Journal of Biomedical Informatics 56 (2015) 229–238

Contents lists available at ScienceDirect

Journal of Biomedical Informatics

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

A comparison of models for predicting early hospital readmissions

Joseph Futoma [a], Jonathan Morris [b], Joseph Lucas [a,c,*]

[a] Dept. of Statistical Science, Duke University, Box 90251, Durham, NC 27708, USA
[b] Quintiles, 4820 Emperor Blvd., Durham, NC 27703, USA
[c] Dept. of Electrical and Computer Engineering, Duke University, Box 90291, Durham, NC 27708, USA

ARTICLE INFO

Article history:
Received 26 November 2014
Revised 13 May 2015
Accepted 24 May 2015
Available online 1 June 2015

Keywords:
Electronic Health Records
Early readmission
Penalized methods
Random forest
Deep learning
Predictive models

ABSTRACT

Risk sharing arrangements between hospitals and payers together with penalties imposed by the Centers for Medicare and Medicaid (CMS) are driving an interest in decreasing early readmissions. There are a number of published risk models predicting 30 day readmissions for particular patient populations, however they often exhibit poor predictive performance and would be unsuitable for use in a clinical setting. In this work we describe and compare several predictive models, some of which have never been applied to this task and which outperform the regression methods that are typically applied in the healthcare literature. In addition, we apply methods from deep learning to the five conditions CMS is using to penalize hospitals, and offer a simple framework for determining which conditions are most cost effective to target.

1. Introduction

Changes in federal regulation of the healthcare industry together with the novel use of payment penalties based on quality of care metrics are leading to substantial changes in business models within healthcare. The availability of large repositories of electronic health data and the continued rise of risk sharing relationships between health systems and payers have created a strong incentive to shift healthcare delivery out of the hospital setting and into lower cost, outpatient services. The double incentive of shared risk and early readmission penalties – imposed both within the United States [1] and abroad [2] – have created a strong incentive for hospital systems to identify, at the time of discharge, those patients who are at high risk of being readmitted within a short period of time.

A hospital readmission is defined as admission to a hospital a short time (typically within 30 days) after an original admission. A readmission may occur for planned or unplanned reasons, and at the same hospital as original admission or a different one. A study conducted by the Medicare Payment Advisory Committee (MedPAC) reported that 17.6% of hospital admissions resulted in readmissions within 30 days of discharge, with 76% of these being potentially avoidable [3]. In total, these readmissions accounted for $15 billion in Medicare spending. In an effort to curb hospital readmission rates, part of the Patient Protection and Affordable Care Act penalizes hospitals with excessive readmissions at 30 days through a program called the Hospital Readmission Reduction Program. In the fiscal year 2013, more than 2000 hospitals were penalized over $280 million. On October 1, 2014, the penalty increased to a minimum of 3% of a hospital's Medicare reimbursement, and also included several more conditions [1].

Hospital leaders recognize that scrutiny over readmission rates will continue to grow over the next few years, and that the financial penalties will only increase. As such, procedures for reducing readmissions have been thoroughly researched and have already started to be implemented at many hospitals. Techniques such as improving patient education, conducting followup visits or phone calls, and transferring discharge information to primary doctors may all reduce readmissions. However, individualized followups can be costly; this raises the question of which patient groups should be targeted in order to most effectively use the resources available for preventing readmissions. Methods that can accurately assess patient readmission risk are in high demand, as hospitals scramble to target the most at-risk patients and reduce their readmission rates in the most cost effective manner.

A variety of literature exists on statistical techniques for assessing patient readmission risk, using many types of available data. Some methods, such as in [4], leverage a variety of data sources, including patient demographic and social characteristics, medications, procedures, conditions, and lab tests. Other methods are

* Corresponding author at: Dept. of Statistical Science, Duke University, Box 90251, Durham, NC 27708, USA. Tel.: +1 919 668 3667.
E-mail addresses: jdf38@stat.duke.edu (J. Futoma), jonmorrismd@gmail.com (J. Morris), joe@stat.duke.edu (J. Lucas).

http://dx.doi.org/10.1016/j.jbi.2015.05.016
1532-0464/

based on only a single source of data, for instance, solely on administrative claims data, as in [5]. A thorough review of past models can be found in [6]. With the exception of [7], all of these methods are logistic regressions on independent variables typically chosen by hand.

Our aim is to compare in detail existing methods used to predict readmission with many other statistical methods. These methods include "local" models tailored to particular patient subpopulations as well as "global" models fit to the entire dataset. We compare penalized linear models as well as non-linear models such as random forests and deep learning. Due to the increased difficulty of training deep models, we conduct a smaller set of experiments to validate their performance.

The remainder of this paper will be organized as follows. Section 2 summarizes our data source. Section 3 presents a variety of statistical methods to predict patient readmissions. Section 4 introduces the experimental setup in applying these methods to hundreds of diverse groups of admissions, and summarizes the results. Section 5 compares deep neural networks to penalized logistic regression for predicting readmissions in the 5 groups that CMS (Centers for Medicare and Medicaid) is using to assign penalties. After a brief introduction to deep learning, we offer simple advice on identifying which conditions to target. We conclude in Section 6 with a brief discussion and directions for future work.

2. Data summary and processing

The dataset used is the New Zealand National Minimum Dataset, obtained from the New Zealand Ministry of Health. It consists of nearly 3.3 million hospital admissions in the New Zealand (NZ) hospital system between 2006 and 2012. New Zealand is an island nation with a national healthcare system. Because of this, we anticipate that we are losing very few patients to outside health systems. However, New Zealand uses ICD-10-AM (Australia modification) medical coding and hospitals in New Zealand are under different regulatory pressures from those in the United States. In addition, healthcare workflow and the utilization of admissions may be very different in the New Zealand healthcare environment. As such, the predictive variables and model parameters we discover will not directly translate to data from the United States. However, this paper is focused on the characteristics of the statistical models, not the learned model parameters; the results we present will be a valuable guide for modeling decisions when addressing the early readmission question with US healthcare data.

We formalize the task of predicting early patient readmissions as a binary classification task. As such, our outcome variable of interest is a binary indicator of whether or not a patient is readmitted again to the NZ hospital system within 30 days. For each visit, we have background information on the patient's race, sex, age, and length of stay. Additionally, we also know the type of facility (public or private), and whether the patient was a transfer. As noted in [5], prior admissions can be predictive of future readmissions, so we also include the number of hospital visits in the past 365 days for each patient visit.

We expect the most informative aspect of the dataset to be the large collection of ICD 10-AM codes assigned to each patient visit. Before preprocessing, this consists of 17,390 binary variables coding the precise diagnosis (12,231) and procedures (5159) relevant to each hospital admission. For each visit we also have a single Diagnosis Related Group (DRG) code, selected from a set of 815 unique DRGs which break down admissions into broader diagnoses classes than the highly specific ICD codes. Table 1 provides a brief summary of the dataset.

Before modeling, we do a small amount of preprocessing of the raw dataset. We first filter out patient visits with entry dates before 2005 and eliminate from the training/validation sets any visits that ended in the patient's death. Censored values are treated as not being readmitted within 30 days. Additionally, we combine patient visits that have overlapping admission and discharge dates; generally these represent episodes where the patient was transferred directly from one institution to another. Finally, we exclude as potential predictors in all models any ICD code that appears 10 times or fewer in the full dataset. This leaves us with a sparse 3,295,775 × 12,045 binary matrix of ICD codes, in addition to the background and demographic variables from Table 1.

Table 1
Full dataset, post-processing.

Characteristic	
Total number of admissions	3,295,775
Number of unique individuals	1,328,384
Percent readmission within 30 days	19.0
Number of unique procedures (ICD-10 AM)	3599
Number of unique diagnoses (ICD-10 AM)	8446
Number of ICD-10 AM codes per visit, mean (SD)	5.1 (3.8)
Number of unique diagnosis related groups (DRGs)	815
Variables used in prediction	
Age (years), mean (SD)	41.2 (14.1)
Male (%)	38.8
White/Islander/Asian/Hispanic/African (%)	62.6/26.9/7.1/0.2/0.5
Public facility (%)	93.9
Transfer (%)	5.6
Length of Stay (days), mean (2.5%, 25%, 50%, 75%, 97.5% quantiles)	2.9 (0, 0, 1, 3, 16)
Number of admissions in past 365 days, mean (2.5%, 25%, 50%, 75%, 97.5% quantiles)	3.7 (0, 0, 0, 1, 25)

3. Methods

3.1. DRG-specific methods

Most published approaches to the prediction of 30 day readmission focus on a single target patient population – typically those that are penalized by CMS. In order to mirror this approach and produce a large scale model comparison, we tested a variety of statistical models on 280 different patient-visit cohorts as determined by the DRGs. In the context of regression, this is equivalent to the inclusion of an interaction effect between disease groups and every predictor. Fig. 3.1 displays a histogram of sample sizes for the 280 patient cohorts we consider. In Section 3.2 we introduce methods that scale seamlessly to the entire dataset of over 3 million admissions.

For each DRG, we test 5 methods, 2 of which are novel for the task of predicting early readmission. Before modeling each group, we exclude as potential predictors any ICD code appearing 10 times or fewer in that group. Table 2 contains an abbreviation and short description of each of the DRG-specific methods considered. All models are trained on the background variables from the lower half of Table 1, as well as all the ICD codes remaining after thresholding. Note that this implies that the matrix of independent variables will be extremely sparse since on average only 5 codes are used per admission.

- **LR** The first method considered is logistic regression with a maximum likelihood estimator for the regression coefficients. Define $y_i \in \{-1, 1\}$ to indicate whether the i'th patient visit resulted in readmission within 30 days (where a 1 denotes readmission), and define $\mathbf{x_i}$ to be the sparse p-dimensional vector of independent variables for patient visit i. Maximum likelihood logistic regression involves the identification of a p-dimensional vector of regression coefficients, $\hat{\boldsymbol{\beta}}$, such that

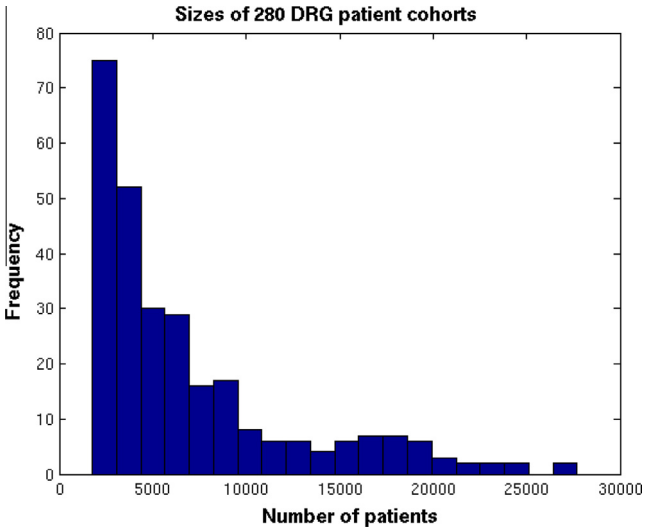

Fig. 3.1. Sample sizes for the 280 diagnosis related groups on which we tested the DRG-specific statistical approaches to prediction of 30 day readmission.

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \sum_{i=1}^{n} \log(1 + \exp(-y_i \boldsymbol{\beta}^\top \mathbf{x_i})). \tag{1}$$

- **LRVS** The next method considered is a form of logistic regression with a multi-step heuristic approach to variable selection that was proposed for the task of predicting early readmission by [5]. This approach, which we label LRVS, consists of 4 steps:
  1. Filter the set of ICD codes so that only codes occurring in more than 1% of observations are retained.
  2. A univariate variable selection is applied. For every predictor, a logistic regression using a single variable at a time is conducted; the variable is retained only if the p-value from a Likelihood Ratio test is below a specified threshold.
  3. A multivariate variable selection procedure is applied. The remaining variables are permuted in a random order, and a stepwise forward variable selection is conducted; a variable is retained only if a Likelihood Ratio test between the larger model with the additional variable and the previous model has a p-value below a specified threshold.

Table 2
DRG-specific methods.

Method name	Abbr.	Short description
Logistic regression	LR	Logistic regression with no additional variable selection. Most frequently used method in literature
Logistic regression with multi-step variable selection	LRVS	Logistic regression with a multi-step variable selection procedure defined in [5]. ICD codes with low prevalence are excluded, followed by univariate and multivariate variable selection
Penalized logistic regression	PLR	Logistic regression with elastic net regularization term, penalizing large coefficient values
Random forest	RF	Tree-based classification method that is able to capture nonlinearity
Support vector machine	SVM	Method maximizes margin between data points and separating hyperplane. Linear kernel and polynomial (order 3) kernel tested, following [7] who used this to predict readmissions

4. Perform logistic regression on the selected variables.

This method has two p-value threshold parameters that must be set. In order to set these, we perform a grid-search and select the pair of p-values with the best performance on a validation set. This is discussed further in Section 4.1.

- **PLR** The third method considered is penalized logistic regression. Penalized methods add a regularization term to the loss function defined in Eq. (1). Ridge regression, first introduced in [8], is a well-known method where the penalty is proportional to the sum of the squares of the coefficients, $\hat{\boldsymbol{\beta}}$, while the LASSO penalty of [9] is proportional to the sum of the magnitudes of the coefficients. Both approaches address the issue of overfitting by shrinking the magnitude of regression coefficients. The LASSO approach is sometimes preferred because it forces some coefficients to be exactly zero. The Elastic Net of [10] is a combination of the two, and is also commonly used. In particular, in elastic net the coefficients $\hat{\boldsymbol{\beta}}$ are found by solving:

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \sum_{i=1}^{n} \log(1 + \exp(-y_i \boldsymbol{\beta}^\top \mathbf{x_i})) + \lambda \sum_{\mathbf{j=1}}^{\mathbf{p}} (\alpha |\beta_\mathbf{j}| + (\mathbf{1} - \alpha) |\beta_\mathbf{j}|^\mathbf{2}). \tag{2}$$

  This formulation is a generalization of both the Ridge and LASSO penalized regression approaches; when $\alpha = 0$ this reduces to ridge regression, while $\alpha = 1$ corresponds to the LASSO. Given a fixed value of α, this approach has a single parameter, λ. We utilize the same approach to setting this parameter as that used for the LRVS method (see Section 4.1 for details). In our experiments, we test three different types of penalties: Ridge, balanced and Lasso ($\alpha = 0.01, 0.5, 1$). Although it is possible to tune the α parameter each time a regression is fit, we chose to directly compare the effect of different penalties instead.
- **RF** The fourth method used for classification is random forests [11]. Random forests are an ensemble learning method, where a large number of binary tree classifiers are trained separately and then combined into a single unified prediction. In the case of a random forest, each classifier is a decision tree that is trained on a random subset of all predictors. A decision tree is a predictive model which maps observations to a class using a tree structure. Each leaf at the bottom of the tree represents a class label, and each internal node of the tree represents a decision to be made based on features. Following common practice, we train a random forest of 500 decision trees, each of them on $\sqrt{p}$ predictors, where p is the total number of predictors. In theory this value could be tuned each time we fit a random forest, which would give slightly better performance. However, [11] suggests this value is relatively robust, and we decided not to tune it to save on computation costs, since our results using this value were quite good. Additionally, each decision tree is trained on a data set that is derived by sampling with replacement from the original data to form a new dataset of the same size (bagging) [12].
- **SVM** Finally, we test a support vector machine (SVM) approach following the methodology in [7] who also utilize them to predict early readmission. The main idea of an SVM, originally introduced in [13], is to maximize the margins between the data points and a hyperplane used to separate two classes. In our case the classes are defined according to whether or not a patient visit resulted in readmission within 30 days. Similar to logistic regression, a linear SVM is formulated as an optimization problem that can then be solved efficiently, but with the inclusion of constraints and a different loss function. It is also possible to consider nonlinear SVMs where the target classifier is a nonlinear function of the inputs. This is accomplished

Table 3
Results for predicting readmission across 280 DRGs.

Method name	Mean (SE) AUC across all 280 DRGs	# DRGs where method had highest mean AUC	*p*-Value, one-sided *t*-test comparing with RF
RF	**0.684 (0.004)**	**80**	–
PLR (α = 0.01)	**0.683 (0.004)**	**27**	0.75
PLR (α = 0.5)	**0.682 (0.004)**	**15**	0.62
PLR (α = 1)	**0.681 (0.004)**	**22**	0.58
SGD (Huber, α = 1)	0.672 (0.004)	39	0.038
SVM (linear)	0.671 (0.004)	47	0.03
SGD (log, α = 1)	0.671 (0.004)	5	0.022
SGD (log, α = 0.5)	0.670 (0.004)	3	0.014
SGD (Huber, α = 0.5)	0.669 (0.004)	7	0.006
SGD (log, α = 0.01)	0.669 (0.004)	4	0.01
LRVS	0.667 (0.004)	28	<0.001
SGD (Huber, α = 0.01)	0.665 (0.004)	1	<0.001
LR	0.648 (0.004)	2	<0.001
SVM (poly)	0.588 (0.005)	0	<0.001

The accuracy of the bold models is statistically indistinguishable from the accuracy of the best model.

through the use of a nonlinear kernel function which is applied to each pair of input data, implicitly mapping observations into a high-dimensional feature space. As in [7] we use both a linear kernel and a polynomial kernel of degree 3. See [14] for a thorough introduction to SVMs.

3.2. Scalable methods for the entire dataset

When fitting a statistical model like logistic regression, typically an optimization problem must be solved to find point estimates for the coefficients. Usually, the optimization problem is solved via a gradient descent algorithm such as Newton's method. The main idea is that at each iteration, the gradient to the loss function is either exactly computed or closely approximated making a calculation that uses the entire dataset. However, when the full dataset is on the order of millions of observations, as in our case, this approach can be intractably slow. In this big data setting, Stochastic Gradient Descent (SGD) is a simple, efficient method for fitting models; [15] provides a good introduction.

SGD is an optimization technique where the true gradient is approximated by instead considering only a single observation at a time. The algorithm iterates over the observations, and for each one updates the model parameters by taking a small step in the direction of a very noisy gradient estimate. Often, a small batch of data (for instance, only 100 observations) will be analyzed in place of a single observation when computing this noisy gradient to speed learning and improve the quality of the noisy gradient. Under mild assumptions, if the step sizes decrease at a certain rate, the algorithm is guaranteed to converge to the true optimum [16].

As in the models in the previous section, both a loss function and penalty should be specified; the only difference now is in how the optimization itself is conducted. In our experiments fitting models to the entire data set we consider the logistic regression loss function with the same Ridge, balanced and LASSO penalties (Elastic Net with $\alpha = 0.01, 0.5, 1$). In addition, we test the modified Huber loss function, another popular loss function in binary classification introduced in [17]. The modified Huber loss is more robust than the logistic regression loss, and better reduces the impact of outliers. Mathematically, the loss is given by

$$L(y, f(x)) = \begin{cases} \max(0, 1 - yf(x))^2 & yf(x) \geqslant -1 \\ -4yf(x) & \text{otherwise} \end{cases} \quad (3)$$

where $y \in \{-1, 1\}$ is the true label and $f(x) \in \mathbb{R}$ is the classifier score for input x. The maximum term is the hinge loss used by SVMs, so the modified Huber loss is a quadratic, smoothed modification to this. By comparing these results to those obtained from the DRG-specific approaches it is possible to gain an understanding of the increase in accuracy obtained from patient stratification based on DRG. We hypothesize that methods for patient stratification that lead to more homogeneous sub-populations of patients will lead to increased predictive accuracy.

4. Experiments

4.1. Experimental setup

We test each of the five methods from Section 3.1 on all of the 280 DRG subsets mentioned previously. Then, we test the methods described in Section 3.2 on the entire dataset, as well as consider their performance on each DRG. In order to fairly compare models with differing levels of flexibility, we use 10-fold cross-validation to estimate predictive accuracy. We train each method 10 times, each time on 90% of the DRG dataset, withholding a different 10% for the final evaluation. For the penalized logistic regression, logistic regression with variable selection, SVM, and SGD methods that require tuning of a parameter, we further divide each 90% training set into a validation set (10% of the total data) and initial training set (80% of the total data). We first train the method on the initial training set for a variety of parameter values, and then select the one that performs the best on the validation set. Then we retrain the method on the original 90% training set before the final evaluation on the test set. For the global SGD methods, the model is fit to 90% of the full dataset (approximately 3 million observations) without using any stratification by DRG, and then tested on the remaining 10%.[1] We then compare the performance of the SGD methods within each DRG to the local methods fit separately to each DRG.

Table 3 reports the average performance of all methods across DRGs, while Table 4 reports the performance of the SGD methods on the full dataset. All methods trained locally on DRGs were trained using built-in functions from the Matlab Statistics Toolbox on a desktop with a 3.5 GHz Intel Core i7 processor. Runtimes were roughly the same, and it took about 3 h to complete the 10-fold cross validation for one method on a single DRG. The SGD methods were trained using the *scikit-learn* library for machine learning in Python [18]. Running on a laptop with a 2.4 GHz Intel Core i5 processor, it took about 2 h to complete the 10-fold cross validation for one global method on the full datatset.

4.2. Results

Following the most common procedure for evaluating models for predicting early readmission, we use the area under the ROC curve (AUC, sometimes referred to as C-statistic) as a quantitative

[1] However, we ensure that within each DRG, the test observations used by SGD within each fold are the same as those for the locally trained methods.

Table 4
SGD results, full dataset.

SGD method (loss, penalty)	AUC: mean (SE)	*p*-Value, one-sided *t*-test comparing with best model
Logistic, $\alpha = 1$	**0.828 (<0.001)**	–
Logistic, $\alpha = 0.5$	0.827 (<0.001)	0.058
Logistic, $\alpha = 0.01$	0.827 (<0.001)	0.01
Huber, $\alpha = 1$	0.827 (<0.001)	<0.001
Huber, $\alpha = 0.5$	0.824 (<0.001)	<0.001
Huber, $\alpha = 0.01$	0.821 (<0.001)	<0.001

The accuracy of the bold models is statistically indistinguishable from the accuracy of the best model.

means of comparing predictive performance among classifiers. A perfect classifier would achieve an AUC of 1, while random guessing corresponds to 0.5. Table 3 compares the performance of each method when evaluated on the 280 DRGs. By contrast, Table 4 shows performance on the entire patient population – only relevant to the models trained on the entire population. For each DRG and for each method, we record the mean AUC across all 10 held-out test sets. We then report the mean and standard error of these 280 average AUCs in the table. We also report the number of DRGs on which each method had the highest mean AUC among all methods. The random forest and penalized logistic regressions are overall the best methods, when comparing their mean AUCs for each of the 280 DRGs. This is verified in the last column of the table. We report the *p*-values from one-sided *t*-tests, testing whether the mean AUC for the random forest is significantly higher than the mean AUC of every other method.

In the left pane of Fig. 4.1, the heterogeneity across DRGs is apparent. Although the average AUC of the best method across DRGs is 0.69, it ranges from 0.57 to .95, indicating that some DRGs are substantially easier or more difficult to model. There is a modest correlation of 0.36 ($p < .001$) between the best AUC and actual readmission rate within each DRG, and a similar correlation of 0.29 between the best AUC and number of readmissions within each DRG ($p < .001$). This indicates that groups with many readmissions are somewhat easier to model. However, there is only a slight correlation of 0.12 ($p = .052$) between best AUC and the sample size within each DRG, indicating that having a much larger set of patients will not necessarily lead to substantial improvements in predictive performance. These results indicate that certain classes of admissions are inherently harder to model than others, an important fact that should be taken in account when using these methods in application.

The right pane of Fig. 4.1 highlights the differences between the best methods fit locally to each DRG compared to the best global SGD methods that were fit to the full data and then tested on each DRG. The histogram is right shifted, indicating that most of the time, the best locally trained method outperforms the best global method. We hypothesize that this is due to increased homogeneity of patient subgroups when compared to the population as a whole. That homogeneity allows the inclusion of variables that are relevant only to patients within the subgroup.

On average, the local methods have an AUC that is 0.02 higher, and on 80% of DRGs the local method performs better. If prediction of early readmission is desired for patients with a particular diagnosis profile, it is best to model those patients separately. However, the global SGD method is scalable and still manages to perform quite well, suggesting that hospitals might want to fit all of their admissions with such a method, as well as fit more specific models on particular subsets of patients (for instance, those on which CMS imposes penalties).

Finally, Table 4 gives results from the SGD methods trained to the full dataset (i.e. training on roughly 3 million admissions and testing on 300,000). Due to the large sample size, the choice of the penalty and loss function is minimal and AUC has low variability across the ten cross-validation runs, although the logistic loss slightly outperforms the Huber loss. However, when the predicted values from these methods are partitioned by DRG, as we saw previously, there is a large degree of variability.

5. Deep learning

The five conditions CMS currently uses to assess penalties to hospitals are listed in Table 5. This makes models for these conditions especially relevant for healthcare systems. In this section we will focus on these five disease states and examine the accuracy of newer deep learning methods for classification. In the appendix in Table A.7 we list the ICD10-AM codes used to identify admissions pertaining to each of these conditions.

5.1. Model description

The goal in deep learning is to model high-level abstractions in the data using nonlinear transformations. Such abstractions can then be used to interpret the data, or to build better predictive models. Our interest in applying these techniques is to develop better models for predicting early readmissions. Deep learning

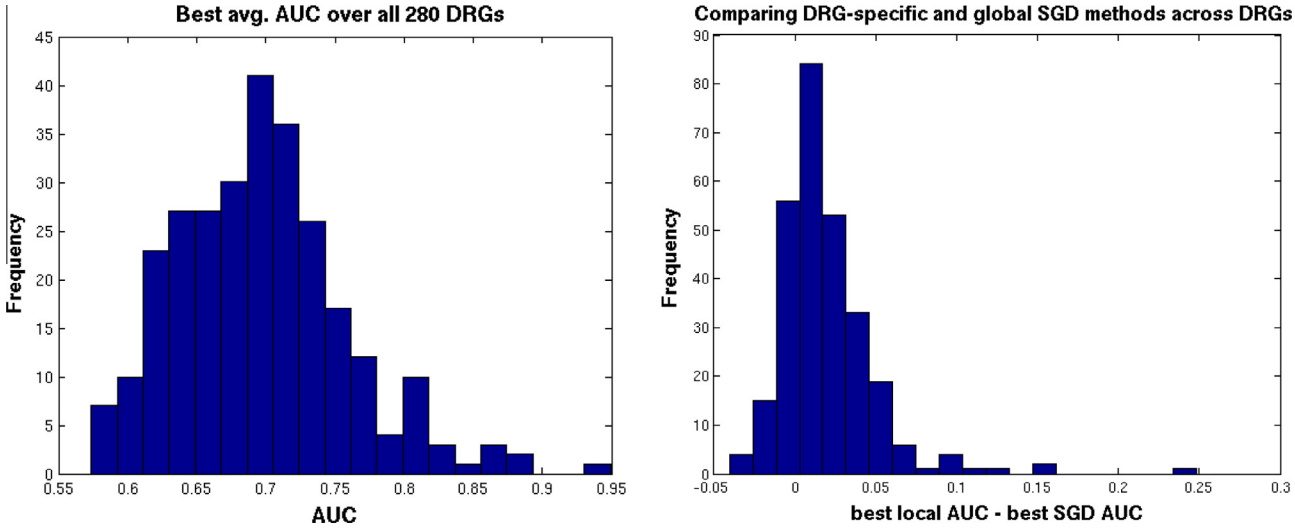

Fig. 4.1. Left: best AUCs for each DRG. Right: difference in local and global (SGD) methods.

Table 5
The five conditions CMS uses to assess penalties.

Condition	Abbr.
Chronic obstructive pulmonary disorder	COPD
Heart failure	HF
Pneumonia	PN
Acute myocardial infarction	AMI
Total hip arthroplasty/total knee arthroplasty	THA/TKA

methods provide an extremely flexible and robust approach for learning complicated, nonlinear decision boundaries between classes. [19] provides an extensive background to deep learning and a detailed overview of the main results in the field (through 2009).

The basic building block in many deep learning models is a feedforward neural network. This model (in Fig. 5.1) consists of three layers of variables: an input layer of observed variables, a hidden layer of unobserved variables, and an output layer consisting of a prediction (e.g. the probability of a particular admission resulting in an early readmission). Information moves from the input nodes, through the hidden nodes, to the output nodes. The input and hidden nodes form a bipartite graph, where each input node is connected only to hidden nodes and not to other input nodes; similarly, the hidden layer and output layer form a bipartite graph. Given an input x, the hidden layer h is computed as $h = f(b^{(1)} + W^{(1)}x)$, where $b^{(1)}$ is a vector of offsets, $W^{(1)}$ is a matrix of weights, and f is a known nonlinear function, typically the sigmoid, $f(x) = 1/(1+e^{-x})$. That is, each node in the hidden layer is computed as some linear combination of the inputs that is then passed through a nonlinear function. The output o is computed similarly, $o = g(b^{(2)} + W^{(2)}h)$, where the function g depends on the problem at hand. In regression problems g is typically the identity function. For classification tasks with K classes, the softmax function is usually used, which is defined as $g_j(h) = e^{h_j}/\sum_{k=1}^{K} e^{h_k}$. Since the softmax function maps a vector of arbitrary real numbers to a set of positive numbers that sum to 1, the corresponding entires $g_j(h)$ are interpreted as class probabilities.

Training a feedforward neural network consists of learning appropriate values for the parameters $b^{(1)}, W^{(1)}, b^{(2)}$, and $W^{(2)}$. Typically, this is done using gradient descent or stochastic gradient descent, as discussed in Section 3.2, where the goal is to minimize a loss function comparing the prediction o for an input x with the true value, y. In order to calculate the gradient, which is then used to update the weights and offsets, a method called backpropagation is used. The main idea is to propagate forwards each training example through the network to generate output activations, and then propagate these activations backwards through the network to update each weight or offset accordingly.

As a simple generalization of this model, many layers of hidden nodes may be stacked, as in Fig. 5.2. The corresponding model is called a multi-layered neural network, or a deep neural network. Through stacking multiple layers, the model is able to capture much richer structure and learn significantly more complicated functions than a neural network with a single layer. There is theory proving that deep networks are capable of compactly representing functions that a shallow architecture with fewer labels could not; see [19] for details. Despite the modeling flexibility associated with such deep neural networks, they are historically very hard to train, and were not widely used until [20] proposed a fast and reliable method for training. Their main discovery was that significantly better results were achieved when the network is first pre-trained using an unsupervised learning algorithm. That is, before training the final network, only the inputs x and not the known labels y are used to pretrain the network. This is accomplished layer by layer in a greedy manner. We treat the lowest two layers connecting the input to the first hidden layer as a Restricted Boltzmann Machine (RBM; a form of neural network with a single input layer and a single hidden layer). We can train this RBM using the Contrastive Divergence algorithm [21]. Once this layer has been trained, we pass the inputs up through the first layer, and treat the resulting output as new inputs to use to learn the weights between the first and second hidden layers. This procedure is continued up the network, until all of the weights connecting layers have been initialized to reasonable values. Finally, backpropagation is used to "fine-tune" the parameters, where now the learning algorithm uses the known labels y.

Despite the numerous advances made in training deep neural networks in the past decade, there are still a large number of parameters that must be tuned to the problem at hand in order for the model to achieve good performance. Unless the model parameters are chosen carefully, the high degree of flexibility exhibited by deep neural networks can lead to overfitting. In what follows, we describe our approach to minimize overfitting in prediction of early readmission. We have relied on several sources of practical knowledge on how to train these models, and they provide much more detail on the topic [22–24]. Additional details on parameter selection and tuning are provided in Appendix B.

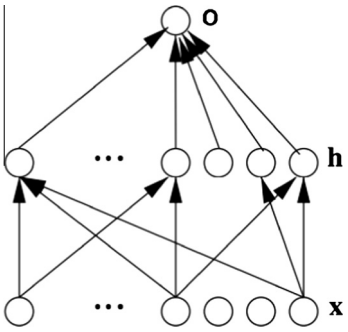

Fig. 5.1. A single layer neural network (image modified from [19]). The input x is fed forward through the hidden layer h to the single output node o, which could be a continuous number for a regression problem or a probability for a classification problem.

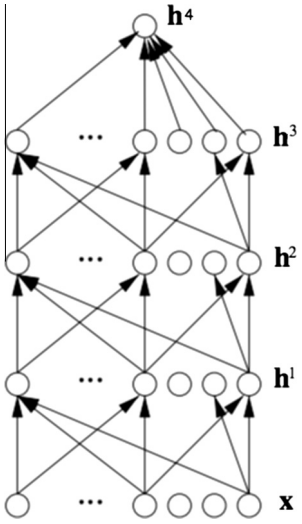

Fig. 5.2. A multilayer neural network (image from [19]). The input x is fed forward through the three hidden layers to the single output node, which could be a continuous number for a regression problem or a probability for a classification problem.

Table 6
Results from 10-fold cross validation. We compare a penalized logistic regression (PLR) with a deep neural network (NN). *p*-Values are for a one-sided *t*-test that NN has significantly higher AUC.

Condition	Size	Readmission rate (%)	PLR AUC: mean (SE)	NN AUC: mean (SE)	*p*-Value, one-sided *t*-test
PN	40,442	27.9	0.715 (0.005)	**0.734 (0.004)**	0.01
COPD	31,457	20.4	0.703 (0.003)	**0.711 (0.003)**	0.086
HF	25,941	19.0	0.654 (0.005)	**0.676 (0.005)**	0.004
AMI	29,060	29.5	**0.633 (0.006)**	**0.649 (0.007)**	0.11
THA/TKA	23,128	8.7	**0.629 (0.005)**	**0.638 (0.006)**	0.264

The accuracy of the bold models is statistically indistinguishable from the accuracy of the best model.

5.2. Results

We focused this analysis on the five patient cohorts for which CMS has imposed early readmission penalties (listed in Tables 6 and A.7). For these five patient populations we trained deep neural networks to predict readmission, and compared against penalized logistic regressions as a baseline (where we tune the α parameter instead of fixing it as before). We decided to compare with penalized regressions to provide a clear link to prior work. Previous methods for predicting early readmission have been almost exclusively regression-based, and in the last section we found penalized methods were the best among regression methods. Table 6 shows the resulting AUCs per group for both methods, along with the sample sizes and readmission rates. The deep neural networks consistently had better AUC (3/5 times they were significantly better at a 0.05 level), but involved a substantial amount of tuning of parameters, requiring a large amount of CPU time. Interestingly, the neural networks significantly outperformed the penalized regressions on the three conditions with highest overall AUCs. We display the ROC curves, averaged over the 10 folds, for each of the 5 deep neural network models considered in Fig. 5.3 (we omit the penalized logistic regression curves because the neural net curves were uniformly higher).

In addition to AUC, we consider the positive predictive value (proportion of those designated high risk who are actually readmitted, also known as precision) for both methods, following [7]. In Fig. 5.4, we plot the positive predictive value in each condition as a function of sensitivity (also known as a precision-recall curve). Somewhat surprisingly, the positive predictive value is higher for the HF and COPD models compared to PN, despite the fact that the PN model had the highest AUC, best ROC curve, and a higher readmission rate than HF or COPD. This is suggests that although AUC is a commonly used general metric of model discriminative performance, it is important that it be supplemented by task-specific measures of discrimination that depend on different penalties for false positive and false negatives. As a final means of comparing models, in Fig. 5.5 we display calibration curves for the 10 models considered. We see that the deep neural networks are much better calibrated than the penalized regressions, which is further reason to prefer them to the regressions. In addition, the THATKA and AMI curves exhibit the worst calibration, reinforcing our earlier findings. It is safe to conclude that THATKA and AMI are the hardest of the five conditions to predict readmissions in our data, although it is not clear that there is one condition that we are definitively best able to predict.

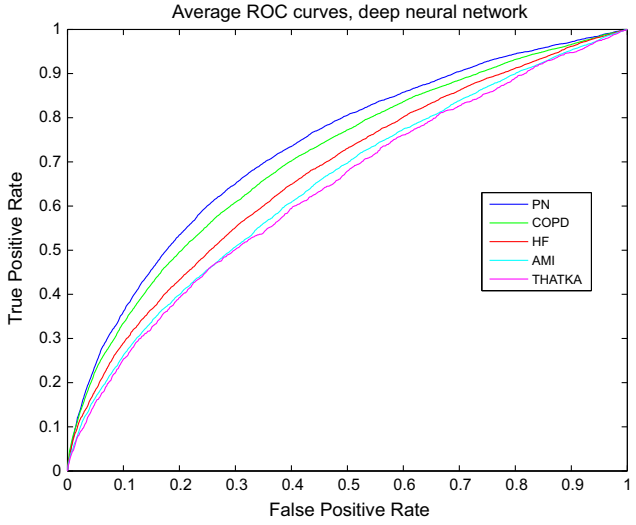

Fig. 5.3. ROC curves for deep neural networks models, fit to the five conditions CMS penalizes.

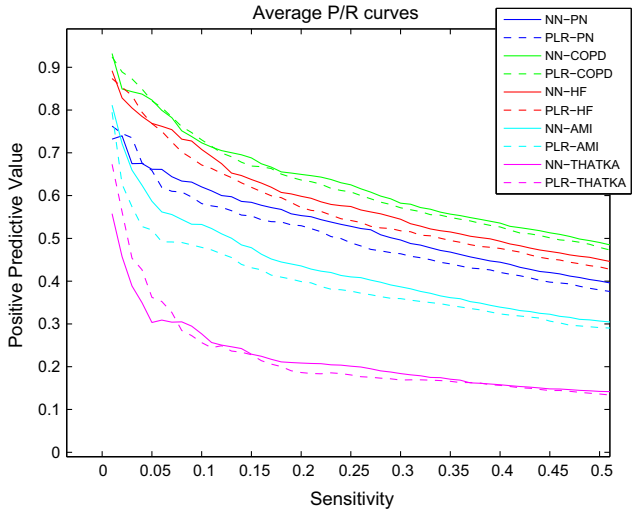

Fig. 5.4. Precision/recall curves for both deep neural network and penalized regression models, fit to the five conditions CMS penalizes.

6. Discussion

In this paper we highlighted a general framework for constructing models for assessing patient readmission risk using only ICD codes and a few background variables. Comparing our proposed methods to others that exist in the literature, we find random forests, penalized logistic regressions, and deep neural networks have significantly better predictive performance than other methods that have been previously applied to this problem.

The impact on overall predictive accuracy that can be obtained by moving from standard logistic regression to more complicated models can be substantial, however these models can also be difficult to tune and are sometimes more challenging to interpret. Although in their current state we found deep learning models to be the most difficult to manage due to the large number of model parameters, they are also the models with the greatest potential to boost predictive accuracy in statistical approaches to predicting

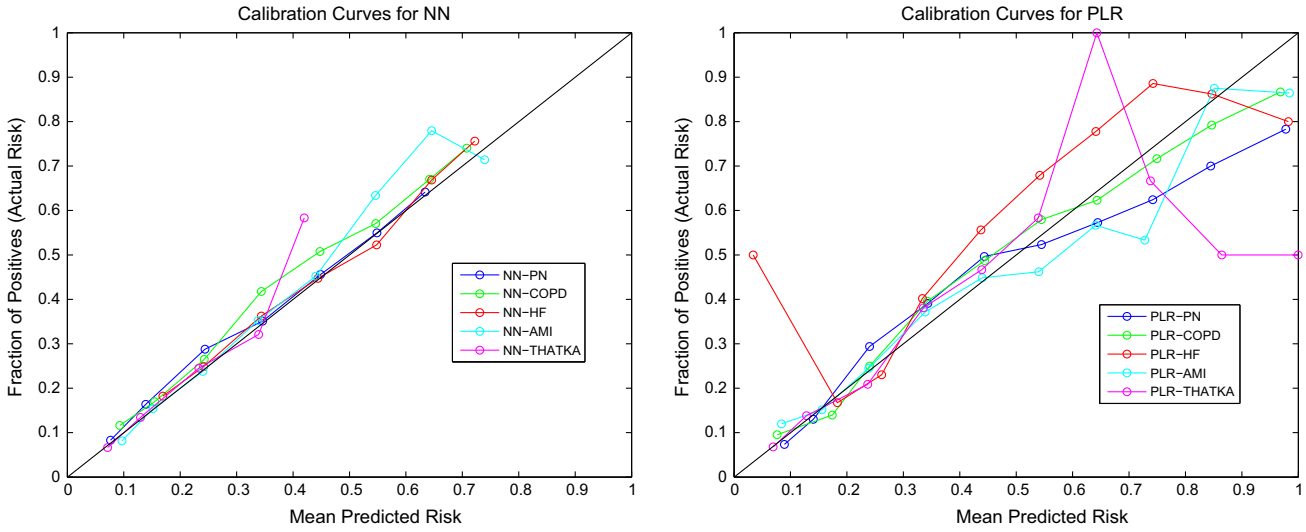

Fig. 5.5. Left: calibration curves for the 5 deep neural networks. Right: calibration curves for the 5 penalized regressions.

early readmission. There is currently a lot of effort being invested by the machine learning community in these models; we expect improvements in algorithms for selecting the depth and size of these models together with improvements in model fitting to make deep learning more effective and "user friendly" in the near future. Due to the fast-moving nature of the field, there are certainly improvements to be made on the way we designed and trained our deep architecture. In particular, some pretraining was done with the entire data set. This was minimal, but it may have introduced a small bias in our estimates of accuracy. We emphasize that practitioners interested in applying deep learning methods to their own problems should spend time fine-tuning the details of their own models rather than exactly copying the final architecture and parameters from our study or from others.

An interesting finding of our study is that typically "local" methods specific to a particular patient population outperform a "global" method that does not take into account the varying nature of different disease groups. This suggests that models should be tailored to specific populations of interest if the goal is to predict readmissions for specific subgroups (e.g. for the five groups CMS uses to assess penalties). However, local methods are more likely to run into power issues if sample sizes are limited, in which case global methods fit to a large heterogeneous population may still prove beneficial. We use DRGs as a means of partitioning patients into homogenous subgroups for the purpose of comparing methods, however, there may be significant room for improvement of overall prediction accuracy by a more careful approach to patient stratification. Developing a method to cluster patients into meaningful subgroups would be useful in settings where an explicit partitioning is not available.

Despite the promise our study offers for improving the performance of existing methods for predicting early readmissions, there are several important limitations that should be taken into consideration. First, EHR datasets such as the one we used contain repeated observations on individuals, and there are many individuals in our dataset with a large number of admissions (see Table 1). We do not explicitly account for potential correlations between repeated observations that are conditionally independent of length of stay, admissions in the past year and demographic variables. Another issue with this type of data is that we have no information on the reliability of the ICD codes. It is likely that there are some of the codes in our data that were the result of recording errors or misdiagnoses. However, there is little that can be done to address this without additional information, which we do not have with this data source. Finally, our modeling approach simplifies a time-to-event problem to a binary regression problem, since we label each observation depending on whether or not a readmission was observed within 30 days. This clearly results in some loss of information; for instance, readmissions occurring at 29 days and 31 days now fall into different classes, despite the closeness. However, treating the problem as a binary regression opens a much wider array of statistical tools that can be used; furthermore, this is the approach CMS uses to assess penalties.

There are a variety of factors involved in hospital readmissions, many of them unpredictable. Often times there may be unseen socioeconomic factors at play that are not readily available in a hospital database. However, statistical models do a reasonable job of predicting readmissions,[2] giving hospitals valuable information about which patient groups to target. In particular, when considering the five conditions CMS uses to assess penalties, we found that some conditions were substantially easier to predict than others. In the future, collaboration with hospitals may allow us to take a model-based approach in order to determine the right type of intervention for each patient cohort, extending the ideas introduced in Section 5.

Conflict of interest

The authors declare that they have no conflicts of interest in regards to the content published in this article.

Acknowledgements

The authors would like to thank Quintiles for partially supporting this work. Additionally, this material was based upon work partially supported by the National Science Foundation under Grant DMS-1127914 to the Statistical and Applied Mathematical Sciences Institute. We would like to acknowledge the Duke Center for Predictive Medicine for the many productive conversations regarding predicting patient outcomes. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

[2] We defer to [25], which states a model is considered "reasonable" if its AUC exceeds 0.7 and "strong" if its AUC exceeds 0.8. By this criterion most of the models considered in our study would be considered reasonable, while the global models fit to the full data (as well as models for some of the DRGs) would be considered strong.

Table A.7
The ICD10-AM codes in our data used to select patients with each condition.

Condition	ICD10-AM codes
Chronic obstructive pulmonary disorder	J440, J441, J448, J449
Heart failure	I500, I501, I509
Pneumonia	J121, J122, J128, J129, J13, J14, J150, J151, J152, J153, J154, J155, J156, J157, J158, J159, J160, J168, J170, J171, J172, J173, J178, J180, J181, J188, J189
Acute myocardial infarction	I210, I211, I212, I213, I214, I219
Total hip arthroplasty/total knee arthroplasty	49312-00, 49315-00, 49318-00, 49319-00, 49324-00, 49327-00, 49330-00, 49333-00, 49339-00, 49342-00, 49345-00, 49346-00, 49518-00, 49519-00, 49521-00, 49521-01, 49521-02, 49524-00, 49527-00, 49530-00, 49530-01, 49533-00, 49554-00

Appendix A. ICD9 based definitions of 5 diseases penalized by CMS

See Table A.7.

Appendix B. Technical details of fitting the deep learning models

In our experiments on predicting early readmissions, the majority of input variables are binary ICD codes. For the continuous variables (age, length of hospital stay, and number of admissions in past year) we standardize them so that all observed and hidden nodes take values between 0 and 1. Throughout, we use the sigmoid function as the activation function between layers, and the softmax function to link the final hidden layer to the output. We again use 10-fold cross-validation to evaluate predictive accuracy. We focus on the five conditions for which CMS has enacted early readmission penalties (listed in Table 6).

One of the first steps in designing a deep neural network is to determine the network architecture. We tried a variety of architectures, ranging from two to five hidden layers of nodes, and varying the sizes of each layer. Ultimately, we found that using three hidden layers performed substantially better than one or two layers, but the large increase in training time when using four or five layers did not justify the slight performance gain. We eventually settled on using the same number of nodes in each hidden layer, selecting this value to be around 75% the number of input nodes (i.e. number of retained ICD codes for this condition). The neural network is first pretrained using the unsupervised learning algorithm mentioned previously. For each of the 5 conditions modeled, we tried pretraining the network using the full dataset with all 3 million observations, as well as using only the observations corresponding to those conditions, and found that the results were similar. In the end, we pretrained each network for 1000 epochs using only the observations for to the condition we intended to model (i.e. made 1000 passes over the dataset using stochastic gradient descent). It took about 24 h to pretrain the model for each of the 5 conditions.

Once the network has been pretrained, there are several parameters to choose that help speed the optimization procedure. We briefly summarize the final approach we took. We trained each model for 2000 epochs on each training set. We initially fix the learning rate, which determines how big the step size will be at each iteration of gradient descent, to 1. After each epoch, we multiply it by 0.998 so the learning algorithm takes progressively smaller steps. We use a fixed batch size of 100 observations for stochastic gradient descent. We also use a momentum parameter to aid the optimization, which smooths our noisy gradients in a manner similar to conjugate gradient methods. We initially set the momentum to 0.5, and let it increase to a final value of 0.99 after 500 epochs, increasing it a fixed amount after each epoch.

Finally, there are several parameters to be tuned that help prevent overfitting. We penalize large values of the weights by using a penalty proportional to the sum of the squares of the weights (i.e. the ridge regression penalty). Next, we encourage the binary hidden units to be active half of the time by incorporating a sparsity penalty that ensures no hidden units are always on or always off. Additionally, on presentation of each batch of 100 observations during training, a proportion of hidden units are randomly omitted from the network, a procedure known as dropout. This prevents hidden units from relying on the presence of other hidden units and helps to prevent overfitting. One final approach we used to combat overfitting was "early-stopping". The idea is to monitor the performance of the network on a validation set (in terms of AUC) during the training. When the network begins to overfit to the training set, and its AUC on the validation set decreases significantly, we stop training early. At this point we evaluate the method on our final test set. To train the deep neural networks, we modified[3] a Matlab toolbox from [26]. The final architectures used had three hidden layers of equal size (400 for HF, COPD, AMI, 200 for THA/TKA, and 750 for PN). The dropout proportion, weight penalty, and sparsity penalty were all determined by using a grid search of possible values, and selecting the values that performed the best on the full training set. Once values were selected, we used 10-fold cross validation for our final evaluation of out-of-sample predictive performance. The final dropout proportion used was 50%, and the final values used for the weight penalty and sparsity penalty were 10^{-5}. It took about 10 h to fine-tune each fold of the final cross validation procedure for each condition.

References

[1] C. for Medicare, M. Services, Readmissions Reduction Program, August 2014. <http://www.cms.gov/Medicare/Medicare-Fee-for-Service-Payment/AcuteInpatientPPS/Readmissions-Reduction-Program.html>.
[2] K.J. Verhaegh, J.L. MacNeil-Vroomen, S. Eslami, S.E. Geerlings, S.E. de Rooij, B.M. Buurman, Transitional care interventions prevent hospital readmissions for adults with chronic illnesses, Health Aff. (Millwood) 33 (2014) 1531–1539.
[3] M.P.A. Committee, Report to Congress: Promoting Greater Efficiency in Medicare, 2007.
[4] S.A. Choudhry, J. Li, D. David, C. Erdmann, R. Sikka, B. Sutariya, A public-private partnership develops and externally validates a 30-day hospital readmission risk prediction model, Online J. Public Health Inf. 5 (2013).
[5] D. He, S.C. Matthews, A.N. Kalloo, S. Hutfless, Mining high-dimensional administrative claims data to predict early hospital readmissions, J. Am. Med. Inf. Assoc. 21 (2014) 272–279.
[6] D. Kansagara, H. Englander, A. Salanitro, D. Kagen, C. Theobald, M. Freeman, S. Kripalani, Risk prediction models for hospital readmission: a systematic review, J. Am. Med. Assoc. 306 (2011) 1688–1698.
[7] S. Yu, A. v. Esbroeck, F. Farooq, G. Fung, V. Anand, B. Krishnapuram, Predicting readmission risk with institution specific prediction models, in: Proceedings of the 2013 IEEE International Conference on Healthcare Informatics, ICHI '13, 2013, pp. 415–420.
[8] A.E. Hoerl, R.W. Kennard, Ridge regression: biased estimation for nonorthogonal problems, Technometrics 12 (1970) 55–67.
[9] R. Tibshirani, Regression shrinkage and selection via the lasso, J. R. Stat. Soc., Ser. B 58 (1994) 267–288.
[10] H. Zou, T. Hastie, Regularization and variable selection via the elastic net, J. R. Stat. Soc., Ser. B 67 (2005) 301–320.
[11] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32.
[12] L. Breiman, Bagging predictors, Mach. Learn. (1996) 123–140.
[13] C. Cortes, V. Vapnik, Support-vector networks, Mach. Learn. (1995) 273–297.
[14] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines and Other Kernel-based Learning Methods, Cambridge University Press, NY, 2000.

[3] We made a few technical modifications to efficiently handle the large, sparse nature of our data, and made AUC the metric for model performance instead of classification error.

[15] L. Bottou, Large-scale machine learning with stochastic gradient descent, in: Proceedings of the 19th International Conference on Computational Statistics (COMPSTAT'2010), 2010, pp. 177–187.
[16] H. Robbins, S. Monro, A stochastic approximation method, Ann. Math. Stat. 22 (3) (1951) 400–407.
[17] T. Zhang, Statistical behavior and consistency of classification methods based on convex risk minimization, Ann. Stat. 32 (2003) 56–134.
[18] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: machine learning in Python, J. Mach. Learn. Res. 12 (2011) 2825–2830.
[19] Y. Bengio, Learning deep architectures for ai, Found. Trends Mach. Learn. 2 (1) (2009) 1–127.
[20] G.E. Hinton, S. Osindero, Y.-W. Teh, A fast learning algorithm for deep belief nets, Neural Comput. 18 (7) (2006) 1527–1554.
[21] G. Hinton, Training products of experts by minimizing contrastive divergence, Neural Comput. 14 (2002) 1771–1800.
[22] G.E. Hinton, A practical guide to training restricted boltzmann machines, in: G. Montavon, G.B. Orr, K.-R. Mller (Eds.), Neural Networks: Tricks of the Trade, second ed., Lecture Notes in Computer Science, vol. 7700, Springer, 2012, pp. 599–619.
[23] G.E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Improving Neural Networks by Preventing Co-Adaptation of Feature Detectors, CoRR, 2012. <http://arxiv.org/abs/1207.0580>.
[24] H. Larochelle, Y. Bengio, J. Louradour, P. Lamblin, Exploring strategies for training deep neural networks, J. Mach. Learn. Res. 10 (2009) 1–40.
[25] D.W. Hosmer, S. Lemeshow, Applied Logistic Regression, second ed., John Wiley & Sons, NY, 2000.
[26] R.B. Palm, Prediction as a Candidate for Learning Deep Hierarchical Models of Data, Master's Thesis, 2012.