

# EMOCS: Evolutionary Multi-objective Optimisation for Clinical Scorecard Generation

Diane P. Fraser  
University of Exeter  
Exeter, United Kingdom  
D.P.Fraser@exeter.ac.uk

Edward Keedwell  
University of Exeter  
Exeter, United Kingdom  
E.C.Keedwell@exeter.ac.uk

Stephen L. Michell  
University of Exeter  
Exeter, United Kingdom  
S.L.Michell@exeter.ac.uk

Ray Sheridan  
RD&E Hospital  
Exeter, United Kingdom  
ray.sheridan@nhs.net

## ABSTRACT

Clinical scorecards of risk factors associated with disease severity or mortality outcome are used by clinicians to make treatment decisions and optimize resources. This study develops an automated tool or framework based on evolutionary algorithms for the derivation of scorecards from clinical data. The techniques employed are based on the NSGA-II Multi-objective Optimization Genetic Algorithm (GA) which optimizes the Pareto-front of two clinically-relevant scorecard objectives, size and accuracy. Three automated methods are presented which improve on previous manually derived scorecards. The first is a hybrid algorithm which uses the GA for feature selection and a decision tree for scorecard generation. In the second, the GA generates the full scorecard. The third is an extended full scoring system in which the GA also generates the scorecard scores. In this system combinations of features and thresholds for each scorecard point are selected by the algorithm and the evolutionary process is used to discover near-optimal Pareto-fronts of scorecards for exploration by expert decision makers. This is shown to produce scorecards that improve upon a human derived example for C.Difficile, an important infection found globally in communities and hospitals, although the methods described are applicable to any disease where the required data is available.

## CCS CONCEPTS

• **Computing methodologies**~Genetic algorithms • **Mathematics of computing**~Mixed discrete-continuous optimization

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the Owner/Author

GECCO '19, July 13–17, 2019, Prague, Czech Republic

© 2019 Copyright is held by the owner/author(s).

ACM ISBN 978-1-4503-6111-8/19/07.

<https://doi.org/10.1145/3321707.3321802>

## KEYWORDS

Multi-objective optimisation, Evolutionary programming, Medicine, Prediction/forecasting.

### ACM Reference format:

Diane P. Fraser, Edward Keedwell, Stephen L. Michell, Ray Sheridan. 2019. EMOCS: Evolutionary Multi-objective Optimisation for Clinical Scorecard Generation. In *Proceedings of ACM GECCO conference, Prague, Czech Republic, July 2019 (GECCO '19)*, 8 pages. DOI: 10.1145/3321707.3321802

## 1 INTRODUCTION

Hospital admissions in the UK increased by 27.5% in the 10 years to March 2017 [1]. In an environment of limited resources, this puts ever greater pressure on clinical staff to understand disease severity and prioritise treatment for those cases that are at greatest risk of mortality. Clinical scorecards are an efficient method of assisting clinicians to stratify patients according to risk using evidence drawn from datasets of previous patient outcomes. A scorecard contains a (usually small) set of clinical measurements that are assigned a score, with higher scores usually indicating increased risk of mortality, or severity of disease. The most widely used scorecard is known as CURB-65. This is used to determine the risk posed to patients who are suffering from pneumonia and is defined as [2]:

- Confusion (Abbreviated Mental Test score  $\leq 8$ )
- Blood Urea Nitrogen  $> 7$  mmol/l (19 mg/dL)
- Respiratory rate  $\geq 30$  breaths per minute
- Blood pressure:  
Systolic BP  $< 90$  mmHg or Diastolic BP  $\leq 60$  mmHg
- Age  $\geq 65$

1 point is awarded for each of these five parameters and summed to provide an estimate of likelihood of mortality. The likelihood increases with the score from 0.6% at score 0 to 27.8% at score 5. By using readily available clinical measurements, this score allows for treatment to be prioritised to those most at risk.

To be effective, scorecards need to be accurate. They also need to be simple to implement by clinicians, to allow for accurate manual scoring in a high pressure clinical environment. As such, clinical scorecards should contain as few variables as possible for clinicians to measure and score, whilst providing the highest accuracy possible in terms of predicting mortality, to allow for differential allocation of limited treatment resources. In this paper we demonstrate the first use of multi-objective evolutionary algorithms to develop a Pareto-front of potential scorecards for experts to select from. The methodology could be applied to any appropriate disease with sufficient clinical data, however, the system is demonstrated here on real-world C.Difficile data as a case study.

## 1.1 C.Difficile

Butt et al [3] derived a clinical scorecard of risk factors associated with mortality outcome using manually applied univariate and multivariate analysis. The data used in that study was a set of real-world data from a UK hospital of 245 patients suffering from C.Difficile. The data consisted of 30 parameters that are routinely collected on admission to hospital when C.Difficile is suspected. ‘‘C.Diff’’ is an antibiotic exposure related form of gastroenteritis that is associated with a high morbidity and mortality. Caused by Clostridium Difficile, it can be difficult to assess severity clinically. We use the same data in this study to develop a method for the automatic derivation of clinical scorecards for C.Diff.

## 1.2 Scorecard Derivation from Data

The creation of scorecards is usually accomplished through the analysis of a dataset of patient variables (features) coupled with disease severity or mortality outcomes. The scorecard will then usually be created by a process of manual interrogation of the data, coupled with the use of various computational tools to determine feature importance within the dataset. A two-step process is required to determine a scorecard: firstly the identification of a set of relevant features from the data; and secondly the application of a scoring mechanism. The scorecard is then usually assessed for accuracy by calculating a receiver operating characteristic (ROC) curve with each scoring point evaluated for false positive and false negative rates. The area under the ROC curve (AUC) is the standard accuracy reporting measure.

Although no single established methodology for the elucidation of scorecards exists, a common approach is to rank the features in order of their correlation with the outcome and then add each feature into the scorecard until an acceptable accuracy level has been found. In [3], the approach used was based on decision tree feature importance, where the top features were discovered by an algorithmic approach. These features were then used to create a scorecard manually by exploring the various potential combinations of points scored with outcome accuracy on a training set. This was then tested on hold-out datasets and a validation cohort from another hospital. The 4-feature scorecard for C.Difficile which was derived using this method is shown in Table 1.

The scorecard shown in Table 1 gave an AUC of 0.754 on the training data set. Note that the last condition required both feature conditions to be true for a point to be awarded. This was found to give a higher AUC than if both were scored separately. A simpler scorecard without Respiratory Rate was also tested since the validation cohort data did not include this feature. This gave an AUC of 0.704 on the training data and an AUC of 0.653 on the validation data. Note that the Outcome used in [3] was ‘‘all mortality’’ and missing data scored 0. Here, the Outcome is ‘‘30-day mortality’’ and scorecards were not scored if any scorecard features were missing from the data. For this modified outcome with the training data set, the AUC obtained for the scorecard in Table 1 is 0.784.

**Table 1: The clinical scorecard devised in Butt et al [3]**

Measure	Condition	Score	
		True	False
Serum albumin levels (g/L)	$\leq 24.5$	1	0
C-reactive protein (mg/L)	$> 228$	1	0
Respiratory rate (resps/min)	$> 17$	1	0
and white cell count ( $10^3/\text{mcL}$ )	$> 12$	1	0

In this work, we have used a multi-objective optimisation algorithm to automate the process of scorecard derivation from data: EMOCS (Evolutionary Multi-Objective Clinical Scorecard) system. We compare this with the human-based approach described above. To our knowledge, this is the first time that an MOEA has been used in this way, generating Pareto-fronts of scorecard size vs accuracy (in this case measured as the AUC). This approach has several advantages over existing methods:

- The full trade-off of scorecard size vs accuracy can be discovered. Clinicians can investigate the trade-off between simplicity of use and the accuracy provided;
- Interactions between variables can be captured. The EMOCS method can investigate the combinatorial effect of combining features within a scorecard, in contrast to the greedy approaches usually used;
- More intricate scoring mechanisms can be explored. A single score of 1 for each feature is usually used in scorecard generation, whereas other scoring mechanisms can be automatically explored by the EMOCS approach;
- The system provides full automation of scorecard generation from a dataset and can be applied to any disease dataset that has clinical features and severity or mortality outcomes.

## 1.2 Previous Work

With improvements in computational and machine learning techniques over the past few years, there have been a number of studies that have used evolutionary algorithms for data mining clinical data. Examples include [4] who used an evolutionary algorithm to find feature subsets in clinical data. This is most closely related to the first of the approaches described below, although the authors in [4] did not go on to create a clinical scorecard. [5–7] used an evolutionary algorithm in combination

with other methods to perform classification on clinical data taken for a range of purposes. The key difference between these approaches and the proposed work is that the previous work creates methods for classifying patients into sufferers and non-sufferers whereas this work develops a particular set of features, combined into a scorecard designed to determine risk of mortality arising from the disease. Furthermore, this approach uses multi-objective optimization to explore the trade-off between scorecard size and accuracy. Although multi-objective approaches have been used for prediction (e.g. [8]), this represents a different use case to that described here. Therefore this is, to the best of our knowledge, the first time that evolutionary algorithms have been applied to the discovery of scorecards from clinical data.

The remainder of this paper is organized as follows: in Section 2 the EMOCS methodology is described; in Section 3 results are shown for a cohort of 245 C.Diff patients; in Section 4 conclusions are drawn and future research directions are explored.

## 2 METHODOLOGY

The application of evolutionary algorithms to this problem presents several possibilities of the integration of evolution and machine learning tools. In this work, three approaches have been developed, with the influence of evolution increasing from feature selector to the derivation of complete scorecard:

- **EMO as a Feature Selector:** In this approach the evolutionary algorithm acts as a feature selector ‘wrapper’ around a fast decision tree approach. Scores are allocated on a simple 1 point per feature basis;
- **EMO as a Scorecard Generator:** In this approach the EA is required to select the features, operators and thresholds for the scorecard (effectively replacing the decision tree). Scores are still allocated on a simple 1 point basis;
- **EMO as a Complete Scoring System:** As above, but scoring systems of greater complexity are permitted, allowing for multiple points to be awarded to more important features within a card.

The EMOCS system uses the well-known NSGA-II algorithm [9] as the basis for the work described here. This work was carried out on an i7-6600U Windows® 7 laptop and an i7-6700 ubuntu workstation within the R [10] statistical software environment. Although NSGA-II implementations are available which can take customised chromosome types, the R package ‘nsga2R’ [11] was used in this study, which requires real valued chromosomes. The encoding is described fully in Section 2.3. The following sections describe in more detail the objectives used, and the three approaches described above.

### 2.1 Objective Functions

All three algorithm variants use the same two objective calculations:

- **Area Under the ROC Curve (AUC) (Maximised):** This is a standard measure used in many fields including scorecard generation that measures a system’s ability to balance true and false positives for a number of decision points. In the case of

scorecards with a single point allocated per feature, the number of points on the ROC curve is equal to the number of features in the scorecard. If multiple points are allowed per feature then the number of points on the ROC curve increases accordingly. The true and false positive rates for each successive point are assessed and used to create the AUC curve;

- **Features Used (Minimised):** This is a simple cardinal measure of the number of features used to build the scorecard. This could be replaced by a measure of scorecard complexity, defined by clinicians (see future work).

### 2.2 EMO as Feature Selector

In this hybrid method, the chromosome only represents the feature selection part of scorecard generation. For a given set of features, a decision tree is used to determine operators and thresholds for each feature. The scorecard is then constructed and scored. This determines the objectives passed back to the nsga2R routine.

The chromosome is constructed as for the full chromosome, shown below in Figure 1, however, only the first ‘Variable’ part is encoded.

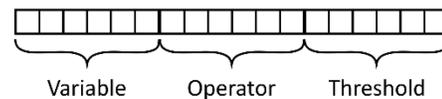
The decision tree used here is the ‘Fast and Frugal Tree’ R package (FFTrees) [12, 13]. The advantage of this package, apart from being fast, is that it is frugal. That is, it produces decision trees which have a single leaf and further decision at each node. A further advantage of this routine is that it can generate a number of trees with a range of sensitivity and specificity. The tree used to generate the scorecard is that with the greatest balanced or weighted accuracy, as determined by the FFTrees routine. Parameters can be set, however, to modify how this is decided. Here, the default is used for which both balanced and weighted accuracy are equal to the average of the sensitivity and specificity.

The FFTrees algorithm returns the collection of decisions which make up the chosen tree. Although the decision tree applies the conditions in a defined order, all of the conditions are used in the scorecard with equal weight. 1 point is assigned to ‘Score True’ and 0 to ‘Score False’.

### 2.3 EMO as Scorecard Generator

The description here is for the chromosome used in the scorecard generator and the full scoring system. The methodology is as follows. See also, Figure 1 and Table 2.

The chromosome represents feature selection, operators and thresholds. The scorecard is constructed from the chromosome with a score of 1 per feature allocated to ‘Score True’. ‘Score False’ is always set to 0. This scorecard is then scored to determine the objectives passed back to the nsga2R routine.



**Figure 1: The full chromosome split into three for the different parts of the scorecard entries.**

**Table 2: Encoding the three parts of the full chromosome for the scorecard generator system (Score = 1) and the full scoring system (Max Score > 1)**

Parameter	Real Range	Mapped Range	Interpretation
Variable			
(Score = 1)	[0, 2)	0, 1	Not Selected/Selected
(Max Score > 1)	[0, (Max Score + 1) )	0, 1, ... Max Score	Not Selected/Score
Operator	[1, 5)	1, 2, 3, 4	'<', '>', '=', '!='
Threshold			
Real	Extent of Data	Extent of Data	Real Value
Integer	Extent of Data	Extent of Data	Integer Value
Factor	[1, (No. of Levels + 1) )	1, 2, ... No. of Levels	Category
Logical	[0, 2)	0, 1	False/True

The chromosome is split into three equal sections. The first section encodes the feature selection and score. The middle section encodes the operator, and the final section encodes the feature threshold values.

A lower and upper bound is provided for each element of the chromosome, as indicated by the “Real Range” in Table 2. Note that many of the intervals are half open. In order to exclude the upper bound from parameters which map to integers, the upper bounds are the required integer upper bounds, less a real tolerance. This tolerance is given by  $\sqrt{mtol}$  where  $mtol$  is the smallest positive floating-point number  $x$  such that  $1 + x \neq 1$ .

The *nsga2R* routine generates a population of initial chromosomes with random values. Subsequent generations of chromosomes are generated by the routine from the previous generation using a series of cross-overs and mutations. In this study a population of 20 was used with standard values of 0.1 for the a mutation probability and 0.7 for the crossover probability. For each generation, scorecards are generated from the population of chromosomes and the objective functions are calculated. These objectives are used by *nsga2R* to determine which chromosomes should be kept in the population, and which cross-overs and mutations to be used. The first objective  $Obj_1 = 1 - auc$  maximises the AUC. The second objective  $Obj_2 = \alpha n_v + \beta$  minimises the number of variables  $n_v$ .  $\alpha = 0.1$  and  $\beta = -0.1$  are chosen to map  $Obj_2$  to a similar range as  $Obj_1$ .

## 2.4 EMO as Complete Scoring System

The chromosome represents feature selection, operators and thresholds as described above. Now, the feature selection includes the score to be allocated to that feature by setting a maximum score. The scorecard is constructed from the chromosome with the score allocated to “Score True”. “Score False” is always set to 0. The scorecard is then scored and the objectives passed back to the *nsga2R* routine.

## 2.5 Validation

To investigate the full potential of the methods, most of the experiments were performed using the full data set as a training set. However, to investigate the possibility of over fitting the data, a set

of scorecard generation experiments was also performed using 5-fold cross validation. Note that due to the nature of the data, this implied that each test data set would have only 49 cases with 8 positive outcomes.

## 3 RESULTS

### 3.1 Experiments

A series of experiments were performed with different random seeds for each of the three methods outlined in Section 2. 30 runs were performed for both the hybrid method and the scorecard generation system; 20 runs were performed for the more complex full scoring system due to increased computational complexity. Each run resulted in up to 20 unique scorecards, typically 6-9 for the hybrid method and 9-12 for both the scorecard and full scoring systems. Run times using a single core of an i7-6700 were typically 50-70 minutes, 1½-5 hours and 5-7 hours, respectively. The hypervolume of the Pareto-front increased with the number of generations. Example Pareto-fronts are given in Figures 2 a)-c) for each method. Also indicated is the AUC for the previous scorecard given in Table 1, shown as a point on these plots.

The results in Figure 2 show that the automated method is capable of discovering scorecards that dominate the human-derived scorecard shown by the point. The plots also reveal the expected relationship between the number of variables and the AUC for each scorecard. For each curve, there is no highly defined ‘knee’ but a scorecard of 4-6 points appears to provide a large degree of the available accuracy. The ability to use a differential scoring system appears to have some benefit, with higher AUCs possible for larger numbers of variables. However, there appears to be little difference in results between the hybrid method and full scorecard generator here, though the hybrid method may have computational advantages as described below.

Figure 3 shows the mean of the normalised hypervolume against the number of generations for each series of 10 experiments. Here the hypervolume reference point is taken at  $auc=1$  and number of features, or variables,  $nv=14$ . The shading gives the standard deviation of the experiments for that method. It is seen that within the experiments for a given method the variation in hypervolume is

lowest for the hybrid method and greatest for the full scoring system. For a given set of features, the decision tree will always return the same set of conditions. So although the method explores the parameter space of the feature selection, no exploration is made of the condition operators or thresholds. In contrast, the two full scorecard methods explore the full scorecard parameter space. This exploration comes at a cost in time. The hybrid method is roughly six times faster than the full scoring system, with the scorecard generator method roughly 2-3 times faster.

Comparing Figure 3 b) and c) the standard deviations show that allowing scores to vary in the full scoring system increases variations in the results obtained and tends to slow down convergence. This does, however, allow the method to produce scorecards with higher AUC values. The 4-feature scorecards with the highest AUC values obtained from the experiments presented here, are given in Table 3 for each method.

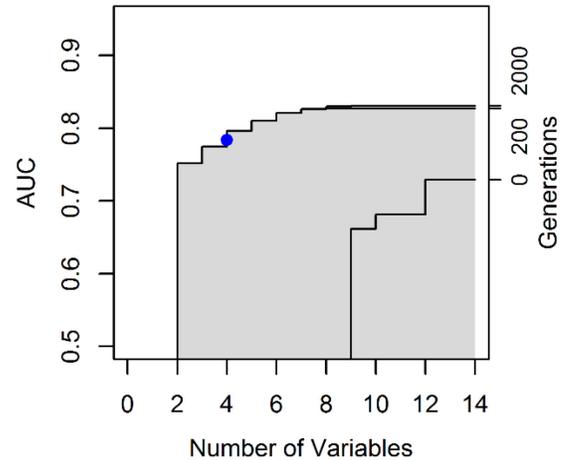
The first feature in each scorecard is the same as for the previous scorecard in Table 1, though the conditions vary. Both the hybrid method and full scorecard system have two further features in common with the scorecard in Table 1, and the full scoring system has one further feature the same. In all cases, though, the conditions vary. The remaining two features from these methods, and the fourth feature from the hybrid method differ from Table 1.

The full scoring system allocated the highest score of 3 points for “Score True” to the feature condition which is common to all scorecards. The next two feature conditions were given scores of 2 points, and the last feature, which does not appear in any of the other scorecards, a score of 1 point. These scores reflect a measure of importance which can be assigned to the features concerned.

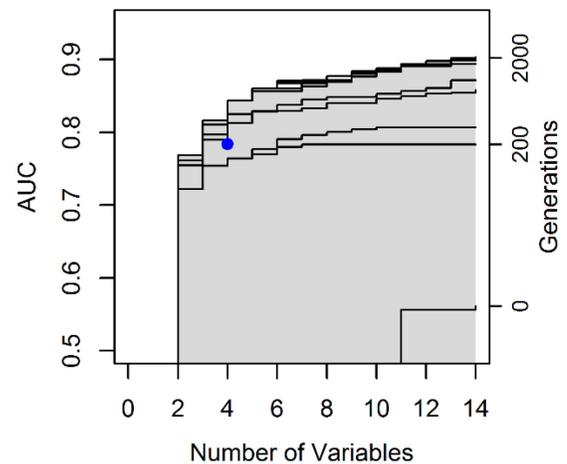
The variations in the conditions for the same features reflects the exploration of parameter space which the methods employ. It also reflects the granularity of the data. It is striking that each of the scorecards use a number of similar features, although the thresholds used do vary significantly. The best scorecards derived by the evolutionary methods significantly exceed the performance of the human-derived scorecard on this data and increasing evolutionary ‘freedom’ appears to improve the potential for this method to find scorecards with higher accuracy, although this may also result in greater overfitting (e.g. see Figure 5).

The ROC curves corresponding to the scorecards in Table 3 are given in Figure 4. The increased number of points in Figure 4 c) compared with b) confirms that the full scoring system allows a greater choice of decision thresholds. Also shown in Figure 4 is the ROC curve for the scorecard from Table 1 for comparison. For each method presented here, the improvement in AUC is seen to result mainly from an improvement in true positive rates at mid to high false positive rates.

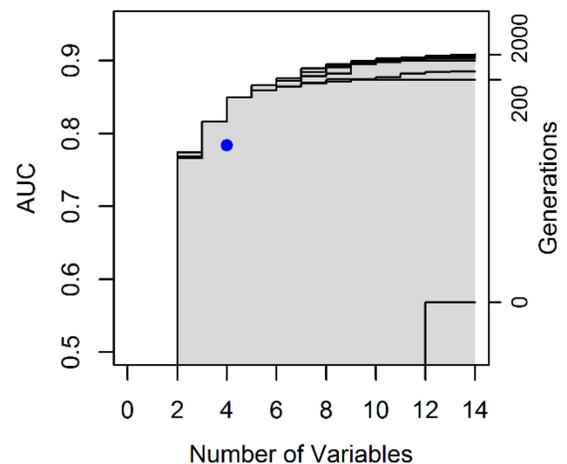
Mean values for the AUC for each method are given in Table 4 along with the standard deviations. The scorecard generator exceeds the performance of the hybrid system for all sizes of scorecard, and the full scoring system for scorecards with only two or three features. The full scoring system is marginally better than the scorecard generator for scorecards with 4 to 9 features, though the difference is well within the standard deviations of the samples.



a)

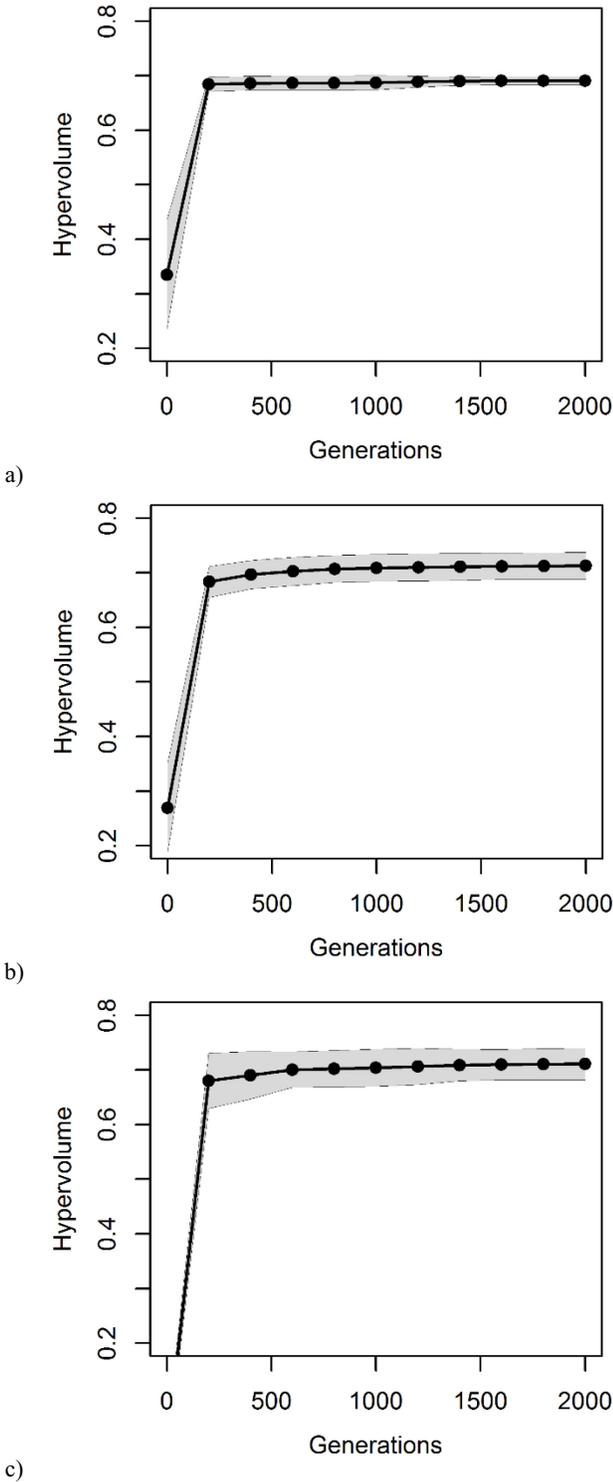


b)

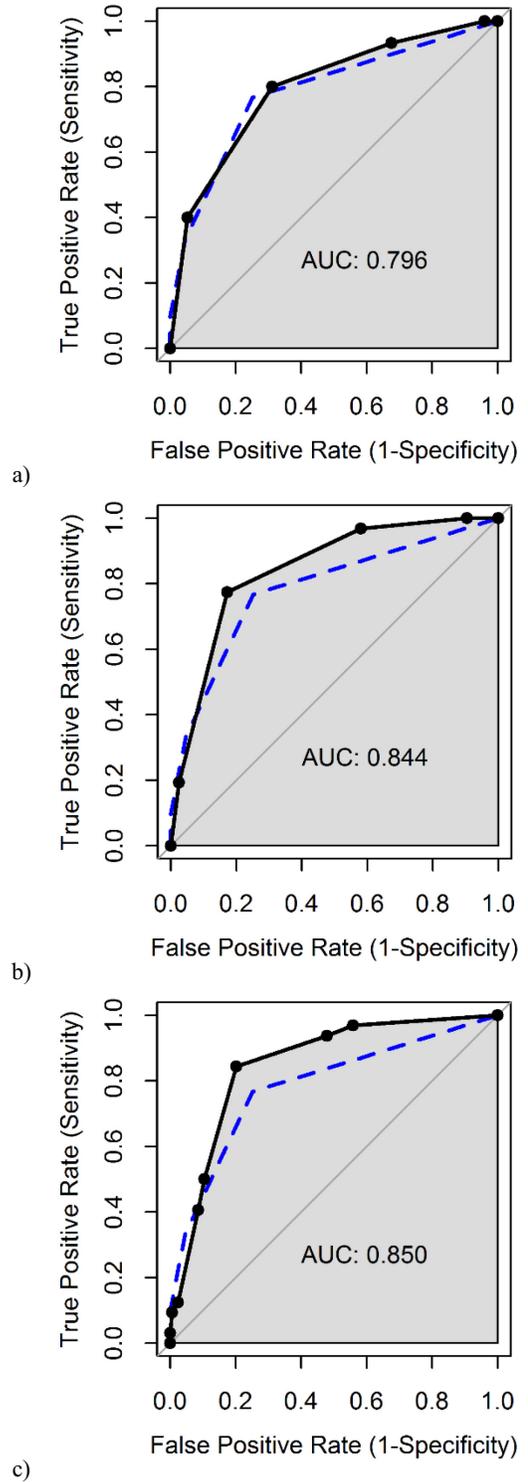


c)

**Figure 2. Example Pareto-fronts for generations from 0 to 2000 at intervals of 200. The lowest curves are for the random start points. The solid point shown in blue is for the scorecard of Table 1. The methods are: a) The hybrid method; b) the full scorecard generator; c) the full scoring system.**



**Figure 3.** The mean hypervolume for each generation at intervals of 200. The shading gives the standard deviation across the total runs for each method. The methods a)-c) are as for Figure 2.



**Figure 4.** ROC curves for the 4-feature clinical scorecards given in Table 3 (solid line) and Table 1 (dashed line). The methods a)-c) are as for Table 3.

**Table 3: The 4-feature clinical scorecards with highest AUC****a) For the hybrid method: AUC=0.796**

Measure	Condition	Score	
		True	False
Serum albumin levels (g/L)	< 30	1	0
C-reactive protein (mg/L)	≥ 90	1	0
Respiratory rate (resps/min)	≥ 16	1	0
Creatinine baseline (mg/DL)	> 66	1	0

**b) For the full scorecard generator: AUC=0.844**

Measure	Condition	Score	
		True	False
Serum albumin levels (g/L)	< 31	1	0
Respiratory rate (resps/min)	> 14	1	0
White cell count (10 <sup>3</sup> /mcL)	> 16	1	0
Diastolic BP (mmHg)	< 61	1	0

**c) For the full scoring system: AUC=0.850**

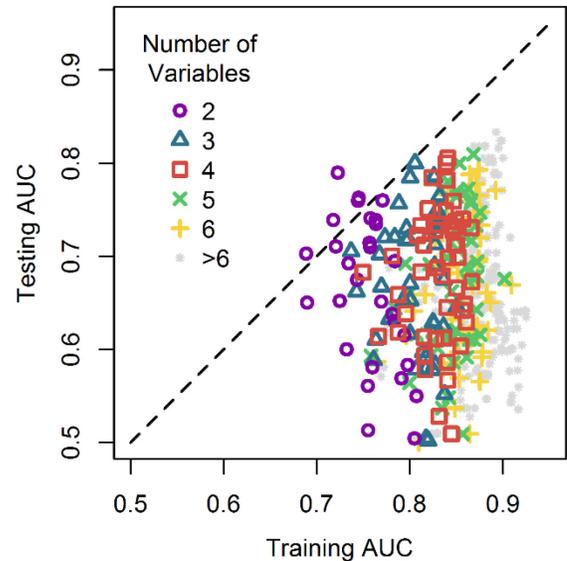
Measure	Condition	Score	
		True	False
Serum albumin levels (g/L)	< 25	3	0
White cell count (10 <sup>3</sup> /mcL)	> 16	2	0
Diastolic BP (mmHg)	< 61	2	0
Age (years)	> 86	1	0

### 3.2 Validation

A series of 10 experiments were performed with 5-fold cross validation for the scorecard generation system described in Section 2.3. Up to 20 unique scorecards (typically 9-12) were generated for each fold within each experiment. Run times using a single core of an i7-6700 were typically 1½-4½ hours per fold. Figure 5 shows Testing AUC against Training AUC for all of the resulting scorecards. The number of features, or variables, on the given scorecard is indicated by the symbol and colour of each point. Mean values for the AUC for the training and testing datasets, averaged over all validation runs, are given in Table 5 along with the standard deviations.

Some reduced performance is expected for the test data, especially given the small size of the test data sets. Figure 5 and Table 5 show that the greatest decline in performance occurs for scorecards with the greater number of features. This is consistent with the greater number of features over-fitting the training data. In contrast, many of the scorecards with smaller numbers of features show less of a decline in performance, or even improved performance, when tested against the test data sets. It is noticeable that the peak in the mean AUC for the test data occurs for scorecards with 4 features, followed closely by sizes of 5 and 3.

This lends support to the accuracy of scorecards with smaller or moderate numbers of features, as discussed in Section 3.1. Scorecards of this size are also more suitable for clinical use.



**Figure 5. AUC for the test data sets against AUC for the training data sets for 5-fold cross validation. The number of variables indicates the number of scorecard features. The dotted line is a guide to the eye and indicates the identity Testing AUC = Training AUC.**

## 4 CONCLUSIONS AND FURTHER WORK

All three methodologies presented here automatically produce viable scorecards from clinical data. Although the hybrid feature selection method explores a full range of features, it relies upon decision trees, a greedy method, to provide the test conditions for the scorecards. The scorecards produced consistently improve upon the manually derived scorecard. Further, clinical decision makers should be able to choose scorecards which give improved true positive rates without compromising false positive rates.

The full scorecard method explores both the full range of features and the test conditions. The best AUC values for a given number of features improve upon the hybrid method, though there is a much greater variation within the results obtained. Similarly, the full scoring system is able to give improved results for most sizes of scorecard, though again with significant variation within the results obtained.

The full scoring system shows a further improvement in performance through the use of differential scores for each feature. This increase in AUC could be useful, although the number of points awarded per feature adds an additional level of complexity and therefore may not be applicable in all clinical situations.

The automated systems presented here could be used for generating clinical scorecards from any suitable clinical data and

**Table 4: Mean AUC and standard deviation for the given number of variables (up to 10) for the three methods**

a) Hybrid Method				b) Scorecard Generator			c) Full Scoring System		
Number of Variables	Number of Scorecards	AUC Mean	AUC Std Dev	Number of Scorecards	AUC Mean	AUC Std Dev	Number of Scorecards	AUC Mean	AUC Std Dev
2	30	0.7436	0.0247	31	0.7538	0.0175	23	0.7328	0.0600
3	30	0.7716	0.0115	32	0.7931	0.0246	20	0.7797	0.0437
4	30	0.7928	0.0084	32	0.8090	0.0306	22	0.8103	0.0365
5	30	0.8052	0.0074	33	0.8255	0.0280	22	0.8278	0.0318
6	29	0.8149	0.0079	30	0.8346	0.0286	22	0.8404	0.0316
7	27	0.8189	0.0085	32	0.8444	0.0275	21	0.8475	0.0316
8	20	0.8221	0.0085	30	0.8488	0.0288	20	0.8523	0.0318
9	19	0.8229	0.0086	29	0.8549	0.0288	17	0.8597	0.0305
10	7	0.8300	0.0038	22	0.8602	0.0334	18	0.8569	0.0334

**Table 5: Mean AUC and standard deviation for the given number of variables (up to 10) for the validation runs**

a) Training data set				b) Testing data set		
Number of Variables	Number of Scorecards	AUC Mean	AUC Std Dev	Number of Scorecards	AUC Mean	AUC Std Dev
2	50	0.7674	0.0289	50	0.6541	0.0800
3	56	0.8096	0.0273	56	0.6753	0.0723
4	53	0.8315	0.0251	53	0.6834	0.0697
5	59	0.8465	0.0263	59	0.6777	0.0721
6	60	0.8532	0.0271	60	0.6619	0.0730
7	55	0.8617	0.0273	55	0.6675	0.0818
8	50	0.8667	0.0290	50	0.6618	0.0746
9	46	0.8686	0.0296	46	0.6653	0.0746
10	35	0.8744	0.0306	35	0.6699	0.0805

represent the first such automated systems to the best of our knowledge. Future work will look at several areas. A simple extension will allow for interactions between features and incorporate a more realistic costing mechanism rather than simply taking the number of features as an objective. This will allow for a balance between the diagnostic power of the features and the cost in terms of simplicity of use, time and resources. Changes to the R code which applies the NSGA-II algorithm to the scorecard systems will be investigated in order to improve performance. In particular, the code could be developed to make better use of multi-threading on multi-core systems. Further to work on the methodology itself, ways to present the results to the clinical practitioner (e.g. via an automated system) will also be considered.

**ACKNOWLEDGMENTS**

This work was funded by a seedcorn grant from the EPSRC Centre for Predictive Modelling in Healthcare (EP/N014391/1).

**REFERENCES**

[1] NHS Digital. 2018. Hospital Admitted Patient Care Activity, 2016-17. Retrieved from <http://digital.nhs.uk/catalogue/PUB30098>

[2] BTS Guidelines for the Management of Community Acquired Pneumonia in Adults. 2001. *Thorax*, 56, suppl 4 (2001), iv1-iv64. DOI: [https://doi.org/10.1136/thx.56.suppl\\_4.iv1](https://doi.org/10.1136/thx.56.suppl_4.iv1)

[3] Emma Butt, Jane A.H. Foster, Edward Keedwell, Julia E.A. Bell, Richard W. Titball, Aneel Bhangu, Stephen L. Michell and Ray Sheridan. 2013. Derivation and validation of a simple, accurate and robust prediction rule for risk of mortality in patients with Clostridium difficile infection. *BMC Infectious Diseases*, 13, 1 (July 12 2013), 316. DOI: <https://doi.org/10.1186/1471-2334-13-316>

[4] Mengmeng Li, M., Zhigang Shang and Caitong Yue. 2017. A Feature Subset Evaluation Method Based on Multi-objective Optimization. In *Simulated Evolution and Learning, SEAL 2017. LNCS. Lecture Notes in Computer Science*, vol 10593. Springer, Cham. DOI: [https://doi.org/10.1007/978-3-319-68759-9\\_47](https://doi.org/10.1007/978-3-319-68759-9_47)

[5] Kumar, S. and Sahoo, G. 2015. Classification of Heart Disease Using Naïve Bayes and Genetic Algorithm. In *Computational Intelligence in Data Mining - Volume 2. Smart Innovation, Systems and Technologies*, vol 32. Springer, New Delhi. DOI: [https://doi.org/10.1007/978-81-322-2208-8\\_25](https://doi.org/10.1007/978-81-322-2208-8_25)

[6] Xiao Liu, Xiaoli Wang, Qiang Su, Mo Zhang, Yanhong Zhu, Qiugen Wang, and Qian Wang. 2017. A Hybrid Classification System for Heart Disease Diagnosis Based on the RFRS Method. *Computational and Mathematical Methods in Medicine*, vol. 2017. DOI: <https://doi.org/10.1155/2017/8272091>

[7] Shabia Shabir Khan, S.M.K. Quadri and M.A. Peer. 2016. Genetic Algorithm for Biomarker Search Problem and Class Prediction. *International Journal of Intelligent Systems and Applications (IJISA)*, 8, 9 (2016), 47-55. DOI: <https://doi.org/10.5815/ijisa.2016.09.06>

- [8] Marta Vallejo, Jeremy Cosgrove, Jane E. Alty, Stuart Jamieson, Stephen L. Smith, David W. Corne, and Michael A. Lones. 2016. A Multi-Objective Approach to Predicting Motor and Cognitive Deficit in Parkinson's Disease Patients. In *Proceedings of Genetic and Evolutionary Computation Conference Companion Denver, Colorado, USA, 2016 (GECCO '18)*, 8 pages. DOI: <https://doi.org/10.1145/2908961.2931731>
- [9] K. Deb, A. Pratap, S. Agarwal and T. Meyarivan. 2002 A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6, 2 (2002), 182-197. DOI: <https://doi.org/10.1109/4235.996017>
- [10] R Core Team. 2018. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria. Retrieved from <https://www.R-project.org/>
- [11] nsga2R: Elitist Non-dominated Sorting Genetic Algorithm based on R. Retrieved from <https://cran.r-project.org/package=nsga2R>
- [12] Nathaniel D. Phillips, Hansjörg Neth, Jan K. Woike and Wolfgang Gaissmaier. 2017. FFTrees: A toolbox to create, visualize, and evaluate fast-and-frugal decision trees. *Judgment and Decision Making*, 12(4), 344-368
- [13] FFTrees: Generate, Visualise, and Evaluate Fast-and-Frugal Decision Trees. Retrieved from <https://cran.r-project.org/package=FFTrees>