Robert Beetel
Udacity
Machine Learning Engineer Nanodegree
October 26, 2017

Capstone Project Proposal

Machine Learning Engineer Nanodegree

Domain Background

Patients admitted into Intensive Care Units (ICU's) are severely ill or injured and require a high level of care. These patients are at much greater risk of dying than typical hospital patients. ICU patients are also among the most expensive patients. The burden of care with regard to cost for ICU patients is disproportionately large with estimates of associated costs in the U.S. approaching 1% of national GDP[1] . In the U.K. ICU care costs account for 0.6% of National Health Expenditures[2].

Problem Statement

The overall goal of the project is to train a classifier to classify ICU patients as survivors and non-survivors given data from the first 24hrs of their ICU stay. The ability to predict survival, or classify patients into high and low risk categories, would result in improved survival rates by identifying high risk patients so that they may receive more aggressive treatment while also achieving a reduction in cost by recognizing lower risk patients so that they may receive less aggressive, and less expensive, treatment more appropriate to their circumstances. The projects two goals, identification of high risk patients and identification of low risk patients, can often be competing aims in classifier performance. Maximizing identification of non-survivors may result in higher rates of false positives while maximizing identification of survivors may result in higher rates of false negatives. To balance these goals a metric will be used which balances those two interests by simultaneously maximizing recall and f-scores for both survivors and non-survivors while minimizing the differences between them.

---

[1] Critical care medicine in the United States 2000-2005: an analysis of bed numbers, occupancy rates, payer mix, and costs. Halpern NA, Pastores SM, Crit Care Med. 2010 Jan; 38(1):65-71.

[2] Cost effectiveness of adult intensive care in the UK., Ridley S, Morris S, Anaesthesia. 2007 Jun; 62(6):547-54.

Datasets and Inputs

Patient data was collected from the the MIMIC-III (Medical Information Mart for Intensive Care III), a large, freely-available database comprising deidentified health-related data associated with patients who stayed in critical care units of the Beth Israel Deaconess Medical Center between 2001 and 2012. In addition to data on juvenile patients, the database contains data associated with 53,423 ICU admissions for adult patients (aged 16 years or above) covering 38,597 distinct adult patients and 49,785 hospital admissions. Data includes information such as patient demographics, vital sign measurements made at the bedside (~1 data point per hour), laboratory test results, procedures, medications, caregiver notes, imaging reports, and mortality (both in and out of hospital). In-hospital mortality for adult patients was observed to be 11.5%.[3,4]

The database tables were comprised of CSV files. Those files were downloaded and a local copy of the database was reconstructed using Postico, a PostGRESQL client for Mac.

Data was queried in three major groups or categories:

- Patient Demographics
- Chart Data
- Lab Data

Data from queries was exported in the form of CSV files. Data samples will be defined as unique ICU stays with mortality during each stay as the sample outcome and data collected within the first 24 hours of the stay as the sample input features.
The data from different sources will be pre-processed and from the larger data sets, a smaller number of features will be selected based on the correlations between the features and outcome.

Solution Statement

The task presents itself as a supervised classification problem with output as two classes, survivors and non-survivors, and inputs as categorical/dummy features. It is believed that given the relatively high dimensional, complex feature space the optimum decision boundary would likely be non-linear. Classifiers selected for evaluation are those that work well in higher dimensional space with categorical data and are capable of generating linear and non-linear decision boundaries.

---

[3] MIMIC-III, a freely accessible critical care database. Johnson AEW, Pollard TJ, Shen L, Lehman L, Feng M, Ghassemi M, Moody B, Szolovits P, Celi LA, and Mark RG. Scientific Data (2016). DOI: 10.1038/sdata.2016.35. Available at: http://www.nature.com/articles/sdata201635

[4] Pollard, T. J. & Johnson, A. E. W. The MIMIC-III Clinical Database http://dx.doi.org/10.13026/C2XW26 (2016).

The candidate classifiers included:

- Support Vector Classifier (SVC)
- Linear Support Vector Classifier(LSVC)
- K-Nearest Neighbor Classifier (KNeighbors)
- Multi-Layer-Perceptron Classifier (MLP)
- Decision Tree Classifier (Tree)

The support vector machine classifiers, SVC and LSVC, fit the criteria in that they they work well in higher dimensional spaces by reducing the feature space to a lower dimensional support vector space. The support vector classifiers are also flexible in that different kernel functions can be employed some of which are capable of generating linear or non-linear decision boundaries. LSVC is a support vector machine that utilizes a linear kernel function while the SVC classifier may employ polynomial, radial basis or sigmoid functions producing non-linear decision boundaries.

Nearest neighbor classifiers store labeled training data and assign class to a test point by simple majority vote of the nearest training data points nearest to the test point. The test point is assigned to the class to which the majority of neighboring training data points are assigned. The classifier is capable of generating highly non-linear decision boundaries. in relatively high dimensional space.

Multi-layer perceptron classifiers are multi-layer neural networks. The multi-layer perceptron classifier evaluated had a 3-layer, input, middle and output layer, architecture. The classifier was evaluated over a range of activation functions and solvers.

Lastly, Decision Tree classifiers are algorithms which infer simple sets of decision rules from training data. Decision Tree decision boundaries can range from very simple to highly complex.

Benchmark Model

To create a benchmark for classifier performance, the K-Nearest Neighbors classifier was selected for it's simplicity. The K-Neighbors Classifier was trained and tested using the full feature set as inputs and a train/test split of 80/20%. The algorithm was trained using default parameter settings. Precision, recall, f1 and support scores for survival and non-survival groups were calculated and are shown below in Table 1.

BENCHMARK PERFORMANCE-1

		precision	recall	f1-score	support
Classifier	**Classes**				
KNeighborsClassifier	**Avg/Total**	0.79	0.83	0.81	533.0
	Non-Survivors	0.31	0.17	0.22	75.0
	Survivors	0.87	0.94	0.90	458.0

Table 1. Precision, f1, recall support scores are shown for the K-Nearest Neighbors classifier. The algorithm was trained and tested using default parameters, 20 input features and an 80/20% train-test split. Of note, survivor recall and f1 scores are very high while corresponding non-survivor scores are very low.

Evaluation Metrics

The metrics we will be using to assess classifier performance include:

Precision = TP / (TP + FP)

Recall = TP / (TP + FN)

F-Scores = Harmonic Means of Precision and Recall:

*F-Beta Score = $(1 + \beta^2)$ * (Precision * Recall) / (β^2 * Precision + Recall)*

*F-1 Score = (Precision * Recall) / (Precision + Recall)*

Support = Number of occurrences of each class in y_test

where:

TP = True Positive (Correctly classify a Patient as ‘High Risk’)

FP = False Positive (Incorrectly classify a patient as ‘High Risk’)

FN = False Negative (Incorrectly classify a patient as ‘Low Risk’)

y_test = Set of results against which classifier predicted values are compared.

The primary goal of this project is to identify high risk patients so that they may receive more aggressive treatment to decrease their likelihood of dying. Given that, we would like our classifier to predict mortality of patients with a maximum number of True Positives and a minimum number of False Negatives. It was noted that, due to the small number of non-survivors, classifiers could achieve a maximum performance in some areas by predicting that a patient would survive no matter the data. The benchmark classifier had the tendency to

maximize average scores by maximizing performance for survivors at the expense of performance for non-survivors. To avoid this, an optimization metric was developed to perform the dual function of maximizing recall and f-scores for both survivors and non-survivors while minimizing the difference between them. The metric is a combined sum of the survivor and non-survivor scores, minus the absolute value of the difference between them.

The equation below shows how the optimization metric for F1-score would be calculated.

$$\text{F1 Optimization Metric} = (\text{F1-Survivor} + \text{F1-Non\_Survivor}) - \text{abs}(\text{F1-Survivor} - \text{F1-Non-Survivor})$$

Project Design

The theoretical project workflow is summarized as follows:

1. Download and build local copy of Mimic III database
2. Query the database for patient demographic data and initial (first 24hrs) of laboratory and chart data for ICU stays with ICU stay defined as a unique instance of a patient being admitted into the ICU.
3. Pre-process data and select features based on the strength of their correlation with outcomes.
4. Train candidate classifiers to predict patient survival, optimizing each classifier along its respective parameter space, number of input features and the train/test split size.
5. Evaluate optimized classifier performance, select final model and compare with benchmark

Patient data will be collected from the the MIMIC-III (Medical Information Mart for Intensive Care III), a large, freely-available database comprising deidentified health-related data associated with patients who stayed in critical care units of the Beth Israel Deaconess Medical Center between 2001 and 2012.

Data will be pre-processed and features will be selected. Preprocessing will involve converting time series, continuous data, constant continuous and categorical variables to categorical dummy variables. Time series data will be summarized using descriptive statistics. The distributions for these statistics as well as for constant continuous variables will be assessed for normality and transformed if necessary. Outliers will also be removed. Continuous data will then be converted to categorical by calculating the quartiles for each features distribution and labeling each data point based on into which quartile fell.

Once all data is converted to categorical, features will then be converted into dummy variables. Finally, missing data will be dropped from the data-set and feature selection will be performed using chi2 values to assess the correlation between features and patient outcome.

It is believed that given the relatively high dimensional, complex feature space the optimum decision boundary would likely be non-linear. Classifiers selected for evaluation were those that work well in higher dimensional space with categorical data and were capable of generating linear and non-linear decision boundaries.

The classifiers evaluated included:
- Support Vector Classifier (SVC)
- Linear Support Vector Classifier(LSVC)
- K-Nearest Neighbor Classifier (KNeighbors)
- Multi-Layer-Perceptron Classifier (MLP)
- Decision Tree Classifier (Tree)

The top twenty features, ranked in order of chi2 scores / P-values, will be organized in a data frame along with corresponding patient mortality outcomes.The data set will be divided into features and outcomes, then split into training and testing data sets. Classifiers will be optimized over their respective feature spaces, feature set sizes and train/test splits ranging from 90/10% to 50/50%. Classifiers will be evaluated for their ability to minimize False Negatives and Maximize False Positives. The highest performing classifier will be selected as the final model and will be compared to the benchmark to determine what, if any, improvement has been made.

All work will be done in Python 2.7 installed via Anaconda.