ELSEVIER

Artificial Intelligence in Medicine 9 (1997) 107–138

Artificial Intelligence in Medicine

# An evaluation of machine-learning methods for predicting pneumonia mortality

Gregory F. Cooper[a,*], Constantin F. Aliferis[a], Richard Ambrosino[a], John Aronis[b], Bruce G. Buchanan[b], Richard Caruana[c], Michael J. Fine[d], Clark Glymour[e], Geoffrey Gordon[c], Barbara H. Hanusa[d], Janine E. Janosky[f], Christopher Meek[e], Tom Mitchell[c], Thomas Richardson[e], Peter Spirtes[e]

[a]*Center for Biomedical Informatics, Suite 8084 Forbes Tower, 200 Lothrop Street, University of Pittsburgh, Pittsburgh, PA 15261, USA*
[b]*Intelligent Systems Laboratory, Department of Computer Science, University of Pittsburgh, Pittsburgh, PA 15213, USA*
[c]*School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, USA*
[d]*Division of General Internal Medicine, Department of Medicine, University of Pittsburgh, Pittsburgh, PA 15213, USA*
[e]*Department of Philosophy, Carnegie Mellon University, Pittsburgh, PA 15213, USA*
[f]*Division of Biostatistics, Department of Family Medicine and Clinical Epidemiology, University of Pittsburgh, Pittsburgh, PA 15261, USA*

Accepted 14 October 1996

## Abstract

This paper describes the application of eight statistical and machine-learning methods to derive computer models for predicting mortality of hospital patients with pneumonia from their findings at initial presentation. The eight models were each constructed based on 9847 patient cases and they were each evaluated on 4352 additional cases. The primary evaluation metric was the error in predicted survival as a function of the fraction of patients predicted to survive. This metric is useful in assessing a model's potential to assist a clinician in deciding whether to treat a given patient in the hospital or at home. We examined the error

* Corresponding author. E-mail: gfc@cbmi.upmc.edu

0933-3657/97/$17.00 Copyright © 1997 Elsevier Science B.V. All rights reserved
*PII* S0933-3657(96)00367-3

rates of the models when predicting that a given fraction of patients will survive. We examined survival fractions between 0.1 and 0.6. Over this range, each model's predictive error rate was within 1% of the error rate of every other model. When predicting that approximately 30% of the patients will survive, all the models have an error rate of less than 1.5%. The models are distinguished more by the number of variables and parameters that they contain than by their error rates; these differences suggest which models may be the most amenable to future implementation as paper-based guidelines.

*Keywords:* Clinical databases; Computer-based prediction; Machine learning; Pneumonia

## 1. Introduction

The construction of computer decision-support systems from patient databases has longed played an important role in medical informatics. For example, starting in the early 1960s, researchers investigated the construction from data of Bayesian diagnostic systems that assume conditional independence of findings given a disease state [14,25]. As clinical information is stored increasingly in computer databases, the opportunities expand for using this information to help improve patient care and reduce health-care costs. Often, standard statistical methods, such as logistic regression, are used currently to construct predictive models in medicine. In the last decade, however, researchers in artificial intelligence have developed new machine-learning methods that construct computer models from data. In the current journal, for instance, numerous articles within the past few years have appeared discussing unsupervised [10,19] as well as supervised [8,16,26,27] machine-learning research in medicine. Such methods have the potential to complement and augment existing statistical techniques. To understand the degree to which this potential can be realized in medicine, it is important to compare the performance of statistical methods and machine-learning methods on real clinical tasks using real clinical data. This paper reports one such investigation in the construction and application of computer models to predict patient mortality. The particular machine-learning methods investigated are neural-network learning, a rule-learning technique, two causal discovery methods, a simple Bayesian classifier, and a generalized decision-tree induction method. Two statistical techniques, namely, logistic regression and a K-nearest neighbor method, are investigated as well.

In the study reported here, we focus on predicting patient mortality in the area of community-acquired pneumonia. Pneumonia is an important disease that affects over 3 million people annually in the US [23]. It is the sixth leading cause of death in this country [24]. In 1987 it was responsible for over 900 000 hospital admissions [23], and the resulting health-care costs that year were more than 3 billion dollars.

In the current study, we concentrate on the task of predicting mortality of hospitalized patients from their findings at initial presentation with pneumonia. Such predictions may be useful to clinicians in making decisions about where to treat patients with pneumonia. We want to identify as large a group of patients as possible who can be treated safely at home for pneumonia, because (1) this strategy is likely to reduce the costs of treating pneumonia, and (2) it will allow patients with mild cases of pneumonia to be treated at home, where typically they would be more comfortable.

## 2. The study database

We used the 1989 MedisGroups Comparative Hospital Database (MCHD), which contains information on inpatients discharged from 78 hospitals in 23 states in the US between July 1987 and December 1988. These data were made available to the project by MediQual. The MCHD was collected by MediQual primarily to derive hospital mortality rates that are adjusted for severity of illness; the adjustment is based on MediQual's proprietary MedisGroups classification system. The MCHD contains over 250 pieces of clinical information, called *key clinical findings* (KCFs), which include patient demographic characteristics, history and physical examination findings, and laboratory and radiological results. The KCFs are collected during three review periods during the hospitalization of a patient; we only used patient data that were collected on the admission reviews, which is defined as the most abnormal findings during the first 48 h of hospitalization. MediQual employs site-specific, trained abstractors to encode for each hospital patient all the above information, based on a review of the patient's medical record. In addition to the KCFs, MediQual records the principal and secondary diagnoses for all patients.

We used the following inclusion criteria to define patients with community-acquired pneumonia from the MCHD: (1) an ICD-9-CM principal diagnosis of pneumonia was present at admission, (2) patient age 18 years or more, and (3) admission from home or a nursing home [9]. We excluded patients with a history of AIDS or a known positive HIV antibody titer. Patients with HIV-related illness were excluded since their pneumonia etiology and outcomes are different than patients without this underlying illness. We also excluded patients who had been in the hospital within 7 days of the current admission or who had been transferred from another acute care hospital, because we wanted to study only primary admissions for community-acquired pneumonia. In total, 14 199 patients from the MCHD met all the eligibility criteria and composed the study cohort.

Among the more than 250 variables in the MCHD, clinical experts (including author MJF) selected 46 variables with either known or postulated association with mortality in patients with community-acquired pneumonia. Table 1 shows the variables that were included. Of these 46 variables, 17 have continuous (real) values. For each continuous variable (e.g. sodium level) the clinical experts defined discrete categories of variable values according to clinically meaningful ranges of

Table 1
The variables that appear in the study database and their range of values

| Variable | Categories of values[a] |
|---|---|
| *Patient-history findings* | |
| Age (years) | 18–106 |
| Gender | male, female |
| A re-admission to the hospital | no, yes |
| Admitted from a nursing home | no, yes |
| Admitted through the ER | no, yes |
| Has a chronic lung disease | no*, yes |
| Has asthma | no*, yes |
| Has diabetes mellitus | no*, yes |
| Has congestive heart failure | no*, yes |
| Has ischemic heart disease | no*, yes |
| Has cerebrovascular disease | no*, yes |
| Has chronic liver disease | no*, yes |
| Has chronic renal failure | no*, yes |
| Has history of seizures | no*, yes |
| Has cancer | no*, yes |
| Number of above disease conditions | an integer from 0* to 10 |
| Pleuritic of chest pain | no*, yes |
| *Physical examination findings* | |
| Respiration rate (resps/min) | ≤29*, ≥30 |
| Heart rate (beats/min) | ≤124*, 125–150, ≥151 |
| Systolic blood pressure (mmHg) | ≤60, 61–70, 71–80, 81–90, ≥91* |
| Temperature (°C) | ≤34.4, 34.5–34.9, 35–35.5, 35.6–38.3*, 38.4–39.9, ≥40 |
| Altered mental status (disorientation, lethargy, or coma) | no*, yes |
| Wheezing | no*, yes |
| Stridor | no*, yes |
| Heart murmur | no*, yes |
| Gastrointestinal bleeding | no*, yes |
| *Laboratory findings* | |
| Sodium level (mEq/l) | ≤124, 125–130, 131–149*, ≥150 |
| Potassium level (mEq/l) | ≤5.2*, ≥5.3 |
| Creatinine level (mg/dl) | ≤1.6*, 1.7–3.0, 3.1–9.9, ≥10.0 |
| Glucose level (mg/dl) | ≤249*, 250–299, 300–399, ≥400 |
| BUN level (mg/dl) | ≤29*, 30 to 49, ≥50 |
| Liver function tests (coded only as normal* or abnormal) | SGOT≤63 and alkaline phosphatase≤499*, SGOT>63 or alkaline phosphatase>499 |
| Albumin level (gm/dl) | ≤2.5, 2.6–3, ≥3.1* |
| Hematocrit | 6–20, 20.1–24.9, 25–29, ≥30* |
| White blood cell count (1000 cells/$\mu$l) | 0.1–3, 3.1–19.9*, ≥20 |
| Percentage bands | ≤10*, 11–20, 21–30, 31–50, ≥51 |
| Blood pH | ≤7.20, 7.21–7.35, 7.36–7.45*, ≥7.46 |
| Blood $pO_2$ (mmHg) | ≤59, 60–70, 71–75, ≥76* |
| Blood $pCO_2$ (mmHg) | ≤44*, 45–55, 56–64, ≥65 |

Table 1 (continued)

| Variable | Categories of values[a] |
|---|---|
| *Chest X-ray findings* | |
| Positive chest X-ray | no*, yes |
| Lung infiltrate | no*, yes |
| Pleural effusion | no*, yes |
| Pneumothorax | no*, yes |
| Cavitation/empyema | no*, yes |
| Lobe or lung collapse | no*, yes |
| Chest mass | no*, yes |

[a] The MediQual database makes no distinction between values that fall within a specified 'normal range', as indicated by an asterisk in this column, and values that are not mentioned in the medical record. Both are recorded at zero. Consequently, in the study database, there is no way to distinguish whether a variable with a value of zero has a normal value or a missing value.

the variable. For each continuous variable, we recorded it both as a continuous and as a categorical variable. Thus, the total variables recorded per patient case is $46 + 17 = 63$.

Each patient case also contains a value for a variable that we call *vital_status*, which indicates whether the patient died during hospitalization. In particular, *vital_status* was given the value *alive* if a patient was either (1) discharged or transferred from the hospital, or (2) alive in the hospital more than 60 days after hospitalization. If a patient died during the first 60 days of hospitalization, then *vital_status* was given the value *expired.* In the remainder of this paper, we will use the term *study database* to refer to the database of 14 199 cases that contains values for the 63 finding variables and the variable *vital_status.*

## 3. Methods

We used a randomly selected 70% of the study database to create a training set of 9847 cases, and we used the remaining 30% of cases as a test set of 4352 cases. Based on the data in the training set, we used each of the six machine-learning methods listed in Section 1, as well as logistic regression and a K-nearest neighbor method, to construct predictive models. For each case in the test set, a patient's findings on admission with pneumonia were given to a model and the model predicted the chance (e.g. a probability) that the patient would survive (or conversely, die) during hospitalization. By our assigning a model-specific threshold to use for predicting survival, each model will predict a set of survivors. (As described in Section 3.3, one of the models predicts survival directly, without using such thresholds in making predictions.) We also examined the predictive accuracy of several hybrid models, which are described in Section 3.9.

Using test data, we evaluated the predictive performance of each of the predictive models constructed from the training data. The primary evaluation metric we applied was the fraction of patients predicted to survive who died (fpsd) as a function of the fraction of patients predicted to survive (fps). We emphasize that fps denotes the fraction of patients predicted to survive by a given model, rather than the fraction of patients who do in fact survive. We applied each model to the test set in order to plot values for fpsd as a function of values for fps. We examined the extent to which the models have different fpsd error rates.

The fpsd versus fps metric is useful in assessing a model's potential to assist a clinician in deciding where to treat a given patient. Suppose, for instance, that we have a computer model that predicts a sizable percentage of patients with pneumonia will live and almost all of them do live. Such a model could be helpful in suggesting which patients are likely to do well when treated at home, rather than in a hospital[1]. In other words, if a predictive model has an fpsd value that is sufficiently low, then the model's prediction of survival might be used as one factor in assessing, for example, which patients are expected to do well clinically if treated at home. The fps level that corresponds to such a low fpsd value would indicate the fraction of all pneumonia patients the model predicts will survive, and thus, could be safely treated at home. The accuracy of such predictions would need to be carefully validated, preferably through randomized trials, but at least through systematic study of the performance of any implementation.

As a partial measure of model complexity, we tabulated the number of variables and parameters in each model. We also investigated the convergence rate of one of the models as the amount of training data was varied. As an indication of the variance of the models in predicting the vital status outcomes of difficult patient cases, we examined the extent to which the models made errors on the same cases. In addition, we examined four hybrid predictive models.

The remainder of this section contains (1) a brief summary of each of the eight methods that we used to infer predictive computer models from the training set of 9847 patient cases, (2) details regarding how we applied each method to the test set of 4352 cases, and (3) a description of the statistical methods we used to compare the models.

### 3.1. A summary of the logistic regression method

Logistic regression derives an equation of the following form:

$$P(V|X_1, X_2,\ldots, X_n) = \frac{e^{(\beta_0+\beta_1X_1+\beta_2X_2+\cdots+\beta_nX_n)}}{1+e^{(\beta_0+\beta_1X_1+\beta_2X_2+\cdots+\beta_nX_n)}}$$

where $V$ is a binary variable to be predicted, and $X_1, X_2,\ldots, X_n$ are discrete or continuous predictor variables [1]. The constants $\beta_0, \beta_1, \beta_2,\ldots, \beta_n$ are estimated from the training data, typically by using an iterative maximum likelihood technique. We can generalize the above equation by including interaction terms in the sum, which are composed of a product of the $X_i$ variables.

[1] We emphasize that other factors also generally would need to be considered, such as (1) expected patient morbidity, (2) expected time to recovery, (3) the patient's home environment (e.g. whether there are family members that could assist the patient) and (4) the patient's preferences.

### 3.2. The application of the logistic regression method

The logistic regression (LR) model we used was originally constructed and evaluated in the study described in [9]. For additional details about the construction of the model and its complete specification, see that paper.

Each of the variables in Table 1 was tested for its ability to predict *vital_status*. For each discrete variable, we used a chi-square test. For a continuous variable, we used a Student's *t*-test on its continuous range and a chi-squared test on its discretized range, as defined in Table 1. These tests were applied to the entire database of 14 199 patient cases using a two-tailed significance level of 0.05[2]. These variables defined the set of significant univariate predictors. Adjacent value categories of the univariate predictor variables with three or more categories were combined if the odds ratios of these categories were not significantly different from each other at the 0.1 level using a likelihood ratio statistic.

Given the univariate predictors and their categories, a logistic regression model was developed as follows from the training data. The BMDP implementation of logistic regression [1] was applied. A forward stepping search procedure was applied to the test data using the set of univariate predictors to identify significant (at the 0.05 level) multivariate predictors of *vital_status*, which we denote as *MP*. We added to the model all two way interactions between the multivariate predictors and the variables *age* and *gender*, which we denote as *AGP*.

We also applied the CART procedure to identify potentially important two and three way interactions among the multivariate predictors in *MP*; we use *CP* to denote this set of predictors. In its most basic form, as applied here, the CART method constructs a classification tree in which each node in the tree represents a model variable and each branch out of a node represents a value range for that variable. Each leaf node of the tree predicts one of the possible values for the outcome variable. Thus, a path in the tree denotes a (partial) patient state, as well as an outcome predicted for patients in that state. The variables along any given path are said to interact. CART was applied with the GINI index and used a cost matrix that penalized making a false prediction of death twice as much as making a false prediction of survival.

Finally, we applied a backward stepping logistic regression procedure to the union of the predictors in *MP*, *AGP*, and *CP* in order to prune those predictors that were not statistically significant at the 0.05 significant level. The remaining predictors formed the terms in the final logistic regression model, which consists of

[2] The use of the entire study database of 14 199 cases (which consists of both the training and the test cases) to select univariate predictors for logistic regression was done for historical reasons. In particular, the researchers who generated the logistic regression model made a decision to use all the database cases in the selection process [9]. Since in the present study we are interested in making a comparison to this highly relevant prior work, we have included the logistic regression model here without modification. We note, however, that when evaluated on the test set, any model that incorporates such predictors, which are selected based on the entire database, has the potential to be biased.

Table 2
This table lists the variables that appear in each of the models, except that the variables used by the RL/OP method are not shown, because RL/OP constructed several models with different sets of variables

| Variable | Model | | | | | | |
|---|---|---|---|---|---|---|---|
| | LR | HME | SB | PCLR | K2MB | NN | KNN |
| *Patient-history findings* | | | | | | | |
| Age | x | x | x | x | | x | x |
| Gender | x | x | x | | | x | x |
| A re-admission to the hospital | | | | | | x | x |
| Admitted from a nursing home | | | | | | x | x |
| Admitted through the ER | | | | | | x | x |
| Has a chronic lung disease | | | | | | x | x |
| Has asthma | | | | | | x | x |
| Has diabetes mellitus | | | | | | x | x |
| Has congestive heart failure | x | x | x | | x | x | x |
| Has ischemic heart disease | x | x | x | | | x | x |
| Has cerebrovascular disease | x | x | x | | | x | x |
| Has chronic liver disease | | | | | | x | x |
| Has chronic renal failure | | | | | | x | x |
| Has history of seizures | | | | | | x | x |
| Has cancer | x | x | x | x | x | x | x |
| Number of above disease conditions | | | | | | x | x |
| Pleuritic chest pain | | | | | | x | x |
| *Physical examination findings* | | | | | | | |
| Respiration rate | x | x | x | x | x | x | x |
| Heart rate | x | x | x | | | x | x |
| Systolic blood pressure | x | x | x | x | | x | x |
| Temperature | x | x | x | | | x | x |
| Altered mental status | x | x | x | x | x | x | x |
| Wheezing | | | | | | x | x |
| Stridor | | | | | | x | x |
| Heart murmur | | | | | | x | x |
| Gastrointestinal bleeding | | | | | | x | x |
| *Laboratory findings* | | | | | | | |
| Sodium level | x | x | x | | | x | x |
| Potassium level | | | | | | x | x |
| Creatinine level | | | | | | x | x |
| Glucose level | x | x | x | | | x | x |
| BUN level | x | x | x | x | x | x | x |
| Liver function tests | x | x | x | x | | x | x |
| Albumin level | x | x | x | x | | x | x |
| Hematocrit | x | x | x | | | x | x |
| White blood cell count | | | | | | x | x |
| Percentage bands | | | | | | x | x |
| Blood pH | x | x | x | x | x | x | x |
| Blood $pO_2$ | x | x | x | | | x | x |
| Blood $pCO_2$ | | | | | | x | x |

Table 2 (continued)

| Variable | Model | | | | | | |
|---|---|---|---|---|---|---|---|
| | LR | HME | SB | PCLR | K2MB | NN | KNN |
| *Chest X-ray findings* | | | | | | | |
| Positive chest X-ray | | | | | | x | x |
| Lung infiltrate | | | | | | x | x |
| Pleural effusion | x | x | x | | | x | x |
| Pneumothorax | | | | | | x | x |
| Cavitation/empyema | | | | | | x | x |
| Lobe or lung collapse | | | | | | x | x |
| Chest mass | | | | | | x | x |

LR, logistic regression; HME, hierarchical mixtures of experts; SB, simple Bayes; PCLR, PC+LR; K2MB, Markov-blanket variant of K2; NN, artificial neural network model; KNN, K-nearest neighbor model.

20 univariate predictors (variables) shown in Table 2, as well as nine interaction terms among those variables[3]. The full model with coefficients is described in [9].

### 3.3. A summary of the hierarchical mixtures of experts method

The hierarchical mixtures of experts (HME) method was introduced in [11,12]. This method divides the set of patients into several populations, then estimates the probability of survival separately for each population. The HME method constructs its model by an iterative expectation–maximization (EM) algorithm: it alternately improves the division of the patients into populations and adjusts the estimated probability of survival for each population. In the final model, the populations can be viewed as the leaves of a binary classification tree. The internal nodes of this tree correspond to logistic regression equations, which estimate the probability that each patient belongs in the left or right subtree, each of which corresponds to separate patient subpopulations. Thus, in contrast to traditional classification-tree methods, the HME method allows probabilistic splits in the tree, so that, for example, a patient could be classified as 73% in population 1 and 27% in population 2. The probability for *vital_status* = alive for such a patient would be the weighted sum of 0.73 times the probabilistic prediction of *vital_status* = alive in population 1 plus 0.27 times the prediction of being in population 2.

### 3.4. The application of the hierarchical mixtures of experts method

We applied the HME method using the 20 variables and nine variable interactions in the logistic regression model described in the previous section; some of the variables in Table 1 correspond to more than one regression parameter in the HME

[3] The nine interaction terms that were included in the model are as follows: *Age × Gender*, *Age × Cancer*, *Age × Liver_function_tests*, *Gender × Systolic_blood_pressure*, *Gender × Albumin_level*, *BUN_level × Altered_mental_status*, *Gender × Sodium_level*, *Gender × pH*, *Altered_mental_status × pH*.

model. Using the entire training set, we initially built an HME model that had a balanced three-level binary tree structure containing eight leaf nodes.

This structure was then pruned as follows. To compare the current model $T$ to a pruned version of $T$, which we denote as $T_j$, we computed the following likelihood ratio $L_j$:

$$L_j = \prod_{i=1}^{9847} \frac{P(vital\_status_i | F_i, T)}{P(vital\_status_i | F_i, T_j)}$$

where $F_i$ is the set of findings for patient $i$, $vital\_status_i$ is the vital status outcome that patient $i$ experienced, and the product is taken over all patients in the training set. Each possible pruning of a single node in $T$ corresponds to some $T_j$; at each step, we choose the $T_j$ with the smallest likelihood ratio value $L_j$. After pruning a node, we ran the HME fitting algorithm on the pruned tree until convergence was reached. The pruned tree then became $T$, and we repeated the above process. As a heuristic, we stopped pruning when $\log(L_j) > 84$ for all $j$, where 84 is twice the number of parameters that each pruning saves.

The final HME model contains three internal nodes and four leaf nodes. Each internal node is a logistic regression model with 40 variables and one constant, making a total of 41 parameters. Each leaf node contains just a single constant parameter. Thus, there are a total of 127 parameters in the HME model. Constructing this model required about 5 h of real time on a DEC Alpha workstation.

We used the HME model to assign a probability to each of the test cases. All the test cases were classified in about 6 s on the same machine. The test cases were sorted by their probability, and a probability threshold was used to partition the cases into those patients predicted to live and those predicted to die. The probability threshold was varied to obtain a plot of fpsd versus fps.

### *3.5. A summary of a rule-based method*

This section describes a method called RL/OP, which combines a rule learning system (RL) with a post-processing rule optimizer (OP).

RL is a knowledge-based, rule induction program under development in the Intelligent Systems Laboratory at the University of Pittsburgh [4]. The rules generated by this program are of the form *if* ⟨*feature 1*⟩ *and* ⟨*feature 2*⟩ *and … then predict the case as* ⟨*class X*⟩. A feature is an *attribute:value* pair. Thus *age* is an attribute, while *age* > 80 is a feature. If a test case satisfies all of the features on the left-hand-side of the rule (the if part), then the rule would cause that case to be predicted as the class specified by the right-hand-side of the rule (the then part). The following is an example of a rule that predicts that a patient will survive: *if* (*age* < *23*) *and* ($pO_2 > 47.5$) *then predict vital_status* = *alive*. An example which predicts expired is as follows: *if (systolic blood pressure* < *60) then predict vital_status* = *expired.*

RL performs a general-to-specific search of the rule-space and selects ‘good rules’ using the following two criteria: (a) that the rule strength be greater than a predetermined threshold (e.g. 0.6), and (b) the heuristic that every new rule must cover at least one new case. Rule strength is determined by a *certainty factor* (CF) which is a function of the correct and incorrect predictions and the class prevalences in the training data.

OP is a ruleset post-processor under development in the Section of Medical Informatics at the University of Pittsburgh [2,3]. It is designed to select a subset of rules from a ruleset (from any source) with the goal to improve classification on unseen test cases. OP selects a subset of rules from a larger ruleset using a two stage process. The first stage, which we call rule ordering, involves assigning an order to the rules. The second stage, which we call ruleset evaluation and selection, involves selecting a ruleset using the ordering from the first stage. The rule ordering stage begins by rating the strength of each of the rules, based on the user-specified costs of incorrect predictions and the actual predictive performance on the training data. The rule with the highest strength is placed at the top of the list. In order to determine the second rule to place on the ordered list, the algorithm recalculates the rule strength for each of the remaining rules, giving more weight to the cases in the database that were incorrectly classified by the first rule. This procedure is repeated, so that in determining the third rule to be placed on the ordered list, the training cases given more weight are those which are incorrectly classified by the first two rules, and so on.

During the ruleset evaluation and selection stage, OP selects the single ruleset which performs best (i.e. has the lowest cost of incorrect predictions) on the training data. The first ruleset evaluated is the empty ruleset. (This corresponds to the rule *predict all cases as expired*, since the default class is *vital_status* = expired.) The second ruleset contains the first rule from the ordered list, the third ruleset contains the first two rules from the ordered list, and so on.

### 3.6. The application of the rule-based method

RL was run using all of the variables in Table 1, for each of certainty factor thresholds of 0.60, 0.80, 0.90, and 0.95. Thus, four rulesets were produced. For continuous variables, it determined rule-specific two-valued discretizations of those variables. For categorical variables, it used the values given in Table 1. The four rulesets were combined to form a large ruleset consisting of 738 rules.

In this study, misclassification costs facilitated the selection of rulesets whose predicted total error cost was minimized. By varying the ratio of the costs of the two prediction errors (i.e. the ratio of number of predictions of *vital_status* = alive in patients who expired to number of predictions of *vital_status* = expired in patients who survive), we were able to build models with varying fraction of patients predicted to survive (*fps*). Because the RL/OP method provides only a categorical output (i.e. *vital_status* = alive or *vital_status* = expired) a different model (ruleset) is required for each *fps* value of interest.

In our study, the rules generated by RL predicted some cases as *vital_status* = alive and others as *vital_status* = expired. When two or more conflicting rules predicted the same case to be both *vital_status* = alive and *vital_status* = expired, it was necessary for OP to use a conflict-resolution strategy to decide the class of the case. The conflict-resolution strategy used in this study was based on a weighted sum of rule strength, which is best illustrated with an example. Assume for a given case that two rules predict expired, and three rules predict alive. The prediction for the case will be *vital_status* = expired, if the sum of the strengths for the rules predicting *vital_status* = expired exceeds the sum of the strengths for the rules predicting *vital_status* = alive. If no rule predicts the class of a case (or if the weighted voting ends in a tie), a default class is chosen. The default class used in this study is *vital_status* = expired, because this class has the least expected cost (i.e. the class with the least cost if all cases were predicted as that class).

OP was run on the 738 rules for different ratios of the costs of the two prediction errors. As this ratio was varied from 1 to 80, OP produced rulesets that had both fewer cases predicted to live and lower error rates for those cases.

RL required about 60 min of real time on a DEC 3000 workstation for each run. OP required approximately 5 min of processing time per cost ratio (model). All the cases in the test set were scored in approximately 10 s on the same machine. The number of rules per model ranged from 90 to 427 (average of 204 over the six models generated). The number of variables used in the models ranged from 35 to 44 (average 41). The number of parameters, defined as the sum of the number of conditions in the left-hand-sides of all the rules in a model, ranged from 192 to 1040 (average 475).

### *3.7. A summary of a simple Bayesian classifier*

The Simple Bayes classifier (SB) is based on the well-known version of Bayes' theorem in which findings are assumed conditionally independent given a patient state [13,14,25]. Let $V$ be a variable that ranges over patient states of interest. For example, $V$ might denote a patient disease state or outcome state. In the experiments reported here, $V$ denotes the variable *vital_status*. Let $F = \{f_1, f_2, \ldots, f_n\}$ be a set of finding variables. For example, $f_j$ could be the patient's age that ranges over the positive integers. Then let $P(f_j|V)$ denote the probability distribution of $f_j$ given patient state $V$. In training SB, we designate a set of findings for inclusion in the model, and then we estimate $P(f_j|V)$ from the training data. Given a set of findings $F$, we can compute the posterior probability $P(V|F)$ as follows (under the assumption of conditional independence of the findings given that the value of $V$ is known):

$$P(\underline{V}|\underline{F}) = \frac{P(\underline{V}) \prod_{f \varepsilon E} P(f|\underline{V})}{\sum_{V} (P(V) \prod_{f \varepsilon E} P(f|V))}$$

where an underlined variable set indicates that each of the variables in the set is instantiated to some particular value. The summation is taken over all possible instantiations of the variables in set $V$.

### *3.8. The application of the simple Bayesian classifier*

We estimated $P(f_j|$*vital_status* = alive) as the number of joint occurrences of $f_j$ and *vital_status* = alive in the training set divided by the number of occurrences of *vital_status* = alive in the training set; the probability $P(f_j|vital_status = \text{expired})$was estimated in an analogous manner. The prior probability $P(vital\_status = \text{alive})$ was estimated as the fraction of the training cases in which *vital_status* has the value alive; the probability $P(vital\_status = \text{expired})$ was estimated similarly.

The variables used in the SB model were taken as the 20 variables found by the logistic regression model (see Table 2). The SB model did not, however, use any of the interaction terms that were used in the logistic regression model. For each of the 20 variables, SB used the discrete values given in Table 1, except the variable *age*, which was discretized into the intervals 0–10, 11–20,..., 90–100. The SB model contained 136 parameters consisting of prior and conditional probabilities. The model was constructed in approximately 3 min of real time using a DEC 5000 workstation. Using the same computer, about 2 min were required to classify all the cases in the test set.

### *3.9. A summary of an independence testing method for learning belief networks*

Let $G$ be a directed acyclic graph over a set of variables $Z$, and let $X$ be in $Parents(Y, G)$ if and only if there is a directed edge from $X$ to $Y$ in $G$. For discrete variables, say that a distribution factors according to $G$ if and only if

$$P(Z) = \prod_{Y \varepsilon Z} P(Y|Parents(Y, G))$$

that is, the joint distribution over $Z$ is the product of the conditional distributions $P(Y|Parents(Y, G))$, for each $Y$ in $Z$. For discrete variables, the parameters of the distribution are just the conditional distributions $P(Y|Parents(Y, G))$. Let a *belief network model* be a pair $\langle G, P\rangle$, where $P$ is a probability distribution that factors according to $G$ [15].

The factorization of $P$ entails a set of conditional independence relations among variables in $Z$. Say that a distribution $P$ is *faithful* to $G$ if and only if it factors according to $G$, and every conditional independence true in $P$ is entailed by the factorization of $P$ according to $G$. We also assume that $P$ is faithful to some directed acyclic graph $G$. There may be more than one directed acyclic graph $G$ to which $P$ is faithful, but for the purposes of predicting mortality based on patient findings (as opposed, for example, to predicting the effects of different therapeutic interventions) these different directed acyclic graphs are equivalent.

The PC algorithm takes as input a database and optional background knowledge such as time order. It uses statistical tests of conditional independence relations to construct a *pattern*, which represents a set of statistically (but not causally) indistinguishable directed acyclic graphs. The user specifies a significance level to use with the statistical tests. If there are hidden variables, the PC algorithm may indicate that it cannot find a model without latent variables to which the data are faithful. The PC algorithm is asymptotically reliable if $P(Z)$ is faithful to some directed acyclic graph that contains only the variables in $Z$ (in the sense that $P(Z)$ is faithful to each of the directed acylic graphs represented by the pattern). The algorithm and correctness proofs are described in detail in [20,21].

### *3.10. The application of the independence testing method for learning belief networks*

The goal was to find a model that minimized fpsd for a given level of fps. The PC algorithm uses statistical tests to construct a pattern, and the output depends upon the significance level chosen. One possible method for constructing a model would be to randomly partition the training data into a subset 1 and a subset 2, and then proceed as follows:

(1) Use the PC algorithm to construct a pattern from the data in training subset 1 for each of the following five significance levels: 0.0001, 0.0005, 0.001, 0.005, and 0.01[4]. In constructing each of these belief networks, search over the all the variables given in Table 1, using the discrete variable values given there (except the variable *Age*, which is discussed below).

(2) For each of the five patterns, select a belief network that is consistent with that pattern and estimate the parameters of that network using the data in training subset 1.

(3) Among the five belief networks, choose the one that minimizes fpsd as a function of fps on training subset 2.

The problem with the above method is two fold. First, we discovered that some of the patterns indicate the presence of hidden variables. Hence the pattern could not be turned into a model that could be estimated without latent variables (step 2 above). Second, there were up to eight variables directly adjacent to *vital_status*, so in order to estimate the probability of *vital_status* conditional on the variables directly adjacent to it would require a very large number of parameters, which would make the parameter estimates unstable. For these reasons, we modified step 2 of the algorithm in the following way. We used the adjacencies in each of the patterns output by the PC algorithm at the different significance levels to select predictor variables for a logistic regression model, under the assumption that the strongest predictors were the variables adjacent to *vital_status* in the PC output. We call this hybrid technique the PC + LR (or PCLR) method. This approach greatly reduced the number of parameters needed to predict *vital_status*. The variables selected at 0.005 and 0.01 significance levels were identical. Using training subset 2, we found that the variables selected at the 0.001, 0.005, and 0.01 significance levels produced very similar predictions, which were much better than the predictions that resulted from using variables selected at the lower significance levels. Since the fewest variables were selected at the 0.001 significance level, we used the belief network that resulted from that level; call it *B*. The nine variables included in B are shown in Table 2. In constructing *B*, the variable *age* was

[4] The FCI algorithm [21] is a more general procedure that takes as input raw data and optional background knowledge, and assuming the data are faithful to some graph, outputs a graphical object that represents a set of directed acyclic graphs with latent variables that entail the conditional independence relations judged to hold in the data. We do not use the FCI algorithm because in many cases it is not obvious how to select a representative model from the class of models represented by the output, or how to estimate such a model.

represented as integer values indicating an age category in years: 0–39, 40–54, 55–63, 64–68, 69–72, 73–75, 76–79, 80–83, 84–88, and above 89. These intervals were chosen because they contain approximately equal numbers of patients.

Using a Sparcstation 20 workstation, the CPU time required to construct $B$ was approximately 7 h and 15 min. The logistic regression model constructed from $B$ contained ten parameters; we did not explore the inclusion of interactions terms in the logistic regression model. Using a Sparcstation 20, the time required for classification on all the test cases using logistic regression was approximately 2 min.

### *3.11. A summary of a Bayesian method for learning belief networks*

This method uses a Bayesian scoring metric and a heuristic search procedure to learn belief networks from data. The scoring metric assigns to a belief-network structure a number that is proportional to the posterior probability of that structure given a database of training cases (and a set of modeling assumptions). A greedy search procedure is used in an attempt to find a belief-network structure that has a relatively high score. The belief-network structure found is then parameterized (i.e. the probabilities associated with the network are estimated) using the training data.

In our experiments we used the Bayesian scoring metric described in [5] in conjunction with a variant of the K2 search algorithm described there. The K2 algorithm requires that the user give it an ordering $O$ of the database variables. A variable $X$ is a potential parent of a variable $Y$ if and only if $X$ is to the left of $Y$ in the ordering. K2 starts with an unconnected graph and for all possible single arc additions consistent with ordering $O$ it adds the arc that leads to the highest score increase. If no score increase can be achieved by a further arc addition, K2 stops. Since a total ordering of the variables is not known in our domain, we apply K2 with many random node orderings. Among all the belief-network structures inferred in this manner, we select the one that has the highest score.

Since the space of random orderings is immense for more than a few variables, we can further constrain the search as follows. We developed an algorithm that takes a single *focus variable* $V$ to be predicted and places it at the rightmost end of ordering $O$. The algorithm then applies K2 to determine the parents of $V$. These parents (call them $\pi_V$) are taken to approximate the Markov Blanket (MB) of $V$[5]. Let $Z$ denote the set of variables consisting of $V$ and $\pi_V$. We apply K2 using multiple random orderings on the variables in $Z$. Thus, K2 will derive a belief-network structure that contains just the variables in $Z$. We call this variant of K2 the K2MB method.

[5] The Markov Blanket MB($V$) of a variable $V$ is a minimal set of variables that makes $V$ probabilistically independent of all other variables in the model, conditioned on knowing the values of the variables in MB($V$).

### 3.12. The application of the Bayesian method for learning belief networks

We applied K2MB with *vital_status* as the focus variable. Table 2 shows the six variables in $\pi_{vital\_status}$ that were selected to approximate the Markov Blanket of *vital_status*. K2MB applied K2 with 5000 random orderings on the variable set consisting on *vital_status* and $\pi_{vital\_status}$. This search required approximately 11 h of real time on an IBM RS6000 Model 320. There are 111 independent probabilities that parameterize the belief-network structure found by K2MB. This resulting belief network was used to perform inference on the test cases, which required a total of approximately 1 min of real time on the RS6000.

### 3.13. A summary of a neural network method

An artificial neural network (NN) is a generalized, non-linear, least-squares curve-fitting procedure that uses a network of idealized neuron-like computing elements as a function approximator [17,18]. Each artificial neuron takes a weighted sum of its inputs and passes this sum through a smooth but non-linear transfer function. The function is smooth so that the function computed by the network is differentiable, which makes training the network easier. It is non-linear to allow the network to approximate functions more complex than simple linear models.

A typical network consists of three layers of neuron-like units. The first layer is the input layer where measurements such as blood pressure or temperature are fed into the network. The second layer (often called the *hidden layer*) is connected to the input layer via an adapting set of weights. The purpose of the hidden second layer is to find a re-representation of the inputs that facilitates the prediction the network is learning to make. The third layer is the output layer where the prediction of the network is made. It is attached to the second layer through a second set of adapting weights. Typically, the adapting weights connecting the first and second layer and the second and third layer are trained by a gradient descent procedure called backpropagation. During training, the inputs are applied to the first layer, the target prediction is applied at the output layer, and the internal weights of the network are adjusted so that the network's prediction more closely matches the target value. This weight adjustment process is repeated by cycling through the training data until no further improvement occurs.

Artificial neural networks are popular because (1) they are appropriate for many different kinds of prediction, (2) they often perform competitively with other more specialized models, and (3) their use often requires minimal manual preprocessing of the data. In other words, they are popular because they are an effective black-box learning procedure. The black-box nature of artificial neural networks, however, is also a weakness. It is difficult, for example, to inspect a trained network to see what it has learned, or to modify a network to incorporate medical expertise not contained in the training data.

### *3.14. The application of the neural network method*

We used the Aspirin/MIGRAINES Neural Network Simulator, Version 6, which is distributed by Mitre Corporation. We will use the abbreviation NN to denote this method and the model it constructed. We gave NN access to all the variables listed in Table 1. Recall that Table 1 lists a total of 46 variables. Of these, 17 are continuous variables that were mapped into discrete categories. Therefore, counting both the continuous and discrete versions of variables, there are a total of $46 + 17 = 63$ variables. NN was given training data on all 63 variables. Thus, there was some redundancy in the variables given to NN as input. The value range of each variable was scaled to the interval [0,1]. Boolean inputs (e.g. Male/Female) are represented as 0/1. Continuous inputs are scaled proportionately from 0.0 to 1.0. So, for example, age is scaled so that 0.0 represents the youngest patient in the training set and 1.0 the oldest. The output of the network predicts patient mortality by assigning a value to the variable *vital_status*. During training, an output for *vital_status* of 0.0 was used to represent survival and 1.0 was used to represent death. Note that although the training set contains only values of 0.0 and 1.0 for *vital_status*, we are interested in the network's ability to predict values between 0.0 and 1.0 so that an estimate of the probability of death can be made.

A single artificial neural network containing 63 inputs, eight hidden units, and one output unit was trained using the 9847 training cases. We selected a momentum of 0.9 and a learning rate of 0.1, because they are typical values that are used in applying neural networks. We used 20 000 epochs during training. The number of epochs was determined by partitioning the training set into two subsets. In particular, for different numbers of epochs a neural network was learned using training subset 1 and then tested using subset 2. Finally, we trained a neural network on the entire training set using 20 000 epochs. Training required about 24 h of real time on a DEC Alpha workstation. The resulting network contained approximately 600 weights (parameters). We used this network to assign a probability of death for each of the test cases. In performing classification, NN used as input all 63 variables. The test cases were sorted by their probability, and a probability threshold was used to partition the cases into those patients predicted to live and those predicted to die.

### *3.15. A summary of a K-nearest neighbor method*

The K-nearest neighbor (KNN) method is a classification/prediction technique which uses the training cases themselves to classify new instances instead of building a computational model (such as a rule base or artificial neural net) to use to make predictions [7]. KNN is a direct implementation of the dictum that similar cases should have similar outcomes. It works by finding the $K$ training cases most similar (in some sense) to the new case (its $K$ nearest neighbors) and then assigning to the new case the predominant classification of those $K$ neighbors. Thus, KNN is a form of case-based reasoning. To use a KNN method, one specifies $K$ (the number of neighbors that will be used to make the prediction), how similarity between cases is to be measured so that the $K$ *nearest* cases can be found, and how the predictions of the $K$ nearest neighbors are to be combined to form the prediction for the new case.

### *3.16. The application of the K-nearest neighbor method*

We used KNN as follows to derive a probability of death (i.e. *vital_status* = expired) for each patient. $K$ is the number of neighbors in the training set used to estimate the probability of death for a new case. To find the $K$ nearest neighbors we used a simple Euclidean distance in the 63 dimensional space defined by the 63 available measurements[6]. Each variable was scaled to the same range so that all variables would have comparable importance in the distance calculation. Let $D$ be the number of the $K$ nearest neighbors that died. The ratio $D/K$ estimates the probability of death of the patient and can be used to rank it when compared with other patients. In computing $D/K$ we do not take into account the distances to the different neighbors; each neighbor affects the estimated probability equally, independent of distance to the new case. (We note that this application of KNN is basic; more sophisticated KNN methods using more complex distance metrics and combination schemes might perform better).

In this application we are using KNN to estimate a continuous quantity, namely a probability. The training set, however, contains only Boolean outcomes indicating whether each patient lived or died. Because of this, we need to use large enough $K$ to provide low variance estimates of the probability and to have sufficient resolution for classification. As an example, if $K = 5$, the possible maximum likelihood estimates of the probability of death are 0.0, 0.2, 0.4, 0.6, 0.8, and 1.0. Since the probability of death is, on average, about 0.11 for the patient population in the test set, estimates made with $K = 5$ would not only have high variance, but would probably yield poor rankings because the probability is quantized too coarsely in the region where most cases will fall (0.00–0.20). Of course if $K$ is too big, the neighborhoods will become too large to accurately sample small clusters in the space.

Since there is no principled way to select $K$, we selected a value for $K$ by using a cross validation with 6855 cases randomly chosen from a training set as a subset 1 and the remaining 2992 cases from the training set as subset 2. We tried $K = 10$, 25, 50, 100, 200, 400, and 800 using subset 1. The performance of each $K$ value was evaluated by classifying the cases in subset 2. By using different probability thresholds for predicting a case as expired versus alive, we computed the fpsd values corresponding to fps values of 0.1, 0.2, 0.3, 0.4, 0.5, and 0.6. KNN models with $K$ equal to 200, 400 and 800 performed well, with $K = 400$ providing the best overall performance.

[6] The Euclidean distance is more appropriate for deriving nearest neighbors when using ordinal variables rather than nominal variables. For simplicity, however, we used the same distance metric for both types of variables.

KNN training consisted of semiautomatically exploring the predictive performance of using different values of $K$. This training required approximately 5 h of real time on a DEC Alpha workstation. The time required to classify all 4352 test cases with $K = 400$ was about 27 min on an unloaded Dec Alpha. KNN is different from the other models in this paper in that it uses the training database directly, rather than construct a model from the training data. In this sense, the number of parameters used by the KNN algorithm is $9847 \times 63$ (number of training cases times number of variables per case) plus 1 (the parameter $K$), which is equal to 620 362. Thus, by far, the KNN method uses more parameters than any other method described in this paper.

We used the KNN model to assign a probability to each of the test cases. The test cases were sorted by their probability, and a probability threshold was used to partition the cases into those patients predicted to live and those predicted to die.

### 3.17. Hybrid methods

We also examined the predictive performance of four hybrid models that combine the prediction of LR, HME, PCLR, NN, and KNN, which we call the *base models*. The first hybrid model (called *mean value*) averages the five posterior probabilities of the base models, which it uses as a posterior probability to derive its predictions. The second hybrid model (called *majority rule*) predicts *vital_status* = alive for a patient case if three out of five of the models predicts alive, otherwise it predicts expired. The third hybrid model (called *consensus*) predicts *vital_status* = alive only if all five models predict alive, otherwise it predicts expired. The fourth hybrid model (called *exception*) predicts *vital_status* = alive if any of the five models predicts alive, otherwise it predicts expired.

### 3.18. Statistical methodology

In this section we describe the methods we used for comparing error (fpsd) rates and for performing a sample-size analysis based on the error rates we observed.

#### 3.18.1. Comparison of fpsd values

Recall that fps denotes the fraction of patients predicted (by a model) to survive, and fpsd denotes the fraction of those patients who in fact died. In this paper the primary approach to comparing models is based on a plot of fpsd versus fps for each of the models. For the purpose of analysis, we assume that in the limit (as the test database grows in size) that for a given fps value the fpsd value of each model will converge to some point value. Thus, a statistical analysis based on the binomial distribution is applicable. We use a statistical analysis based on such ratios of events, that is, proportions. In particular, based on the binomial distribution, we (1) compute confidence intervals around fpsd values and (2) derive the statistical significance of the difference in fpsd values of different models at a given fps value (or small range of fps values).

A standard method for comparing the difference between two proportions is to use the $z$-approximation. Suppose, for example, that we have the proportion $q_1 = m_1/n_1$, and the proportion $q_2 = m_2/n_2$. In this paper, $q_i$ is equal to fpsd of model $i$ and $n_i$ is the sample size corresponding to a given fps value. The value for $z$ is computed as follows [6]:

$$z = (q_1 - q_2)/\sqrt{p(1-p)(1/n_1 + 1/n_2)} \tag{1}$$

where $p$ is a pooled probability estimate that is computed as $p = (m_1 + m_2)/(n_1 + n_2)$. Using standard statistical tables, a value of $z$ can be mapped into a two-sided $P$ value, which gives the likelihood that the same or more extreme differences would be observed under the assumption of the null hypothesis that there is no difference in the proportions in the limit.

The $z$-approximation method assumes that the proportions being compared are derived based on independent (non-overlapping) samples. In the current study, however, for a given fps value, there typically is partial overlap between the samples used to derive the fpsd values of different models. The partial overlap is due to each method selecting from a common test set those patient cases to classify as survivors. In general, two models so constructed will predict some, but not all, the same cases as survivors. If there were complete overlap, which generally there is not, then the McNemar test would be applicable. Suppose we assume that independent samples would yield the same fpsd values as those values we observe (by using partially independent samples), then Eq. (1) would derive $P$ values that indicate the statistical significance of the differences in error rates. In analyzing the results we obtained, we will make this assumption, realizing, however, that it provides only a heuristic indication of statistical significance. We will use $P^*$ to denote a $P$ value computed using Eq. (1) under the assumption.

### 3.18.2. Sample-size analysis

We perform a sample-size analysis, which derives the number of patient cases we would expect to need in a future study in order to detect fpsd differences between LR and each of the other seven models. To do so, we use the standard formula for computing a sample size [6] when the true proportions are assumed to be $p_c$ and $p_t$. The proportion $p_c$ is the fpsd value for LR, and the proportion $p_t$ is the fpsd value for one of the other models. We used a two-tailed Type I error of 0.10, and a lower one-tailed Type II error of 0.10. The sample size is the number of cases needed for estimating the fpsd value for LR for a given fps level; the same number of cases would be needed for estimating the fpsd value for model $i$.

## 4. Results

### 4.1. Model variables

From Table 2 in the Section 3, we observe that models contained from 6 to 46 variables (not including discrete versions of continuous variables). The HME and

SB models adopted the variables used by LR[7]. The NN and KNN models use all 46 variables. All the models in Table 2 contain the following five variables: *has cancer*, *respiration rate*, *altered mental status*, *BUN level*, and *blood pH*. All the models, except K2MB, contain the variable *age*. The PCLR and K2MB models use no chest X-ray findings.

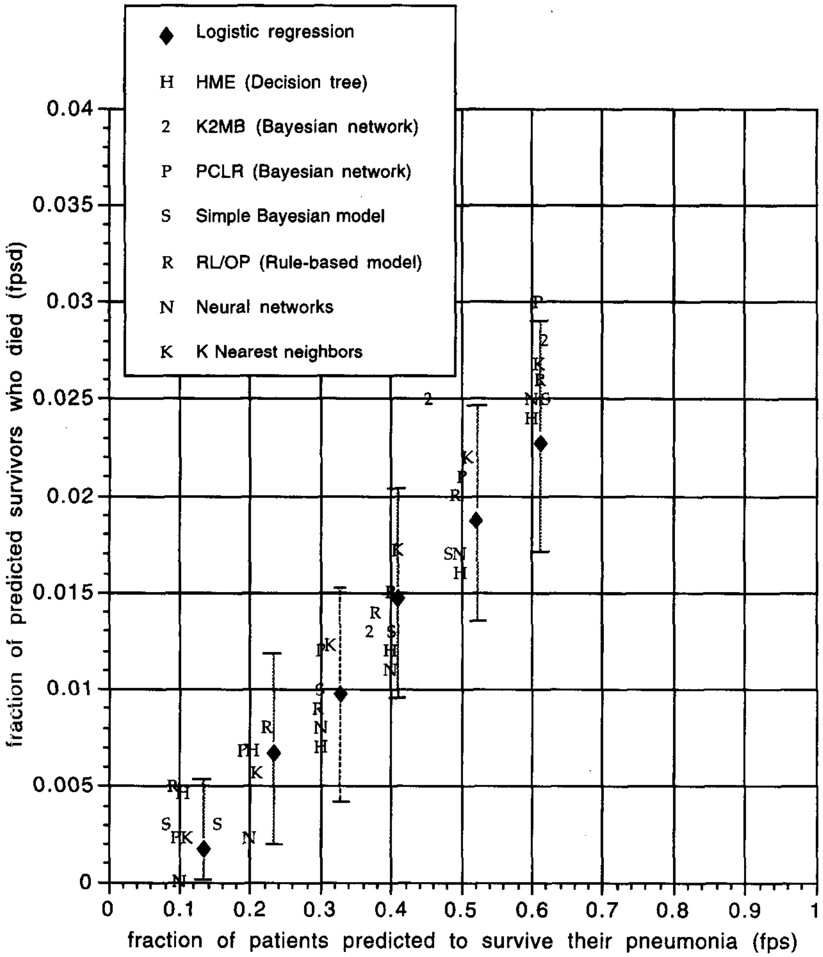

Fig. 1. A plot of the fpsd error rate as a function of fps for the eight models named in the legend. Each vertical bar through a logistic regression data point indicates a 95% confidence interval around that point. Some of the letters in the plot have been moved slightly to the right or left to prevent their overlapping each other.

[7] Note that this adoption by HME and SB of the variables used by LR implies that the HME and SB models are subject to the same possible variable-selection bias as is LR, which is discussed in footnote 2.

### *4.2. Errors in predicting patient mortality*

Fig. 1 contains a plot of fpsd versus fps for each of the models[8]. In the test set, 11% of the patients died. Thus, the highest that an fpsd value can possibly be is 11%. A vertical bar through each logistic regression data point gives the 95% confidence interval around that point. We used logistic regression as a reference model, because it is a standard method of prediction in medical studies.

Let the term *fps level* denote an fps value of 0.1, 0.2,..., or 0.6. For the data plotted in Fig. 1, we grouped the fpsd errors of each model according to fps level to which it was closest. Thus, for example, for an fps level of 0.6, the group consists of the eight data points on the far right of Fig. 1. For each fps level, the standard deviation for the fpsd value of each model is always between 0.002 and 0.003, except for fpsd level 0.5 where K2MB has a standard deviation of 0.004.

For each fps level, we compared the difference between the fpsd value of logistic regression (LR) to the fpsd values for each of the other methods. In every case, the $P^*$ value was greater than 0.10[9].

For each fps level, we also compared the difference between the lowest fpsd value of any model at that level to the fpsd value of every other model at the level. In only the following two instances was the $P^*$ value less than 0.10: (1) at fps level 0.2 the difference between the NN model and the RL/OP model has a $P^*$ value of 0.06, and (2) at fps level 0.5 the difference between the HME model and the K2MB model has a $P^*$ value of 0.04. Not surprisingly, the lowest fpsd values occurred with an fps level 0.1. For all the models (except K2MB), when fps is equal to approximately 0.1 (which corresponds to 435 patient cases) the fpsd value is no more than 0.0054 (which corresponds to two erroneous predictions of death). Thus, at an fps level of approximately 0.1, out of the 4352 cases in the test set, each model predicted that about 435 patients will survive, and of those patients, about 433 or 434 did survive.

### *4.3. Sample-size analysis*

Given the performance data displayed in Fig. 1, we now quantify how many cases we would expect to need in a future study to show a statistically significant difference between the LR model and each of the other models. Table 3 shows the total number of cases needed for each pairwise comparison. This analysis does not take into account that performing multiple pairwise comparisons generally requires a larger sample size for each comparison (to maintain given Type I and Type II

[8] Only three points are plotted for K2MB, because this model did not predict fps levels lower than 0.36. A posthoc analysis indicated that this was due to the variable *age* not appearing in the K2MB model. The seven other models each contained the variable *age*.

[9] For all statistical comparisons discussed in this section, the values for $m_1$ and $m_2$ at fps level 0.1 were each less than 5. For higher fps levels, these values were at least 5, except for the value for the NN model at fps level 0.2, which was 2. The statistical tests discussed in Section 3.18 may not be reliable when $m_i$ is less than 5.

Table 3
A sample-size analysis that estimates the number of patient cases needed to show statistically significant differences in the fpsd values between LR and each of the other seven models, under the assumptions of a Type I and Type II error of 0.10

| fps level | HME | RL/OP | SB | PCLR | NN | KNN | K2MB |
|---|---|---|---|---|---|---|---|
| 0.1 | 97 195 | 125 675 | 744 319 | * | 128 364 | * | ** |
| 0.2 | * | 844 410 | 63 836 | * | 39 222 | 1 136 689 | ** |
| 0.3 | 115 594 | 2 239 013 | * | 290 261 | 266 329 | 290 261 | ** |
| 0.4 | 132 518 | 1 229 013 | 302 736 | * | 73 384 | 320 494 | 302 736 |
| 0.5 | 135 054 | 2 588 132 | 307 480 | 321 568 | 307 480 | 144 449 | 37 240 |
| 0.6 | 1 275 815 | 206 885 | 321 829 | 27 419 | 321 829 | 81 878 | 52 852 |

An asterisk indicates that LR and the model to which it is being compared have the same fpsd value, and therefore a sample size estimate is not defined. A double asterisk indicates those fps levels for which K2MB did not produce an fpsd value.

errors) than does a single pairwise comparison; thus, the sample sizes in Table 3 should be interpreted as lower bounds on the number of cases needed if all seven pairwise comparisons are made. The sample sizes in Table 3 range from 27 419 to 2 588 132, with the mean size being 450 665 cases.

### 4.4. Hybrid models for predicting patient mortality

If the models are making errors on the same cases (or many of the same cases), then there would be little hope in combining the models to form a hybrid model that has improved predictive performance. If most such cases are different, however, a hybrid model might do relatively well, when compared with the eight base models. We examined the degree to which the following five models made errors on the same cases: LR, HME, PCLR, NN, and KNN.

For each of the six fps levels, we listed each case that resulted in an fpsd error for each of the five models. Moreover, for each fps level we tabulated the number of erroneously predicted cases that are shared by the five models. Each fps level is represented by a column in Table 4. Moreover, the errors are recorded in the rows

Table 4
An entry in the table indicates the number of erroneous patient survival predictions that are common to a given number of methods for a given fps level

| No. models | fps levels | | | | | |
|---|---|---|---|---|---|---|
| | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 |
| 1 | 1 | 10 | 12 | 21 | 50 | 58 |
| 2 | 0 | 3 | 8 | 7 | 11 | 22 |
| 3 | 1 | 2 | 7 | 13 | 8 | 18 |
| 4 | 0 | 0 | 3 | 5 | 16 | 13 |
| 5 | 0 | 1 | 1 | 7 | 14 | 33 |

Table 5
Values for fps and fpsd for each of four hybrid predictive models as applied to the test set

| Hybrid mean value | | Hybrid majority rule | | Hybrid consensus | | Hybrid exception | |
|---|---|---|---|---|---|---|---|
| fps | fpsd | fps | fpsd | fps | fpsd | fps | fpsd |
| 0.102 | 0.002 | 0.101 | 0.002 | 0.107 | 0.002 | 0.101 | 0.002 |
| 0.199 | 0.003 | 0.200 | 0.003 | 0.199 | 0.002 | 0.207 | 0.003 |
| 0.302 | 0.007 | 0.301 | 0.007 | 0.298 | 0.008 | 0.298 | 0.008 |
| 0.397 | 0.016 | 0.402 | 0.015 | 0.404 | 0.016 | 0.401 | 0.013 |
| 0.498 | 0.017 | 0.500 | 0.018 | 0.507 | 0.020 | 0.498 | 0.018 |
| 0.600 | 0.021 | 0.602 | 0.023 | 0.595 | 0.027 | 0.600 | 0.020 |

The three models are hybrids of the LR, HME, PCLR, NN, and KNN models.

1 through 5 according to the exact number of models that made that error (at that fps level). For instance, column 4 of the table tabulates the errors made by the models at fps level 0.4. We see that 21 erroneous case predictions were made by exactly one system, seven were made by two systems, and so on.

The results in row 1 of Table 4 indicate that often the five base models are making erroneous predictions on different cases. We examined the predictive performance of four hybrid models, which are described in Section 3.17. Table 5 indicates that the four models performed comparably, although the hybrid-exception model appears to perform slightly better than the other three. Fig. 2 shows a plot of the hybrid-exception performance data in Table 5, as well as a plot of the data from Fig. 1 for the five base models that were used to construct the hybrid-exception model. The plot suggests that the hybrid-exception model is predicting well, but not significantly better than the other five models on which it is based.

For each fps level, we compared the difference between the fpsd value of the hybrid-exception model to the fpsd values for each of the other model results shown in Fig. 2. In only the following two instances was the $P^*$ value less than 0.10: (1) at fps level 0.6 the difference between the hybrid-exception model and the PCLR model has a $P^*$ value of 0.02, and (2) at fps level 0.6 the difference between the hybrid-exception model and the KNN model has a $P^*$ value of 0.09.

### 4.5. The number of model parameters and variables

We tabulated the number of variables and parameters in each model. A variable is a patient finding. The number of variables is an indication of the burden of a model regarding data input. For this purpose, we counted a continuous variable as only one variable; we did not count the discrete version of a continuous variable, since the discrete version can be computed directly from the continuous version. A parameter is a numerical term in a model. The number of parameters provides an indication of whether a computer would be needed to apply the full model in a feasible manner. As shown in Fig. 3, some of the models differ significantly in the number of variables and parameters that they contain. The number of variables ranges from 6 to 46, and the number of parameters ranges from 10 to over 600. The K-nearest neighbor model is not included in Fig. 3, because in a sense the model uses the entire training database as a set of parameters, which translates to 620 362 parameters; such a large number would make scaling the *y* axis of Fig. 3 difficult. As we discuss in Section 5, these differences in the number of variables and parameters suggest that some of the models may be more readily converted to paper-based guidelines than other models.

### 4.6. Additional results

The results in Sections 4.1–4.5 suggest additional questions for study. Although it was not practical for us to address these questions at this time using all the

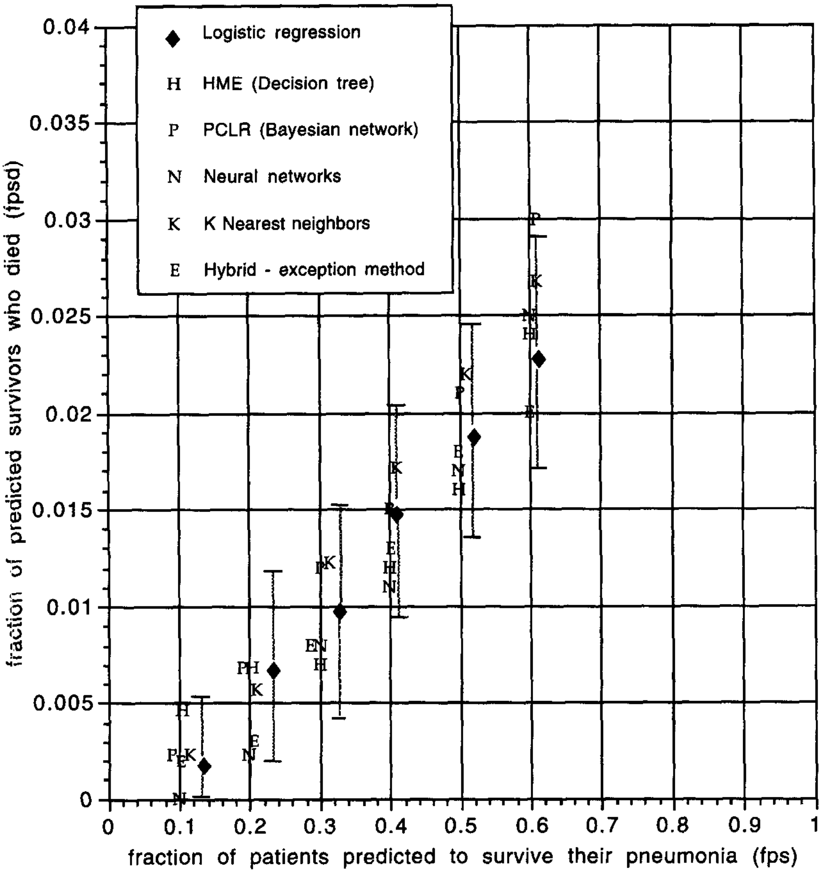

Fig. 2. A plot of the fpsd error rate as a function of fps for the five base models named in the legend, plus the hybrid-exception model shown as E.

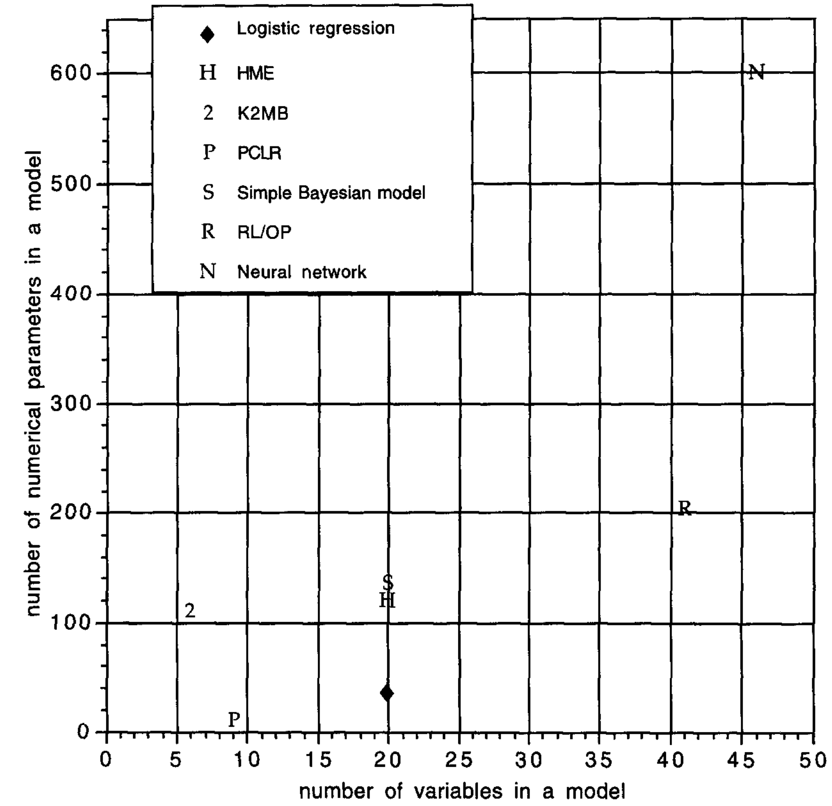

Fig. 3. A scattergram showing the number of variables and parameters in seven of the models. The data point for RL/OP shows the mean values over six RL/OP models (see Section 3.6). The K-nearest neighbor model is not shown (see the text).

prediction methods, we have addressed them using the NN and the KNN methods. This section describes the questions and the results we obtained. These results may be helpful in planning future extensions to the current study.

### 4.7. Predictive error of neural network models as a function of training set size

How likely would the results in Fig. 1 improve if additional training data were available? We partially addressed this question by performing additional experiments with the neural-network method. In particular, we randomly partitioned the training set into a subset 1 containing 6855 cases and a subset 2 containing 2992 cases. From the subset 1 we randomly selected training sets of size 100, 250, 1000, 2000, 3000, 4000, and 6000 cases. We repeated this selection process five times in order to generate five training sets of a given size. For each of the five training sets of a given size, we constructed a neural-network model and evaluated the model on the subset 2. We computed the mean fpsd over the five models constructed from a training set of a given size. Fig. 4 shows the results, which suggest that beyond about 3000 cases further significant reduction of the fpsd error rate is unlikely. Since we used a training set of 9847 cases in the experiments summarized in Fig. 1, it is unlikely that additional training data (on the same variables for the same patient population) would further improve the performance of the NN model.

### 4.8. Cross validation results for the neural network and K-nearest neighbor models

We constructed an additional ten pairs of training and test sets of size 9847 and 4352 cases, respectively, by random partitioning of the entire 14 199 cases in the study database. We constructed a NN model using each of the ten training sets, and we applied the model, as described in Section 3.14, to its paired test set. Thus, we derived ten additional fpsd data points for each fps level for the NN method.

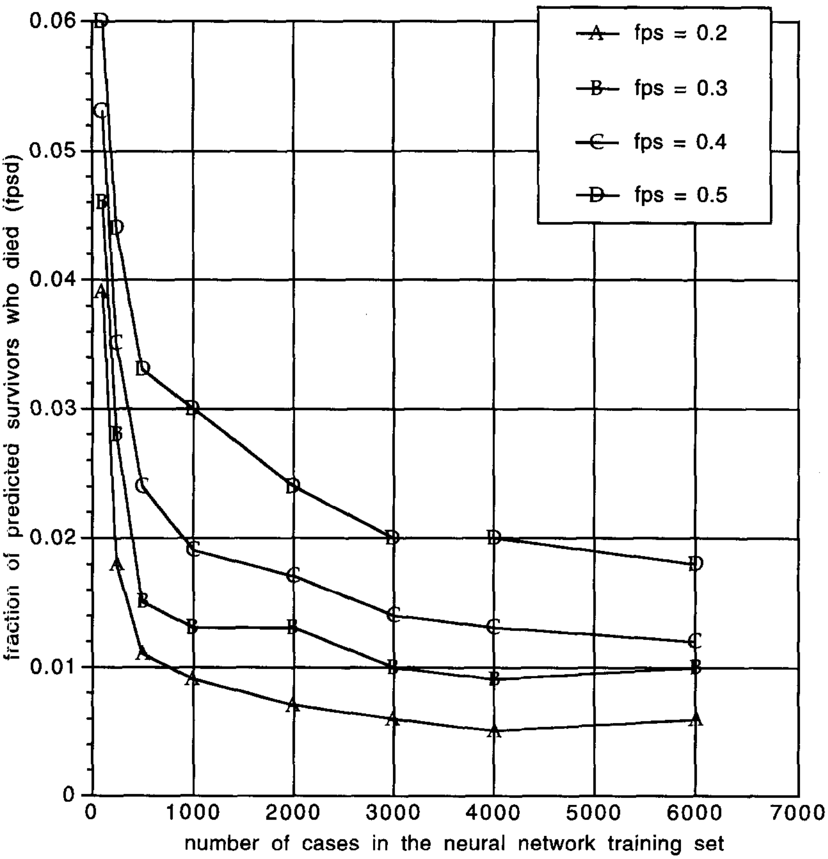

Fig. 4. A plot of the fpsd error rate for neural-network models that were constructed using different size training sets.

Table 6
Means and standard deviations derived using cross validation with 11 trials for the neural network (NN) method and 101 trials for the K-nearest neighbor (KNN) method

| fps value | NN mean fpsd | NN fpsd S.D. | KNN mean fpsd | KNN fpsd S.D. |
|---|---|---|---|---|
| 0.1 | 0.001 | 0.002 | 0.004 | 0.003 |
| 0.2 | 0.004 | 0.003 | 0.009 | 0.003 |
| 0.3 | 0.009 | 0.002 | 0.015 | 0.003 |
| 0.4 | 0.015 | 0.002 | 0.020 | 0.003 |
| 0.5 | 0.021 | 0.003 | 0.027 | 0.003 |
| 0.6 | 0.029 | 0.003 | 0.034 | 0.003 |

Furthermore, we constructed an additional 100 training and test sets that we used to develop 100 additional fpsd data points for each fps level for the KNN method.

Table 6 shows that the NN method outperforms the KNN method at every fps value. For each fps value, a $t$-test indicates that the difference between the mean fpsd of NN and KNN is significant at $P < 0.01$[10]. This result supports the pattern of relative performance between NN and KNN that is shown in Fig. 1. The standard deviations in Table 6 are close to those computed under the assumption of a binomial distribution, as described in Section 4.2; this correspondence increases our confidence in the validity of the statistical analysis described in that section. A comparison of Fig. 1 and the data in Table 6 shows that the performance of both NN and KNN are better in Fig. 1 than in Table 6. This result suggests that for the test cases used to derive Fig. 1, it was somewhat easier to predict mortality than we would expect for a wider sample of this domain population. We attribute this result to chance selection of the test cases. This pattern does suggest the possibility that the other methods also would perform slightly worse on a wider population sample.

## 5. Discussion

The ability to accurately predict mortality in patients with pneumonia may be useful to clinicians in making decisions about where to treat patients with pneumonia. We would like to identify as large a group of pneumonia patients as possible who could be treated safely at home. We studied a population in which approximately 11% of the patients died of pneumonia. For the eight different models we studied, the results show that when 10% of the patient population is predicted to survive, the predictive error rate is no more than half a percent. Additional analysis suggests that this rate may be slightly better than we would expect on a larger sample of patients than the ones we tested. The lowest error rate for each fps level was attained by one of the three following methods: neural networks, HME, and

[10] Cross validation increases the power of the statistical comparison. Strictly speaking, the independence assumptions underlying the $t$-test are not valid; thus, these results, while informative, should be considered heuristic.

logistic regression. The results also show, however, that all eight models tested have absolute error rates within 1% of each other. Significantly, most of the models' error rates are sufficiently close such that a very large test database would be needed to reliably establish whether the rates differ statistically.

We have compared the predictive performance of eight models constructed by eight methods. Thus, the results directly reflect the performance of the models, rather than the methods. Nonetheless, for at least one method (neural networks) resampling results suggest that the model's performance is a good indication of the method's performance for this domain (relative to the assumptions made in applying the method). Our data do not indicate how the eight methods would perform in making predictions in other medical areas, nor would it be expected of them to do so; additional studies would be needed to address such issues of robustness.

We examined four simple hybrid models that combine the predictions of five of the eight base models. There was one hybrid model that predicted as well or better than logistic regression for all six levels of fps. While these results look promising, we are not able to conclude at this point that this performance difference is statistically reliable. Recent results do lend hope that hybrids can be found that perform significantly better [22] than base models. The space of possible hybrid models is immense and largely unexplored. As one example, consider that the neural network method performed relatively well in the current study, but used all the variables. This suggests using one of the other methods to identify the variables for the neural network method to use in constructing a predictive model. Many other hybrid combinations are possible.

We found that the models differ markedly in the number of variables and parameters they contain. We emphasize, however, that in the current study we did not direct our efforts toward reducing the number of variables and parameters in models generated by the eight methods. Future research may show, for example, that it is possible to reduce the number of variables and parameters used in developing a neural network model, and yet, maintain the low predictive error rate of that model. Such reductions are of interest because models with a smaller number of variables may be more quickly and easily used by clinicians, since those models would require that less data be input. Models with a relatively small number of parameters may be more readily converted into paper-based models that could be used widely in current medical practice. As medical information systems become more widespread and comprehensive in the data they capture, the need for paper-based decision tools is likely to decline. Even so, paper-based models may be useful as medical teaching tools. One model (PCLR) in the current study required only nine variables and ten parameters. We have performed a preliminary conversion of this model, as well as several others described in the current paper, into paper-based models. In a future study, we plan to test the accuracy of these paper-based models for predicting mortality.

The current paper focuses on predictions of *inpatient* mortality. The applicability of such predictions in helping decide whether a patient should be treated at home versus in the hospital requires making the following methodological assumption:

*If a hospital-treated pneumonia patient has a very low probability of death, then that patient would also have a very low probability of death if treated at home.*

This assumption is needed for pragmatic reasons. In particular, our study database contains only information about inpatients, which is not ideal methodologically. Some inclusion of outpatients in the database is preferable. From the standpoint of predictive accuracy, an ideal database would be based on a clinical trial in which patients were randomized to be treated either as inpatients or outpatients. Such a trial would, however, be unethical, because it is likely that (1) some of the more severely ill patients would be randomized to be treated at home, (2) they would have fared better if hospitalized, and (3) clinicians would have elected to treat them in the hospital. Appropriately then, such an ideal database is unlikely to ever be available. One approach to using inpatient data would be to make the above stated methodological assumption, with the understanding that the models we build will eventually need to be evaluated as clinical decision aids. As one possibility, suppose that a trial is performed in which clinicians are randomized either to have or not to have access to such a decision aid in making decisions about where to treat patients who present with pneumonia. Suppose that (1) patients who are treated by clinicians using the aid do as well or better than patients who are not (in terms of mortality and significant morbidity) and (2) using the aid increases the number of patients treated at home, relative to not using the aid. Such a result would lend support to the decision aid being useful. In actual practice, the effectiveness of the tool would of course depend on whether physicians can and will use it. Thus, high predictive accuracy is likely to be a necessary, but insufficient, condition for producing decisions aids that have a positive impact on the cost and quality of patient care.

The ideal database described in the previous paragraph is unlikely to ever exist. On the other hand, the MCHD database that we used in the current study contains only information about inpatients. In between would be databases that contain extensive observational data on pneumonia patients treated in both inpatient and outpatient settings. Indeed, such data have been collected by a PORT Pneumonia study. Using this data, we plan to generate and evaluate models that predict outcomes conditioned on whether a patient is treated as an inpatient or an outpatient. The outcomes we intend to examine include patient mortality, as well as symptom resolution, complications, and length of time to recovery.

## Acknowledgements

This work was funded by grant BES-9315428 from the National Science Foundation (NSF) as part of a joint initiative by NSF and the Whitaker Foundation to support research on cost-effective health care. Support also was provided by grant LM05291-02 from the National Library of Medicine. Dr Fine is supported as a Robert Wood Johnson Foundation Faculty Scholar.

## References

[1] A.A. Afifi and V. Clark, *Computer-Aided Multivariate Analysis* (Van Nostrand Reinhold, New York, 1990).

[2] R. Ambrosino, *The Development and Evaluation of a Method for Rule Post-processing, Report SMI-95-02, Section of Medical Informatics*, Department of Medicine, University of Pittsburgh, 1995.

[3] R. Ambrosino, B.G. Buchanan, G.F. Cooper and M.J. Fine, The use of misclassification costs to learn rule-based decision support models for cost-effective hospital admission strategies, in: R.M. Gardner, ed., *Proc. Symp. Computer Applications in Medical Care* (Hanley and Belfus, Philadelphia, 1995) 304–308.

[4] S. Clearwater and F. Provost, RL4: A tool for knowledge-based induction, in: *Proc. 2nd Int. Conf. Tools for Artificial Intelligence* (IEEE Computer Society Press, Los Alamitos, CA, 1990) 24–30.

[5] G.F. Cooper and E. Herskovits, A Bayesian method for the induction of probabilistic networks from data, *Machine Learning* 9 (1992) 309–347.

[6] B. Dawson-Saunders and R.G. Trapp, *Basic and Clinical Biostatistics* (Appleton and Lange, Norwalk, Connecticut, 1990).

[7] R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis* (Wiley, New York, 1973).

[8] N. Ezquerra and A. Pazos, Neural computing in medicine, *Artif. Intell. Med.* 6 (1994) 355–357.

[9] M.J. Fine, B.H. Hanusa, J.R. Lave, D.E. Singer, R.A. Stone, L.A. Weissfeld, C.M. Coley, T.J. Marrie and W.N. Kapoor, Comparison of severity of illness measures in patients with community-acquired pneumonia, *J. Gen. Int. Med.* 10 (1995) 349–368.

[10] M. Hadzikadic, Automated design of diagnostic systems, *Artif. Intell. Med.* 4 (1992) 329–342.

[11] M.I. Jordan, A statistical approach to decision tree modelling, *Proc. 11th Int. Conf. Machine Learning* (Morgan Kaufmann Publishers, San Francisco, CA, 1994) 363–370.

[12] M.I. Jordan and R.A. Jacobs, Hierarchical mixtures of experts and the EM algorithm, *MIT Computational and Cognitive Science Technical report 9301*, Cambridge, MA, 1993.

[13] P. Langley, W. Iba, and K. Thompson, An analysis of Bayesian classifiers, *Proc. 11th Nat. Conf. Artificial Intelligence* (AAAI Press, Menlo Park, CA, 1992) 223–228.

[14] M.C. Miller III, M.C. Westphal, J.R. Reigart II, and C. Barner, *Medical Diagnostic Models: A Bibliography* (University Microfilms International, Ann Arbor, Michigan, 1977).

[15] J. Pearl, *Probabilistic Reasoning in Intelligent Systems* (Morgan Kaufmann, San Mateo, CA, 1988).

[16] J.A. Reggia, Neural computation in medicine, *Artif. Intell. Med.* 5 (1993) 143–157.

[17] D.E. Rumelhart, G.E. Hinton, and R.J. Williams, Learning representations by back-propagating errors, *Nature* 323 (1988) 533–536.

[18] D.E. Rumelhart and J.L. McClelland, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* (MIT Press, Cambridge, MA, 1986).

[19] V.W. Soo, J.S. Wang and S.P. Wang, Learning and discovery from a clinical database: An incremental concept formation approach, *Artif. Intell. Med.* 6 (1994) 249–261.

[20] P. Spirtes, C. Glymour and R. Scheines, An algorithm for fast recovery of sparse causal graphs, *Soc. Sci. Comp. Rev.* 9 (1991) 62–72.

[21] P. Spirtes, C. Glymour and R. Scheines, *Causation, Prediction, and Search* (Springer-Verlag, New York, 1993).

[22] P. Spirtes and C. Meek, Learning Bayesian networks with discrete variables from data, in: U.M. Fayyad and R. Uthurusamy, *Proc. 1st Int. Conf. Knowledge Discovery and Data Mining* (AAAI Press, Menlo Park, 1995) 294–299.

[23] USDHHS, *Health US 1988, Public Health Services DHHS Publication No. (PHS)* 89–1232, 1988.

[24] USGPO, *Statistical Abstract of the US*, 108th edn., US Government Printing Office, US Department of Commerce, Bureau of the Census, 1988.

[25] H.R. Warner, A.F. Toronto, L.G. Veasy and R. Stephenson, A mathematical approach to medical diagnosis: Application to congenital heart diseases, *J. Am. Med. Assoc.* 177 (1961) 177–183.

[26] G.I. Webb and J.W.W. Agar, Inducing diagnostic rules for glomerular disease with DLG machine learning algorithm, *Artif. Intell. Med.* 4 (1992) 419–430.

[27] A.K.C. Wong and K.C.C. Chan, Automating the knowledge acquisition process in the construction of medical expert systems, *Artif. Intell. Med.* 2 (1990) 267–292.