Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Biomedical Informatics 38 (2005) 347–366

Journal of Biomedical Informatics

www.elsevier.com/locate/yjbin

Predicting dire outcomes of patients with community acquired pneumonia

Gregory F. Cooper [a,*], Vijoy Abraham [b], Constantin F. Aliferis [c], John M. Aronis [d], Bruce G. Buchanan [e], Richard Caruana [f], Michael J. Fine [g], Janine E. Janosky [h], Gary Livingston [i], Tom Mitchell [j], Stefano Monti [k], Peter Spirtes [j,l]

[a] *Center for Biomedical Informatics, University of Pittsburgh, Suite 8084 Forbes Tower, 200 Lothrop Street, Pittsburgh, PA 15213, USA*

[b] *Academic Computing, Stanford University, 560 Escondido Mall, Meyer Library 260, Stanford, CA 94305-3093, USA*

[c] *Department of Biomedical Informatics, Vanderbilt University, Room 412 Eskind Library, 2209 Garland Avenue, Nashville, TN 37232, USA*

[d] *Department of Computer Science, University of Pittsburgh, Pittsburgh, PA 15260, USA*

[e] *P.O. Box 68, 315 Evergreen Way, Orcas, WA 98280, USA*

[f] *Department of Computer Science, Cornell University, 4157 Upson Hall, Ithaca, NY 14853, USA*

[g] *Center for Health Equity Research and Promotion, VA Pittsburgh Healthcare System (151-C), University of Pittsburgh, University Drive C, Building 28, Suite 1A102, Pittsburgh, PA 15240, USA*

[h] *Department of Family Medicine and Clinical Epidemiology, University of Pittsburgh, 3518 Fifth Avenue Pittsburgh, PA 15261, USA*

[i] *Department of Computer Science, University of Massachusetts Lowell, One University Avenue, Lowell, MA 01854, USA*

[j] *Center for Automated Learning and Discovery, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA*

[k] *The Broad Institute of MIT and Harvard University, 320 Charles Street, Cambridge, MA 02141, USA*

[l] *Department of Philosophy, Carnegie Mellon University, Pittsburgh, PA 15213, USA*

Received 22 February 2005
Available online 17 March 2005

Abstract

Community-acquired pneumonia (CAP) is an important clinical condition with regard to patient mortality, patient morbidity, and healthcare resource utilization. The assessment of the likely clinical course of a CAP patient can significantly influence decision making about whether to treat the patient as an inpatient or as an outpatient. That decision can in turn influence resource utilization, as well as patient well being. Predicting dire outcomes, such as mortality or severe clinical complications, is a particularly important component in assessing the clinical course of patients. We used a training set of 1601 CAP patient cases to construct 11 statistical and machine-learning models that predict dire outcomes. We evaluated the resulting models on 686 additional CAP-patient cases. The primary goal was not to compare these learning algorithms as a study end point; rather, it was to develop the best model possible to predict dire outcomes. A special version of an artificial neural network (NN) model predicted dire outcomes the best. Using the 686 test cases, we estimated the expected healthcare quality and cost impact of applying the NN model in practice. The particular, quantitative results of this analysis are based on a number of assumptions that we make explicit; they will require further study and validation. Nonetheless, the general implication of the analysis seems robust, namely, that even small improvements in predictive performance for prevalent and costly diseases, such as CAP, are likely to result in significant improvements in the quality and efficiency of healthcare delivery. Therefore, seeking models with the highest possible level of predictive performance is important. Consequently, seeking ever better machine-learning and statistical modeling methods is of great practical significance.

Keywords: Machine learning; Community acquired pneumonia; Outcome prediction; Quality and cost of healthcare delivery

[*] Corresponding author. Fax: +1 412 647 7190.
E-mail address: gfc@cbmi.pitt.edu (G.F. Cooper).

1532-0464/$ - see front matter © 2005 Elsevier Inc. All rights reserved.
doi:10.1016/j.jbi.2005.02.005

1. Introduction

This paper describes a retrospective evaluation of machine-learning and statistical methods that predict the chance of dire outcomes in patients who present with community acquired pneumonia (CAP). We use the term *dire outcome* to denote a severe complication, death within 30 days of presentation, or an admission to the intensive care unit (ICU) for either respiratory failure, respiratory or cardiac arrest, or shock/hypotension.[1]

CAP is an important clinical condition, both from the point of view of resource-utilization and patient outcomes. Previous studies have estimated that each year in the US there are about 4.1 million CAP patients of whom approximately 1.2 million are hospitalized [1,2]. Taken together, pneumonia and influenza have been ranked as the sixth leading cause of death in this country [3]. In the US, CAP has been responsible for 64 million days of restricted activity, 39 million days of bed confinement, and 10 million days of work lost annually [4]. The aggregate cost of hospitalization of CAP patients is estimated to total almost $9 billion per year in the US [5].

One use of predicting the probability of dire outcomes of CAP patients would be to assist clinicians in making the admission decisions for those patients. The admission decision is one of the most cost-sensitive decisions made in the care of CAP patients, with the average cost of care for inpatients being 18–28 times greater than the cost of care for outpatients [5]. Ideally, patients who could be safely treated as outpatients (typically at home) on oral antibiotics would be so treated, while usually the remaining patients would be admitted to the hospital and treated with intravenous antibiotics.

Researchers have previously developed models that predict mortality as an outcome of patients with CAP. The PSI model in particular was developed to predict patient mortality within 30 days of presentation with CAP [7], based on 20 demographic and clinical variables. The models described in the current paper predict dire outcomes more broadly than mortality, to include severe complications and admission to the ICU. It seems likely that decisions about where to treat CAP patients are based not just on mortality, but also on other possible dire outcomes. Thus, predicting dire outcomes more broadly seems useful.

To learn computer models that predict dire outcomes of CAP patients, we applied several induction methods to a training set (i.e., derivation set) of CAP patient records. These models were then evaluated using a CAP test set (i.e., validation set). The modeling methods we used are logistic regression, rule-based learning, neural networks, finite mixture model techniques, simple (i.e., naïve) Bayes methods, and a combination of finite mixture modeling and simple Bayes. All of these methods have been previously described in the literature, although some of them are likely to be unfamiliar to many readers. Our primary goal was not to compare these learning algorithms as a study end point; rather, it was to develop the best possible model to predict dire outcomes and then estimate the impact of its use on healthcare quality and cost.

We describe the predictive performance of each model on the test set, where the area under the ROC curve is used as a measure of performance. These results suggest which of the tested models are likely to perform best in predicting dire outcomes in future patients with CAP. We also describe performance when the training set contains more and more cases. These results indicate whether we can expect further improvement in the models' performance if the training set were expanded by collecting additional cases of patients with CAP. Finally, for the model M with the best predictive performance on the full training set, we estimate the impact that M would have on healthcare quality and cost, if it were applied in helping guide decisions about whether to treat CAP patients at home or in the hospital.

2. The CAP database

This section describes the CAP database used in this project. The description borrows from previous literature that reports other analyses of that database [6,7]. The pneumonia PORT database on CAP patients was collected using a prospective cohort study of hospitalized and ambulatory care patients. The study was conducted from October 1991 to March 1994 at five medical institutions in three geographic locations: the University of Pittsburgh Medical Center (UPMC), a 942-bed university teaching hospital and St. Francis Medical Center (SFMC), a 427-bed community teaching hospital, in Pittsburgh, Pennsylvania; Massachusetts General Hospital (MGH), an 899-bed university teaching hospital, and Harvard Community Health Plan-Kenmore Center (HCHP), a 44,931-member health-care center within a staff-model health maintenance organization (only ambulatory patients were enrolled from this site), in Boston, Massachusetts; and Victoria General Hospital (VGH), a 637-bed university teaching hospital, in Halifax, Nova Scotia, Canada. Pittsburgh and Boston were selected because prior work by the investigators suggested that patient management strategies differed between these areas.

Eligible patients were identified by trained research assistants through daily reviews of emergency, admitting, and radiology department records and patient logs. Clinical eligibility was determined by patient interview

[1] A detailed description of the definition we use for a dire outcome is given in Appendix B. Unless stated otherwise, the term *dire outcome* (in italics) in this paper refers specifically to that definition.

and review of medical records. Eligibility criteria were that a patient must: (1) be at least 18 years of age, (2) have one or more symptoms suggestive of pneumonia, (3) have radiographic evidence of pneumonia within 24 h of presentation, and (4) provide informed consent for base-line and follow-up interviews [7]. For radiologic evidence of pneumonia, the report of the local clinical radiologist was used. Patients with one or more of the following criteria were not eligible for enrollment: (1) hospitalized within 10 days prior to initial presentation with CAP; (2) clinical diagnosis of AIDS or known positive antibody titre for HIV; (3) definitive diagnosis other than pneumonia that was the likely explanation for the pulmonary infiltrate (e.g., pulmonary edema or pulmonary embolus); or (4) previous enrollment in the cohort study.

During the study enrollment period, 4002 individuals who satisfied all the criteria for study eligibility were identified, of whom 2287 (57.1%) were enrolled. Based on chart review, hundreds of data items were collected for each of the 2287 patients. In the research reported here, we used 158 clinical variables in the PORT database that are often available just before the admission decision, including demographic information, history and physical examination information, laboratory results, and chest X-ray findings.[2] Variables that are continuous typically have associated discretized variable versions (e.g., age appears as a continuous variable and as a discrete variable containing six age ranges), leading to a total of 196 variables. Appendix A contains a list of the 196 variables. Patients with records in the database were followed prospectively to assess their vital status and a variety of outcomes 30 days after the radiographic diagnosis of pneumonia. A patient was considered to have experienced a dire outcome if any of the following occurred to the patient: (1) death within 30 days of presentation, (2) an initial ICU admission for respiratory failure, respiratory or cardiac arrest, or shock, or (3) the presence of one or more severe complications. Appendix B contains additional details about the criteria used to define dire outcomes.

3. Predictive models

This section describes the predictive models that were applied in the experiments reported here, as well as how the models were induced from the PORT CAP database. Table 1 gives a short summary of all the models. While there are many alternative machine-learning techniques to the ones in Table 1, we believe these methods represent a diverse sample of capable machine-learning methods. The remainder of this section provides additional details about the methods we used.

Throughout this section, the term *feature* denotes a variable-value pair. Thus *age* is a variable, while *age* = 80 is a feature. Also, f denotes an arbitrary variable, while f' denotes an arbitrary variable-value pair.

3.1. The Simple Bayes (SB) model

This section describes the Simple (aka Naïve) Bayes model and how we constructed several such models.

3.1.1. Summary of the Simple Bayes model

The Simple Bayes classifier (SB) [8–10] is based on the well-known version of Bayes' rule in which findings are assumed to be conditionally independent given a patient state.

Given a set of patient features $F' = \{f'_1, \ldots, f'_n\}$ we can compute the posterior probability of a patient state S' as follows (under the assumption of independence of the findings given the state of S):

$$P(S'|F') = \frac{P(S')\prod_{f' \in F'} P(f'|S')}{\sum_S P(S)\prod_{f' \in F'} P(f'|S)},$$

where an accented symbol indicates that a variable or set of variables has a particular state to which they are instantiated. The summation is taken over all possible states of S. Fig. 1 graphically depicts the conditional independence of the finding variables given the patient state variable S.

Often, the conditional probability $P(f'|S')$ is estimated as $(N_{f',S'} + a)/(N_{S'} + b)$, where $N_{f',S'}$ is the number of patient cases containing both finding f' (e.g., cough = yes) and state S' (e.g., *dire outcome* = yes), $N_{S'}$ is the number of cases containing state S', and the parameters a and b are positive real numbers that act to smooth the probability estimate when $N_{S'}$ is small. Similarly, the prior probability $P(S')$ can be estimated as $(N_{S'} + c)/(N + d)$, where N is the total number of patient cases, and c and d are positive real numbers. The values we used for a, b, c, and d are described in the next section.

3.1.2. Induction of Simple Bayes models without variable selection (SB.D and SB.C)

We constructed simple Bayes models that include all the available variables. We built one model based only on discrete variables (SB.D) and another based on continuous and discrete variables (SB.C). The SB.D model contains a total of 158 predictors, obtained by removing the 38 continuous variables. Among the 38 continuous variables, all but three (AGEPRES, DYMDSYM, and MENTSTAT) have associated discretized variables.[3]

[2] Information that we used on vital signs and laboratory results represent the first values available to physicians after patient presentation.

[3] For our purpose, the *COPD Severity* variable (COPDSEV) was treated as a discrete variable, since it has only seven values.

Table 1
A brief description of the predictive models evaluated in this paper

Model abbreviation	Description
SB.D	A simple Bayes model that uses just the 158 discrete variables.
SB.C	A simple Bayes model that uses all 161 database variables, including 35 continuous variables.
SB.VS.D	A simple Bayes model that uses 46 variables selected based on a greedy search procedure that attempted to find the most predictive set of discrete variables.
FM.D and FM.C	Finite mixture models containing 158 discrete (FM.D) or 161 continuous and discrete (FM.C) variables.
FAN.D and FAN.C	A model that combines a simple Bayes model with a finite mixture model. One version contains 158 discrete variables (FAN.D) and the other contains 161 continuous and discrete variables (FAN.C).
RL.BS	A rule-based system that uses the training set to tune various parameter thresholds that influence the rules being learned. All 196 database variables are used.
LR.DIRE	A logistic regression model that was developed using the PORT CAP database. It uses 102 database variables (when trained on 1601 cases), with some of these variables continuous and some discrete.
NN.STL	A neural network model constructed using traditional backpropagation methods. All 196 of the PORT CAP database variables are included.
NN.MTLR	A neural network model constructed using two new techniques. One technique involves learning to predict dire outcomes by learning also to predict related outcomes. The other technique involves focusing on a learning measure that is directly related to the area under the ROC curve. All 196 database variables are used.

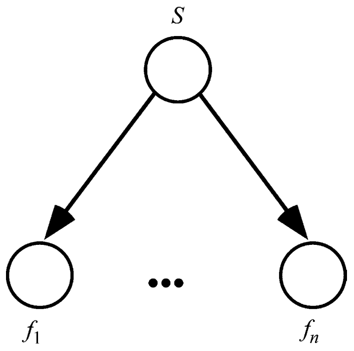

Fig. 1. In the Simple Bayes model, the variables $f_1,\ldots,f_n$, which represent patient findings, are assumed conditionally independent given the patient state variable S.

The SB.C model contains a total of 161 predictors, obtained by removing the discretized versions of 35 continuous variables.

For parameter estimation for discrete variable f, we used a equal to 1 and b equal to the number of states of f. For estimating $P(S)$, we used c equal to 1, and d equal to 2, since the *dire outcome* variable has two states. In SB.C, continuous variables were modeled using conditional Gaussian distributions [11].

3.1.3. Induction of a Simple Bayes model using variable selection (SB.VS.D)

The system was applied to the entire set of variables, including continuous and discrete. Continuous variables were not modeled on a numeric scale, but represented as a single discrete state per value for each unique real value.

We performed variable selection by dividing the training data into two smaller sets called train–train (70%) and train–test (30%). We used a wrapper approach [12] that searched over the train–train cases for the subset of variables that give the best performance on the train–test cases, given that the probabilities were estimated using the train–train cases. Since an exhaustive search through the entire power set of the training variables is computationally infeasible, we performed a forward stepping greedy search [12]. That is, at each step of the search we added the one variable that improved the performance the most on the train–test set, until adding any single variable did not improve performance. Performance was measured as area under an ROC curve.

The model selected 46 variables for prediction. The entire set of variables included (in order of significance) is AGEPRES6, CPCO2, CBUN, ALERT, ALB, COUGHY, O2SATC, CHGB, HTNA, CVDA, INTUBATE, CBPSYS, HEADACHY, COPDICUA, NOPNEPIS, PTEDUC, CTEMPCCO, FEVERY, COBEFORY, CXRINF, LASTDCDY, PULDULL, ASPEVENT, LUNGOUTA, ASPLENA, FIO2ABG, FLUSHOT, SEX, SWEATSY, CXREFFBL, FLU, UNSBLANA, CPRESTIM, CURSPUTY, PRIATBRT, VALVDISA, DMA, DIARRHEY, PTHISP, PNHOSPNO, ALC, NEUTA, MENT4, BEFCPBRY, PSYCHDXA, and PULFREM.[4]

Estimation of conditional probabilities was done using parameter values $a = 0$ and $b = 0$; if $N_{S'} = 0$, then the conditional probabilities were taken to be uniform (e.g., if f' is the state of a binary variable, then $P(f'|S') = 0.5$). Similarly, estimation of prior probabilities was done using $c = 0$ and $d = 0$; if $N = 0$, then the prior probabilities were taken to be uniform.

[4] See Appendix A for the meaning of these variables.

3.2. Finite Mixture (FM) and FAN models

This section describes the Finite Mixture and the FAN models, as well as how we constructed several such models.

The finite mixture (FM) model [13–15] is obtained by using a discrete latent variable to model dependencies among the findings and between the findings and the outcome variable. That is, conditioned on the latent variable, all the observable variables (findings and the outcome) are assumed independent. A graphical representation of a FM model is similar to the one for the Simple Bayes model, with the only difference that in a FM model the common parent is the latent variable H rather than the outcome variable S, which is being predicted (Fig. 2). The most important practical difference with the simple Bayes model is that in FM the findings generally are not restricted to be conditionally independent given the predicted variable S.

Learning an FM model from data consists of two steps: (1) the determination of the number of values of the latent variable, referred to as the cardinality of the model; and (2) the estimation of the relevant prior and conditional probabilities, in particular, the estimation of the prior probability distribution of the latent variable, and of the conditional probability distributions of each finding and of the outcome variable, given the latent variable.

The determination of the model cardinality is the most difficult computational step. We can formulate it as a search problem, whereby a scoring function over model cardinality of FM is defined, and a search for the model cardinality that maximizes the given score is performed. A well established class of scores is based on asymptotic approximations of the marginal likelihood of the model, defined as the probability $P(D \mid M)$ of the training set D given the model M. Intuitively, searching for the model that maximizes the marginal likelihood corresponds to searching for the model that best "explains" the data. Scoring functions that can be used include: (i) the Bayesian information criterion (BIC); (ii) the Akaike information criterion (AIC); and (iii) the Cheeseman–Stutz score (see [16] for a detailed description of these scoring methods). These scoring functions can be viewed as different versions of "penalized likelihood," since they have the general form

$$P(D \mid M) = P(D \mid \hat{\theta}, M) - penalty,$$

where $\hat{\theta}$ is the maximum likelihood estimator of the parameters of the model, and *penalty* can in general be interpreted as a term that penalizes model complexity with its particular form differing among the various scoring functions.

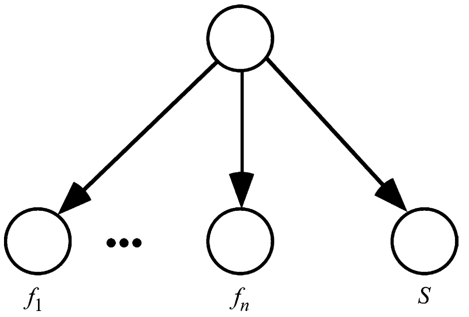

Fig. 2. The Bayesian network structure used to represent a finite mixture model, where H is a latent variable, S is an outcome variable, and $f_1, \ldots, f_n$ are variables that represent patient findings.

The exact estimation of the parameters of the model is computationally intractable because of the presence of the latent variable. Therefore, approximate methods need to be used. We use the Expectation Maximization (EM) algorithm for this task [17,18]. The EM algorithm is an iterative algorithm that is guaranteed to converge to a local maximum. The EM algorithm formalizes a quite intuitive idea. Starting from some initial parameterization of the model: (i) the values of the latent variable are replaced by their expectation according to the probability specified by the model; and (ii) the new parameters are estimated based on the "complete" data obtained by assuming that the missing data are given by their estimated values. These two steps are repeated until convergence to a local maximum is reached. Notice that at each step, since (artificially) complete data are available, parameter estimation for FM is the same as parameter estimation for the Simple Bayes model.

3.2.1. Summary of a model that combines the Finite Mixture and simple Bayes models (FAN)

The Finite Mixture Augmented Naive Bayes (FAN) model [11] is obtained by superimposing a finite mixture model on the set of findings of a simple Bayes model (Fig. 3). That is, given a Simple Bayes model defined over the outcome variable and the findings, a latent variable is introduced to model the residual probabilistic dependencies between the findings that are not captured by the outcome variable. At the same time, in an attempt to improve over the FM model, the FAN model reduces the burden on the latent variable by modeling part of the dependencies among findings through the outcome variable. Notice that the Simple Bayes model is subsumed by the FAN model, since it corresponds to a FAN model with a one-valued latent variable.

Parameter estimation for a FAN model, as well as the selection of the latent variable's cardinality, are very similar to the corresponding tasks for the FM model. The only difference is that in the FAN model there are two parent variables, the latent variable and the outcome variable, on which to condition the probability estimates of the findings.

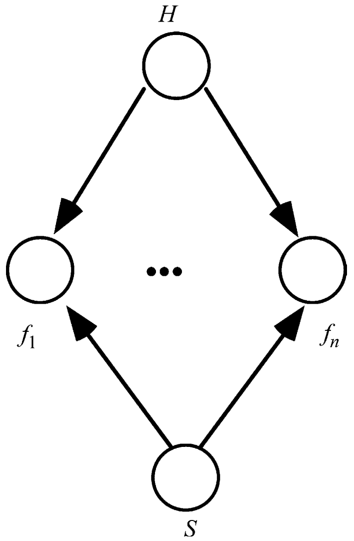

Fig. 3. The graphical representation of a FAN model.

3.2.2. Induction of the Finite Mixture (FM) and FAN models

In a manner parallel to the SB.C and SB.D model induction described above, both continuous and discrete versions of FM (FM.C and FM.D) and FAN (FAN.C and FAN.D) were constructed. In all the experiments, model selection (i.e., determining the number of values of the latent variable) was based on a greedy step-forward search that used BIC and AIC scores for ranking models. In particular, given the set of models considered in the search, the models were ranked according to the BIC score and the AIC score, and the model that had the best average rank over the two scores was selected.

3.3. Rule-based learning with bias search (RL.BS)

This section describes a method for rule-based learning with bias search (RL.BS), as well as how we applied it.

3.3.1. Summary of rule-based learning with bias search

This section describes a rule learning system that searches over various learning parameters in attempting to find a set of rules that has good predictive performance. In the machine learning field, such a procedure is known as *bias search*. We performed bias search using version 5.5 of the RL rule learning system [19,20].

The rules generated by RL are of the form: "if f_i' and…and f_j' then predict the patient state as S'." If a test case satisfies all of the features on the left-hand-side of the rule (the *if* part), then the rule predicts the class specified by the right-hand-side of the rule (the *then* part). As an example, the following rule predicts that a patient will not experience a dire outcome:

$$\text{if}(age < 23) \text{ and } (pO_2 > 48) \text{ then predict } \textit{dire outcome} = \text{no}.$$

Rule strength is determined by a certainty factor (CF), which is the following smoothed estimate of the positive predictive value of the rule: $(\text{TP} + 1)/(\text{TP} + \text{FP} + k)$, where TP is the number of true positive predictions of the rule on a training set, FP is the number of false positive predictions, and k is the number of values of the variable being predicted (e.g., $k = 2$ for the binary *dire outcome* variable).

RL selects rules using the following three criteria: (a) the CF must be greater than the *CF-thresh* threshold (e.g., CF-thresh = 0.6), (b) the number of cases covered by the rule must exceed a threshold called *min-POS*, and (c) every new rule must cover at least one new case not covered by previous rules.

RL-bias-search (RL.BS) is an extension of the RL system. RL.BS randomly divides the training data into train–train (~67%) and train–test (~33%) datasets. The train–train dataset is used to learn the rules using a selected parameter setting (i.e., bias) and the train–test is used to evaluate the performance of a learned rule set. RL.BS explores various combinations of the *CF-thresh* and *min-POS* settings in search of the settings that optimize the predictive performance of the induced rule set; the optimization criteria are discussed in the next paragraph. The user specifies the minimum and maximum values for *CF-thresh* and *min-POS*, as well as the bias step size, which is added to or subtracted from the *CF- thresh* and *min-POS* values of a selected pair of settings to create new bias settings to evaluate. We used a simulated-annealing algorithm [21] to generate new parameter settings; for additional details about this approach, see [22]. The search yields parameter settings that then are used to learn a final rule set from the entire set of training cases.

The final rule set is used to make a prediction for a case using a method called *weighted voting*. In weighted voting, the rules matching a patient case are grouped according to the state predicted by the rules (thus, for a binary prediction problem, there will be two groups). Then, the strengths of the rules (the rules' CFs) of each group are summed; each group's sum is the amount of evidence provided for the state corresponding to the group (therefore, for a binary prediction problem, this yields two sums). The state with the largest amount of evidence is then predicted as the class for that case. For a given patient case, the probability of its state is calculated as the evidence for the state of interest divided by the total of the evidence given for all states. For example, for a binary prediction with the states $S = \text{A}$ and $S = \text{B}$, if the sum of the CFs of the rules predicting state A is 350 and the sum of the CFs of the rules predicting state B is 150, then the evidence for class A is 350, the evidence for class B is 150, and the estimate of $P(S = \text{A}|\text{case findings})$ is $350/(350 + 150) = 0.7$.

3.3.2. Induction of a rule-based model using bias search (RL.BS)

All 196 of the pneumonia PORT variables were made available for rule learning. For all of the experiments, we used a parameter step size of 0.02. A parameter setting was evaluated by running RL on the train–train dataset using the setting, then evaluating the performance of the rule set induced by RL on the train–test dataset. Using the settings that yielded the best performance on the train–test dataset, a final model was induced from the entire training set. This rule set was then applied to the test set to evaluate the performance of the rules.

Using the training set containing all 1601 cases, the induced rule set consisted of 35 rules that used the following 22 variables, which are listed in alphabetical order: AGEPRES6, AGEPRESB, BPSYS, CBUN, CGLU, CHCO3, CHGB, CNUMCOMO, CPCO2, CPH, CPO2, CR, CRESPRAT, FIO2ABG, HEADACHY, KP, NUMSYM, PO2, PTEMPLOA, RTEMP, SOXYGEN, and SWEATSY.

3.4. Logistic regression model

This section describes logistic regression models, as well as how we constructed a particular model for use in the studies reported here.

3.4.1. Summary of logistic regression

Logistic regression derives an equation of the following form:

$$P(S) = \frac{e^{(\beta_0+\beta_1 f_1+\beta_2 f_2+\cdots+\beta_n f_n)}}{1 + e^{(\beta_0+\beta_1 f_1+\beta_2 f_2+\cdots+\beta_n f_n)}},$$

where S is a binary variable to be predicted, and $f_1,f_2,\ldots,f_n$ are discrete or continuous predictor variables [23]. 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 products of the f_i variables.

3.4.2. Application of logistic regression to predict dire outcomes (LR.DIRE)

LR.DIRE is a logistic regression model that we trained on the PORT database in a way similar to how the logistic regression model underlying the PSI model was developed by Fine et al. [7] using an earlier CAP database. Thus, LR.DIRE is closely related (although not identical) to what would result if a PSI model were constructed based on the PORT database.

The logistic regression models developed for the experiments in this paper were all constructed using a two step process. For each experiment, the predictor variables were each run in a simple univariate regression on the training dataset. Significant variables, defined as those with p values of 0.05 or less for an F test, were then entered into a multivariate logistic regression, which also was built using the training set. We did not include any interaction terms. To avoid strong covariate effects, variables measuring similar concepts were removed based on the following rule: if a continuous variable and its categorical counterpart were both significant in the univariate analysis, the one with the smaller p value was used for the multivariate run; in cases where both variables had the same value, the continuous version was used.

Forward stepwise logistic regression was used for all experiments. The analysis employed BMDP Dynamic, v7.0 running on Windows NT 4.0. For the experiment using 1200 training cases, the default BMDP settings were used [24]. For experiments using 400, 800, and 1601 cases, some of the default parameter settings were changed. These changes involved the p value limits for entry and removal of terms into the model, and the total number of times a term could be moved into or out of the model (as well as some other minor changes). The changes are noted below:

- Following the PSI model, AGEPRES (age of presentation) and SEX were included the model at the start of the forward stepping process and were not allowed to be removed.
- All other terms (besides AGEPRES and SEX) were allowed to move a maximum of two (2) times—either entered or removed from the model a total of no more than 2 moves.
- The following syntax was added to the "*/Regression paragraph*": ENTER = 0.05, 0.04. REMOVE = 0.06, 0.05.

In addition to these changes, for the experiment with 1601 cases certain categorical variable codings were reordered in the syntax such that baseline or "normal" values were represented by the initial entry. This was to insure BMDP was comparing non-normal values to the appropriate baseline value, since the software denotes the initial entry as the "normal" entry.

For the remainder of the experiments, the exact parameters and methodology were used as listed above (default settings plus three alterations).

The variables considered significant from pass 1 (univariate regression results, not shown here) were then entered into the multivariate regression for pass 2, which was run using a training set. The results of pass 2 for the training set containing 1601 cases resulted in the following 102 variables being included in the model:

Discrete variables ($n = 86$):

FEVERY, RIA, COPDICUA, MUSKELA, NUTRSTAA, RTEMP, ALERT, CPOLYS, CHGB, PTE-

DUC, CNA, O2SATABG, CPO2, SOXYGEN, CXRLOBES, CXRINFBL, COUGHY, SPUTBFOY, SWEATSY, CONFSDYY, MYALGIAY, CADA, CVDA, CCOPDSEV, MIA, DNRA, PTLIVLOA, MENT4, CBANDS, CPLT, CCR, O2SATC, CPH, FIO2ABG, CO2RETEN, CXRCLOBE, COBEFORY, HEADACHY, COPDA, CHFA, NTCAGEA, SVENTARA, DEMENTA, CONFUSA, PTEMPLOA, FLUSHOT, CRESPRAT, PULCLEAR, CHCT, CHCO3, FIO2POX, FIO2ABGA, CXREFF, BFPNSOBY, CHILLSY, VENTARRA, BEFFATGY, ASPEVENT, PRIHOSPA, CBPSYS, INTUBATE, CPULSE, CXREFFBL, HTNA, PULDULL, PTMRSTAA, CPY, PULCYAN, CWEIGHT, DMA, THROTSOY, CPNHOSPN, SEX, PULRALES, PULDEC, FEV15A, PULRHONC, ACTIVCAA, CANCERA, PTRACEA, STEROIDA, CBPDIAS, HOMEO2A, CXRINF, UNSBLANA, PULBRONC.

Continuous variables ($n = 16$):

LASTDCDY, RESPRATE, KP, O2SAT, AGEPRES, NUMCOMOR, WBC, GLU, SGOT, ALB, NUMSYM, BUN, ALKPHOS, PCO2, LDH, TEMPCCOR.

3.5. Artificial neural networks

In this section, we briefly describe the standard feed forward neural network model, as well as extensions we used in the study reported here. We then describe how we applied both the standard method and the newly developed ones.

3.5.1. The standard neural network method (STL)

The standard approach to applying artificial neural networks trained with backpropagation to problems such as pneumonia risk prediction is to use sum-squared-error (SSE) with "0" coding for "no dire outcome" and "1" coding for "a dire outcome" as the target values at the output of the network. This is supervised learning where the network learns to predict values near zero for patients at low risk, and values near 1 for patients at high risk.

Given a large training set, this method learns to predict the *probability* of dire outcomes (instead of the Boolean 0/1 values it is trained on). When the training set is small, however, the method will generally overfit the training set and learn to output values near 0 and 1 for the cases it is trained on. This yields low error on the training set, but generalizes poorly to new cases in a test set.

Early stopping is a method that stops training the neural network before it begins to overfit the training data. This is done by examining the performance of the network during training on an independent set of cases. When performance on this set begins to worsen (instead of continuing to improve), we assume the backprop network has begun overfitting the training data and we stop training.

Early stopping requires that part of the training data (the train–test dataset) not be used for training and be held aside for testing to detect overfitting. This is unfortunate because most learning methods train better with more training data. Using a smaller training set (the train–train dataset), because some of the training data must be held aside to detect overfitting, can reduce the performance of the trained backprop networks. We use a simple form of model averaging to help mitigate this effect. Suppose we have 1601 cases to use for training. We use 75% of this data for training (the train-train dataset), and 25% as the early stopping test set (the train–test set). Thus we train the network on 1201 cases and test it on 400 cases to determine when to stop training. A network trained this way is only trained on 1201 cases, not the full 1601 cases available for training. To reduce the negative impact of using a smaller training set, we train 10 networks on training sets of size 1201 instead of just one. Each network uses a different random sample of 1201 (from the 1601 available) cases as a train–train set (and thus also different samples of 400 cases as the train–test set.) After training all ten networks, each on somewhat different train–train sets, we combine the predictions of the ten networks by averaging their predictions. This means that the number of parameters in the final model is ten times the number of parameters in any one backprop network.

3.5.2. The rankprop technique

In this and the following sections, we describe two new neural network learning techniques that we developed and combined [25].

In the experiments reported in this paper, the performance criterion is the area under the ROC curve. ROC area is a sort-based measure. It depends only on the ordering it induces on the data, not on the particular value it predicts for any one case. But the sum-squared-error we used in the previous section is sensitive to the values predicted for each case, not the ordering induced on the data. We developed a method called *Rankprop* that learns to rank patients by relative risk instead of predict specific risk values for them.

The term *Rankprop* is short for "backpropagation using sum of squares errors on estimated ranks." The basic idea of Rankprop is to rank (i.e., sort and then number consecutively) the cases in the train-train set based on the outcomes predicted, scale the ranks (we scale uniformly to $[0.25, 0.75]$ with sigmoid output units), and apply standard SSE backpropagation using scaled ranks as target values instead of using the 0/1 values in the train–train data.

Ideally, we would rank cases by the true frequency of an outcome, such as *dire outcome*. Unfortunately, we do not know the true frequencies. During training, all we know is which patients in the training set had dire outcomes. There are many possible sorts consistent with these values. Which should backprop try to fit? It is the large number of possible sorts of the train–train set that makes backpropagating ranks challenging. Rankprop solves this problem by using the neural net's model as it is being learned to order the train–train set whenever target values are tied. In this database, where there are many ties because there are only two target values, finding a proper ranking of the training set is a serious problem. Rankprop learns to adjust the target ranks of the dataset at the same time it is learning to predict ranks from that dataset.

To do so, Rankprop alternates between rank passes and backprop passes. On the rank pass it records the output of the network for each training pattern. It then sorts the training patterns using the target values (0 or 1 for *dire outcome*), but using the network's predictions for each pattern as a secondary sort key to break ties. The basic idea is to find the legal rank of the target values (0 or 1) maximally consistent with the ranks the current model predicts. This closest matching ranking of the target values is then used to define the target ranks used on the next backprop pass through the training set.

Why might Rankprop be useful in learning a model? We are given data from a target function $f(x)$. Suppose the goal is not to learn a model of $f(x)$, but to learn to sort patterns by $f(x)$. Must we learn a model of $f(x)$ and use its predictions for sorting? No. It suffices to learn a function $g(x)$ such that for all x_1, x_2: $[g(x_1) \leqslant g(x_2)] \Rightarrow [f(x_1) \leqslant f(x_2)]$. There can be many such functions $g(x)$ for a given $f(x)$, and some of these may be easier to learn than $f(x)$. Rankprop tries to learn simple functions that directly support ranking. One difficulty with this approach is that rankprop must learn a ranking of the training data while also training the model to predict ranks. It is not yet known under what conditions this parallel search will converge. We conjecture that when rankprop does converge, it will yield simpler models than it would have been learned from the original target values (0/1), and that these simpler models will often generalize better.

Another way of looking at this is to consider what a traditional neural network trained with SSE on 0/1 targets tries to learn. If we were predicting mortality, for example, it would attempt to drive the mortality prediction for every patient who lived to a value of 0, and every patient who died to a value of 1, regardless of their (unknown) probability of death. Now compare this with rankprop on the same database. Assume 90% of the patients lived and 10% died. Consider the patients who survive. Rankprop does not have to drive all patients that live to one fixed value. Instead, it has to find some ordering of the patients that live. This means it is possible that the patients who live and have low probability of death will be sorted to the left of patients who live but have high(er) probability of death. The same is true for patients who die. If such orderings can be found by rankprop, the function that is to be learned should be less nonlinear than the function that would have to be learned by SSE on 0/1 targets.

3.5.3. The multitask learning technique

There are a number of attributes available in historical databases that are not suitable for use as inputs. For example, the pneumonia PORT database contains variables such as the *total cost of hospitalization and the length of stay*, as well as the values of the variables that define *dire outcome* such as *ICU admission for respiratory failure*. It is inappropriate to use these attributes as inputs when learning to predict risk because these values would never be available for future test cases when risk prediction is to be done. They represent future measurements.

Although these variables will not be available when the model is used, it is possible to use the values of these variables in the training set in ways that do not assume that they will be available in the test set. Multitask learning (MTL) is one method of doing this. Multitask learning is a method that improves generalization performance by having a learner simultaneously learn many extra related tasks at the same time it learns to make predictions for the main task. It does this parallel learning while using a shared representation; what is learned for each task can benefit other tasks because they share what is being learned.

In this application, we use MTL to benefit from future lab results and from any other future information we have available about the training cases that could be related to the dire outcome prediction task. These extra attribute values are used as extra backprop outputs as shown in Fig. 4. The extra outputs bias the shared hidden layer towards representations that better capture important features of the domain. See [25–27] for details about MTL and [28] for other ways of using extra outputs to bias learning.

It is beyond the scope of this paper to explain in detail how MTL works. An example, however, will give an intuition for what MTL does. The pneumonia PORT database that we used contains an attribute that gives the length of stay in the hospital (in days) for patients who are hospitalized. In learning to predict the probability of a *dire outcome* in patients with pneumonia, we use this *length of stay* variable as an extra task for multitask learning. Let us examine why learning to predict *length of stay* might improve the ability to predict a dire outcome as well. In general, we expect patients with low

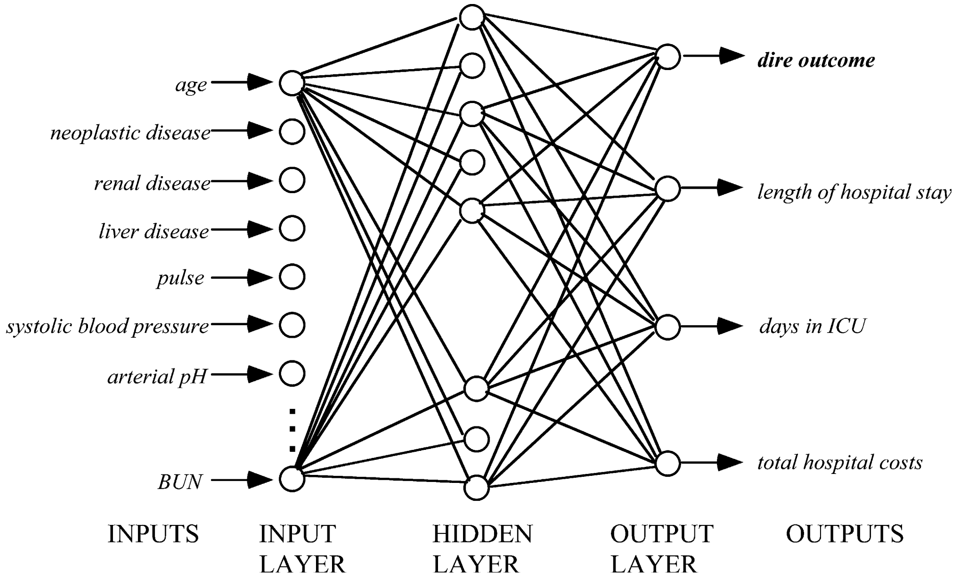

Fig. 4. A hypothetical example of a neural network that can be applied to predict dire outcomes. The network is trained with outcomes that are plausibly related to a *dire outcome*, including in this example: length of hospital stay, days in the ICU, and total hospital costs.

risk of a dire outcome to not experience a dire outcome (*dire outcome* = 0), and patients at high risk to have *dire outcome* = 1. But some patients that are high risk will have *dire outcome* = 0, and some patients that are low risk will have *dire outcome* = 1. This happens because *dire outcome* is the outcome itself, not the underlying probability of the outcome. Unfortunately, we have no direct measure of a patient's probability of *dire outcome*. We only see the Boolean outcome.

This is where learning to predict *length of stay* can help. A patient's *length of stay* is related to their risk. A patient who stays in the hospital for a long time presumably is at greater risk than a patient who stays in the hospital a short time (ignoring patients who die after a short stay). If we train an artificial neural network to predict *length of stay*, some of what it learns may help it create intermediate assessments of patient risk for a dire outcome. Since the network shares what it learns for the *length of stay* task with the main risk prediction task (*dire outcome*), some of this shared assessment may help the network to learn to better differentiate intermediate risks of *dire outcome*. This learning of related tasks, which can inform the main task, is the central idea behind multitask learning.

3.5.4. Application of the STL and MTLR methods

We applied two different neural network methods to the experiments with the PORT database. The first the *standard learning method* (STL) uses traditional sum-squared-error on 0/1 targets. The second method uses Multitask and Learning along with Rankprop learning (MTLR). For MTLR, we used the following variables as extra outputs for neural network training:

type of care (outpatient versus inpatient)
whether detailed baseline data were collected (y/n)
whether proxy respondent provided baseline data (y/n)
regression model risk based on baseline data
3-level risk score, derived from the regression model
5-level risk score, derived from the regression model
count of cardiologic morbid complications
count of central nervous system morbid complications
count of dermatologic or allergic morbid complications
count of gastrointestinal morbid complications
count of hematologic morbid complications
count of liver morbid complications
count of pulmonary morbid complications
count of renal morbid complications
count of suppurative morbid complications
count of total morbid complications
discretized count of total morbid complications
count of total severe morbid complications
count of bothersome symptoms
count of missing values of bothersome symptoms
ICU care or telemetry
reason for ICU admission
length of stay for hospitalized patients
discrete version of length of stay
total patient hospital cost
discretized version of total patient hospital cost
days to return to daily household activities
discretized version of days to return to daily household activities

With both methods we trained on 75% of the training data and used the remaining 25% of the training data for early stopping. To reduce the negative impact of training on less data, we average the predictions of ten networks, each of which was trained on a randomly chosen 75% sample of the training set. In all experiments we used all 196 variables as inputs.

We trained the networks with standard gradient descent using the Aspirin/Migraines Neural Network Simulator developed by Mitre Group. The backprop networks we used for all our experiments have 64 hidden units, sigmoidal hidden and output units, and are fully connected at the input-to-hidden and the hidden-to-output layers. There were 15,104 weights per network. A momentum of 0.9 and learning rate of 0.1 were applied. Networks were trained a maximum of 100 epochs, but early stopping often selected models trained with 25–75 epochs.

4. Experimental methodology

In this section, we describe the training and test datasets we used, as well as the evaluation measures.

4.1. Training and test datasets

A training (derivation) set of 1601 patient cases (70%) was created by randomly sampling from the 2287 CAP patients in the pneumonia PORT database. The following size training subsets were created by using the first *n* cases from a list of the 1601 cases: 100, 200, 400, 800, 1200, and 1601. Missing data were filled-in using an iterated *k*-nearest neighbor method, which is described in Appendix C.

A test (validation) set of size $2287 - 1601 = 686$ was used for the evaluation. For each of the six training sets, the method used to fill in missing data in that training set also was used to fill-in data in a corresponding test set. Thus, there were six versions of the test dataset, corresponding to the six training sets mentioned above.

4.2. Evaluation measures

The primary measure we used in comparing the 11 models described in Section 3 is the area under the ROC curve, which is a common measure for comparing the discriminative performance of models [29–31]. An area of 1 corresponds to perfect prediction of dire outcome for the cases in the test set. An area of 0.5 corresponds to random guessing of the probability of dire outcome for each case in the test set.

We analyzed in detail the performance of NN.MTLR, which had the highest ROC curve area among the 11 models. To assess performance differences at particular points on the ROC curve, one of us (author MJF, whose clinical research focus includes CAP) assessed a range of probability thresholds of dire outcomes that he believes to be clinically relevant in influencing decisions about where to treat CAP patients. This range spanned from 1 to 5%. We identified a point on the ROC curve for NN.MTLR that was close to the 1% threshold; more specifically, at that point on the curve, for the test set of patient cases recommended by NN.MTLR for home treatment, about 1% (more precisely, 1.3%) experienced dire outcomes.[5]

5. Results and discussion

We first describe the predictive performance of the models we constructed for this study. Next, we analyze the best performing model.

5.1. Predictive performance of the constructed models

Of the 686 CAP patients in the test set, 79 (11.5%) experienced a *dire outcome*. Table 2 subdivides the 79 cases to show more specifically why the dire outcomes occurred. The table indicates that the most common cause of a dire outcome was admission to the ICU (for respiratory failure, respiratory or cardiac arrest, or shock/hypotension). Only 20% of the dire outcomes were due to death alone, while an additional 23% were due to death plus another dire event. Fifty-seven percent of the dire outcomes were for reasons that did not include death. The fact that the majority of the dire outcome cases had a non-mortality outcome suggests that predicting mortality is not a good surrogate for predicting dire outcomes more broadly. It seems likely that a clinical decision about where to treat a CAP patient (inpatient versus outpatient) would be influenced by concern about multiple types of clinical dire outcomes, not just death. This line of reasoning supports the importance having a model that predicts dire outcomes more broadly than mortality.

In Table 3, for each system and training set size, the area under the ROC curve is shown. For 1601 training cases, the bold entry for NN.MTLR indicates that it has the highest ROC area. For the models constructed from 1601 training cases, an asterisk indicates that a comparison of its ROC area with NN.MTLR has a two-sided *p* value that is less than 0.05, suggesting that NN.MTLR performed better. We used the method described by Hanley and McNeil to perform these statistical tests [32].

Fig. 5 shows the models from Table 3 that have ROC curve areas that are not statistically different (at the 0.05 level) from those of the best model (NN.MTLR). Fig. 6 shows the results of the remaining methods from Table 3 that are statistically significantly worse than

[5] We assume that CAP patients with a probability of death (at 30 days) below 1.3% would likely be treated at home, unless there were extenuating circumstances.

Table 2
The 79 cases of dire outcomes in the test set are subdivided into seven different classes

Severe complications	Death	Admission to ICU	Number of patient cases
No	No	Yes	27 (34%)
No	Yes	No	16 (20%)
No	Yes	Yes	3 (4%)
Yes	No	No	14 (18%)
Yes	No	Yes	4 (5%)
Yes	Yes	No	12 (15%)
Yes	Yes	Ues	3 (4%)

Each class describes a joint set of conditions that constitute a dire outcome.

NN.MTLR. Interestingly, four of these five models do not improve monotonically as a function of the training set size. The reasons for these non-monotonic patterns will require further study. For example, it could be that the method of variable selection in LR.DIRE led to its decreased performance when using 1601 training cases.

Most of the 11 models are close to their peak performance after only 400 training cases. The NN.MTLR and SB.D models attain close to their best performance after only 200 training cases. The asymptotic character of the curves in Figs. 5 and 6 suggests that the addition of further training cases beyond 1601 is not likely to improve the models' predictive performance appreciably.

Table 3
The area under the ROC curve for predicting a *dire outcome* as a function of the modeling methodology and the training set size

	100	200	400	800	1200	1601
FAN.C	0.690	0.724	0.766	0.756	0.771	0.814*
FAN.D	0.831	0.819	0.805	0.820	0.838	0.849
FMM.C	0.773	0.722	0.780	0.810	0.812	0.815*
FMM.D	**0.840**	0.827	0.821	0.783	0.784	0.813*
LR.DIRE	—	—	0.818	0.829	0.828	0.774*
NN.MTLR	0.830	**0.848**	0.836	**0.862**	**0.866**	**0.863**
NN.STL	0.726	0.828	0.829	0.834	0.848	0.854
RL.BS	0.726	0.765	0.814	0.839	0.823	0.851
SB.C	0.754	0.790	0.833	0.815	0.850	0.854
SB.D	0.831	0.838	**0.843**	0.850	0.854	0.851
SB.VS.D	0.529	0.708	0.745	0.769	0.806	0.809*

Entries in bold represent the best performance for a given training set size. For 1601 training cases, asterisks indicate that the performance of a model is significantly worse (see text) than the best performing model (NN.MTLR). We were not able to obtain acceptable convergence for LR.DIRE with only 100 or 200 cases, and therefore, its performance is not reported for these training set sizes.

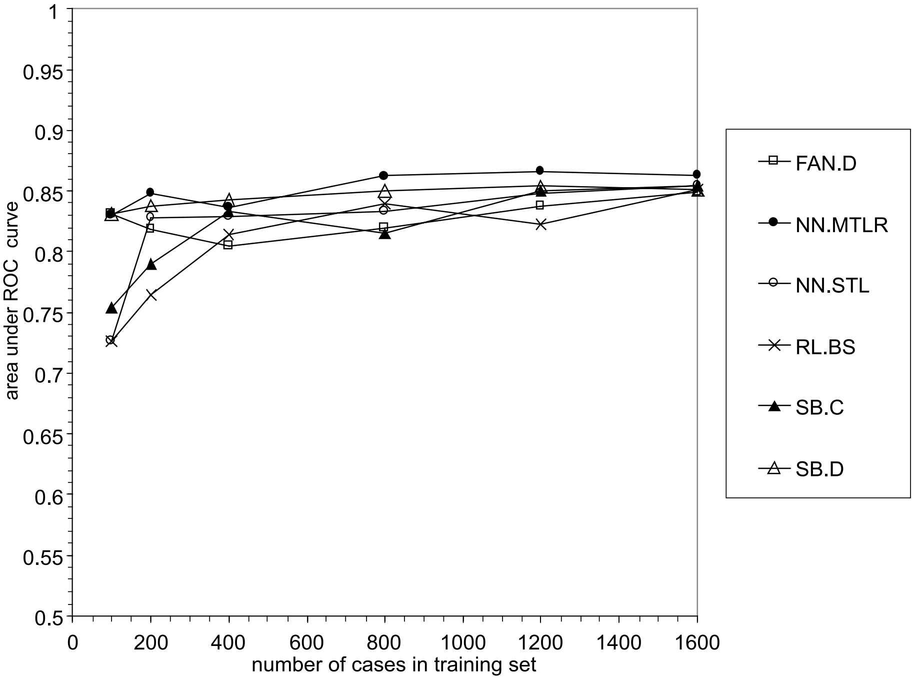

Fig. 5. The ROC curve areas for the models that perform the best when using 1601 training cases.

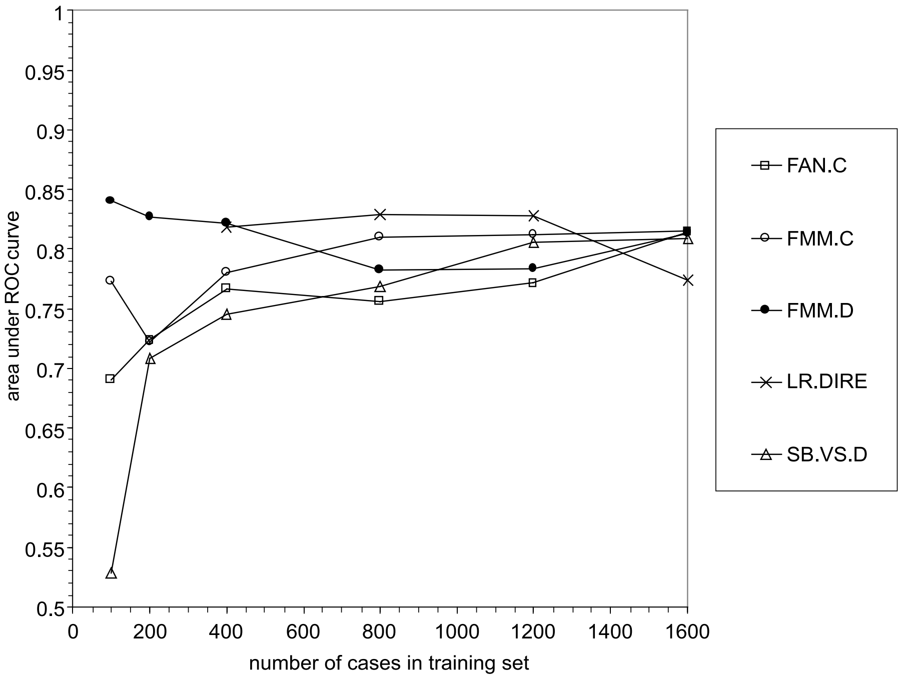

Fig. 6. The ROC curve areas for the models that perform significantly worse than the best method (NN.MTLR) when using 1601 training cases.

With 1601 training cases, the mean difference in ROC curve area between the models in Fig. 5 and those in Fig. 6 is 0.8535 − 0.8049 = 0.0486. Although this difference may seem inconsequential, we discuss below how small (but real) differences in predictive performance can have major implications for healthcare delivery.

5.2. Analyzing the predictive performance of NN.MTLR

In this section, we analyze the predictive performance of the NN.MTLR model, which for brevity we call NN. The NN model was the best predictor of dire outcomes we found when using the full 1601 training cases. We examine its predictions to gain insight into how well it might perform as an aid to clinicians making decisions about where to treat CAP patients. In particular, we are interested in estimating the impact the NN model might have on CAP patient quality of care and healthcare costs, if the decisions about where to treat the patients (home versus hospital) were based on the model's predictions. We emphasize that the results of this analysis are preliminary.

Fig. 7 shows the ROC curve for the NN model applied to the 686 test cases. This NN model was trained with 1601 cases to predict a *dire outcome*, as defined in Appendix B. The true positive rates and false positive rates for the curve are based on whether or not a *dire outcome* was predicted to occur, given whether or not a patient actually experienced a *dire outcome*. The area under the ROC curve is 0.863. In terms of clinical practicality, however, only a portion of the ROC curve is relevant to making a decision about whether to admit a CAP patient. In the remainder of Section 5.2 we focus our analysis on a relevant portion of the NN ROC curve. In doing so, for the purpose of analysis, we

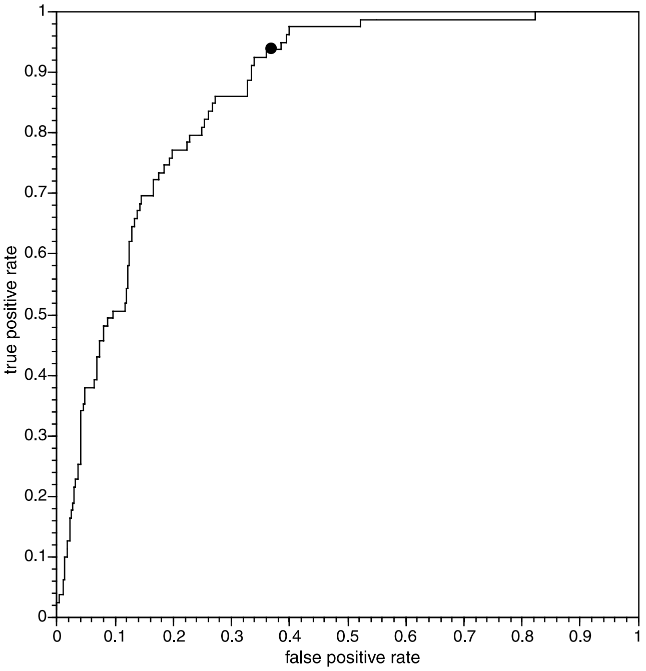

Fig. 7. ROC curve for the prediction by NN of a *dire outcome* in each of 686 test cases of patients with CAP. The large black dot indicates the point on the curve (tpr = 0.937 and fpr = 0.359) that is discussed in detail in the text.

assume that those patients predicted by NN not to have a dire outcome would be treated at home, and the patients predicted to have a dire outcome would be admitted to the hospital.

5.2.1. Predictive performance at a specific point on the ROC curve

In this section, we examine the point on the ROC curve in Fig. 7 at which approximately 1.3% of the test cases predicted by NN to *not* have a dire outcome in fact *had* a dire outcome. We call this percentage the *error rate*. At this 1.3% error rate, NN has a true positive rate (tpr) of 0.937 and a false positive rate (fpr) of 0.359 (see Fig. 7). At that point on the ROC curve, NN recommends treating at home 394 patient cases (because they are predicted not to have a dire outcome), and of those, a total of five patients (1.3% [0.17%, 2.37%]) subsequently developed a dire outcome, where the 95% confidence interval is shown in square brackets. Based on the care actually received, 280 patients were treated at home, and of those, a total of five patients (1.8% [0.25%, 3.33%]) developed a dire outcome. The difference between the 1.3% error rate by NN and the 1.8% error rate of the actual treatment is not statistically significant ($p = 0.29$). We discuss these error rates further in Section 5.2.3.

At the ROC point described in the previous paragraph, the NN model recommends admitting fewer patients to the hospital than were admitted in actual practice. In particular, the model recommends admitting 292 of the 686 test patients (42.6%), while 406 (59.2%) of the 686 cases were actually admitted to the hospital. Thus, the NN model recommended admitting about 16.6% fewer (of the total 686 cases) than were actually admitted, which is a highly statistically significant difference ($p < 0.0001$). In the next section, we estimate the cost savings that might result from such a reduction in admissions.

5.2.2. Estimated cost savings

In this section, we estimate the dollar-cost savings of using the NN to inform decisions about whether to treat CAP patients at home (outpatients) or in the hospital (inpatients). To do so, we must introduce several assumptions, which we make explicit. The validity of these assumptions, as well as the accuracy of the estimates of the performance of NN, are issues that will require further evaluation. Nonetheless, we believe the analysis below provides insight. In particular, the analysis highlights that even small improvements in predictive performance can have significant, positive healthcare consequences.

We focus our analysis on CAP patients in the US who were seen in the ED, because most of the pneumonia PORT patients (upon which we base on analysis) were seen in the ED. From 1993 to 1995, the average annual number of CAP visits was about 4,487,000, of which about 1,256,000 (28%) occurred at emergency departments [34]. Since we confine our analysis to CAP patients seen in the ED, our analysis will tend to underestimate the total national impact of a given strategy. Nonetheless, we believe that an analysis for this important subset of patients is informative.

As described in Section 5.2.1, the NN model is expected to recommend 16.6% fewer admissions (when used as a decision aid for the 1,256,000 CAP patient-cases seen in the ED) than the treatment actually used (42.6% vs. 59.2%). The estimated cost of an inpatient CAP case in the US in 1994 was $7517,whereas the highest estimated cost of an outpatient case was just $421[5]. Thus, the difference in cost is $\$7517 - \$421 = \$7096$. If the use of the NN recommendations translated into $1{,}256{,}000 \times 16.6\%$ fewer CAP admissions (than the admissions expected to occur without using a decision model), then the expected savings of following those recommendations would be $1{,}256{,}000 \times 0.166 \times \$7096 = \$1.479$ billion (in 1994 dollars).

More generally, viewed as a function of the percentage decrease in admissions, the US national cost savings would be about $1{,}256{,}000 \times 0.01 \times \$7096 = 89$ million dollars per percent decrease in hospital admissions of CAP patients seen in emergency departments. If such a decrease in admissions could be achieved while not increasing the number dire outcomes that occur in patients treated at home, the overall impact would be positive and significant. *This perspective highlights that even small (but real) improvements in predictive modeling may lead to enormous healthcare cost savings. Therefore, the development of better predictive models is an undertaking of considerable importance.*

We conclude this section with a discussion of the major assumptions on which the above cost analyses are based.

- *For each patient recommended to be treated at home by some model M according to decision threshold t, if he/she had been so treated then his/her dire outcome status would have been the same as the outcome he/she actually experienced, even if he/she was actually admitted.*

This assumption is made more tenable by the use of a decision threshold near one percent, in which only very low risk patients were being recommended for treatment at home by a model. Also, the NN model controlled for numerous patient covariates that might otherwise confound the relationship between place-of-treatment and dire outcome status. In related research, a controlled trial involving 1743 patients with CAP showed that the availability to clinicians of the output of a mortality-prediction model "reduced the use of institutional resources without causing adverse effects on the well-being of patients" [33].

We make an analogous assumption to the one above for patients recommended to be treated in the hospital.

- *Patients in emergency departments throughout the US would experience similar results to those found in this analysis, which is based on data from a small set of US and Canadian hospitals*. The patient base used here has a higher fraction of patients who were seen at academic medical centers than does the US pneumonia population at large. Attenuating somewhat this concern about possible population heterogeneity, Fine et al. [7] found that the PSI mortality risk model, which was constructed using data from a wide mix of 78 US hospitals, generalized well in predicting mortality of the 2287 patients in pneumonia PORT dataset.
- *The predictions of a model would be readily available and heeded*. It could well be that clinicians would appropriately (or inappropriately) ignore some of a model's predictions. It could also be that the treatment decision is based on multiple decision makers in addition to the treating clinician, including the patient and his or her family.
- *The financial and time costs of having and using a predictive model are negligible*. Such a model might be a part of a larger clinical information system that serves many other purposes. Thus, the incremental cost of making the model available for clinical use could be relatively small. In particular, it likely will be important that the predictor variables in a predictive model be available in electronic health records, which are populated with information as part of the normal workflow of healthcare delivery; current trends make this scenario seem plausible in the next 10–15 years.
- *There are no hidden health-quality costs in using the model*. For example, we assume that by using a model, a physician's own clinical skills would not decrease over time, thereby possibly decreasing patient quality of care.

Almost certainly, the above assumptions do not hold exactly. We therefore emphasize that the results in Section 5.2 are preliminary and suggestive, but by no means conclusive. Ultimately, the overall clinical and monetary impact of a given predictive model will need to be validated by experimental study. The preliminary analyses in this paper suggest that such studies, if carefully designed, may be well worthwhile.

5.2.3. A preliminary analysis of the impact on quality of care

Monetary cost savings are appropriate only if they do not decrease the overall quality of health care. We would like to develop predictive models that decrease cost and increase quality of care. While the present study does not address this important issue in detail, we provide the following relevant analyses.

Plausibly the most negative outcome is for a CAP patient to be sent home for treatment and subsequently experience a dire outcome. As mentioned above, of the 280 patients actually treated at home, only 5 (1.8%) experienced a dire outcome. Home treatment was recommended for 394 patient cases by NN, and also only 5 patients (1.3%) experienced a dire outcome. Thus, by this measure, the model's recommendations are not be expected to decrease the quality of health care.[6]

As mentioned, 1.8% of the patients actually treated at home experienced a dire outcome. We examined the outcomes of those patients who in reality were treated at home, but for which the NN model recommended treatment in the hospital; in doing so, we used the tpr and fpr described in Section 5.2.1, which corresponds to an error rate of 1.3%. There were 18 such cases, of which 3 (16.7%) had a dire outcome, which is a 9.3-fold increase in risk over the 1.8% risk for the total population of patients actually treated at home. These results suggest that using the NN model's recommendations might improve the quality of care for some high-risk patients who otherwise would be treated at home. It is possible, however, that some or all of these patients would prefer to be treated at home.

6. Summary and conclusions

On a training and test dataset of CAP patients, we found that there were statistically significant differences in how well different induced models predicted dire outcomes. After approximately 400 training cases, there was little improvement in the ROC area for most of the models induced by most of the methods. Thus, the results appear stable in the large training sample limit. It will be useful to validate these results by testing whether randomly re-sampling subsets of the 1601 training cases leads to similar asymptotic patterns of predictive performance.

An innovative neural network learning method induced a model (NN.MTLR) that had the largest ROC area when using all the training cases; statistically, its performance was significantly different from five models constructed using other methods, and not significantly different from five other models. Additional research will be required to determine whether other machine-learning methods that were not tested in our investigation might perform even better than this neural network model.

[6] Of the 394 patients recommended for home treatment by NN, 132 were actually treated in the hospital. We do not know what the dire outcome status of these inpatients would have been, had they been treated at home. As discussed in Section 5.2.2, we assume that the dire outcome status of these low-risk patients would have been the same with home treatment as the actual outcomes they experienced in the hospital.

Our study does not provide direct evidence for how NN.MTLR would perform on other clinical prediction problems; rather, it focuses specifically on finding a good predictor of a *dire outcome* in CAP patients. Also, additional research is needed to validate the *dire-outcome* predictive performance between NN.MTLR and other methods and to determine with confidence the reasons for those differences.

The paper presents a preliminary analysis that supports that the predictive performance of NN.MTLR could—*under assumptions*—yield a substantial decrease in the cost of healthcare delivery without any decrease in healthcare quality. The data needed to construct and apply NN.MTLR are not yet readily available in electronic health records. We believe it plausible, however, that such data will be electronically available in the next 10–15 years. As such data are increasingly captured in electronic form, the number of local patient cases available for model induction and the clinical coverage of those cases will increase dramatically. Furthermore, in time, we can expect that patient outcomes will be measured in increasing detail, thus making it feasible to model complex outcomes of interest, such as a *dire outcome*. It will therefore become more and more feasible to routinely construct detailed predictive models, such as NN.MTLR, which are based on local patient data. More generally, the induction of such models might use both national data (for its large sample size) and local data (for its ability to locally tailor models). Once constructed, such models could be applied seamlessly to make predictions on new local cases.

Acknowledgments

We thank Scott Obrosky for help in accessing and interpreting the pneumonia PORT dataset. We also thank Richard Ambrosino for help in structuring the original dataset into the database that was used in the research reported here. This research was supported in part by Grant BES-9315428 from the National Science Foundation.

Appendix A. A list of the variables in the pneumonia PORT database that were used in the present study

An alphabetized listing of the baseline clinical variables contained in the pneumonia PORT database are listed below. Each variable name is followed by a brief description.

The term *categ* indicates that a variable is a categorical (i.e., discretized) version of a continuous variable. The categories were created based on clinical judgment of pneumonia specialists on the pneumonia PORT research project.

ABPAINY	abdominal pain y/n
ACTIVCAA	active cancer at presentation
AGEPRES	age at presentation
AGEPRES6	(categ) age at presentation
AGEPRESB	(categ) age at presentation
ALB	albumin g/dl
ALC	alcohol abuse (new or old)
ALERT	alertness
ALKPHOS	alkaline phosphatase IU/L
ALLATBXR	allergic antibiotics y/n
ANOREXIY	loss of appetite y/n
ASMAICUA	asthma ICU admission past year
ASPEVENT	aspiration event
ASPLENA	splenectomy
ASTHMAA	asthma
BANDS	wbc differential percent bands
BEFCPBRY	chest pain y/n before admission
BEFFATGY	fatigue retro y/n before admission
BFPNSOBY	shortness of breath y/n before admission
BILIR	total bilirubin mg/dl
BPDIAS	diastolic blood pressure mm hg
BPSYS	systolic blood pressure mm hg
BUN	blood urea nitrogen mg/dl
CADA	coronary artery disease
CALB	(categ) albumin
CALKPHOS	(categ) alkaline phosphatase
CANCERA	ever had cancer
CBANDS	(categ) wbc % bands
CBILIR	(categ) total bilirubin
CBPDIAS	(categ) diastolic blood pressure
CBPSYS	(categ) systolic blood pressure
CBUN	(categ) blood urea nitrogen
CCOPDSEV	(categ) copd severity
CCR	(categ) creatinine
CFATGY	fatigue y/n
CGLU	(categ) glucose
CHCO3	(categ) HCO_3
CHCT	(categ) hematocrit
CHFA	congestive heart failure
CHGB	(categ) hemoglobin
CHILLSY	chills y/n
CKP	(categ) potassium
CLASTDCD	(categ) days since last discharge
CLDH	(categ) ldh lactic dehydrogenase
CNA	(categ) sodium
CNOPNEPI	(categ) number prior episodes pneumonia
CNUMCOMO	(categ) number of comorbid conditions (exc. HTN)
CNUMSYM	(categ) number of symptoms
CO2RETA	copd CO_2 retainer
CO2RETEN	CO_2 retention
COBEFORY	cough retro y/n before admission
CONFSDYY	confusion y/n
CONFUSA	confusion noted in chart
COPDA	chronic obstructive pulmonary disease

COPDICUA	copd icu admission
COPDSEV	copd severity
COUGHY	cough y/n
CPCO2	(categ) pCO_2
CPH	(categ) arterial pH
CPLT	(categ) platelet count
CPNHOSPN	(categ) number of prior hospitalizations for pneumonia
CPO2	(categ) PO_2
CPOLYS	(categ) wbc % polys
CPRESTIM	(categ) time at presentation
CPULSE	(categ) heart rate beats per minute
CPY	chest pain y/n
CR	creatinine mg/dl
CRESPRAT	(categ) respiratory rate breaths/min
CSGOT	(categ) sgot ast
CSOBY	shortness of breath y/n
CSTERDUR	(categ) steriod duration
CTEMPC	(categ) temperature centigrade
CTEMPCCO	(categ) temperature corrected
CURSPUTY	sputum y/n
CVDA	cerebrovascular disease
CWBC	(categ) wbc
CWEIGHT	(categ) weight
CWTLOSS	weight loss—y/n/missing
CXRCLOBE	(categ) number of lobes involved
CXREFF	pleural effusion
CXREFFBL	bilateral pleural effusion
CXRINF	radiographic evidence of pneumonia (infiltrate)
CXRINFBL	bilateral infiltrate
CXRLOBES	number of lobes involved
CXRPRCNT	max % of involvement of a single lobe
CXRTYPE	type of infiltrate
DEMENTA	prior md documented dementia
DIARRHEY	diarrhea y/n
DMA	diabetes mellitus
DNRA	dnr status
DYMDSYM	days from first seeing an MD to presentation to study
FEV15A	copd fev1 <1 L
FEVERY	fever y/n
FIO2ABG	FiO_2 at time of abg
FIO2ABGA	FiO_2 at time of abg
FIO2POX	pulse oximetry FiO_2
FLU	flu 6 weeks prior to presentation
FLUSHOT	flu shot in past year
GLU	glucose mg/dl
HCO3	HCO_3 meq/L
HCT	hematocrit %
HEADACHY	headache y/n
HEMOPTY	coughing up blood y/n
HGB	hemoglobin g/dl
HOMEO2A	copd chronic home O_2 user
HTNA	hypertension
HYPOAMA	hypogammaglobulinemia
IMMSUPP1	"immunosuppression1 (y/n) includes: NEUTA,HYPOAMA, ASPLENA"
IMMSUPP2	"immunosuppression2 (y/n) includes: IMMSUPP1, MYEL90A, TRANSPA, STERHDC"
INSTLUNA	interstitial restrictive lung
INTUBATE	patient intubated
IVDRUGA	IV drug use
KP	potassium meq/L
LASTDCDY	days since most recent hospital discharge
LDH	ldh lactic dehydrogenase IU/L
LIVERDIA	liver disease
LUNGOUTA	pneumonectomy
MENT4	mental status questionnaire-4 levels
MENTSTAT	mental status questionnaire
MIA	myocardial infarction
MUSKELA	musculoskeletal problems
MYALGIAY	muscle pain y/n
MYEL90A	myelosuppressive drugs past 90 days
NA	sodium meq/L
NAUSEAY	nausea y/n
NEUTA	neutropenia
NOPNEPIS	number prior episodes pneumonia
NTCAGEA	neuromuscular thoracic cage disorder
NUMCOMOR	number of comorbid conditions; includes:
NUMIMMUN	number of factors for immunosuppression
NUMSYM	number of symptoms
NUTRSTAA	documented poor nutrional status or malutrition
O2SAT	O_2 saturation pulse oximetry %
O2SATABG	which done first (ABG or pulse ox)
O2SATC	(categ) O2SAT
PCO2	pCO_2 mmHg
PH	arterial pH
PLT	platelet count $\times 10^3/\mu l$
PNEUVACC	ever had vaccination for pneumonia
PNHOSPNO	number of prior hospitalizations for pneumonia
PO2	pO_2 mmhg
POLYCA	copd polycythemia
POLYS	wbc differential percent polys
PREG	patient pregnant
PRESTIME	time at presentation—hours
PRIATBNM	number of prior antibiotics
PRIATBRT	route of prior antibiotics
PRIATBXA	antibiotic 30 days prior present
PRIHOSPA	prior hospitalization 30 days
PSYCHDXA	formal psychiatric diagnosis
PTEDUC	patient education
PTEMPLOA	current employment status
PTHISP	pt hispanic descent or origin
PTLIVLOA	living arrangements
PTMRSTAA	marital status
PTRACE	pt racial background

PTRACEA	pt racial background (white/non-white)
PULBRONC	bronchial breath sounds
PULCLEAR	lungs clear
PULCYAN	cyanosis
PULDEC	decreased breath sounds
PULDULL	dullnes to percussion
PULETOA	E to A changes
PULFREM	fremitus
PULMHTNA	copd pulmonary hypertension
PULRALES	rales
PULRHONC	rhonchi
PULSE	heart rate beats per minute
PULWHEEZ	wheezing
RESPRATE	respiratory rate breaths minute
RIA	chronic renal insufficiency
RTEMP	route temperature centigrade
SEX	sex of patient
SGOT	sgot ast IU/L
SMOKE	does patient smoke tobacco
SOXYGEN	$pO_2 < 60$ or $O_2sat < 90$
SPUTBFOY	sputum retro y/n before admission
STBLANGA	chronic stable angina
STERDUR	corticosteroids yes number of days
STERHDC	high-dose chronic corticosteroids
STEROIDA	corticosteroids past 90 days
SVENTARA	dysrhythmias supraventricular
SWALLDIA	swallowing disorders
SWEATSY	sweats y/n
SZA	seizures
TEMPC	temperature centigrade
TEMPCCOR	temp deg C corrected (for oral/rectal/axillary)
THROTSOY	sore throat y/n
TRANSPA	transplant y/n (solid organ transplant)
UNEATY	unable to eat y/n
UNSBLANA	unstable angina
VALVDISA	valvular disease
VENTARRA	dysrhythmias ventricular
VOMITY	vomiting y/n
WBC	white blood cell count × 1000/μl
WEIGHT	weight in pounds

Appendix B. A detailed definition of dire outcome

In the text that follows, a number in square brackets denotes the integer used in the pneumonia PORT database to encode the variable value that immediately follows the brackets. An alphabetic label in square brackets denotes a variable name.

A *dire outcome* is defined as having been present if one or more of the following three events occurred with a patient:

(1) death within 30 days of presentation [DEAD30PR]: [1] yes

(2) an initial ICU admission for one or more of the following three reasons:
- respiratory failure [RESPFALI]: [1] documented as reason for the first admission
- respiratory or cardiac arrest [CARRESTI]: [1] documented as reason for the first admission
- shock/hypotension [SHOCKI]: [1] documented as reason for the first admission

(3) the presence of one or more of the following severe complications:
- bleed at site of procedure [BLEEDI]: [4] vascular repair or >2 units of packed red blood cells
- congestive heart failure [CHFI]: [3] new or worsening CHF or pulmonary edema with intubation
- IV site phlebitis [IVSITEI]: [4] requiring surgery
- pulmonary embolus [PULEMBI]: [1] $pO_2 \geqslant 60$ or no pO_2 done, or [2] with hypoxemia ($pO_2 < 60$), or [3] with hypotension
- myocardial infarction [INFARCI]: [1] subendocardial, or [2] transmural without arrhythmias or CHF, or [3] transmural with arrhythmias or CHF
- shock [CSHOCKI]: [2] hypotension requiring pressors or IABP
- ventricular tachycardia or fibrillation [VTACHYI]: [3] new onset or worsening of V-tach with syncope, chest pain or hypotension
- cardiac/respiratory arrest [CARDIACI]: [1] successful resuscitation, or [2] unsuccessful resuscitation (patient expired), or [3] no resuscitation performed (patient expired)
- cerebrovascular [CEREBI]: [2] new stroke with no deficits, or [3] new stroke with neurological deficits
- encephalopathy [ENCEPHI]: [3] coma
- other allergy [OTALEGYI]: [3] systemic anaphylaxis
- gastrointestinal bleeding [GBLEEDI]: [4] new or worsening and required 3–4 units of blood, or (5) new and worsening and required >4 U of blood
- anemia [HEMANEI]: [5] Hct < 20
- leukopenia [LEUKI]: [4] WBC < 1000
- thrombocytopenia [THROMI]: [4] <50 K
- pneumothorax [PNTHRXI]: [2] treated with chest tube, or [3] tension pneumothorax
- respiratory failure [PULFAILI]: [3] intubated with PEEP $\leqslant$ 5 mm, or [4] ARDS or intubated with PEEP >5 mm
- renal insufficiency [RENALINI]: [3] new or worsening and requiring dialysis or transplant
- urinary tract infection [UTII]: [4] urosepsis with positive blood cultures
- brain abscess/parameningeal focus [BRABSCI]: [1] present at presentation or [2] developed after presentation

- empyema [EMPYI]: [1] present at presentation or [2] developed after presentation
- endocarditis [ENDOI]: [1] present at presentation or [2] developed after presentation
- meningitis [MENINGI]: [1] present at presentation or [2] developed after presentation
- osteomyelitis [OSTEOMI]: [1] present at presentation or [2] developed after presentation
- septic arthritis [SEPARTI]: [1] present at presentation or [2] developed after presentation
- line-related sepsis [LINESEPI]: [2] definite

Appendix C. A method to fill-in missing values

The method we describe in this appendix was used to address the missing values problem in the PORT database.[7] The method is an iterated *k*-nearest neighbor method. It is, essentially, a nonparametric EM-style algorithm using Gibbs sampling.

Consider a data set comprising a set of cases, each represented as a set of attributes with values. Most of the values are known; some are initially missing. We will refer to these initially-missing values as "unknown," even after the program has filled in estimated values for them. We use the term "missing" for values that do not yet have an estimate.

C.1. The basic algorithm (one-nearest-neighbor version)

For each case C in the data set,
For each variable X in case C,

If the value of variable X is unknown, then

Produce a new estimate of the value of variable X of case C using the Nearest Neighbor algorithm. (See below for details.) The basic idea is to find the other case in the data set that most closely resembles case C, considering only those cases that have a known value for variable X. Call this "nearest neighbor" case C'. Variable X of case C' now becomes the new estimate for the unknown variable X of case C.

Continue until the whole data set has been scanned and a new estimate has been created for every unknown variable in the data set.

Repeat this entire procedure until either there is no further change or until a fixed limit on the number of iterations has been reached.

As the example below will show, often there is an advantage to iterating this procedure more than once. The estimated values filled in during the first pass may alter the nearest neighbor choices during the second pass. This, in turn, may change some of the estimated values. Normally, only a small number of iterations are needed before the filled-in values converge.

C.2. Finding the nearest neighbor

Given a case C and an variable X whose value is unknown, we want to find the nearest neighbor to C. To do this, we must compute the distance between C and every other case in the data set that has a known value for variable X. Then we simply pick the case whose distance is smallest. We used simple unweighted Euclidean distance to measure the distance between cases.

During the first iteration of the algorithm, we will not yet have computed an estimate for some of the unknown values. Provision must be made for computing the difference between the values of some variable X when that value is unknown for one of the cases. In this case, we compute the average of the X value for every case in which X is known, and we use that average as the estimated value for the unknown X.

It is also possible that the value of some variable X will be missing for both of the cases being compared. (This can occur if case C has two unknown variables.) In this case, we compute the average difference between X values over the entire data set, and we use this difference as an estimate of the distance between the two unknown X values.

[7] Caruana, Rich, "Iterated *K*-Nearest Neighbor Method and Article of Manufacture for Filling in Missing Values." United States Patent 6,047,287. Assignee: Justsystem Pittsburgh Research Center, Pittsburgh, Pennsylvania. Filed May 5, 1998, granted April 4, 2000.

References

[1] Adams P, Marano M. Current estimates from the National Health Interview Survey, 1994. National Center for Health Statistics 1995;10:1–483.

[2] Graves E, Gillum B. 1994 Summary: National Hospital Discharge Survey, Advance Data, No. 278. National Center for Health Statistics. October 1996.

[3] Pneumonia and influenza death rates—United States, 1979–1994. Morbidity Mortality Weekly Rep (MMWR) 1995;44:535–7.

[4] Current estimates from the National Health Interview Survey, 1994. US Department of Health and Human Services 1995; PHS 96-1521, Series 10:1528–935.

[5] Lave JR, Lin CC, Fine MJ. The cost of treating patients with community-aquired pneumonia. Semin Respir Crit Care Med 1999;20:189–98.

[6] Kapoor WN. Assessment of the Variation and Outcomes of Pneumonia: Pneumonia Patient Outcomes Research Team (PORT) Final Report: Agency for Health Policy and Research (AHCPR); 1996.

[7] Fine MJ, Auble TE, Yealy DM, Hanusa BH, Weissfeld LA, Singer DE, et al. A prediction rule to identify low-risk patients with community-acquired pneumonia. N Engl J Med 1997;336:243–50.

[8] Langley P, Iba W, Thompson K. An analysis of Bayesian classifiers. In: Proceedings of the AAAI National Conference on Artificial Intelligence; 1992. p. 223–8

[9] Langley P, Sage S. Induction of selected Bayesian classifiers. In: Proceedings of the Conference on Uncertainty in Artificial Intelligence; 1994. p. 399–406

[10] Hand DJ, Yu K. Idiot's Bayes—not so stupid after all. Int Stat Rev 2001;69:385–99.

[11] Monti S, Cooper GF. A Bayesian network classifier that combines a finite mixture model and a naive Bayes model. In: Proceedings of the Conference on Uncertainty in Artificial Intelligence; 1999. p. 447–56

[12] Kohavi R, John GH. Wrappers for feature subset selection. Artif Intell 1997;97:273–324.

[13] Cheeseman P, Stutz J. Bayesian classification (AutoClass): theory and results. In: Fayyad UM, Piatetsky-Shapiro G, Smyth P, editors. Advances in knowledge discovery and data mining. Cambridge, MA: MIT Press; 1996.

[14] Kontkanen P, Myllymaki P, Silander T, Tirri H. On the accuracy of stochastic complexity approximations. In: Proceedings of the Causal Models and Statistical Learning Seminar; 1997. p. 103–17

[15] Kontkanen P, Myllymaki P, Tirri H. Constructing Bayesian finite mixture models by the EM algorithms. NeuroCOLT Technical Report NC-TR-97-003 (www.neurocolt.org/abs/1997/abs97003.html); 1997

[16] Chickering DM, Heckerman D. Efficient approximation for the marginal likelihood of Bayesian networks with hidden variables. Mach Learn 1997;29:181–212.

[17] Dempster AP, Laird NM, Rubin DB. Maximum likelihood from incomplete data via the EM algorithm. J Roy Statist Soc B 1977;39:1–38.

[18] McLachlan GJ, Krishnan T. The EM algorithm and extensions. New York: Wiley; 1997.

[19] Provost FJ. Policies for the selection of bias in inductive machine learning. Doctoral Dissertation, Department of Computer Science, University of Pittsburgh; 1992

[20] Provost FJ, Buchanan BG. Inductive policy: the pragmatics of bias selection. Mach Learn 1995;20:35–61.

[21] Russell S, Norvig P. Artificial intelligence: a modern approach. Englewood Cliffs, NJ: Prentice Hall; 2002.

[22] Livingston G. A framework for autonomous knowledge discovery from databases. Doctoral Dissertation, Department of Computer Science, University of Pittsburgh; 2001

[23] Afifi AA, Clark V. Computer-aided multivariate analysis. New York: Van Nostrand Reinhold; 1990.

[24] Dixon WJ. In: BMDP statistical software manual. Ewing, NJ: University of California Press; 1992. p. 1136–44

[25] Caruana R, Baluja S, Mitchell T. Using the future to 'Sort Out' the present: rankprop and multitask learning for medical risk evaluation. Adv Neural Inform Process Syst 1996:959–65.

[26] Caruana R. Multitask learning. Mach Learn 1997;28:41–75.

[27] Suddarth SC, Holden ADC. Symbolic-neural systems and the use of hints for developing complex systems. Int J Man–Mach Stud 1991;35:291–311.

[28] Abu-Mostafa YS. Learning from hints in neural networks. J Complexity 1989:192–8.

[29] Weinstein MC, Fineberg HV. Clinical decision analysis. Philadelphia, PA: W.B. Saunders; 1980.

[30] Sox HC, Blatt MA, Higgins MC, Marton KI. Medical decision making. Stoneham, MA: Butterworths; 1988.

[31] Hanley J, McNeil B. The meaning and use of the area under a reciever operating characteristic (ROC) curve. Radiology 1982;143:29–36.

[32] Hanley JA, McNeil BJ. A method of comparing the areas under receiver operator characteristic curves from the same cases. Radiology 1983;148:839–43.

[33] Marrie TJ, Lau CY, Wheeler SL, Wong CJ, Vandervoort MK, Feagan BG. A controlled trial of a critical pathway for treatment of community-acquired pneumonia. J Am Med Assoc 2000;283:749–55.

[34] Niederman M, McCombs J, Unger A, Kumar A, Popovian R. The cost of treating community-acquired pneumonia. Clin Ther 1998;20:820–37.