Are we modelling the right thing? The impact of incorrect problem specification in credit scoring

Steven Finlay ∗

Department of Management Science, Lancaster University Management School, Lancaster University, Lancaster LA1 4HW, United Kingdom

Abstract

Classification and regression models are widely used by mainstream credit granting institutions to assess the risk of customer default. In practice, the objectives used to derive model parameters and the business objectives used to assess models differ. Models parameters are determined by minimising some function or error or by maximising likelihood, but performance is assessed using global measures such as the GINI coefficient, or the misclassification rate at a specific point in the score distribution. This paper seeks to determine the impact on performance that results from having different objectives for model construction and model assessment. To do this a genetic algorithm (GA) is utilized to generate linear scoring models that directly optimise business measures of interest. The performance of the GA models is then compared to those constructed using logistic and linear regression. Empirical results show that all models perform similarly well, suggest that modelling and business objectives are well aligned.

© 2008 Published by Elsevier Ltd.

Keywords:
Genetic algorithms
Credit scoring
Forecasting

1. Introduction

All mainstream credit granting institutions use credit scoring – mechanically derived forecasting models of customer behaviour – to make decisions about to whom to extend credit to and on what terms. The most widely used credit scoring models predict a simple binary outcome; that is, the likelihood that an individual will be a ‘good’ customer who repays the credit advanced to them, or a ‘bad’ customer who defaults. Despite much research into the applicability of a wide variety of classification and regression methods to credit scoring problems, logistic regression remains the most widely used method in practice (Crook, Edelman, & Thomas, 2007; Finlay, 2008). This is mainly attributed to the fact that logistic regression produces simple models that are easily interpretable, as well as empirical evidence suggesting that the performance of simple linear models is only fractionally worse than more complex model forms such as neural networks and support vector machines (Baesens et al., 2003).

In many real world situations, the objective lender is trying to optimise through the use of a credit scoring model is different from the objective used during model development. Therefore, a key question – that has not been widely considered by the credit scoring community – is; are we modelling the right thing? And if not, what is impact of not doing so? As a simple illustration, consider logistic regression applied to a binary classification problem, where the dependent variable, y, takes values of 0 or 1. Through the application of an appropriate algorithm, a model is derived that maximises likelihood over the set of n observed cases:

$$\prod_{i=1}^{n} \left( \frac{y_i}{1-y_i} \right)$$

where $P_i$ is the posterior probability that $y_i = 1$, calculated as a function of independent variables. Yet, for many practitioners the actual point estimate for an observation is of little interest. What is of primary importance is the relative performance at specific points in the distribution of ranked model scores (Thomas, Banasik, & Crook, 2001). It is also true that for some decisions (such as where a fixed accept rate policy is in operation) the only concern is that observations fall on the correct side of the decision rule applied. Whether, an individual only just passes the cut-off score or exceeds it by a great margin is irrelevant (Hand, 2005). This can be demonstrated by considering a hypothetical example. Imagine that there exist two models that generate probabilistic estimates of credit applications being good credit risks. Two credit applications are scored by each model to produce the results shown in Table 1.

Now assume that both cases are revealed to be good payers. From a maximum likelihood perspective, Model 1 outperforms Model 2. Yet, if a lender was using these models to make credit granting decisions, say on the basis of accepting only those where the estimated probability of being good exceeds 0.8, then Model 2 is better because both cases would be accepted. Maximising likelihood is therefore no guarantee of optimal model performance in this case.

A genetic algorithm (GA) is a data driven, non-parametric heuristic search process, where the training algorithm can be chosen to...
Previous studies where GAs have been used to develop credit scoring models have reported mixed findings. Fogarty and Ireson (1993/4) took a sample of over fifty thousand accepted credit card applications and compared a GA derived Bayesian classifier with decision rules derived from a number of techniques including a nearest neighbour clustering algorithm, a decision tree and a simple Bayesian classifier. They found that the GA derived classifier performed better than other methods when assessed on classification rates, but did not perform better than a simple decision rule to classify all cases as good. Desai, Conway, Crook, and Overstreet (1997) looked at a three-way classification problem where accounts were classified as good, poor or bad payers. They reported that a GA approach was marginally better at classifying the worst accounts (bad payers) than linear discriminant analysis, logistic regression and a variety of neural network models, but did not perform as well when measured in terms of classification performance on good and poor paying accounts. Yobas, Crook, and Ross (2000) reported that while a GA derived model performed better than neural networks and decision trees on the development sample (no validation sample performance was available for the GA derived model), all three methods were outperformed by linear discriminant analysis. While the results and methodologies applied in these previous studies differ, one feature that they all have in common is that they only considered misclassification performance metrics for which the non-GA approaches used in the study were generally known to provide good levels of performance. It is, therefore, no surprise that a GA approach was not found to significantly outperform the alternative model development approaches examined.

In this paper, a GA approach is again explored, but incorporating a number of features that differentiate it from previous studies. First, the objective is primarily to determine the sensitivity of models developed using standard approaches to differences between modelling and business objectives. The actual performance of GA derived models is only a secondary consideration. Second, rather than simply judging performance of competing models on the basis of a single misclassification measure, model performance is assessed using several different criteria.

- The maximisation of the GINI coefficient (a measure of the area under the receiver operator curve) which for a discrete population of observations that fall into one of two classes and ranked by model score, can be calculated using the Brown formula:

\[ 1 - \sum_{i=1}^{n} \frac{G(i) + G(i-1)}{B(i) + B(i-1)} \]

where \( G \) and \( B \) represent the cumulative proportion of cases falling into each class respectively.

- The minimisation of the proportion of bads within the highest scoring \( x \% \) of the population; that is, the number of bads scoring \( \geq \) where can be cut-off score at or above \( x \% \) of the population score. For the purposes of this study values of \( x \) of 5, 10, 25 and 50 percent were considered.

In each case a GA is applied to generate a scoring model that maximised each objective independently, whereas a single competing model was constructed and assessed using the competitor approaches. Second, two large real world data sets are used, whereas previous studies have been based on relatively low dimensional data sets and small samples (with the exception of Fogarty and Ireson’s study). Third, solutions are considered with and without seeding – the process whereby a genetic algorithm is initialised using a number of pre-existing solutions found using some alternative technique. The GA is then applied in an attempt to improve upon the performance of the original seed solution(s).

Empirical results are presented for the two data sets; with the performance of the GA derived models compared to models constructed using logistic regression and multiple OLS regression.

2. Overview of genetic algorithms

The theory of GAs was developed in the late 1960s and early 1970s by John Holland and his associates as a means to study evolutionary processes in nature (Holland, 1975), but they were quickly adopted as a heuristic approach applicable to a wide range of optimisation problems (De Jong, 1975; Hollstien, 1971). The general principles of GAs are analogous to Darwinian principles of natural selection and survival of the fittest, and the terminology employed to describe GA training and selection is taken from the biological analogy.

With GAs, a set of possible solutions to a given problem is analogous to a population of individuals in the natural world. The goal is to combine together and mutate different solutions so that over time fitter (better) solutions evolve. Individual solution within the population is represented in the form of a finite length string, comprising a finite alphabet where the string and its component characters are analogous to chromosomes and genes, respectively (Goldberg, 1989a). From an initial (usually randomly generated) population of strings, new populations are created over a number of generations (iterations) through the application of the following genetic operators:

- Selection: from the existing population, a number of strings are selected for breeding, with selection favouring those strings that represent the best solutions found to date.
- Crossover: from the selected population, pairs of strings are matched for breeding. ‘Child’ strings are created by selecting and combining different characters from each of the parent strings.
- Mutation: each character within a string has the chance to undergo mutation, based on some random process. If selected, then the value of the character within the string is randomly reassigned to one of the possible values defined by the encoding alphabet.

The algorithm terminates when a given number of generations have occurred, or when the improvement from one generation to the next falls below a specified threshold. Despite the relative simplicity of genetic algorithms, they have been successfully applied to a wide range of diverse and complex optimisation problems (Coley, 1999; Mitchell, 1996).

3. Design and implementation of genetic algorithms

A number of parameters need to be selected for GA training, and as with methods such as neural networks, the parameters that deliver the best solution tend to be problem specific. Consequently, a good deal of trial and error can be required to find the most appropriate training parameters. The first question is how to en-
code solutions to a given problem? The approach favoured by Holland (1975) is to use binary strings. For example, if the objective is to optimise some function of two independent variables $x_1$ and $x_2$, where $x_1$ and $x_2$ take positive integer values between 0 and 31, then any possible combination of values of $x_1$ and $x_2$ can be encoded using a string of ten binary characters representing a base two numbering system. A drawback of simple binary encoding is that in some situations a large change is required in the structure of a string to obtain a small increment in the value that the string represents. The most widely adopted method for overcoming this problem is the application of binary (reflective) Gray encoding – a process that transforms a simple binary string into a form where a by a string of length $s$ is a Hamiltonian path in the binary hyper-cube ($0,1$) (Reeves & Rowe, 2003). Non-binary encodings are also possible, including integer and real number representations, and some studies have shown that a high cardinality alphabet can lead to better GA performance than using simple binary encoding (Wright, 1991).

As to the size of the population, some theoretical work has indicated that the relationship between optimal population size and string length is exponential (Goldberg, 1989b). However, empirical research has reported that population sizes as small as 30 can lead to reasonable solutions for many problems and a population size of between $s$ and $2s$, where $s$ is the string length, is generally sufficient to allow an optimal solution to be found for binary encoded problems (Reeves, 1995b).

The population is usually initialised to a set of random solutions, but an alternative is to use seeding, where the population is initialised using a set of existing solutions found using some other method. It has been claimed that finding a good seed can be an important determinant of the quality of the final solution (Ahuja & Orlin, 1997; Ahuja, Orlin, & Tiwari, 2000) and that seeding can lead to reduced run times in what can be a computationally expensive exercise (Reeves, 1995a). However, it has also been reported that seeding can lead to sub-optimal solutions (Kapsalis, Smith, & Rayward-Smith, 1993; Levine, 1997). This is because the area of search may focus on the regions of the problem domain close to the seed solutions, and thus the GA may converge on a local rather than global optimum.

Most methods of selecting individuals for crossover are based on random selection, where the probability of selection is a function of a solution's fitness. With fitness proportional (roulette wheel) selection, the probability of selection is directly proportional to the relative fitness of the individual compared to the population average. One drawback of the fitness proportional approach is that for problems where many similarly good solutions exist, or where the flat maximum effect is prevalent – as is found in many credit scoring problems (Lovie & Lovie, 1986), fitness proportional selection can be little better than uniform random sampling; i.e. the principle of survival of the fittest is lost. While it is possible to apply scaling or a non-linear transformation to place greater emphasis on fitter individuals, alternative selection processes that place emphasis on relative rather than absolute differences between competing solutions tend to be more widely applied. With rank selection the probability of each individual being selected is based on a function of its rank:

$$ P_j = \alpha + \beta f_j^s \quad \text{subject to} \quad \sum_{j=1}^{n} (\alpha + \beta f_j^s) = 1 $$

In general a linear ranking scheme with $\beta > 1$ is sufficiently flexible for most applications (Reeves & Rowe, 2003). A variety of crossover methods have been proposed. With $k$-point crossover, $k$ cut-points are chosen at $k$ different loci along the length of the two parent strings. The characters within each string are then copied and exchanged. While $k$ is commonly chosen to take a value of $1$ or $2$, with uniform crossover the value of $k$ is set equal to $s$, the string length, so that the maximum diversity is achieved in the offspring. The perceived benefit of uniform crossover is that it removes representational bias that can arise when $k$ is low. However, it creates a high degree of disruption which can lead to movement away from the optimal solution. A less disruptive approach is parameterised multi-point crossover where a probability is assigned for crossover occurring at each locus in the string (Spears & De Jong, 1991).

After selection and crossover, mutation is randomly applied to each character of each string within the population. If the probability of mutation is too high, then it is likely that good quality solutions will be lost. However, if the probability of mutation is low, then the population may tend to stagnate and progression towards the optimal solution will be slow. Mutation rates between $1/s$ and $1/(n + \sqrt{s})$ where $s$ is the string length and $n$ is the population size, have been widely used across a wide range of optimisation problems (Coley, 1999). If elitism is applied then the very best solution(s) will always be selected for propagation into the next generation without mutation being applied. This ensures that the best solution found to date is always maintained within the current population.

Overall GAs tend to be remarkably robust, and a wide range of parameter values will lead to good solutions for many problems (Reeves, 1995b). Therefore, from a practical perspective when applying a GA approach to real world problems, it is not usually necessary (or feasible due to computational requirements) to go to great lengths to find the best possible set of parameters to use, but rather to identify the range of possible parameter values for which good solutions are likely to exist, and to then (perhaps rather arbitrarily) choose values that lie somewhere within these ranges. For those with a desire to know more about the different approaches to the design of GA’s, I suggest Coley (1999), for a practical focused introduction to the subject and either Reeves and Rowe (2003) or Goldberg (1989a) for a more theoretical treatment.

4. Data

Two data sets were available for study. The first data set (Set A) was supplied by Experian UK and contained details of credit applications made between April and June, 2002, and for which performance information was attached 12 months after the application date. After removal of outliers and indeterminates the sample contained 88,792 observations of which 75,528 were classified as good credit risks and 13,264 as bad credit risks. Goods were classified as no more than 1 month in arrears, bads as 3 months or more in arrears. This is consistent with good/bad definitions commonly reported in the literature as being applied by practitioners, based on bads being three or more cycles delinquent and goods as up-to-date or no more than one cycle delinquent (Hand and Henley 1997; Lewis 1992; McNab and Wynn 2000). Independent variables of 37 were available in Set A. These included common application form characteristics such as age, residential status and income, as well as a set of commonly used UK credit bureau variables including number, value and time since most recent CCJ/bankruptcy, current and historic account performance, recent credit searches, Electoral Roll and MOSAIC postcode level classifiers. A full list of the independent variables in Set A is contained in Appendix A.

The second data set (Set B) was a behavioural scoring data set from a company providing revolving credit, with performance data attached as at 12 months after the sample date. After exclusions such as newly opened accounts (<3 months old), dormant accounts (no significant balance within the last 3 months) accounts already in a serious delinquency status (already bad) and those classified as...
being indeterminate, the sample contained 28,517 observations. Here 24,053 were classified as good and 4,464 as bad. The good/bad definition for Set B was similar to that for Set A. Goods were defined as being no more than one month in arrears, bads as being three months or more in arrears. Set B contained 54 independent variables. These included variables such as current and historic balances, current and historic arrears status, payment to balance ratios etc. For reasons of commercial confidentiality it is not possible to provide a complete list of the variables provided for Set B.

For both data sets, the independent variables were a mixture of categorical and continuous. These were pre-processed to code them as a set of dummy variables. The dummy variable approach is generally found to provide a good approximation to non-linear features of a data set (Fox, 2000) and is a standard practice in the development of credit scoring models (Hand and Henley, 1997). It has also been observed that the coding of continuous variables as a set of dummy variables can lead to greater discrimination in credit scoring models than using either raw or transformed versions of the continuous variables (Hand and Adams, 2000). After pre-processing 218 dummy variables were available for model construction for Set A and 308 for Set B.

5. Methodology

The goal of the exercise was to apply a GA to create linear scoring functions in the form $Y = B^T X$, where $X$ is a column vector of independent variables and $B$ a column vector of parameter coefficients. It should be noted that the resulting score, $Y$, should not be interpreted as an estimate of individual performance, but merely as the relative score of each observation within the dataset. Although other types of scoring function exist, such as a multi-layer perceptron, a linear function of the independent variables was considered appropriate because linear scoring rules have continuing popularity with credit scoring practitioners, perform well when compared to a wide variety of alternative modelling techniques (Baesens et al., 2003) and are desirable for a number of reasons in addition to their overall predictive performance. This includes having an intuitive structure so that model parameters can be used to provide a qualitative explanation of the model to the layperson in support of legislative requirements, and being easily implementable within the decisioning systems employed by lenders.

For the GA derived models four competing encoding/mutation methodologies were employed:

- **GAbin**: simple binary encoding.
- **GAGray**: gray reflective binary encoding.
- **GAint1**: integer encoding.
- **GAint2**: integer encoding with adaptive mutation.

The parameters of each GA were chosen, after considering the findings reported within the literature, and after performing a number of experimental training runs on each of the two data sets. For the binary representations, $GAbin$ and $GAGray$, parameter coefficients, $B$, were defined as integer values in the range $\pm 32,767$ ($\pm 2^{15} - 1$). They were then encoded as a 16 bit binary numbers using a 0/1 alphabet. Given that many commercial credit scoring models generate integer scores in the range 0–1000 this provided a more than sufficient level of granularity. For the integer encoded schemas, $GAint1$ and $GAint2$, model parameters were coded as a string of signed integers, each of which was allowed to take values in the range $\pm 32,767$.

The population sizes for Set A and Set B were chosen as 1500 and 2250, respectively. Selection was performed using linear proportional ranking with replacement. Parameterised multi-point crossover was then applied to generate child solutions, using probabilities of between 0.25 and 0.5 for crossover at each loci of the string. A mutation rate of 1/s, where $s$ is the string length, was applied. For $GAbin$ and $GAGray$, mutation was applied by flipping the value of the mutated character to its alternative value; i.e. from 0 to 1 or vice versa, with each character in each string having an equal chance of being mutated in each generation. For integer encoding $GAint1$ mutation was applied through the use of a uniform random number, allowing the mutated character an equal probability of being assigned any value within the range of possible values ($\pm 32,767$). For the $GAint2$ encoding, an adaptive mutation strategy was applied. First, in order to try maintaining diversity within the population, while the average mutation rate was set at 1/s, the mutation rate in each generation was allowed to vary between 0 and 2/s, based on a cosine function with a period of 25 generations. Second, to encourage progression towards the optima, instead of mutation resulting in a completely random assignment of a character’s value, mutation was applied by adding or subtracting a random amount to the current value of the character selected for mutation. The mutation applied was based on a uniform random number in the range $(-M_s + D_s, +M_s + D_s)$ where $|M_s|$ is the maximum mutational change permitted in each character in generation 0, $D$ is the reduction factor and $g$, the cumulative number of generations over the current run for which no increase in performance has been seen between successive generations. For the purposes of this exercise $M_s$ and $D_s$ were set equal to 16,383 and 0.97, respectively.

The maximum number of generations was set to be either 1000 or when no improvement in validation performance had been seen for 50 generations. These values were chosen after results from a number of test runs showed performance tended to plateau somewhere between 100 and 250 generations in most cases.

While the previous paragraphs describe the final parameters chosen, it is worth noting that preliminary results tended to support the case for a wide range of parameters leading to good solutions. Population sizes >400, crossover probabilities of between 0.125 and 0.75, and mutation rates of between 0.25/s and 5/s, all yielded results very close to those generated by the final set of parameters chosen for both data sets.

Where seeding was applied, several hundred seed models were constructed using a number of variations of linear and logistic regression. This included forward and backward stepwise procedures with randomly chosen entry/exit criteria, as well as a set of models developed using different, randomly selected, 90% subsets of the main development dataset.

Given the large samples available, a development/validation/holdout methodology was adopted. The population was segmented 60/20/20 and in all cases models were developed using the 60% development sample. Where a choice of models existed (as with the GA approaches) the best model was selected based on performance on the 20% validation sample. Model performance for all competing models was then compared using the 20% holdout sample.

The GA derived models were compared with that of other four models that could also be represented in the form $Y = B^T X$ and were constructed by the following means:

- $Col_{step}$: OLS regression (forward stepwise) with a 5% criteria for model entry.
- $Col_{bst}$: OLS regression (non-stepwise).
- $Log_{step}$: Logistic regression (forward stepwise) with a 5% criteria for model entry.
- $Log_{bst}$: Logistic regression model (non-stepwise).
Stepwise models were examined as well as non-stepwise ones to reduce the likelihood of over-fitting (there was some evidence of this with Set B, but not with Set A). Correspondingly, each GA was run twice. First using the full set of available variables, and then using only those variables that featured within either of the logistic or linear stepwise regression models. This resulted in a reduced set of 116 variables for Set A and 77 variables for Set B for the second GA run. Results are reported for the best performing GA in each case.

6. Results

The performance of the GA derived models produced without seeding, and that of the competitor models are shown in Table 2. The performance of the GA derived models produced with seeding and that of the competitor models are shown in Table 3. The best performing model for each measure is highlighted in bold.

The first point to note from Tables 2 and 3 is how similar the performance of many of the competing models was across all performance measures for both data sets. It is also the case that no one type of model could be said to dominate the others entirely, with models created using different modelling techniques performing slightly better/worse in relative terms for each measure. This finding applied equally across the set of GA methodologies, with each of the four different types of GA outperforming the others in some situations, but not significantly so. From Table 2, it can be seen that the GA derived models without seeding, while competitive, were not superior to the best competitor model in any case. As can be seen from Table 3, the GA derived models using seeding did show a propensity to outperform other approaches for three out of the five performance measures, albeit by a small margin. However, for the other two performance measures, the GA lead to a model that was worse or no better than any of the seed solutions, or simply retained the best seed solution without any better solutions being found.

Overall, these results would suggest that while a GA approach can produce models that are competitive with traditional methods, they do not significantly outperform them. These results came as something of a surprise, as the original hypothesis was that the GA derived models should perform significantly better than the competitors, given the additional prior information that was included in the objective function used for model construction. An initial thought was that the parameters selected for use with the GA might not be the most appropriate, resulting in sub-optimal models being arrived at. However, the results are consistent across all four of the GA variants reported upon, and considerable preliminary testing was performed using a wide range of crossover and mutation values, using large populations and different encoding strategies. In all cases very similar levels of performance resulted which would tend to discount this idea. Another thought was that the high dimensionality and resulting sparseness of data within the problem domain may have been a factor. To investigate this, the relative performance resulting from using the full set of independent variables was compared to the performance obtained

<table>
<thead>
<tr>
<th>Table 2</th>
<th>Performance of competing models without seeding.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance measure</td>
<td>GA derived models</td>
</tr>
<tr>
<td></td>
<td>GA&lt;sub&gt;bin&lt;/sub&gt;</td>
</tr>
<tr>
<td>Set A (Application scoring data set)</td>
<td></td>
</tr>
<tr>
<td>GINI</td>
<td>0.7221</td>
</tr>
<tr>
<td>50% cut-off (%)</td>
<td>72.90</td>
</tr>
<tr>
<td>10% cut-off (%)</td>
<td>56.75</td>
</tr>
<tr>
<td>25% cut-off (%)</td>
<td>29.20</td>
</tr>
<tr>
<td>Set B (Behavioural scoring data set)</td>
<td></td>
</tr>
<tr>
<td>GINI</td>
<td>0.6544</td>
</tr>
<tr>
<td>50% cut-off (%)</td>
<td>81.52</td>
</tr>
<tr>
<td>10% cut-off (%)</td>
<td>68.39</td>
</tr>
<tr>
<td>25% cut-off (%)</td>
<td>35.94</td>
</tr>
<tr>
<td>50% cut-off (%)</td>
<td>10.20</td>
</tr>
</tbody>
</table>

Notes: (1) For cut-off measures performance is calculated as the proportion of total bads scoring above the cut-off for the desired cut-off strategy. (2) For GINI measure higher = better. For cut-off measures lower = better. (3) Indicates GA model using reduced set of variables.

<table>
<thead>
<tr>
<th>Table 3</th>
<th>Performance of competing models with seeding.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance measure</td>
<td>GA derived models</td>
</tr>
<tr>
<td></td>
<td>GA&lt;sub&gt;bin&lt;/sub&gt;</td>
</tr>
<tr>
<td>Set A (Application scoring data set)</td>
<td></td>
</tr>
<tr>
<td>GINI</td>
<td>0.7247</td>
</tr>
<tr>
<td>50% cut-off (%)</td>
<td>72.90</td>
</tr>
<tr>
<td>10% cut-off (%)</td>
<td>56.75</td>
</tr>
<tr>
<td>25% cut-off (%)</td>
<td>29.20</td>
</tr>
<tr>
<td>Set B (Behavioural scoring data set)</td>
<td></td>
</tr>
<tr>
<td>GINI</td>
<td>0.6727</td>
</tr>
<tr>
<td>50% cut-off (%)</td>
<td>81.75</td>
</tr>
<tr>
<td>10% cut-off (%)</td>
<td>65.65</td>
</tr>
<tr>
<td>25% cut-off (%)</td>
<td>34.01</td>
</tr>
<tr>
<td>50% cut-off (%)</td>
<td>8.73</td>
</tr>
</tbody>
</table>

Please cite this article in press as: Finlay, S. Are we modelling the right thing? The impact of incorrect ... Expert Systems with Applications (2008), doi:10.1016/j.eswa.2008.12.016
when using the reduced set of independent variables (those that featured within the stepwise logistic or linear regression models).

For Set A, both sets of models gave very similar levels of performance, and in some cases the results from using the reduced set were inferior to that when the full set of variables were used. For Set B the models resulting from using the reduced set of variables did tend to show better performance than when the full set was used, but the level of improvement only compared to that seen when differentiating between stepwise and non-stepwise regressions, which would tend to suggest that this was an over-fitting issue. Overall, this would suggest that the dimensionality of the problem domain and sparsity of data was not a significant contributory factor to the performance of the GA for these data sets.

7. Concluding remarks

In this paper, the suitability of the modelling objectives used to create credit scoring models have been drawn into question, given that they differ from the business measures that are widely used to assess model performance. To explore this issue genetic algorithms were used to create a set of linear scoring models that directly optimised individual measures of business interest. In all cases, there was no significant differences between the performances of the GA derived models and models constructed using logistic and linear regression. This is a somewhat surprising finding given that each GA derived models was trained to optimise a single business objective, while the linear and logistic regression models were developed to optimise more generic functions; that is, minimised sum of squared error and maximise likelihood respectively. This might be viewed as a negative finding from some quarters, but from a practitioner perspective it is very positive indeed. This is because it suggests two things. First, those modelling and business objectives are highly correlated and therefore optimising one leads to optimal or near optimal performance of the other. Second, standard modelling techniques are robust to changes in lender’s business objectives. Therefore, model performance remains close to optimal even when a lender changes the business rules (i.e. the cut-off strategies) applied to the model, and consequently models do not necessarily require redevelopment when such changes are made.

Acknowledgements

Author would like to thank the ESRC and Experian UK for their support for this research, and Professor Robert Fieldes of Lancaster University for his comments on an early draft of the paper. Author is also grateful for the contribution from a third organisation that has requested that its identity remains anonymous.

Appendix A. Variables used in model construction (Set A)

A.1. Application form variables

1. Age of applicant
2. Accommodation status
3. Council tax banding
4. Employment status
5. Employment type
6. Gross annual income
7. Home phone indicator
8. Marital status
9. Mortgage indicator
10. Number of credit cards
11. Number of dependents
12. Time at address
13. Time with bank

A.2. UK credit bureau variables (same person)

1. Director indicator (Y/N)
2. Electoral roll confirmation indicator
3. MOSAIC postcode level classifier
4. Number of active credit accounts (excluding mail-order)
5. Number of active credit accounts (including mail-order)
6. Number of credit account status 8/9s
7. Number of credit searches in the last 3 months
8. Number of credit searches in the last 6 months
9. Number of CCJs/bankruptcies
10. Number of delinquent accounts
11. Number of settled good credit accounts
12. Number of settled good credit accounts in last 12 months
13. Outstanding balance on all active credit accounts (excluding mortgages)
14. Outstanding balance on all active credit accounts (mortgages)
15. Time registered on Electoral Roll at current address
16. Time since most recent CCJ/bankruptcy
17. Time since most recent credit account status 8/9s
18. Time since most recent delinquent account
19. Value of outstanding CCJs/bankruptcy
20. Value of outstanding credit account status 8/9s
21. Value of delinquent accounts
22. Worst current arrears status on all active credit accounts
23. Worst arrears status last 6 months on all active credit accounts.

References


