Analysis of Diabetic Hospital Admissions with Machine Learning Techniques

Henry Lin
Computer Science
Stanford University
henryln1@stanford.edu

Sharman Tan
Computer Science
Stanford University
sharmant@stanford.edu

Joseph Yen
Electrical Engineering
Stanford University
joyen98@stanford.edu

1. Introduction

Machine learning techniques have become more prominent in recent years as they have been utilized in a wide range of fields, from Netflix movie recommendations to autonomous vehicles to facial recognition. These techniques are broadly divided into two main categories: supervised and unsupervised learning. In this paper, we explore several techniques and their effectiveness as we apply them to a dataset containing hospital admissions for diabetic patients [6]. The adoption of machine learning in the medical field has led to some promising work in areas such as tumor classification [7] and breast cancer detection [2] [1]. In this paper, we concentrate on diabetic patients and predict whether or not each patient will be readmitted in the future based on a wide range of input features, from age to current medication prescriptions. To do so, we use binary and multi-class logistic regression. Using K-means Clustering and PCA, we also strive to identify traits that may cause these patients to be predisposed to readmission.

2. Related Work

This has not been the first time machine learning techniques have been applied to the medical field, and [3] is one example of machine learning being used in the context of hospital admissions. Using techniques like logistic regression and deep neural networks, they apply they techniques in a very similar fashion to what we do. Using a dataset containing over 500,000 patient admission records, they used 900+ features to help predict hospital admission, and they ultimately achieved 90% accuracy with a 90/10 split between training and test sets. This work shows the effectiveness of machine learning in studying hospital admissions patterns, and they also incorporated feature selection (subsetting the 900+ features and measuring performance) to help better understand what features may be better predictors than others. Another avenue they explored extensively is the impact of varying amounts of training data, a practical question in a field when data collection is oftentimes expensive and relatively scarce. They ultimately find that there is some threshold before more training data does not help, and is simply redundant computation. In many ways, this paper tackles our problem except in a different field (emergency department), and it is a valuable predecessor to our work. We follow their feature subsetting to try and reduce computation and examine whether some features may be more noise than helpful.

While hospital admission is a rather broad field in terms of medical diseases, researchers have also taken interest in applying machine learning to specific disease types such as in [7] and [1]. One of the main roadblocks in this area of study is the access to a large dataset to train and test on, oftentimes contained to large hospitals or medical schools. This leads to datasets that are much smaller in size, typically in the thousands, as in [7]. On the other hand however, the high performance achieved within these fields suggest that machine learning is very strong at learning classification given the right set of features, placing a large burden on feature selection. This emphasis on feature engineering thus becomes a major part in determining the success of an algorithm, an issue we explore a bit later.

The focus has not solely been on producing the best accuracy on a validation or test set however, and in [1], the researchers were also able to identify a distinct set of features that highly correlated with breast can-

cer detection. This information is valuable for future work as it significantly reduces computation costs by simplifying the feature matrix and focusing the training on more important traits.

3. Data Cleaning

We found a dataset [5] containing information for patients who were admitted and then readmitted into a hospital for diabetes. This dataset is large (100k+ patient instances) with many distinct features (50+ attributes), and is split into 60% training, 20% validation, and 20% test. We first removed irrelevant features in the dataset such as patient and encounter IDs. We also removed features with high percentages of missing data, such as patient weight (97%) and payer codes (52%). For values that listed a specific range, we replaced the cells with the average of the range. We also created modified datasets with normalized features for future learning purposes. This gives us the 12 robust continuous features (gender, age, time, etc.) that will be used to train our models in this paper, as shown below:

Description	Value
gender	0
age	45
time_in_hospital	9
num_lab_procedures	47
num_procedures	2
num_medications	17
number_outpatient	0
number_emergency	0
number_inpatient	0
number_diagnoses	9
change	0
diabetes_med	1
readmitted	2

In addition, the output string labels listing whether or not the patient was readmitted and if they were readmitted under or over 30 days were converted to quantitative values. We employ two different classifications, a multi-class one sensitive to the readmission time and a binary one that only notes readmission occurrences, as shown below.

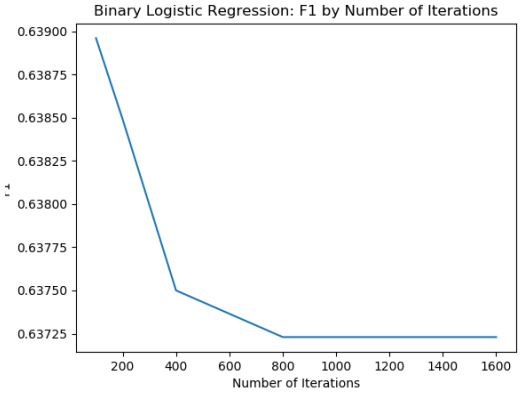

Figure 1. Binary Logistic Regression F1 by Number of Iterations

Original	New Value
No readmission	0
Less than 30 days	1
More than 30 days	2

Original	New Value
No readmission	0
Readmission	1

Categorical data outside of our readmission output was not used to train our models, as it is difficult to train accurately over data with mixed types of features, discrete and continuous. One solution that we proposed was to represent each category as a feature containing one-hot representation values that determine whether or not the patient is in that category. However, since our dataset already contained many continuous features, we decided to exclude categorical data.

4. Methods

We employ both supervised and unsupervised machine learning methods to help glean underlying patterns from our data and learn what features may be important in predicting hospital readmission. Here, we give an overview of the main techniques we have used.

4.1. Supervised Learning

We conducted both multi-class and binary logistic regression classification. Currently, we have a multi-class logistic regression classifier that uses 3 separate one-vs-all classifiers as well as a binary logistic regression classifier that simply predicts whether each

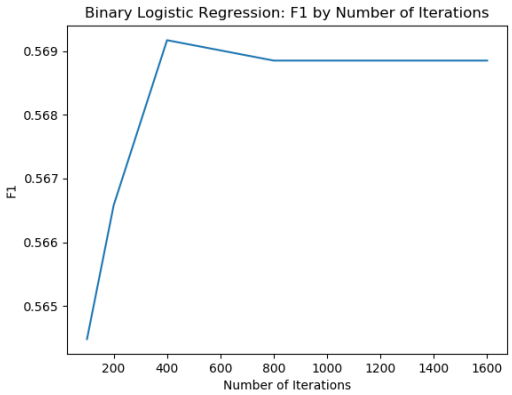

Figure 2. Multi-class Logistic Regression F1 by Number of Iterations

patient will be readmitted. Our model trains on the 12 features and output (whether patient is remitted within 30 days or after 30 days or not at all), and the goal is for it to do well predicting whether an unseen patient will suffer from remission of diabetes (and if so within or after 30 days for the multi-class classifier) from the 12 features of the patient. We currently run experiments on our baseline model to determine optimal hyperparameters such as the number of iterations and the regularization parameter.

An important aspect of training our multi-class logistic regression baseline model is hyperparameter tuning. In order to decide on what value to use for the regularization parameter lambda, we ran the model for 250 iterations for a variety of values from 0.001 to 30, increasing by factors of 3 and 10. We found that the precision, recall, and F1 scores do not change by much with different regularization values, but the value of 0.1 seemed to be a good fit.

After finding 0.1 as an optimal value for the regularization parameter lambda, we varied the number of iterations from 100 to 1600, increasing by a factor of 2. Figures 1 and 2 show the F1 score of each binary class for each number of iterations. The number of iterations that maximized the binary F1 score was 8000.

4.2. Unsupervised Learning

One of the main bottlenecks for machine learning models is the availability of high quality, labeled data for supervised tasks such as spam classification or weather prediction. Services like Amazon Mechanical Turk alongside online datasets have significantly increased the number of datasets available, but creating large datasets is still expensive. This motivates the second type of learning we explore: unsupervised learning. In contrast to supervised learning, unsupervised learning does not rely on a specific correct label or value that we can base a cost function off of and train. Instead, we use the information we are given and run models to try and detect some underlying, interesting structure in the data (e.g. clustering).

One of the main tools for unsupervised learning is principal component analysis (PCA) [4], and it has a wide variety of important uses such as data visualization and feature removal. The main idea of the model is to use singular vector decomposition to try and understand along which dimensions the data displays the most variance. It also works best for continuous variables, and we apply it here to try to elucidate the structure of the data for further analysis.

Another main tool for unsupervised learning is K-means Clustering. We used K-means to see if we could identify features that could predispose diabetes patients to readmission. Our algorithm clustered the data into K = 3 different clusters and used various subsets of the 12 features to determine if any of the features were particularly indicative of readmission likelihood. We also tested the algorithm with all 12 features. The K-means method we used is described below. First, we randomly initialized the centroids to be three of the existing patients in the dataset. Then, we assign all the patients to one of the three clusters based on which cluster centroid the patient is closest to. For each point, we used the Euclidean distance between the patient and the respective cluster centroid, taking into account the entire set of features associated with the patient, as shown below:

$$Distance(p, q) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$$

After all the patients are assigned to a specific cluster, we set each cluster centroid to the average of all the points in the respective cluster. We continue reassigning the patients to their new clusters and resetting the cluster centroids to new values until the centroids converge and retain the same values between consecutive iterations.

Binary Linear Regression Scores				
Dataset	Class	Precision	Recall	F1
Train	No readmission	0.59756	0.76243	0.67000
	Readmission	0.61293	0.42284	0.50044
Validation	No readmission	0.59425	0.68692	0.63723
	Readmission	0.62235	0.52381	0.56885
Test	No readmission	0.69210	0.67640	0.68416
	Readmission	0.52267	0.54077	0.53157

Table 1. Binary Logistic Regression Performance

Multiclass Linear Regression Scores				
Dataset	Class	Precision	Recall	F1
Train	No readmission	0.52631	0.954940	0.678611
	Readmission (< 30 days)	0.42500	0.023077	0.043777
	Readmission (> 30 days)	0.53558	0.110646	0.183403
Validation	No readmission	0.52634	0.955135	0.678679
	Readmission (< 30 days)	0.41803	0.023077	0.043739
	Readmission (> 30 days)	0.53538	0.110266	0.182869
Test	No readmission	0.62655	0.955270	0.756757
	Readmission (< 30 days)	0.54902	0.025842	0.049361
	Readmission (> 30 days)	0.44045	0.112413	0.179113

Table 2. Multi-class Logistic Regression Performance

5. Results and Discussion

5.1. Logistic Regression

The binary and multi-class precision, recall, and F1 scores with lambda set to 0.1 and 8000 iterations are displayed in Tables 1 and 2, respectively. Our scores for multi-class linear regression vary significantly from class to class. Notably, the recall for the *No readmission* class is extremely high (0.96), while the recall for the other two classes are extremely low. This means that we are correctly predicting a large number of the patients who are not readmitted, but we fail to predict the vast majority of patients who are readmitted, at any time period. Our precision scores are relatively the same for all three classes, but the F1 scores follow trends similar to that of recall. Since more patients are indeed not readmitted compared to those who are, we achieve high recall and fair precision and F1 scores just by predicting most of the patients as *No readmission*. Unfortunately, we seem to therefore be classifying few patients as readmission within or after 30 days, so we are achieving very low recall and F1 scores for those patients.

Because multi-class logistic regression did an excellent job classifying patients who are not readmitted but did a poor job of classifying patients who are readmitted within or after 30 days, we decided to implement binary linear regression to see whether we would achieve better results if we just had to classify whether or not patients were readmitted (regardless of when).

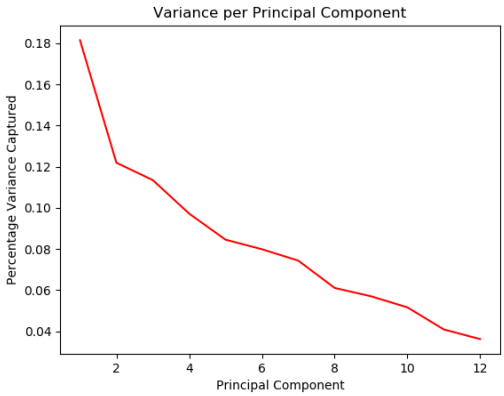

Figure 3. Percentage Variance Captured by each Principal Component

Our precision, recall, and F1 scores for binary linear regression were fairly high, ranging from around 0.4 to 0.8. Therefore, our model does relatively well classifying whether or not patients are readmitted, but it does very poorly when distinguishing between patients who are readmitted within 30 days and those readmitted after 30 days. This means that it is easiest to predict that patients will not be readmitted, and it is hardest to predict that patients will be readmitted within or after 30 days. This may be because the characteristics of patients that are readmitted within 30 days and those of patients that are readmitted sometime after 30 days are similar, and we have fewer datapoints for each of the latter two classes to train on.

5.2. PCA

After running PCA on the twelve continuous features, we plotted the percentage variance captured in each principal component as shown in Figure 3. We found that it was impossible to capture a large amount of the variance using a small number of principal components. This leads us to believe that these features each independently capture a significant part of the variance. If there had been a small number of important principal components, there would have been a much steeper drop in the visualization.

5.3. K-Means Clustering

We were able to perform K-means clustering, as we removed discrete features from the dataset and concentrated on continuous features so we could calculate euclidean distance. In addition to calculating it with

all continuous features, we also did several runs of the algorithm using a manually selected subset to try and remove features that may have contributed more noise than useful information. Ultimately though, this manual selection did not improve results and the results presented below use the full set of continuous features. We also tested multiple times using random initializations of the centroids due to the random convergence of K-means, and we found that this did not affect the results.

To evaluate our K-means results, we calculated the purity score along with manually inspecting the distribution of data points in each cluster compared to the ground truth labels we had in the original dataset. Purity score is a clustering algorithm metric defined in [5], and it is a measure to what extent a cluster contains a single class. In other words, it helps us examine how well our detected clusters align with prelabeled groupings. Formally, purity is defined as:

$$P = \frac{1}{N} \sum_{m \in M} \max_{d \in D} |m \cap d|$$

where M represents the set of clusters, D represents the set of classes, and N data points. This results in a score between 0 and 1, with higher values signaling better alignment between classes and clusters. Running it on our K-means results, we found a purity score of 0.539, indicating some overlap, and we expand on this in Figure 4.

In Figure 4, we provide a visualization of the how the data points in each cluster were distributed across the predefined classes. For example, the blue bar indicates that in cluster 1, approximately 55% of the data points came from the *No Readmission* class, 10% from the *within 30 Days* class, and 35% from the *after 30 Days* class. It is interesting how the distribution remains relatively constant across the clusters, suggesting that the clusters are fragmenting the data points completely differently from the classes.

6. Conclusion and Future Work

Our binary logistic regression model performs relatively well at predicting whether or not patients are readmitted. However, our multi-class logistic regression model performs very well at predicting patients who are not readmitted, but it has difficulty predicting patients who are readmitted and in what time period.

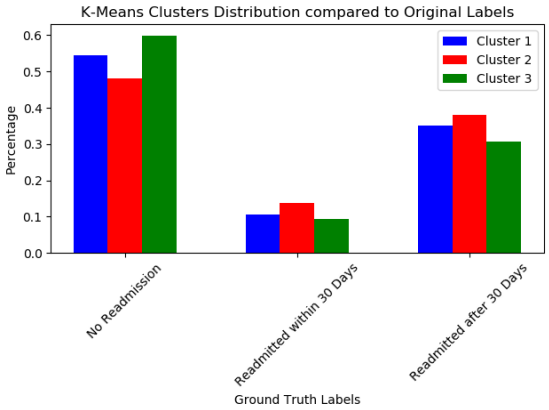

Figure 4. K-Means Clusters distributed among Ground Truth Classes

Through PCA and K-means, we attempted to better understand the underlying feature patterns pervading our dataset, and we found several interesting traits of our dataset. With PCA, we discovered that each feature we chose contributed a significant part of the variance, meaning removing any single one would cause a significant drop in the variance. We also discovered that the groupings detected via the K-means algorithm differed significantly from that of the labeled classes, which is why our supervised learning techniques may not have achieved higher performance.

A promising way to more significantly improve the performance of our model is to convert it to a neural network. This would allow us to automatically learn important features and better classify datapoints with non-linear relationships. As discussed previously with works like [1], feature selection can be time consuming, and neural networks solve that problem through automatic feature learning. With neural networks, there are a variety of hyperparameters we can tune, such as the size of each hidden layer, the number of hidden layers, the regularization parameter, and the activation functions between layers. In addition, a potentially less computationally expensive way to improve our logistic regression models is to convert them to SVMs. Since this is a situation where the number of examples is large (over 100K) while the number of features is small (12), we can create or add more features and then rerun our logistic regression models or implement and run a linear SVM.

7. Repository

All of our code can be found at https://github.com/henryln1/CS229A.

References

[1] A. H. Beck, A. R. Sangoi, S. Leung, R. J. Marinelli, T. O. Nielsen, M. J. Van De Vijver, R. B. West, M. Van De Rijn, and D. Koller. Systematic analysis of breast cancer morphology uncovers stromal features associated with survival. *Science translational medicine*, 3(108):108ra113–108ra113, 2011.

[2] R. C. Deo. Machine learning in medicine. *Circulation*, 132(20):1920–1930, 2015.

[3] W. S. Hong, A. D. Haimovich, and R. A. Taylor. Predicting hospital admission at emergency department triage using machine learning. *PloS one*, 13(7):e0201016, 2018.

[4] I. Jolliffe. Principal component analysis. In *International encyclopedia of statistical science*, pages 1094–1096. Springer, 2011.

[5] M. Sanderson. Manning christopher d., raghavan prabhakar, schütze hinrich, introduction to information retrieval, cambridge university press. 2008. isbn-13 978-0-521-86571-5, xxi+ 482 pages. *Natural Language Engineering*, 16(1):100–103, 2010.

[6] B. Strack, J. P. DeShazo, C. Gennings, J. L. Olmo, S. Ventura, K. J. Cios, and J. N. Clore. Impact of hba1c measurement on hospital readmission rates: analysis of 70,000 clinical database patient records. *BioMed research international*, 2014, 2014.

[7] E. I. Zacharaki, S. Wang, S. Chawla, D. Soo Yoo, R. Wolf, E. R. Melhem, and C. Davatzikos. Classification of brain tumor type and grade using mri texture and shape in a machine learning scheme. *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, 62(6):1609–1618, 2009.