0
2. DO while t<T
3. Construct classifier C_t from M_t
4. e_{ti}=$f(y_i, C_t(x_i))$, $x_i \in M_0$, i=1.. n_0
5. sort elements of M_0 in descending order of e_{ti}
6. n_{t+1}= $INT(n_t * \lambda)$
7. Create set M_{t+1} containing first n_{t+1} elements of M_0
8. t=t+1
9. END while
10. Assign final classification decision as $C^* = \phi(C_0 \ldots C_{T-1})$

Fig. 2. Error Trimmed Boosting (ET Boost) algorithm.

multi-category or continuous outcomes. For the purpose of this study, the error function is chosen to be absolute error: $|y_i - C_t(x_i)|$. Simple majority vote is applied as the combination function to put ET Boost on equal footing with bagging and cross validation. Exploratory analysis showed ET Boost to yield good performance for a wide range of values of λ and T. These parameters were therefore chosen on a somewhat arbitrary basis to be 0.975 and 50 respectively.

MS3-LR utilizes the individual classifiers developed for MS2-LR. In each case the classification decision is made using the single best classifier from the set of T available classifiers. MS3 can be considered a generalization of the 2-stage strategy utilized by Myers and Forgy, 1963). MS1-LDA,...,MS3-LDA, MS1-CART,...,MS3-CART, MS1-NN,...,MS3-NN and MS1-KNN,...,MS3-KNN were generated using LDA, CART, NN and KNN respectively.

Two local accuracy dynamic classification systems where explored. For the first, DCS-LA1-*L*-LR, the ranked scores from the baseline logistic regression model were used to segment the population into L segments, with segment boundaries chosen to maximize the entropy across the L segments. Logistic regression was then applied to generate a separate model for each segment. The process was then repeated for LDA, CART, NN and KNN. For the purposes of this exercise values of L = 2 and L = 4 were considered. For DCS-LA2-*K-LR* a clustering algorithm was applied using nearest centroid sorting to create K clusters, with values of K = 2 and 4 considered. Logistic regression was then applied to each of the resulting clusters. The process was repeated with LDA, CART, NN and KNN. A single DCS-GA strategy was considered, DCS-GA-*K*. For each of the K clusters defined for DCS-LA2-*K*, the performance of the five baseline classifiers was measured and the best one chosen for K = 2 and 4 to generate solutions DCS-GA-2 and DCS-GA-4 respectively.

5. Evaluating performance

The performance of each classifier system was assessed on the misclassification rate for the cut-off score, S, where S was the score for which $100^* N_b(N_g + N_b)$ percent of the population scores S or below. N_g and N_b are the numbers of goods and bads respectively. Therefore, a perfect classifier would correctly assign all goods scores greater than S and all bads scores less than or equal to S. For dynamic classifier systems that consider several independent sub-regions, separate values of the cut-off score, S, were defined for each region.

Performance metrics were calculated using *kj*-cross validation, a variant on standard *k*-fold cross validation. The population is segmented into k equally sized folds, $F_1, \ldots, F_k$ using stratified random sampling. A model is then constructed with fold F_i, $i \in (1, \ldots, k)$ assigned as the validation fold. F_i is then segmented into j sub-folds for which performance metrics are calculated. The process is repeated, k times, generating a population of $k^* j$ performance measures. The performance of two classifier systems is then compared using a paired t-test. *kj*-cross validation is computationally cheap compared to many other validation methods. It is suitable for situations where large data sets are available and experiments are concerned with the comparative performance of competing classification techniques, rather than obtaining the best performance across the entire data set upon which classifiers are being constructed. Another desirable of feature of *kj*-cross validation is that for the special case where k = 2, each model is constructed and validated using independent data sets. This overcomes the problem of non-independence of the training data associated with standard *k*-fold validation (k > 2) which can lead to inflated values of test statistics when applying paired t-tests (Dietterich, 1998). For the purpose of this study k = 2, j = 25.

6. Empirical data

Two real world data sets were available. Data set A was supplied by Experian UK. It contained details of retail credit applications made to several lending institutions between April and June 2002. It can therefore, be taken as providing a good example of the UK population of retail credit applications made during this period. 12 month performance data was provided and used to generate $a+1$, -1 target variable (Good = 1, bad = -1). Those <=1 month in arrears, and no history of seriously delinquency within the last 6 months (3 + months in arrears) were classified as good. Those currently 3 + months in arrears, or 3+ months in arrears at any time within the last 6 months were classified as bad. All other cases were classified as "indeterminate" and excluded. This definition of good/bad was chosen because it is consistent with default definitions widely applied by practitioners, based on bads being 3 + cycles delinquent and goods as up-to-date or no more than 1 cycle delinquent (Hand and Henley, 1997; Lewis, 1992; McNab and Wynn, 2003; McNab and Taylor, 2008; Finlay, 2010). It is noted however, that alternative good/bad definitions could have been applied, and that the precise definition of good/bad and the forecast horizon are lender and problem specific.

The final data set contained 88,789 observations of which 75,528 cases were classified as good and 13,261 as bad. 39 independent variables were available. These included applicant provided information such as age, residential status and income, as well as a common credit reference data such as number, value and time since most recent delinquency, account performance history, number of recent credit searches, time on electoral roll and Experian's MOSAIC postcode level classifier.

Data set B was a behavioural scoring data set provided by a supplier of revolving credit. A random sample of existing customer accounts was taken from single point in time during 2002. Therefore, some accounts were new for customers, while others represented customers that had been on the books for many years. Account performance data was attached 12 months after the sample point. A similar good/bad definition to Data set A was applied, with goods defined as <=1 month in arrears, bads as 3 + months in arrears. After removing exclusion such as indeterminates, dormant accounts, accounts less than 3 months old and those already classified as bad, the data set contained 120,508 goods and 18,098 bads. 54 independent variables were available, examples of which were current and historic statement balance, arrears status, payments and various ratio's of these variables over 3, 6 and 12 months.

7. Balancing and data pre-processing

Data sets A and B are unbalanced. In general better model performance is obtained when undersampling or oversampling is applied to create a balanced sample (Weiss, 2004). For the study reported in this paper 6 times oversampling was applied for data set A and 7 times oversampling for data set B so that the number of goods and bads were approximately equal. Oversampling was chosen in preference to undersampling on the basis of preliminary experiments showing it to outperform undersampling. Balancing was not applied to the validation part of the data sets used to calculate performance metrics. Retaining an unbalanced validation data set (or a balanced validation data set to which a suitable weighting factor is applied) is important because the classification rules employed by lenders are dependent upon the properties of the "through the door" population of credit applicants, which is highly unbalanced in all practical cases.

Data pre-processing was performed using the weights of evidence transformation. This is a standard pre-processing technique applied to credit scoring problems (Crook et al., 2007; Finlay, 2010; Hand and Henley, 1997; Siddiqi, 2005). Continuous variables such as income were sorted in ascending order, and binned into ten equally sized groups, each containing ten percent of the population. Observations with default values or containing unusual values (outliers) were assigned to separate bins. The weight of evidence in each bin was calculated as $LN\left(\frac{(g_i/G)}{(b_i/B)}\right)$, where G and B are the number of goods and bads in the population, and g_i and b_i are the number of goods and bads within the *ith* bin. The weight of evidence was then substituted for the true attribute value. For categorical variables each category was allocated to a separate bin. In cases where a category contained relatively few observations, coarse classing, as described by Thomas et al. (2002) was applied and the category merged with another with which the good:bads odds were most similar. Note that when applying methods that perturb the data, such as bagging and boosting, the weights of evidence must be recalculated each time the data is modified to take into account the changing proportion of goods and bads within each bin.

After balancing and pre-processing, a preliminary variable selection exercise was undertaken. Stepwise logistic regression was applied to the full population with a 5% significance level applied for model entry/exit. The data was segmented into 5 folds, with a model built using the first four folds. The process was repeated five times with a different fold excluded on each occasion. Any variable appearing in any of the 5 models was retained for the main modelling exercise. After pre-processing and variable selection, 29 of original 39 variables were available for model construction for data set A. For data set B 36 variables were selected.

8. Results and findings

Table 1 shows the percent of misclassified cases, and the improvement over the baseline for each classifier system.

In Table 1 the performance of the single best baseline model is taken as the reference point in the "Absolute improvement" columns. Figures refer to incremental improvement. If system X misclassifies 20% of cases and system Y misclassifies 19% of cases, then the improvement of Y over X is 5%. The "Relative improvement" columns are for classifier systems that utilize only a single form of classifier. The figures in this column refer to the improvement over the baseline classifier of that type. For example, for SP5-NN (Bagging with neural networks), for data set A the improvement over the baseline neural network model is 1.53%. Shaded cells indicate that performance is superior to the baseline model at a 99% level of significance.

Let us begin by examining the performance of the baseline classifiers. NN is the best for both data sets. The difference between NN and the next best classifier is small, but statistically significant for Data set B, but not for data set A. For both data sets the performance of CART is considerably worse than other methods. This is despite a number of preliminary experiments that looked at different pruning/data pre-processing strategies. These preliminary experiments suggested that for credit scoring data sets, using discretized (weights of evidence) data gave superior performance than using continuous variables in their raw form or dummy variables. This is a conclusion supported by Baesens et al., 2003a) who also found that using discretized data yielded better results for CART. Overall, the results for the baseline classifiers are not particularly surprising and are compatible with previous empirical studies of classifier performance for credit scoring data sets. In particular, the failure of the nonlinear models to outperform their linear competitors by a more significant margin, lends support to the argument that credit scoring data sets are only weakly nonlinear in nature (Baesens et al., 2003a).

Table 1
Performance of competing classifier systems

Classifier system	Data set A			Data set B		
	% Misclassified	% Absolute improvement over baseline	% Relative improvement over baseline	% Misclassified	% Absolute improvement over baseline	% Relative improvement over baseline
Baseline models						
LR	13.12%	−0.31%	N/A	15.15%	−1.54%	N/A
LDA	13.21%	−0.99%	N/A	15.11%	−1.27%	N/A
CART	14.38%	−9.94%	N/A	15.83%	−6.10%	N/A
NN	13.08%	0.00%	N/A	14.92%	0.00%	N/A
KNN	13.67%	−4.51%	N/A	15.59%	−4.49%	N/A
Static parallel						
SP1 (simple majority vote)	13.03%	0.38%	N/A	14.69%	1.52%	N/A
SP2 (Logistic regression combination)	12.21%	6.65%	N/A	14.18%	4.96%	N/A
SP3 (Neural network combination)	12.16%	7.03%	N/A	14.17%	5.03%	N/A
SP4-LR (cross validation)	13.13%	−0.38%	−0.08%	15.15%	−1.54%	0.00%
SP4-LDA (cross validation)	13.17%	−0.69%	0.30%	15.11%	−1.27%	0.00%
SP4-CART (cross validation)	13.23%	−1.15%	8.00%	14.78%	0.94%	6.63%
SP4-NN (cross validation)	13.10%	−0.15%	−0.15%	14.73%	1.27%	1.27%
SP4-KNN (cross validation)	13.66%	−4.43%	0.07%	15.61%	−4.62%	−0.13%
SP5-LR (bagging)	13.11%	−0.23%	0.08%	15.14%	−1.47%	0.07%
SP5-LDA (bagging)	13.20%	−0.92%	0.08%	15.07%	−1.01%	0.26%
SP5-CART (bagging)	12.28%	6.12%	14.60%	14.27%	4.36%	9.85%
SP5-NN (bagging)	12.88%	1.53%	1.53%	14.66%	1.74%	1.74%
SP5-KNN (bagging)	13.67%	−4.51%	0.00%	15.51%	−3.95%	0.51%
Multi-stage						
MS1-LR (adaboost)	13.04%	0.31%	0.61%	15.00%	−0.54%	0.99%
MS1-LDA (adaboost)	13.08%	0.00%	0.98%	14.95%	−0.20%	1.06%
MS1-CART (adaboost)	14.39%	−10.02%	−0.07%	15.44%	−3.49%	2.46%
MS1-NN (adaboost)	12.95%	0.99%	0.99%	14.78%	0.94%	0.00%
MS1-KNN (adaboost)	13.56%	−3.67%	0.80%	15.27%	−2.35%	2.05%
MS2-LR (ET Boost)	12.06%	7.80%	8.08%	13.13%	12.00%	13.33%
MS2-LDA (ET Boost)	12.47%	4.66%	5.60%	13.70%	8.18%	9.33%
MS2-CART (ET Boost)	11.87%	9.25%	17.45%	13.15%	11.86%	16.93%
MS2-NN (ET Boost)	12.02%	8.10%	8.10%	13.32%	10.72%	10.72%
MS2-KNN (ET Boost)	12.20%	6.73%	10.75%	13.37%	10.39%	14.24%
MS3-LR (adaboost, best single model)	13.07%	0.08%	0.38%	14.99%	−0.47%	1.06%
MS3-LDA (adaboost, best single model)	13.18%	−0.76%	0.23%	15.11%	−1.27%	0.00%
MS3-CART (adaboost, best single model)	14.38%	−9.94%	0.00%	15.84%	−6.17%	−0.06%
MS3-NN (adaboost, best single model)	13.05%	0.23%	0.23%	14.92%	0.00%	0.00%
MS3-KNN (adaboost, best single model)	13.56%	−3.67%	0.80%	15.40%	−3.22%	1.22%
MS4-LR (ET Boost, best single model)	13.03%	0.38%	0.69%	14.92%	0.00%	1.52%
MS4-LDA (ET Boost, best single model)	13.10%	−0.15%	0.83%	15.11%	−1.27%	0.00%
MS4-CART (ET Boost, best single model)	14.39%	−10.02%	−0.07%	15.84%	−6.17%	−0.06%
MS4-NN (ET Boost, best single model)	13.06%	0.15%	0.15%	14.86%	0.40%	0.40%
MS4-KNN (ET Boost, best single model)	13.50%	−3.21%	1.24%	15.51%	−3.95%	0.51%
Dynamic classifier selection						
DCS-LA1–2-LR (score based segmentation)	14.50%	−10.86%	−10.52%	15.74%	−5.50%	−3.89%
DCS-LA1–4-LR (score based segmentation)	16.05%	−22.71%	−22.33%	16.75%	−12.27%	−10.56%
DCS-LA1–2-LDA (score based segmentation)	14.67%	−12.16%	−11.05%	15.75%	−5.56%	−4.24%
DCS-LA1–4-LDA (score based segmentation)	15.73%	−20.26%	−19.08%	16.75%	−12.27%	−10.85%
DCS-LA1–2-CART (score based segmentation)	15.28%	−16.82%	−6.26%	16.93%	−13.47%	−6.95%
DCS-LA1–4-CART (score based segmentation)	15.95%	−21.94%	−10.92%	17.68%	−18.50%	−11.69%
DCS-LA1–2-NN (score based segmentation)	14.69%	−12.31%	−12.31%	15.72%	−5.36%	−5.36%
DCS-LA1–4-NN (score based segmentation)	16.24%	−24.16%	−24.16%	17.20%	−15.28%	−15.28%
DCS-LA1–2-KNN (score based segmentation)	14.82%	−13.30%	−8.41%	16.19%	−8.51%	−3.85%
DCS-LA1–4-KNN (score based segmentation)	15.34%	−17.28%	−12.22%	17.25%	−15.62%	−10.65%
DCS-LA2–2-LR (cluster based segmentation)	13.68%	−4.59%	−4.27%	15.36%	−2.95%	−1.39%

Table 1 (*continued*)

Classifier system	Data set A			Data set B		
	% Misclassified	% Absolute improvement over baseline	% Relative improvement over baseline	% Misclassified	% Absolute improvement over baseline	% Relative improvement over baseline
DCS-LA2–4-LR (cluster based segmentation)	14.04%	−7.34%	−7.01%	15.64%	−4.83%	−3.23%
DCS-LA2–2-LDA (cluster based segmentation)	13.74%	−5.05%	−4.01%	15.39%	−3.15%	−1.85%
DCS-LA2–4-LDA (cluster based segmentation)	14.08%	−7.65%	−6.59%	15.67%	−5.03%	−3.71%
DCS-LA2–2-CART (cluster based segmentation)	15.51%	−18.58%	−7.86%	16.72%	−12.06%	−5.62%
DCS-LA2–4-CART (cluster based segmentation)	16.39%	−25.31%	−13.98%	17.20%	−15.28%	−8.65%
DCS-LA2–2-NN (cluster based segmentation)	13.88%	−6.12%	−6.12%	15.29%	−2.48%	−2.48%
DCS-LA2–4-NN (cluster based segmentation)	14.59%	−11.54%	−11.54%	15.74%	−5.50%	−5.50%
DCS-LA2–2-KNN (cluster based segmentation)	13.98%	−6.88%	−2.27%	15.86%	−6.30%	−1.73%
DCS-LA2–4-KNN (cluster based segmentation)	15.22%	−16.36%	−11.34%	16.03%	−7.44%	−2.82%
DCS-GA-2	14.27%	−9.10%	N/A	15.65%	−4.89%	N/A
DCS-GA-4	13.40%	−2.45%	N/A	15.63%	−4.76%	N/A

All of the static parallel systems show some potential to significantly outperform the baseline. However, cross validation and Bagging are weakest, with only ensembles using CART showing any major improvement (SP4-CART and SP5-CART). Simple majority vote of the baseline classifiers is also relatively poor, showing at best modest improvements. The two best performing static parallel systems are SP2 & SP3 (neural network and logistic regression combination of the baseline classifiers), and SP5-CART (Bagging with decision trees). For data set A SP3 is best, closely followed by SP2 and SP5-CART, with improvements of 7.03%, 6.65% and 6.12% respectively. For data set B the situation is identical. With 5.03%, 4.96% and 4.36% improvement for SP3, SP2 and SP5-CART respectively.

Multi-stage systems also provide statistically significant benefits over the baseline models. The clear winner in this category (and best across all categories and both data sets) is ET boost, which provides similarly large and significant improvements over the baseline and other multiple classifier systems for all methods considered. The performance of ET boost, compared with bagging and AdaBoost at each iteration of boosting is shown in Fig. 3.

ET boost outperforms other methods, but with data set A, there is evidence of rapid deterioration in performance for NN and LR once a certain number of boosts is exceeded. However, in some situations, it would appear that performance has not peaked after 50 boosts. This suggests that the number of boosts and the decay rate for ET Boost should be chosen after performing several preliminary runs to obtain optimal or near optimal performance.

An important question is why does ET boost outperforms other methods by such significant margins, and in particularly what differentiates it from AdaBoost? One reason may be the level of "inertia" displayed by each system. Both algorithms display adaptive behaviour and look to perturb the data at each iteration of the boosting algorithm. However, as Friedman et al. (2000) point out, with AdaBoost, once the weight associated with an observation becomes trivial it may as well be excluded from further iterations of the algorithm because the weight is unlikely to recover to a significant level. With ET boost this is not the case. Observations effectively have weights of 0 or 1 dependant upon the magnitude of the error generated by the classifier at the previous iteration. An observation can therefore, flip between inclusion and exclusion between iterations. Another reason may be the performance of the individual classifiers comprising each boosting ensemble. With ET boost, each individual classifier outperformed the corresponding AdaBoost classifier for the first few boosts, and this may give ET boost an added advantage.

All of the dynamic classifier systems performed poorly, and performance with four sub-regions was consistently worse than with two sub-regions. This result was somewhat surprising, but can be attributed to three factors. First, it suggests that the relationships within the data, following the weights of evidence transformations, are linear and there are no significant interactions. Evidence for this is the relatively close performance of the logistic regression and neural network baseline classifiers, and concurs with the majority of studies that have compared logistic regression and neural networks in credit scoring. Second, both of the local accuracy dynamic classifier systems (DCS-LA1 and DCS-LA2) resulted in several regions where there were relatively few bads (<1,000 cases) which is considered a small number in credit scoring terms. The sparsity of data in these regions meant that the fitted models may have been less efficient predictors than models constructed on the entire population. Over-fitting may also have been an issue in the smallest regions. Third, the decision rule defining the cut-off was applied to each region independently. This imposed a constraint not present for the single models; i.e. better classification might have resulted if the only constraint was that the sum of observations below the cut-off across all segments was equal to the total number of bads. It may be the case that alternative segmentation and/or cut-off rules would improve the performance of the dynamic classifier systems considered. However, this is beyond the scope of this paper and is recommended as a subject for further research.

9. Increasing complexity and diminishing returns. Of what practical value are multi-classifier systems within real world lending environments?

Many academic texts that discuss credit scoring argue that new classification methodologies are justified on the grounds that very small, but significant increments in classifier performance translate into large bottom line financial benefits. For example (Thomas et al., 2002; Baesens et al., 2003a; West et al., 2005) to name but a

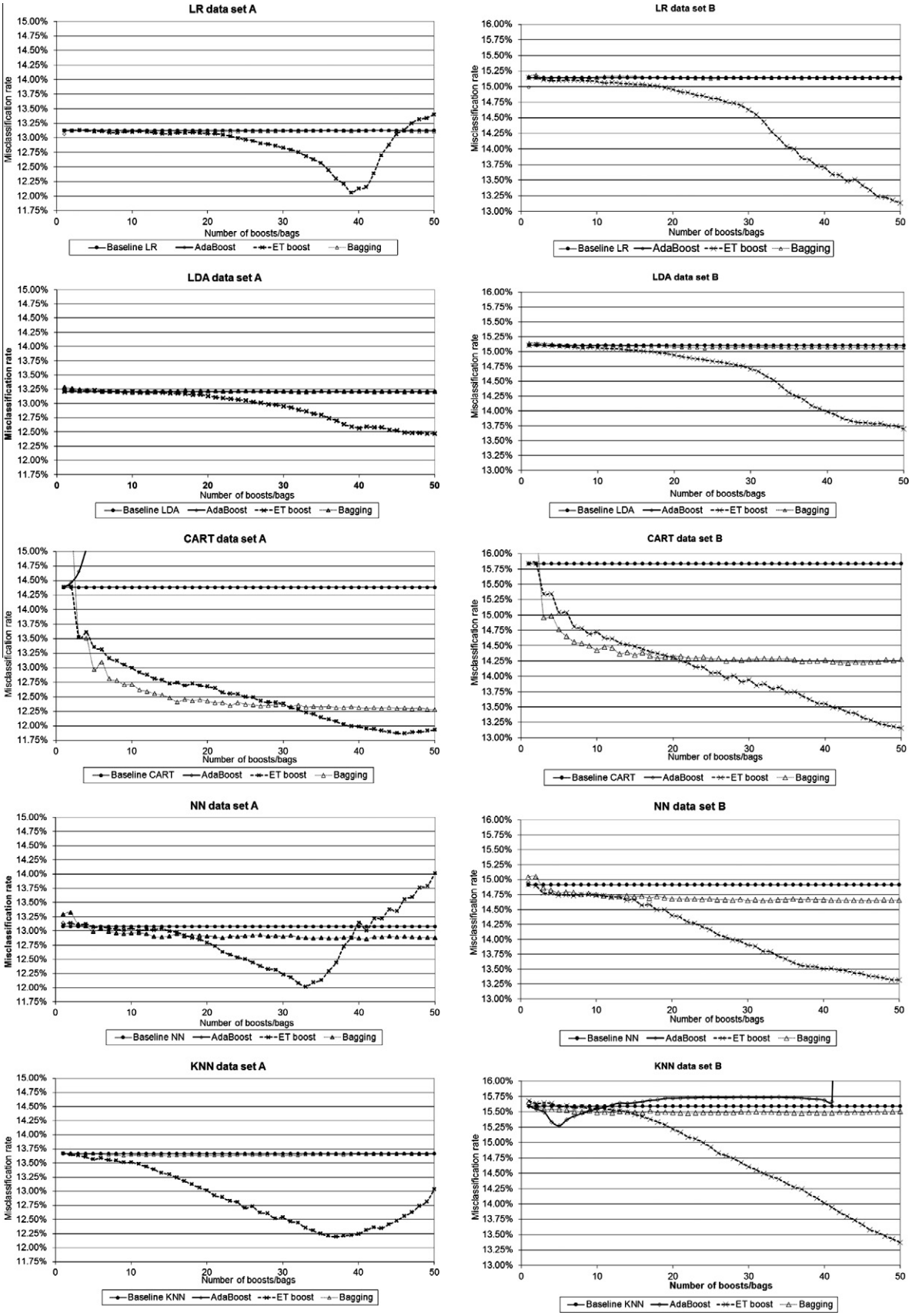

Fig. 3. Performance of ET boost, AdaBoost and Bagging.

few. Indeed, I myself make this point in the introduction to this paper. If one is purely concerned with classifier performance, then I believe this and other papers present a good case for using more complex classifiers (of which multi-classifier systems can be

considered a subset). However, the evidence from the financial services industry is that practitioners continue to favour simple linear/log linear models (Thomas et al., 2001a,b; Siddiqi, 2005; Crook et al., 2007; Thomas, 2009; Finlay, 2010) and it is not unreasonable to ask why more advanced classifiers are not employed in preference to simple linear/log linear ones. One reason is that classification performance is just one of many criteria that lending institutions apply within the cost/benefit case used to decide what type of model to employ. One other criteria is cost. This includes development cost, implementation cost and ongoing maintenance costs over the life of the model. A related issue is the rate at which classifier performance deteriorates (Long, 1976) and there is evidence to suggest that the more complex a classification model is, the more rapidly its performance deteriorates (Hand, 2006). This implies that if, as in the vast majority of empirical studies (including this one), classifier performance is evaluated using a holdout sample taken from the same time frame as the development sample, over-optimistic results will be reported for systems with faster decay rates compared to slower ones.

A counter argument to the increased costs associated with complex classifiers is that one can anticipate a point in the future when the costs are mitigated as improvements in software and hardware continue to accrue, and greater automation is applied to classifier development and deployment. Given past trends one might expect that eventually the cost differential between "simple" and "complex" classifiers will become negligible. In the late 1980s, for example, anyone wanting to implement even the simplest credit scoring model had to acquire significant IT resource, at considerable cost, to implement their models, and once implemented, monitoring the model could be a logistical nightmare. Similarly, one reason why the parameters of credit scoring models were originally represented by integers was that the CPU overhead required to carryout floating point calculations resulted in a significant cost overhead (Finlay, 2010). Today, decision engine software enables many model forms, including decision trees, neural networks and linear models to be embedded into an organization's business infrastructure at the touch of a button. Likewise, the cost of calculating credit scores is now negligible, even for organizations scoring many millions of accounts each day.

Perhaps more difficult hurdles to overcome are ones of human understanding and the explicability of the scoring process encapsulated within the classifier in question. Senior managers and regulators are unlikely to sanction the implementation of models that they do not understand. Likewise, in some regions there is a legal requirement to be able to explain the workings of a model, to provide consumers with quantitative explanations for the scores they receive (Hand, 2006). However, progress is being made in this area too. For example, one line of research that shows considerable promise is rule extraction, whereby a set of simple and comprehensible rules are found to explain the behaviour of a classifier (Baesens et al., 2003b; Martens et al., 2007).

Finally, it is perhaps worth thinking about what one means by a "complex classifier." For the purposes of discussion I assume that "complexity" is a function of the number of parameters each classifier contains and the method used to combine the parameters together, and that linear parameter combination functions are taken to be less complex than nonlinear combination functions. Although, I accept that other/extended definitions of classifier complexity could be defined. On this basis it is easy to argue that a boosted/bagged classifier system comprising N linear/log linear models, or a (weighted) majority vote of N such classifiers, are no more complex than a single neural network model with N units in the hidden layer. Indeed, several examples of such classifiers are demonstrated to give superior performance over a single neural network in this paper. One can go further and say that if the individual classifier functions are associative, and the classifier combination function is also associative, then a boosted or bagged ensemble of classifiers can be represented as a single classifier in which the parameter coefficients have been created by applying the relevant combination rule(s) to the individual parameters of each component classifier in the ensemble. For example, a set of logistic regression derived classifiers combined via weighted majority vote is equivalent to a single classifier where each parameter is the weighted sum of the parameters in the other models.

To summarise, I believe that there are real barriers to the implementation of some types of multi-classifier systems within real world lending environments, but that these barriers are not insurmountable given time. Therefore, continued research into complex multi-classifier systems is justified from a practical as well as an academic perspective.

10. Conclusion

In this paper a study of multi-classifier systems has been presented and a new boosting algorithm, ET Boost, introduced. An empirical study was then undertaken, comparing the performance of ET boost with a number of other popular multi-classifier systems using two real world credit scoring data sets. ET Boost consistently outperformed all other multiple classifier systems, across both data sets, regardless of the type of base classifier employed.

Most static parallel and multi-stage combination strategies, while inferior to ET Boost, provided statistically significant improvements over the single best classifier. Bagging with decision trees performed well, as did weighted majority voting systems that combined the outputs of models constructed using different methodologies. Adaboost did provide statistically significant improvements in performance over the single best classifier in a number of experiments, but was relatively poor compared to the best multi-stage and static parallel methods. This concurs with previous comparisons of bagging and boosting applied to credit scoring (West et al., 2005). Dynamic classifier systems, that look to segment the population into a number of sub-regions, consistently performed poorly.

With regard to future research, further analysis of ET Boost is proposed. In particular, further empirical studies of ET Boost are required to assess its performance across a more general set of classification problems, for example response modelling for direct marketing and insurance claim propensity. Comparisons should also be made between ET boost and other forms multiple-classifier systems that are not covered in this paper, such as Random Forests and Multiple adaptive regression splines (MARS). It would also be logical to extend the study to include some of the newer classification methods that have received considerable academic attention in recent years, such as support vector machines and radial basis function networks. In addition, other measures of performance should also be considered. In particular, measures of group separation such as GINI, KS and divergence that are commonly used to assess classifier performance in the consumer credit industry, but which are not necessarily well correlated with cut-off based measures (Hand, 2005).

Acknowledgements

The author thanks the ESRC and Experian for their support for this research, and Professor Robert Fildes of Lancaster University for his comments on the paper. I am also grateful for the data contribution made by another organisation that has requested to remain anonymous. The three anonymous reviewers also provided many helpful, detailed and insightful comments for which I would like to express my thanks.

References

Baesens, B., Gestel, T.V., Viaene, S., Stepanova, M., Suykens, J., Vanthienen, J., 2003a. Benchmarking state-of-the-art classification algorithms for credit scoring. Journal of the Operational Research Society 54, 627–635.

Baesens, B., Setiono, R., Mues, C., Vanthienen, J., 2003b. Using neural network rule extraction and decision tables for credit-risk evaluation. Management Science 49, 312–329.

Banasik, J., Crook, J.N., Thomas, L.C., 1996. Does scoring a sub-population make a difference? International Review of Retail Distribution and Consumer Research 6, 180–195.

Banasik, J., Crook, J.N., Thomas, L.C., 2001. Scoring by usage. Journal of the Operational Research Society 52, 997–1006.

Bank of England (2010a). Statistical Interactive Database: series LPMVZRF. Bank of England.

Bank of England (2010b). Trends in Lending June 2010. Bank of England.

Boyle, M., Crook, J.N., Hamilton, R., Thomas, L.C., 1992. Methods Applied to Slow Payers. In: Thomas, L.C., Crook, J.N., Edelman, D.B. (Eds.), Credit Scoring and Credit Control. Clarendon Press, Oxford, pp. 75–90.

Breiman, L., 1996. Bagging predictors. Machine Learning 24, 123–140.

Breiman, L., 1998. Arcing classifiers. The Annals of Statistics 28, 149–801.

Breiman, L., 2001. Random forests. Machine Learning 45, 5–32.

Chandler, G.G., Ewert, D.C. (1976). Discrimination on the basis of sex under the equal credit opportunity act. Credit Research Centre, Purdue University.

Crook, J.N., Edelman, D.B., Thomas, L.C., 2007. Recent developments in consumer credit risk assessment. European Journal of Operational Research 183, 1447–1465.

Desai, V.S., Conway, D.G., Crook, J., Overstreet, G., 1997. Credit-scoring models in the credit union environment using neural networks and genetic algorithms. IMA Journal of Mathematics Applied in Business and Industry 8, 323–346.

Desai, V.S., Crook, J.N., Overstreet, G.A., 1996. A comparison of neural networks and linear scoring models in the credit union environment. European Journal of Operational Research 95, 24–37.

Dietterich, T.G., 1998. Approximate statistical tests for comparing supervised classification learning algorithms. Neural Computation 10, 1895–1923.

Durand, D., 1941. Risk elements in consumer instalment financing, first ed. National Bureau of Economic Research, New York.

Finlay, S., 2006. Modelling Issues in Credit Scoring. Lancaster University, Lancaster.

Finlay, S., 2010. The Management of Consumer Credit: Theory and Practice, second ed. Palgrave Macmillan, Basingstoke, UK. Appendix A.

Finlay, S.M., 2009. Are we modelling the right thing? The impact of incorrect problem specification in credit scoring. Expert Systems with Applications 36, 9065–9071.

Freund, Y., 2001. An adaptive version of the boost by majority algorithm. Machine Learning 43, 293–318.

Freund, Y., Schapire, R.E., 1997. A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences 55, 119–139.

Friedman, J.H., Hastie, T., Tibshirani, R., 2000. Additive logistic regression: a statistical view of boosting. The Annals of Statistics 28, 337–374.

Hand, D.J., 2001. Modelling consumer credit risk. IMA Journal of Management Mathematics 12, 139–155.

Hand, D.J., 2005. Good practice in retail credit scorecard assessment. Journal of the Operational Research Society 56, 1109–1117.

Hand, D.J., 2006. Classifier technology and the illusion of progress. Statistical Science 21, 1–15.

Hand, D.J., Henley, W.E., 1997. Statistical classification methods in consumer credit scoring: a review. Journal of the Royal Statistical Society, Series A-Statistics in Society 160, 523–541.

Hand, D.J., Sohn, S.Y., Kim, Y., 2005. Optimal bipartite scorecards. Expert Systems with Applications 29, 684–690.

Henley, W.E., 1995. Statistical Aspects of Credit Scoring. Open University, Milton Kynes.

Henley, W.E., Hand, D.J., 1996. A k-nearest-neighbour classifier for assessing consumer credit risk. Statistician 45, 77–95.

Huysmans, J., Baesens, B., Vanthienen, J., 2005. A Comprehensible SOM-Based Scoring System. In: Perner, P., Atsushi, I. (Eds.), Machine Learning and Data Mining in Pattern Recognition, Proceedings, 3587. Springer, Berlin, pp. 80–89.

Kittler, J., 1997. Statistical classification. Vistas in Astronomy 41, 405–410.

Kittler, J., 1998. Combining classifiers: a theoretical framework. Pattern Analysis and Applications 1, 18–27.

Kuncheva, L.I., 2004. Combining Pattern Classifiers. Methods and Algorithms. Wiley, Hoboken, New Jersey.

Kuncheva, L.I., 2002. Switching between selection and fusion in combining classifiers: an experiment. IEEE Transactions on Systems, Man and Cybernetics - Part B: Cybernetics 32, 146–156.

Kuncheva, L.I., Bezdek, J.C., Duin, R.P.W., 2001. Decision templates for multiple classifier fusion: an experimental comparison. Pattern Recognition 34, 299–314.

Lewis, E.M., 1992. An Introduction to Credit Scoring. Athena Press, San Rafael.

Lin, Y., 2002. Improvement on behavioural scores by dual-model scoring system. International Journal of Information Technology and Decision Making 1, 153–165.

Liu, R., Yuan, B., 2001. Multiple classifiers combination by clustering and selection. Information Fusion 2, 163–169.

Long, M.S., 1976. Credit screening systems selection. Journal of Financial and Qualitative Analysis 11, 313–328.

Martens, D., Baesens, B., Van Gestel, T., Vanthienen, J., 2007. Comprehensible credit scoring models using rule extraction from support vector machines. European Journal of Operational Research 183, 1479–1488.

McNab, H., Wynn, A., 2003. Principles and Practice of Consumer Risk Management, second ed. Financial World Publishing.

McNab, H., Taylor, P., 2008. Consumer Credit Risk Management. Global Professional Publishing, London.

Myers, J.H., Forgy, E.W., 1963. The development of numerical credit evaluation systems. Journal of the American Statistical Association 50, 799–806.

Narain, B., 1992. Survival Analysis and the Credit Granting Decision. In: Thomas, L.C., Crook, J.N., Edelman, D.B. (Eds.), Credit Scoring and Credit Control. Clarendon Press, Oxford, pp. 109–122.

Paleologo, G., Elisseeff, A., Gianluca, A., 2010. Subagging for credit scoring models. European Journal of Operational Research 201, 490–499.

Quinlan, J.R., 1992. C4.5: Programs for Machine Learning. Morgan–Kaufman, San Mateo, CA.

Rosenberg, E., Gleit, A., 1994. Quantitative methods in credit management: a survey. Operations Research 42, 589–613.

Ruppert, D., Carroll, R.J., 1980. Trimmed least squares estimation in the linear model. Journal of the American Statistical Association 75, 828–838.

Schapire, R., 1990. Strength of weak learnability. Journal of Machine Learning 5, 197–227.

Sewart, P.J., 1997. Graphical and Longitudinal Models in Credit Analysis. Lancaster University, Lancaster.

Siddiqi, N., 2005. Credit Risk Scorecards: Developing and Implementing Intelligent Credit Scoring. John Wiley and Sons, Hoboken, New Jersey.

Stepanova, M., Thomas, L.C., 2001. PHAB scores: proportional hazards analysis behavioural scores. Journal of the Operational Research Society 52, 1007–1016.

The Federal Reserve Board. (2010a). Federal Reserve Statistical Release G.19. The Federal Reserve Board.

The Federal Reserve Board. (2010b). Charge-off and Delinquency Rates. The Federal Reserve Board.

Thomas, L.C., 2009. Consumer Credit Models: Pricing, Profit and Portfolios. Oxford University Press, Oxford.

Thomas, L.C., Banasik, J., Crook, J.N., 2001a. Recalibrating scorecards. Journal of the Operational Research Society 52, 981–988.

Thomas, L.C., Ho, J., Scherer, W.T., 2001b. Time will tell: behavioural scoring and the dynamics of consumer credit assessment. IMA Journal of Management Mathematics 12, 89–103.

Thomas, L.C., Edelman, D.B., Crook, J.N., 2002. Credit Scoring and Its Applications. Siam, Philadelphia.

Toygar, O., Acan, A., 2004. Multiple classifier implementation of a divide and conquer approach using appearance-based statistical methods for face recognition. Pattern Recognition Letters 25, 1421–1430.

Weiss, G.M., 2004. Mining with rarity: a unifying framework. ACM SIGKDD Explorations Newsletter 6, 7–19.

West, D., 2000. Neural network credit scoring models. Computers and Operations Research 27, 1131–1152.

West, D., Dellana, S., Qian, J., 2005. Neural network ensemble strategies for financial decision applications. Computers and Operations Research 32, 2543–2559.

Whittaker, J., Whitehead, C., Somers, M., 2005. The neglog transformation and quantile regression for the analysis of a large credit scoring database. Journal of the Royal Statistical Society Series C-Applied Statistics 54, 863–878.

Zhu, H., Beling, P.A., Overstreet, G., 2001. A study in the combination of two consumer credit scores. Journal of the Operational Research Society 52, 974–980.

Zhu, H., Beling, P.A., Overstreet, G.A., 2002. A Bayesian framework for the combination of classifier outputs. Journal of the Operational Research Society 53, 719–727.