# Identification of Suitable Predictive Models for Classifying Terminal Patients

Roberto Cancel, Kyle Esteban Dalope, and Nicholas Lee

University of San Diego

Master of Science, Applied Data Science

ADS-503 Applied Predictive Modeling

Section 01

June 27, 2022

## Introduction

One of the most daunting tasks faced in the medical field is the decision of the order in which patients are admitted. The severity of the same illness can vary from person to person; thus, mortality also varies between individuals. Early models that attempted to predict severity and mortality found limited success. For example, Green et al. (1990) the Health Care Financing Administration (HCFA) constructed a model that could not accurately predict patient mortality across multiple hospitals. Since the release of the study by Green et al., with a deeper understanding of machine learning and medicine, it has been shown that machine learning models can accurately define severity and a patient's probability of mortality. One study by Steele et al. (2018) utilized "Cox models, random forests, and elastic net regression" models to predict patient mortality with regards to coronary artery disease. Another study utilized "Support Vector Machine (SVM), Artificial Neural Networks, Random Forest, Decision Tree, Logistic Regression, and K-Nearest Neighbor (KNN)" algorithms to accurately identify 89.98% of COVID-19-related mortality rates (Pourhomayoun & Shakibi, 2021). While the models above could accurately predict mortality and severity, those models were limited to a single illness. In cases where illnesses are diverse, such as in the emergency room or intensive care unit, there is a need for a model to rank severity and predict mortality across multiple illnesses at once. To do so, the Acute Physiology and Chronic Health Assessment Evaluation scoring system (APACHE) is commonly used. This study aims to utilize APACHE scores, along with other descriptive features of patients, to create a model that can accurately identify the risk of death in the presence of multiple illnesses.

## Data Description

The dataset provided by Kaggle, "dataset.csv", was initially created as a data frame with the read.csv() function to undergo pre-processing and convert any blank observations with NA. It consisted of 84 predictors and one binary response variable from 91,713 patient observations. Predictor features consisted of patient and hospital identifiers, patient demographic features, patient lab results, patient vitals, APACHE comorbidities, APACHE groupings, and APACHE predictions (see Data Dictionary in Appendix). Due to PHI compliance concerns, the four ID columns were dropped (encounter_id, hospital_id, patient_id, and icu_id). Since this is a classification prediction problem, the response variable (hospital_death) was converted into a factor ("death" and "no_death") and the "gender" variable into a binary factored feature. Next, the table() function was used to identify the proportion of each class in our response variable - the classes suffered from a severe imbalance with only 8.63% of observations resulting in hospital death (0-No death N = 83,781 and 1-Death N = 7,907). This class imbalance is addressed when the data is prepared for modeling.

## Data Splitting, Wrangling, and Preprocessing

To avoid data leakage concerns, the dataset was split before any wrangling or preprocessing steps were taken using a stratified random sampling by class methodology with an 80% training and 20% test set. By splitting the data before wrangling, the test set remains "unseen" since all preprocessing techniques use only the information in the training set. When working with large datasets that include missing values, outliers, and correlated features, there are multiple strategies to prepare the data for modeling and analysis. For example, datasets centered around Healthcare, Biomedical, and/or Biology topics often contain many observations and features; thus, data preprocessing is an important factor in the modeling process. Depending

on the data being analyzed and modeled, users have a variety of methods to choose from. This can include data transformations, centering and scaling, Box-Cox transformations, and data reduction and feature extraction such as Principal Component Analysis (PCA).

Upon observing the data set with the summary() function, there were quite a large number of missing (NA) observations (288,046). Schafer (1999) asserted that a missing rate of 5% or less is inconsequential and vice versa; therefore, it was decided to remove any missing data features within each column above a 5% threshold of missingness. This resulted in the removal of 25 features. It was also observed that 11 APACHE features had the same number of missing observations. After subsetting the observations with the missing APACHE features, several of the features included binary diagnoses (e.g., AIDS and cirrhosis). Since imputing diagnoses can introduce severe bias concerns and these observations represented only 0.76% of all observations, the observations with missing diagnoses were dropped. Then, observations with missing ethnicity values were re-coded to the already available “Other/Unknown" value. Finally, the remaining missingness in the data set was imputed by the median of the training set since missingness was now constrained to vital sign data. Near-zero variance features and highly correlated features with a cutoff of 80% were then dropped before conducting Exploratory Data Analysis (EDA).

## Exploratory Data Analysis (EDA)

Given the objective, the target feature is hospital_death, a binary feature where zero represents a patient who has not died, and one represents a patient who will pass or has passed. The upper bar chart of Figure 1 illustrates the proportion of instances by the target feature. In addition, the plot shows how imbalanced the data set is, where instances associated with death make up less than 10% of the data set. The subsequent plots help promote insight into the

relationship between mortality rate and patient characteristics by plotting the target features against some predictors. For example, the bottom bar chart in Figure 1 shows the total proportion of death in each ICU Stay Type category. While the proportion of death may not be different by category, this plot indicates that there is not much of a difference in mortality risk whether the patient is newly admitted to the ICU, released and readmitted, or transferred to the ICU from another department.

As mentioned previously, the Apache scoring system measures the risk for mortality in the ICU. The Apache system has several categories in that patients receive a score. The scores from each category are summed into a cumulative score, and the overall risk is determined from the sum. In the data set, different versions of the Apache system were taken into consideration, as the system by hospitals globally varies. Apache II versus Apache III differ in that Apache III has been updated; and, according to Hsu et al. (2001), Apache III has slightly more discriminative power, performing better in terms of its ability to predict high-risk patients than Apache II does. In Figure 2, the bar charts show the proportion of death and non-death patients for each category of the two Apache systems. From these bar charts, it can be deduced that patients that receive a score for sepsis with the Apache III system have a higher risk for mortality. Similarly, with the Apache II system, the three categories with the highest proportion of mortality are respiratory, hematologic, and cardiovascular. A plot, such as Fig. 2, can assist in determining whether a patient is at high risk for death if they receive a score for a certain category.

Since physical characteristics play a role in health-related risks, it is important to examine the relationship between the features related to physical traits and the target variable. For example, age can significantly affect recovery time, illness severity, and general health quality. Figure 3 shows the histogram for age in the data set, colored by the response variable for each

bin and a boxplot. Both the histogram and boxplot show that most of the data lie around the age range of 50 to 75. The data is skewed left; thus, the results from this study may be slightly more applicable to those in that age range, as those outside the age range are not as highly represented. Although from a logical perspective, this distribution of data makes sense; it is not commonly expected that those in a younger age range would be hospitalized. The boxplot reveals four outliers on the lower end of the distribution, where the histogram shows that the data is skewed left. While uncommon, there may be instances where a patient falls in the same age range as the outlier points. Therefore, the outliers were not removed from the data set.

Similar to age, height and weight are physical traits used to measure a patient's general health. While there are some exceptions based on muscle mass and other factors, in general, an individual's weight can indicate their health status based on their height. Figure 4 shows a scatterplot, colored by the target variable, of the relationship between height and weight in the data set. Typically, an individual determined to be unhealthy is either heavier set and shorter or taller and lighter in weight. If this relationship had a confounding effect on the data, it is expected that the points representing death follow a diagonal line with a negative slope. However, the plot shows that instances where the patient died are randomly scattered throughout the plot. Thus, the relationship between height and weight, described above, does not influence the data set. Figures in the appendix show the individual distributions of height and weight, with their relationship to the target variable.

One aspect of health is the patient’s ethnic background. It is known that ethnicity, and therefore genetics, can lead to an inherent predisposition for illnesses and immunity. Thus, when examining the demographics of patients in the data set, it is key to also study the relationship between ethnicity and the target feature. Figure 5 shows two sets of bar charts. On the left, there

is a bar chart of frequency for each category in the variable, ethnicity. On the right, the normalized shows the proportion of the target variable by each category in the predictor. The plots reveal that the majority of data is represented by Caucasians. While the differences in total count of each ethnicity may lead to some selective biases, the proportion of death and non-death instances are roughly the same across ethnic classes. Thus, there is limited influence by ethnicity on mortality rate, as the probability of death is approximately equal by each ethic class.

## Preparing the Data for Modeling

The severe class imbalance was resolved using down sampling since all models had difficulty predicting the minority class with the class imbalance. Before resampling, our training set consisted of 71,195 observations. Upsampling would limit our modeling capabilities due to computing and memory constraints related to working with a training set with more than 120,000 observations. After downsampling to balance our classes, our training set consisted of 12,332 observations - resulting in significant and proportional model performance and computational time improvement.

Based on the aim to explore multiple modeling approaches, training and test sets were developed with respect to varying model assumptions. Some algorithms require all features to be numeric; therefore, non-numeric categorical features were encoded using the One-Hot encoding methodology. The first encoding for each feature was dropped to mitigate collinearity concerns and assist with feature reduction. Next, linear dependence was evaluated to mitigate multi-collinearity concerns. Twelve features were identified as linear dependent - these features represent duplicate observations of APACHE II and III body system values. Since APACHE III represented the most updated versions, they were retained and the APACHE II versions were

dropped. This resulted in a second training and test set named training_with_dummies_r and testing_with_dummies_r.

To summarize our training and testing sets, X_train, X_test, y_train, and y_test sets consisted of the non-encoded categorical features for use in tree-based classifiers after converting categorical features to factors. The training_with_dummies_r and testing_with_dummies_r with their corresponding y_train and y_test sets consisted of the one-hot encoded, non-linear dependent features for use in the non-tree-based classifiers.

## Model Strategies

The main objective of this study was to accurately classify patients into a binary category, “no_death” or “death”. Given this objective, a logistic regression model seemed reasonable. Thus, logistic regression was the first model constructed, and acted as the baseline model. Discriminant analysis and logistic regression are similar in that they both produce probabilities for classification into each outcome class. However, discriminant analysis assumes a normal distribution in the predictors. In addition, linear discriminant analysis (LDA) and partial least squares discriminant analysis (PLSDA) estimate the coefficients differently than logistic regression (Antonogeorgos et al., 2009). Mixture Discriminant Analysis (MDA) was also explored as a possible solution since it is a non-linear form of discriminant analysis. It was also thought that penalizing the logistic regression model may enhance its performance.

To evaluate a full range of classification models, decision trees, a random forest, a boosted tree, a K-Nearest Neighbors (KNN), and a neural network model were also constructed and evaluated. Unlike logistic regression, LDA, and PLSDA, decision trees were chosen based on their ability to make informed decisions through evaluations of the most significant features at each node. However, because a simple decision tree model can be prone to overfitting, a random

forest model and boosted tree model were also utilized. While the two models suffer from a limited interpretability, it was necessary to evaluate models that would reduce the weaknesses of a standard decision tree, such as the decision tree's high variance.

KNN was utilized to determine if the data had any underlying groupings that could predict the target variable. Where logistic regression was implemented based on the nature of the target variable, a KNN model was used based on the nature of the situation. In healthcare, illnesses are often diagnosed by symptoms. The more symptoms a patient has to a known disease, the more likely that patient has that disease. Thus, it is reasonable to say that the more similar a patient is to another, whose target level is known, the more likely the patient is to fall in the same target level. Because KNN is a similarity based model, it was deemed appropriate for the purposes of this study.

Finally a Neural Network (NN) approach was explored. NN's were developed to loosely mimic the human brain and consist of a set of algorithms designed to recognize patterns. Since labeled training data exists, the NN can be used for our classification problem. NN's help to cluster and classify by recognizing patterns and finding the correct manner of transforming predictors into response variables to optimize the classification. When additional layers (or nodes) are introduced its findings are summed and passed through a node's activation function to measure progress in classification performance. In this way, NN's identify the most significant features with regard to how the NN classifies and clusters input. It should be noted that NN's are computationally exhaustive and not very interpretable; however, their predictive capabilities are often exemplary, so it was explored.

All models were developed with consistent training control parameters which included 10-fold cross validation, two class summaries, and calculation and saving of class probabilities.

Also, given the objective of correctly classifying patients likely to die in order to provide life-saving treatment, models were trained to optimize for ROC. Sensitivity is used for final model selection since it is the proportion of true positives tests out of all patients with a condition. In other words, it is the ability of the model to yield a positive result for a subject that is likely to die.

## Validation and Testing

To combat the phenomena of underfitting and overfitting our models, the trade-off between bias and variance was highly considered to minimize the risk when validating the classification models on the test data sets. Given the large data set and down sampling due to the risk of overfitting and constraint of computational time, the chosen models were selected that would best predict the response variable of "death" and "no death". Whereas, for model tuning all the models had hyperparameters that allowed adjustment for prediction accuracy and computational time as the dataset included many features and observations. These adjustments were initially applied to the training data set to identify optimal performance metrics to run the testing data set for evaluation of performance on classification.

With the ten chosen models, the hyperparameters varied between each to decrease computational time, increase the performance of accuracy, and chosen metric of ROC. These hyperparameters include setting the training control (ctrl) with cross-validation, two class summary, and centering and scaling for each model. The Logistic Regression model does not have hyperparameters to adjust. The Penalized Logistic Regression included had tuneGrid to optimize the alpha (.2, .4, .6) and lambda (length = 2) parameters for regularization. LDA only varied with the preproc() function with Box-Cox transformations, PLSDA was adjusted with the number of components (ncomp = 1:10), and MDA had the parameter of subclasses with tuneGrid

(subclasses=1:3) to set the number of distributions per class. For the Decision Tree (CART), the tuneLength was set to a value of 30. The Random Forest model had the mtry parameter which was set to 7 (mtry = sqrt(p)). The Boosted Tree included the tuneGrid() function, where the the maximum number of nodes per tree, number of trees, shrinkage parameter, and minimum number of observations in trees' terminal nodes can be set (interaction depth = c(5), n.trees = (1:20)*5, shrinkage = c(.01, .1), and minobsinnode = 5). As for KNN, the hyperparameter of the number of neighbors (k) was set between 111 to 151 with increments of 2. Finally, the Neural Net model with hyperparameters of size (number of units in hidden layer) and decay (regularization parameter) are set within the tuneGrid() function (size=1:2 and decay=c(0,0.1,1). To evaluate the models, it can be observed in Table 1 of the *Results and Final Model Selection* where the ROC, Specificity, and Sensitivity metrics were compared between the training and testing data sets.

## Results and Final Model Selection

After training each model, an ROC curve was produced. Figure 6 shows all the ROC curves plotted on the same graph. Ideally, a model would have an ROC curve that follows the y-axis vertically, then moves horizontally once sensitivity equals one, thus maximizing the area under the curve (AUC). However, the form of the curve described previously is not always achievable. Fig. 6 shows that the ROC curves do take on a positive, concave shape, an expected shape for non-ideal situations. Based on these curves, the best models would be those with the largest AUC. Without exact metrics however, the curves look nearly identical. Table 1 shows that the best three models, based on the ROC curves, are the boosted tree model (AUC = 0.8746), the random forest model (AUC = 0.8634), and the logistic regression mode (AUC = 0.8579).

In addition to AUC, sensitivity is another metric that can be used to measure a model's predictive ability. As described previously, sensitivity is a measure of the ability to correctly predict the positive class. In this study, sensitivity is a measure of how well the model can predict a patient whose target level is "death", given that the patient does belong to the class "death." Of the top three models, based on ROC, the random forest and boosted tree models had the highest sensitivity on the training set (sensitivity = 0.7955). Of the two models, based on the test data, the random forest model had the highest sensitivity (0.7988), whereas the sensitivity for the boosted tree model was equal to 0.7917.

Though the random forest model was a higher sensitivity metric on the test data, the value was not that different from the sensitivity metric of the boosted tree model. Even though the random forest model outperformed the boosted tree model in one metric, the boosted tree model performed as good or better than the random forest model for every other metric, presented in Table 1. Due to computational constraints, a wider array of hyperparameters could not be examined for each model. A wider exploration of hyperparameters could potentially help distinguish the better model between the two. However, with the constraints, the boosted tree model performed better on all but one metric. Therefore, for that reason and the fact that its weakest metric was not far off from the best value, the boosted tree model was selected as the best model.

## Discussion and Conclusions

The objective of this study was to identify suitable model(s) that could correctly identify high risk patients. Of the 10 models tested, nine of which had sensitivity values above 75% percent; the KNN model had a sensitivity value of 0.6030 on the test data. Thus, this study found multiple models capable of achieving the objective. Given greater computational power, the

study could have expanded the number of hyperparameters tested for each model, thus bettering each model's performance. The work done here contributes to the ongoing discussion of machine learning algorithms in the healthcare industry. With over 75% probability of correctly classifying high risk cases, the models in this work show that machine learning is capable of assessing and identifying patients that have a high risk for mortality. Where previous models, as discussed earlier, were highly successful at identifying patients on the basis of a common illness, these models indicate that, given the appropriate features and computational strength, such models can be expanded to apply to patients with varying conditions. Having such a technology could greatly help healthcare workers decide where to allocate resources. Through this technology, common features that are shared among high risk patients can be identified, thus highlighting areas of research that would benefit from increased funding. Continuing this study and improving upon the work done here will have numerous benefits for many industries and populations.

**References**

Antonogeorgos, G., Panagiotakos, D. B., Priftis, K. N., & Tzonou, A. (2009). Logistic regression and linear discriminant analyses in evaluating factors associated with asthma prevalence among 10-to 12-years-old children: divergence and similarity of the two statistical methods. *International journal of pediatrics, 2009*.

Green, J., Wintfeld, N., Sharkey, P., & Passman, L. J. (1990). The importance of severity of illness in assessing hospital mortality. *JAMA, 263*(2), 241-246.

Hsu, C. W., Wann, S. R., Chiang, H. T., Lin, C. H., Kung, M. H., & Lin, S. L. (2001). Comparison of the APACHE II and APACHE III scoring systems in patients with respiratory failure in a medical intensive care unit. *Journal of the Formosan Medical Association, 100*(7), 437-442.

Kuhn, M., & Johnson, K. (2013). *Applied Predictive Modeling*. New York: Springer.

Pourhomayoun, M., & Shakibi, M. (2021). Predicting mortality risk in patients with COVID-19 using machine learning to help medical decision-making. *Smart Health, 20*, 100178.

Schafer JL. Multiple imputation: a primer. *Stat Methods in Med*. 1999;8(1):3–15. doi: 10.1191/096228099671525676.

Steele, A. J., Denaxas, S. C., Shah, A. D., Hemingway, H., & Luscombe, N. M. (2018). Machine learning models in electronic health records can outperform conventional survival models for predicting patient mortality in coronary artery disease. *PloS one, 13*(8), e0202344.

# Appendix

## Figure 1

*Bar Charts of the Target Feature Isolated and Associated with ICU Stay Type*

## Figure 2

*Stack Bar Charts of Categories for Each Apache System*

**Figure 3**

*Distribution of Age*

**Figure 4**

*Scatterplot of Height and Weight By the Target Feature*

**Figure 5**

*Bar Charts of Ethnicity*

**Figure 6**

*ROC Curves for Each Model Based on Training Data*

**Table 1**

*Performance Metrics of Each Model*

| Models | ROC..Train | Sens. . . .Train | Spec. . . .Train | Sens. . . .Test | Spec. . . .Test |
|---|---|---|---|---|---|
| Logistic Regression | 0.8579 | 0.7683 | 0.7830 | 0.7611 | 0.7972 |
| Penalized Logistic Regression | 0.8576 | 0.7613 | 0.7835 | 0.7578 | 0.7982 |
| LDA | 0.8574 | 0.7624 | 0.7848 | 0.7533 | 0.7991 |
| PLSDA | 0.8565 | 0.7606 | 0.7843 | 0.7507 | 0.7985 |
| MDA | 0.8559 | 0.7588 | 0.7836 | 0.7513 | 0.7989 |
| Decision Tree | 0.8167 | 0.7634 | 0.7499 | 0.7533 | 0.7547 |
| Random Forest | 0.8634 | 0.7955 | 0.7699 | 0.7988 | 0.7777 |
| Boosted Tree | 0.8746 | 0.7955 | 0.7851 | 0.7917 | 0.8000 |
| KNN | 0.8401 | 0.5981 | 0.8766 | 0.6030 | 0.8795 |
| Neural Net | 0.8571 | 0.7833 | 0.7681 | 0.7910 | 0.7710 |

*Note.* Sens. is an abbreviation for sensitivity and Spec. is an abbreviation for specificity. Columns ending in “Train” indicate that the values are based on the training data, whereas columns ending in “Test” are based on the test data set.

# ADS-503 Final Project

Roberto Cancel, Kyle Estaban Dalope, Nicholas Lee

6/27/2022

## Data Import

```r
## Load in Data - Convert Blank Cells to NA
patient_data <- read.csv("dataset.csv", na.strings = c("","NA"))
```

```r
## Load in File Containing Varible Definitions ##
data_dictionary <- read.csv("Dataset Dictionary - updated.csv")
```

## Handling Missing Data

```r
summary(patient_data)
```

```
##   encounter_id      patient_id      hospital_id         age
##  Min.   :     1   Min.   :     1   Min.   :  2.0   Min.   :16.00
##  1st Qu.: 32852   1st Qu.: 32830   1st Qu.: 47.0   1st Qu.:52.00
##  Median : 65665   Median : 65413   Median :109.0   Median :65.00
##  Mean   : 65606   Mean   : 65537   Mean   :105.7   Mean   :62.31
##  3rd Qu.: 98342   3rd Qu.: 98298   3rd Qu.:161.0   3rd Qu.:75.00
##  Max.   :131051   Max.   :131051   Max.   :204.0   Max.   :89.00
##                                                    NA's   :4228
##       bmi        elective_surgery  ethnicity           gender
##  Min.   :14.85   Min.   :0.0000   Length:91713       Length:91713
##  1st Qu.:23.64   1st Qu.:0.0000   Class :character   Class :character
##  Median :27.66   Median :0.0000   Mode  :character   Mode  :character
##  Mean   :29.19   Mean   :0.1837
##  3rd Qu.:32.93   3rd Qu.:0.0000
##  Max.   :67.81   Max.   :1.0000
##  NA's   :3429
##      height      icu_admit_source       icu_id      icu_stay_type
##  Min.   :137.2   Length:91713       Min.   : 82.0   Length:91713
##  1st Qu.:162.5   Class :character   1st Qu.:369.0   Class :character
##  Median :170.1   Mode  :character   Median :504.0   Mode  :character
##  Mean   :169.6                      Mean   :508.4
##  3rd Qu.:177.8                      3rd Qu.:679.0
##  Max.   :195.6                      Max.   :927.0
##  NA's   :1334
```

```
##    icu_type         pre_icu_los_days        weight       apache_2_diagnosis
##  Length:91713       Min.   :-24.94722   Min.   : 38.60   Min.   :101.0
##  Class :character   1st Qu.:  0.03542   1st Qu.: 66.80   1st Qu.:113.0
##  Mode  :character   Median :  0.13889   Median : 80.30   Median :122.0
##                     Mean   :  0.83577   Mean   : 84.03   Mean   :185.4
##                     3rd Qu.:  0.40903   3rd Qu.: 97.10   3rd Qu.:301.0
##                     Max.   :159.09097   Max.   :186.00   Max.   :308.0
##                                         NA's   :2720     NA's   :1662
##  apache_3j_diagnosis apache_post_operative   arf_apache    gcs_eyes_apache
##  Min.   :   0.01     Min.   :0.0000        Min.   :0.000   Min.   :1.000
##  1st Qu.: 203.01     1st Qu.:0.0000        1st Qu.:0.000   1st Qu.:3.000
##  Median : 409.02     Median :0.0000        Median :0.000   Median :4.000
##  Mean   : 558.22     Mean   :0.2011        Mean   :0.028   Mean   :3.465
##  3rd Qu.: 703.03     3rd Qu.:0.0000        3rd Qu.:0.000   3rd Qu.:4.000
##  Max.   :2201.05     Max.   :1.0000        Max.   :1.000   Max.   :4.000
##  NA's   :1101                              NA's   :715     NA's   :1901
##  gcs_motor_apache gcs_unable_apache gcs_verbal_apache heart_rate_apache
##  Min.   :1.000    Min.   :0.0000    Min.   :1.000     Min.   : 30.00
##  1st Qu.:6.000    1st Qu.:0.0000    1st Qu.:4.000     1st Qu.: 86.00
##  Median :6.000    Median :0.0000    Median :5.000     Median :104.00
##  Mean   :5.471    Mean   :0.0095    Mean   :3.995     Mean   : 99.71
##  3rd Qu.:6.000    3rd Qu.:0.0000    3rd Qu.:5.000     3rd Qu.:120.00
##  Max.   :6.000    Max.   :1.0000    Max.   :5.000     Max.   :178.00
##  NA's   :1901     NA's   :1037      NA's   :1901      NA's   :878
##  intubated_apache   map_apache     resprate_apache  temp_apache
##  Min.   :0.0000   Min.   : 40.00   Min.   : 4.00   Min.   :32.10
##  1st Qu.:0.0000   1st Qu.: 54.00   1st Qu.:11.00   1st Qu.:36.20
##  Median :0.0000   Median : 67.00   Median :28.00   Median :36.50
##  Mean   :0.1512   Mean   : 88.02   Mean   :25.81   Mean   :36.41
##  3rd Qu.:0.0000   3rd Qu.:125.00   3rd Qu.:36.00   3rd Qu.:36.70
##  Max.   :1.0000   Max.   :200.00   Max.   :60.00   Max.   :39.70
##  NA's   :715      NA's   :994      NA's   :1234    NA's   :4108
##  ventilated_apache d1_diasbp_max    d1_diasbp_min   d1_diasbp_noninvasive_max
##  Min.   :0.0000    Min.   : 46.00   Min.   :13.00   Min.   : 46.00
##  1st Qu.:0.0000    1st Qu.: 75.00   1st Qu.:42.00   1st Qu.: 75.00
##  Median :0.0000    Median : 86.00   Median :50.00   Median : 87.00
##  Mean   :0.3257    Mean   : 88.49   Mean   :50.16   Mean   : 88.61
##  3rd Qu.:1.0000    3rd Qu.: 99.00   3rd Qu.:58.00   3rd Qu.: 99.00
##  Max.   :1.0000    Max.   :165.00   Max.   :90.00   Max.   :165.00
##  NA's   :715       NA's   :165      NA's   :165     NA's   :1040
##  d1_diasbp_noninvasive_min d1_heartrate_max d1_heartrate_min   d1_mbp_max
##  Min.   :13.00             Min.   : 58      Min.   :  0.00   Min.   : 60.0
##  1st Qu.:42.00             1st Qu.: 87      1st Qu.: 60.00   1st Qu.: 90.0
##  Median :50.00             Median :101      Median : 69.00   Median :102.0
##  Mean   :50.24             Mean   :103      Mean   : 70.32   Mean   :104.7
##  3rd Qu.:58.00             3rd Qu.:116      3rd Qu.: 81.00   3rd Qu.:116.0
##  Max.   :90.00             Max.   :177      Max.   :175.00   Max.   :184.0
##  NA's   :1040              NA's   :145      NA's   :145      NA's   :220
##    d1_mbp_min     d1_mbp_noninvasive_max d1_mbp_noninvasive_min d1_resprate_max
##  Min.   : 22.00   Min.   : 60.0          Min.   : 22.00         Min.   :14.00
##  1st Qu.: 55.00   1st Qu.: 90.0          1st Qu.: 55.00         1st Qu.:22.00
##  Median : 64.00   Median :102.0          Median : 64.00         Median :26.00
##  Mean   : 64.87   Mean   :104.6          Mean   : 64.94         Mean   :28.88
##  3rd Qu.: 75.00   3rd Qu.:116.0          3rd Qu.: 75.00         3rd Qu.:32.00
```

```
##  Max.   :112.00   Max.   :181.0          Max.   :112.00         Max.   :92.00
##  NA's   :220      NA's   :1479           NA's   :1479           NA's   :385
##  d1_resprate_min   d1_spo2_max      d1_spo2_min      d1_sysbp_max
##  Min.   :  0.00   Min.   :  0.00   Min.   :  0.00   Min.   : 90.0
##  1st Qu.: 10.00   1st Qu.: 99.00   1st Qu.: 89.00   1st Qu.:130.0
##  Median : 13.00   Median :100.00   Median : 92.00   Median :146.0
##  Mean   : 12.85   Mean   : 99.24   Mean   : 90.45   Mean   :148.3
##  3rd Qu.: 16.00   3rd Qu.:100.00   3rd Qu.: 95.00   3rd Qu.:164.0
##  Max.   :100.00   Max.   :100.00   Max.   :100.00   Max.   :232.0
##  NA's   :385      NA's   :333      NA's   :333      NA's   :159
##   d1_sysbp_min    d1_sysbp_noninvasive_max d1_sysbp_noninvasive_min
##  Min.   : 41.00   Min.   : 90.0            Min.   : 41.03
##  1st Qu.: 83.00   1st Qu.:130.0            1st Qu.: 84.00
##  Median : 96.00   Median :146.0            Median : 96.00
##  Mean   : 96.92   Mean   :148.2            Mean   : 96.99
##  3rd Qu.:110.00   3rd Qu.:164.0            3rd Qu.:110.00
##  Max.   :160.00   Max.   :232.0            Max.   :160.00
##  NA's   :159      NA's   :1027             NA's   :1027
##   d1_temp_max     d1_temp_min    h1_diasbp_max    h1_diasbp_min
##  Min.   :35.10   Min.   :31.89   Min.   : 37.00   Min.   : 22.00
##  1st Qu.:36.90   1st Qu.:36.10   1st Qu.: 62.00   1st Qu.: 52.00
##  Median :37.11   Median :36.40   Median : 74.00   Median : 62.00
##  Mean   :37.28   Mean   :36.27   Mean   : 75.36   Mean   : 62.84
##  3rd Qu.:37.60   3rd Qu.:36.66   3rd Qu.: 86.00   3rd Qu.: 73.00
##  Max.   :39.90   Max.   :37.80   Max.   :143.00   Max.   :113.00
##  NA's   :2324    NA's   :2324    NA's   :3619     NA's   :3619
##  h1_diasbp_noninvasive_max h1_diasbp_noninvasive_min h1_heartrate_max
##  Min.   : 37.00            Min.   : 22.00            Min.   : 46.00
##  1st Qu.: 63.00            1st Qu.: 52.00            1st Qu.: 77.00
##  Median : 74.00            Median : 62.00            Median : 90.00
##  Mean   : 75.81            Mean   : 63.27            Mean   : 92.23
##  3rd Qu.: 87.00            3rd Qu.: 74.00            3rd Qu.:106.00
##  Max.   :144.00            Max.   :114.00            Max.   :164.00
##  NA's   :7350              NA's   :7350              NA's   :2790
##  h1_heartrate_min   h1_mbp_max       h1_mbp_min    h1_mbp_noninvasive_max
##  Min.   : 36.00   Min.   : 49.00   Min.   : 32.0   Min.   : 49.00
##  1st Qu.: 69.00   1st Qu.: 77.00   1st Qu.: 66.0   1st Qu.: 77.00
##  Median : 82.00   Median : 90.00   Median : 78.0   Median : 90.00
##  Mean   : 83.66   Mean   : 91.61   Mean   : 79.4   Mean   : 91.59
##  3rd Qu.: 97.00   3rd Qu.:104.00   3rd Qu.: 92.0   3rd Qu.:104.00
##  Max.   :144.00   Max.   :165.00   Max.   :138.0   Max.   :163.00
##  NA's   :2790     NA's   :4639     NA's   :4639    NA's   :9084
##  h1_mbp_noninvasive_min h1_resprate_max h1_resprate_min   h1_spo2_max
##  Min.   : 32.00         Min.   :10.00   Min.   :  0.00   Min.   :  0.00
##  1st Qu.: 66.00         1st Qu.:18.00   1st Qu.: 14.00   1st Qu.: 97.00
##  Median : 79.00         Median :21.00   Median : 16.00   Median : 99.00
##  Mean   : 79.71         Mean   :22.63   Mean   : 17.21   Mean   : 98.05
##  3rd Qu.: 92.00         3rd Qu.:26.00   3rd Qu.: 20.00   3rd Qu.:100.00
##  Max.   :138.00         Max.   :59.00   Max.   :189.00   Max.   :100.00
##  NA's   :9084           NA's   :4357    NA's   :4357     NA's   :4185
##   h1_spo2_min      h1_sysbp_max    h1_sysbp_min   h1_sysbp_noninvasive_max
##  Min.   :  0.00   Min.   : 75.0   Min.   : 53.0   Min.   : 75.0
##  1st Qu.: 94.00   1st Qu.:113.0   1st Qu.: 98.0   1st Qu.:113.0
##  Median : 96.00   Median :131.0   Median :115.0   Median :130.0
```

```
##  Mean   : 95.17   Mean   :133.2   Mean   :116.4   Mean   :133.1
##  3rd Qu.: 99.00   3rd Qu.:150.0   3rd Qu.:134.0   3rd Qu.:150.0
##  Max.   :100.00   Max.   :223.0   Max.   :194.0   Max.   :223.0
##  NA's   :4185     NA's   :3611    NA's   :3611    NA's   :7341
##  h1_sysbp_noninvasive_min d1_glucose_max  d1_glucose_min  d1_potassium_max
##  Min.   : 53.0            Min.   : 73.0   Min.   : 33.0   Min.   :2.800
##  1st Qu.: 98.0            1st Qu.:117.0   1st Qu.: 91.0   1st Qu.:3.800
##  Median :115.0            Median :150.0   Median :107.0   Median :4.200
##  Mean   :116.5            Mean   :174.6   Mean   :114.4   Mean   :4.252
##  3rd Qu.:134.0            3rd Qu.:201.0   3rd Qu.:131.0   3rd Qu.:4.600
##  Max.   :195.0            Max.   :611.0   Max.   :288.0   Max.   :7.000
##  NA's   :7341             NA's   :5807    NA's   :5807    NA's   :9585
##  d1_potassium_min apache_4a_hospital_death_prob apache_4a_icu_death_prob
##  Min.   :2.400    Min.   :-1.000                Min.   :-1.000
##  1st Qu.:3.600    1st Qu.: 0.020                1st Qu.: 0.010
##  Median :3.900    Median : 0.050                Median : 0.020
##  Mean   :3.935    Mean   : 0.087                Mean   : 0.044
##  3rd Qu.:4.300    3rd Qu.: 0.130                3rd Qu.: 0.060
##  Max.   :5.800    Max.   : 0.990                Max.   : 0.970
##  NA's   :9585     NA's   :7947                  NA's   :7947
##       aids         cirrhosis      diabetes_mellitus hepatic_failure
##  Min.   :0e+00   Min.   :0.0000   Min.   :0.0000    Min.   :0.000
##  1st Qu.:0e+00   1st Qu.:0.0000   1st Qu.:0.0000    1st Qu.:0.000
##  Median :0e+00   Median :0.0000   Median :0.0000    Median :0.000
##  Mean   :9e-04   Mean   :0.0157   Mean   :0.2252    Mean   :0.013
##  3rd Qu.:0e+00   3rd Qu.:0.0000   3rd Qu.:0.0000    3rd Qu.:0.000
##  Max.   :1e+00   Max.   :1.0000   Max.   :1.0000    Max.   :1.000
##  NA's   :715     NA's   :715      NA's   :715       NA's   :715
##  immunosuppression    leukemia         lymphoma
##  Min.   :0.0000    Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0000    1st Qu.:0.0000   1st Qu.:0.0000
##  Median :0.0000    Median :0.0000   Median :0.0000
##  Mean   :0.0262    Mean   :0.0071   Mean   :0.0041
##  3rd Qu.:0.0000    3rd Qu.:0.0000   3rd Qu.:0.0000
##  Max.   :1.0000    Max.   :1.0000   Max.   :1.0000
##  NA's   :715       NA's   :715      NA's   :715
##  solid_tumor_with_metastasis apache_3j_bodysystem apache_2_bodysystem
##  Min.   :0.0000              Length:91713         Length:91713
##  1st Qu.:0.0000              Class :character     Class :character
##  Median :0.0000              Mode  :character     Mode  :character
##  Mean   :0.0206
##  3rd Qu.:0.0000
##  Max.   :1.0000
##  NA's   :715
##     X           hospital_death
##  Mode:logical   Min.   :0.0000
##  NA's:91713     1st Qu.:0.0000
##                 Median :0.0000
##                 Mean   :0.0863
##                 3rd Qu.:0.0000
##                 Max.   :1.0000
##
```

```r
# Obtain and Drop id Columns #
id_columns <- grep("_id", colnames(patient_data))
patient_data <- patient_data[, -id_columns]
# Convert Outcome Variable to factor #
patient_data$hospital_death <- as.factor(patient_data$hospital_death)
# Drop NAs from gender #
patient_data <- patient_data[!is.na(patient_data$gender), ]
patient_data$gender <- ifelse(patient_data$gender  == "M", "1", "0")
patient_data$gender <- as.numeric(as.character(patient_data$gender))

# Distribution of Target Variable #
table(patient_data$hospital_death)
```

```
##
##     0     1
## 83781  7907
```

```r
## Roughly 8.6% of the data set is the positive class (1 = death)
```

```r
# Stratified Sampling #
set.seed(100)
trainingRows <- createDataPartition(patient_data$hospital_death,
                                    p = 0.80,
                                    list = FALSE)
## Subset Data into Training and Test Sets ##
patient_train <- patient_data[trainingRows, ]
patient_test <- patient_data[-trainingRows, ]
```

```r
summary(patient_train)
```

```
##       age            bmi        elective_surgery  ethnicity
##  Min.   :16.0   Min.   :14.84   Min.   :0.0000   Length:73351
##  1st Qu.:52.0   1st Qu.:23.64   1st Qu.:0.0000   Class :character
##  Median :65.0   Median :27.65   Median :0.0000   Mode  :character
##  Mean   :62.3   Mean   :29.18   Mean   :0.1833
##  3rd Qu.:75.0   3rd Qu.:32.93   3rd Qu.:0.0000
##  Max.   :89.0   Max.   :67.81   Max.   :1.0000
##  NA's   :3367   NA's   :2682
##      gender           height      icu_admit_source   icu_stay_type
##  Min.   :0.0000   Min.   :137.2   Length:73351       Length:73351
##  1st Qu.:0.0000   1st Qu.:162.5   Class :character   Class :character
##  Median :1.0000   Median :170.0   Mode  :character   Mode  :character
##  Mean   :0.5396   Mean   :169.6
##  3rd Qu.:1.0000   3rd Qu.:177.8
##  Max.   :1.0000   Max.   :195.6
##                   NA's   :1034
##    icu_type         pre_icu_los_days        weight       apache_2_diagnosis
##  Length:73351       Min.   :-24.94722   Min.   : 38.60   Min.   :101.0
##  Class :character   1st Qu.:  0.03542   1st Qu.: 66.70   1st Qu.:113.0
##  Mode  :character   Median :  0.13889   Median : 80.20   Median :122.0
##                     Mean   :  0.83731   Mean   : 84.01   Mean   :185.3
##                     3rd Qu.:  0.40903   3rd Qu.: 97.10   3rd Qu.:301.0
```

```
##                     Max.   :159.09097   Max.   :186.00   Max.   :308.0
##                                          NA's   :2138     NA's   :1330
##  apache_3j_diagnosis apache_post_operative   arf_apache     gcs_eyes_apache
##  Min.   :   0.01     Min.   :0.0000        Min.   :0.0000   Min.   :1.000
##  1st Qu.: 203.01     1st Qu.:0.0000        1st Qu.:0.0000   1st Qu.:3.000
##  Median : 409.02     Median :0.0000        Median :0.0000   Median :4.000
##  Mean   : 558.26     Mean   :0.2011        Mean   :0.0283   Mean   :3.462
##  3rd Qu.: 703.03     3rd Qu.:0.0000        3rd Qu.:0.0000   3rd Qu.:4.000
##  Max.   :2201.05     Max.   :1.0000        Max.   :1.0000   Max.   :4.000
##  NA's   :883                               NA's   :570      NA's   :1523
##  gcs_motor_apache gcs_unable_apache gcs_verbal_apache heart_rate_apache
##  Min.   :1.000    Min.   :0.0000    Min.   :1.000     Min.   : 30.00
##  1st Qu.:6.000    1st Qu.:0.0000    1st Qu.:4.000     1st Qu.: 86.00
##  Median :6.000    Median :0.0000    Median :5.000     Median :104.00
##  Mean   :5.467    Mean   :0.0096    Mean   :3.989     Mean   : 99.79
##  3rd Qu.:6.000    3rd Qu.:0.0000    3rd Qu.:5.000     3rd Qu.:120.00
##  Max.   :6.000    Max.   :1.0000    Max.   :5.000     Max.   :178.00
##  NA's   :1523     NA's   :825       NA's   :1523      NA's   :693
##  intubated_apache   map_apache     resprate_apache  temp_apache
##  Min.   :0.0000   Min.   : 40.00   Min.   : 4.00   Min.   :32.10
##  1st Qu.:0.0000   1st Qu.: 54.00   1st Qu.:11.00   1st Qu.:36.20
##  Median :0.0000   Median : 67.00   Median :28.00   Median :36.50
##  Mean   :0.1515   Mean   : 88.08   Mean   :25.81   Mean   :36.41
##  3rd Qu.:0.0000   3rd Qu.:125.00   3rd Qu.:36.00   3rd Qu.:36.70
##  Max.   :1.0000   Max.   :200.00   Max.   :60.00   Max.   :39.70
##  NA's   :570      NA's   :779      NA's   :963     NA's   :3230
##  ventilated_apache d1_diasbp_max    d1_diasbp_min   d1_diasbp_noninvasive_max
##  Min.   :0.0000    Min.   : 46.00   Min.   :13.00   Min.   : 46.00
##  1st Qu.:0.0000    1st Qu.: 75.00   1st Qu.:42.00   1st Qu.: 75.00
##  Median :0.0000    Median : 86.00   Median :50.00   Median : 87.00
##  Mean   :0.3269    Mean   : 88.49   Mean   :50.16   Mean   : 88.61
##  3rd Qu.:1.0000    3rd Qu.: 99.00   3rd Qu.:58.00   3rd Qu.: 99.00
##  Max.   :1.0000    Max.   :165.00   Max.   :90.00   Max.   :165.00
##  NA's   :570       NA's   :120      NA's   :120     NA's   :839
##  d1_diasbp_noninvasive_min d1_heartrate_max d1_heartrate_min   d1_mbp_max
##  Min.   :13.00             Min.   : 58      Min.   :  0.00   Min.   : 60.0
##  1st Qu.:42.00             1st Qu.: 87      1st Qu.: 60.00   1st Qu.: 90.0
##  Median :50.00             Median :101      Median : 70.00   Median :102.0
##  Mean   :50.25             Mean   :103      Mean   : 70.35   Mean   :104.7
##  3rd Qu.:58.00             3rd Qu.:117      3rd Qu.: 81.00   3rd Qu.:116.0
##  Max.   :90.00             Max.   :177      Max.   :160.00   Max.   :184.0
##  NA's   :839               NA's   :108      NA's   :108      NA's   :165
##    d1_mbp_min     d1_mbp_noninvasive_max d1_mbp_noninvasive_min d1_resprate_max
##  Min.   : 22.00   Min.   : 60.0          Min.   : 22.00         Min.   :14.00
##  1st Qu.: 55.00   1st Qu.: 90.0          1st Qu.: 55.00         1st Qu.:22.00
##  Median : 64.00   Median :102.0          Median : 64.00         Median :26.00
##  Mean   : 64.85   Mean   :104.6          Mean   : 64.92         Mean   :28.89
##  3rd Qu.: 75.00   3rd Qu.:116.0          3rd Qu.: 75.00         3rd Qu.:32.00
##  Max.   :112.00   Max.   :181.0          Max.   :112.00         Max.   :92.00
##  NA's   :165      NA's   :1196           NA's   :1196           NA's   :296
##  d1_resprate_min   d1_spo2_max      d1_spo2_min      d1_sysbp_max
##  Min.   :  0.00   Min.   :  0.00   Min.   :  0.00   Min.   : 90.0
##  1st Qu.: 10.00   1st Qu.: 99.00   1st Qu.: 89.00   1st Qu.:130.0
##  Median : 13.00   Median :100.00   Median : 92.00   Median :146.0
```

```
##  Mean   : 12.85   Mean   : 99.25   Mean   : 90.46   Mean   :148.3
##  3rd Qu.: 16.00   3rd Qu.:100.00   3rd Qu.: 95.00   3rd Qu.:164.0
##  Max.   :100.00   Max.   :100.00   Max.   :100.00   Max.   :232.0
##  NA's   :296      NA's   :263      NA's   :263      NA's   :116
##   d1_sysbp_min    d1_sysbp_noninvasive_max d1_sysbp_noninvasive_min
##  Min.   : 41.00   Min.   : 90.0            Min.   : 41.03
##  1st Qu.: 83.00   1st Qu.:130.0            1st Qu.: 84.00
##  Median : 96.00   Median :146.0            Median : 96.00
##  Mean   : 96.91   Mean   :148.2            Mean   : 96.99
##  3rd Qu.:110.00   3rd Qu.:164.0            3rd Qu.:110.00
##  Max.   :160.00   Max.   :232.0            Max.   :160.00
##  NA's   :116      NA's   :829              NA's   :829
##   d1_temp_max     d1_temp_min    h1_diasbp_max    h1_diasbp_min
##  Min.   :35.10   Min.   :31.89   Min.   : 37.00   Min.   : 22.00
##  1st Qu.:36.90   1st Qu.:36.10   1st Qu.: 62.00   1st Qu.: 52.00
##  Median :37.11   Median :36.40   Median : 74.00   Median : 62.00
##  Mean   :37.29   Mean   :36.27   Mean   : 75.36   Mean   : 62.81
##  3rd Qu.:37.60   3rd Qu.:36.66   3rd Qu.: 86.00   3rd Qu.: 73.00
##  Max.   :39.90   Max.   :37.80   Max.   :143.00   Max.   :113.00
##  NA's   :1815    NA's   :1815    NA's   :2885     NA's   :2885
##  h1_diasbp_noninvasive_max h1_diasbp_noninvasive_min h1_heartrate_max
##  Min.   : 37.00            Min.   : 22.00            Min.   : 46.00
##  1st Qu.: 63.00            1st Qu.: 52.00            1st Qu.: 77.00
##  Median : 74.00            Median : 62.00            Median : 90.00
##  Mean   : 75.81            Mean   : 63.24            Mean   : 92.31
##  3rd Qu.: 87.00            3rd Qu.: 74.00            3rd Qu.:106.00
##  Max.   :144.00            Max.   :114.00            Max.   :164.00
##  NA's   :5850              NA's   :5850              NA's   :2231
##  h1_heartrate_min   h1_mbp_max       h1_mbp_min     h1_mbp_noninvasive_max
##  Min.   : 36.00   Min.   : 49.00   Min.   : 32.00   Min.   : 49.00
##  1st Qu.: 69.00   1st Qu.: 77.00   1st Qu.: 66.00   1st Qu.: 77.00
##  Median : 82.00   Median : 90.00   Median : 78.00   Median : 90.00
##  Mean   : 83.72   Mean   : 91.63   Mean   : 79.37   Mean   : 91.61
##  3rd Qu.: 97.00   3rd Qu.:104.00   3rd Qu.: 92.00   3rd Qu.:104.00
##  Max.   :144.00   Max.   :165.00   Max.   :138.00   Max.   :163.00
##  NA's   :2231     NA's   :3702     NA's   :3702     NA's   :7247
##  h1_mbp_noninvasive_min h1_resprate_max h1_resprate_min   h1_spo2_max
##  Min.   : 32.00         Min.   :10.00   Min.   :  0.00   Min.   :  0.00
##  1st Qu.: 66.00         1st Qu.:18.00   1st Qu.: 14.00   1st Qu.: 97.00
##  Median : 79.00         Median :21.00   Median : 16.00   Median : 99.00
##  Mean   : 79.68         Mean   :22.64   Mean   : 17.22   Mean   : 98.05
##  3rd Qu.: 92.00         3rd Qu.:26.00   3rd Qu.: 20.00   3rd Qu.:100.00
##  Max.   :138.00         Max.   :59.00   Max.   :189.00   Max.   :100.00
##  NA's   :7247           NA's   :3450    NA's   :3450     NA's   :3329
##   h1_spo2_min      h1_sysbp_max    h1_sysbp_min   h1_sysbp_noninvasive_max
##  Min.   :  0.00   Min.   : 75.0   Min.   : 53.0   Min.   : 75.0
##  1st Qu.: 94.00   1st Qu.:113.0   1st Qu.: 98.0   1st Qu.:113.0
##  Median : 96.00   Median :131.0   Median :115.0   Median :130.0
##  Mean   : 95.19   Mean   :133.2   Mean   :116.3   Mean   :133.1
##  3rd Qu.: 99.00   3rd Qu.:150.0   3rd Qu.:134.0   3rd Qu.:150.0
##  Max.   :100.00   Max.   :223.0   Max.   :194.0   Max.   :223.0
##  NA's   :3329     NA's   :2877    NA's   :2877    NA's   :5843
##  h1_sysbp_noninvasive_min d1_glucose_max  d1_glucose_min  d1_potassium_max
##  Min.   : 53.0            Min.   : 73.0   Min.   : 33.0   Min.   :2.800
```

```
##  1st Qu.: 98.0            1st Qu.:117.0   1st Qu.: 90.0   1st Qu.:3.800
##  Median :115.0            Median :150.0   Median :107.0   Median :4.200
##  Mean   :116.5            Mean   :174.6   Mean   :114.3   Mean   :4.252
##  3rd Qu.:134.0            3rd Qu.:201.0   3rd Qu.:131.0   3rd Qu.:4.600
##  Max.   :195.0            Max.   :611.0   Max.   :288.0   Max.   :7.000
##  NA's   :5843             NA's   :4630    NA's   :4630    NA's   :7650
##  d1_potassium_min apache_4a_hospital_death_prob apache_4a_icu_death_prob
##  Min.   :2.400    Min.   :-1.000                Min.   :-1.000
##  1st Qu.:3.600    1st Qu.: 0.020                1st Qu.: 0.010
##  Median :3.900    Median : 0.050                Median : 0.020
##  Mean   :3.936    Mean   : 0.088                Mean   : 0.045
##  3rd Qu.:4.300    3rd Qu.: 0.130                3rd Qu.: 0.060
##  Max.   :5.800    Max.   : 0.980                Max.   : 0.970
##  NA's   :7650     NA's   :6356                  NA's   :6356
##       aids         cirrhosis      diabetes_mellitus hepatic_failure
##  Min.   :0e+00   Min.   :0.0000   Min.   :0.0000    Min.   :0.000
##  1st Qu.:0e+00   1st Qu.:0.0000   1st Qu.:0.0000    1st Qu.:0.000
##  Median :0e+00   Median :0.0000   Median :0.0000    Median :0.000
##  Mean   :9e-04   Mean   :0.0156   Mean   :0.2255    Mean   :0.013
##  3rd Qu.:0e+00   3rd Qu.:0.0000   3rd Qu.:0.0000    3rd Qu.:0.000
##  Max.   :1e+00   Max.   :1.0000   Max.   :1.0000    Max.   :1.000
##  NA's   :570     NA's   :570      NA's   :570       NA's   :570
##  immunosuppression    leukemia         lymphoma
##  Min.   :0.0000    Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0000    1st Qu.:0.0000   1st Qu.:0.0000
##  Median :0.0000    Median :0.0000   Median :0.0000
##  Mean   :0.0264    Mean   :0.0071   Mean   :0.0041
##  3rd Qu.:0.0000    3rd Qu.:0.0000   3rd Qu.:0.0000
##  Max.   :1.0000    Max.   :1.0000   Max.   :1.0000
##  NA's   :570       NA's   :570      NA's   :570
##  solid_tumor_with_metastasis apache_3j_bodysystem apache_2_bodysystem
##  Min.   :0.0000              Length:73351         Length:73351
##  1st Qu.:0.0000              Class :character     Class :character
##  Median :0.0000              Mode  :character     Mode  :character
##  Mean   :0.0207
##  3rd Qu.:0.0000
##  Max.   :1.0000
##  NA's   :570
##     X           hospital_death
##  Mode:logical   0:67025
##  NA's:73351     1: 6326
##
##
##
##
##
```

```
sort(sapply(patient_train, function(x) sum(is.na(x))), decreasing = TRUE)
```

```
##                             X              d1_potassium_max
##                         73351                          7650
##              d1_potassium_min        h1_mbp_noninvasive_max
##                          7650                          7247
##        h1_mbp_noninvasive_min apache_4a_hospital_death_prob
```

```
##                          7247                          6356
##      apache_4a_icu_death_prob     h1_diasbp_noninvasive_max
##                          6356                          5850
##     h1_diasbp_noninvasive_min      h1_sysbp_noninvasive_max
##                          5850                          5843
##      h1_sysbp_noninvasive_min                d1_glucose_max
##                          5843                          4630
##                d1_glucose_min                    h1_mbp_max
##                          4630                          3702
##                    h1_mbp_min                h1_resprate_max
##                          3702                          3450
##               h1_resprate_min                           age
##                          3450                          3367
##                   h1_spo2_max                   h1_spo2_min
##                          3329                          3329
##                  temp_apache                 h1_diasbp_max
##                          3230                          2885
##                 h1_diasbp_min                   h1_sysbp_max
##                          2885                          2877
##                  h1_sysbp_min                           bmi
##                          2877                          2682
##              h1_heartrate_max              h1_heartrate_min
##                          2231                          2231
##                        weight                   d1_temp_max
##                          2138                          1815
##                   d1_temp_min                gcs_eyes_apache
##                          1815                          1523
##              gcs_motor_apache              gcs_verbal_apache
##                          1523                          1523
##            apache_2_diagnosis          apache_3j_bodysystem
##                          1330                          1330
##           apache_2_bodysystem        d1_mbp_noninvasive_max
##                          1330                          1196
##        d1_mbp_noninvasive_min                     ethnicity
##                          1196                          1103
##                        height                resprate_apache
##                          1034                           963
##           apache_3j_diagnosis     d1_diasbp_noninvasive_max
##                           883                           839
##     d1_diasbp_noninvasive_min      d1_sysbp_noninvasive_max
##                           839                           829
##      d1_sysbp_noninvasive_min              gcs_unable_apache
##                           829                           825
##                    map_apache              heart_rate_apache
##                           779                           693
##                   arf_apache               intubated_apache
##                           570                           570
##             ventilated_apache                          aids
##                           570                           570
##                     cirrhosis              diabetes_mellitus
##                           570                           570
##               hepatic_failure              immunosuppression
##                           570                           570
##                      leukemia                      lymphoma
```

```
##                          570                          570
##   solid_tumor_with_metastasis               d1_resprate_max
##                          570                          296
##               d1_resprate_min                   d1_spo2_max
##                          296                          263
##                   d1_spo2_min                   d1_mbp_max
##                          263                          165
##                    d1_mbp_min                 d1_diasbp_max
##                          165                          120
##                 d1_diasbp_min                  d1_sysbp_max
##                          120                          116
##                  d1_sysbp_min              d1_heartrate_max
##                          116                          108
##              d1_heartrate_min              icu_admit_source
##                          108                           86
##              elective_surgery                       gender
##                            0                            0
##                 icu_stay_type                     icu_type
##                            0                            0
##              pre_icu_los_days         apache_post_operative
##                            0                            0
##                hospital_death
##                            0
```

Given the high number of missing values, we will remove all features with more that 5% missingness. This is based on Schafer (1999) asserted that a missing rate of 5% or less is inconsequential.

```r
# Remove Columns with 5% or more NA #
patient_train_r <- patient_train[, !sapply(patient_train,
                                           function(x) mean(is.na(x)) > 0.05)]
# Drop the same Columns from the Test Set #
patient_test_r <- patient_test[, colnames(patient_train_r)]
```

```r
subset <- patient_train_r[is.na(patient_train_r$aids), ]
sort(colSums(is.na(subset)), decreasing = TRUE)
```

```
##                        age          apache_2_diagnosis
##                        570                         570
##         apache_3j_diagnosis                  arf_apache
##                        570                         570
##             gcs_eyes_apache            gcs_motor_apache
##                        570                         570
##           gcs_unable_apache           gcs_verbal_apache
##                        570                         570
##           heart_rate_apache            intubated_apache
##                        570                         570
##                  map_apache             resprate_apache
##                        570                         570
##                 temp_apache           ventilated_apache
##                        570                         570
##                       aids                    cirrhosis
##                        570                         570
##           diabetes_mellitus             hepatic_failure
```

```
##                        570                        570
##          immunosuppression                   leukemia
##                        570                        570
##                   lymphoma solid_tumor_with_metastasis
##                        570                        570
##       apache_3j_bodysystem        apache_2_bodysystem
##                        570                        570
##                h1_spo2_max                h1_spo2_min
##                         49                         49
##              h1_diasbp_max              h1_diasbp_min
##                         43                         43
##               h1_sysbp_max               h1_sysbp_min
##                         43                         43
##            h1_resprate_max            h1_resprate_min
##                         42                         42
##           h1_heartrate_max           h1_heartrate_min
##                         34                         34
##                        bmi                     weight
##                         23                         19
##                d1_temp_max                d1_temp_min
##                         16                         16
##                     height     d1_mbp_noninvasive_max
##                         13                         12
##     d1_mbp_noninvasive_min  d1_diasbp_noninvasive_max
##                         12                         11
##  d1_diasbp_noninvasive_min   d1_sysbp_noninvasive_max
##                         11                         11
##   d1_sysbp_noninvasive_min                  ethnicity
##                         11                          6
##                d1_spo2_max                d1_spo2_min
##                          6                          6
##            d1_resprate_max            d1_resprate_min
##                          5                          5
##              d1_diasbp_max              d1_diasbp_min
##                          4                          4
##           d1_heartrate_max           d1_heartrate_min
##                          4                          4
##                 d1_mbp_max                 d1_mbp_min
##                          4                          4
##               d1_sysbp_max               d1_sysbp_min
##                          4                          4
##           elective_surgery                     gender
##                          0                          0
##           icu_admit_source              icu_stay_type
##                          0                          0
##                   icu_type           pre_icu_los_days
##                          0                          0
##      apache_post_operative             hospital_death
##                          0                          0
```

Missing data appears to be related to the missingness in illness-related features.

```r
patient_train_r <- patient_train_r[!is.na(patient_train_r$aids), ]
patient_test_r <- patient_test_r[!is.na(patient_test_r$aids), ]
```

```r
#Drop the observations with missing apache_2 data
patient_train_r2 <- patient_train_r[!is.na(patient_train_r$apache_2_diagnosis), ]
patient_test_r2 <- patient_test_r[!is.na(patient_test_r$apache_2_diagnosis), ]

#Drop the observations with missing apache_3j_diagnosis
patient_train_r3 <- patient_train_r2[!is.na(patient_train_r$apache_2_diagnosis), ]
patient_test_r3 <- patient_test_r2[!is.na(patient_test_r$apache_2_diagnosis), ]
```

```r
patient_train_r3$ethnicity[is.na(patient_train_r3$ethnicity)] <- "Other/Unknown"
patient_test_r3$ethnicity[is.na(patient_test_r3$ethnicity)] <- "Other/Unknown"
```

```r
imp <- preProcess(patient_train_r3, method = c("medianImpute", "nzv", "corr"), cutoff = .8)
train_pr = predict(imp, patient_train_r3)
test_pr = predict(imp, patient_test_r3)
```

```r
train_pr <- train_pr[complete.cases(train_pr), ]
sum(is.na(train_pr))
```

```
## [1] 0
```

```r
test_pr <- test_pr[complete.cases(test_pr), ]
sum(is.na(test_pr))
```

```
## [1] 0
```

# EDA

```r
p1 <- ggplot(train_pr, aes(x = '', fill = hospital_death)) +
    geom_bar(position = "fill") +
    ggtitle("Proportion of Hospital Death Factors") +
    xlab("Hospital Death") + coord_flip()

p2 <- ggplot(train_pr, aes(x = icu_stay_type, fill = hospital_death)) +
    geom_bar(position = "fill") +
    ggtitle("Proportion of Death by ICU Stay Type") +
    xlab("ICU Stay Type") + coord_flip()

p1 / p2
```

```
p3 <- ggplot(train_pr, aes(x = apache_3j_bodysystem, fill = hospital_death)) +
    geom_bar(position = "fill") +
    ggtitle("Proportion of Death by Apache III System") +
    xlab("Apache III Diagnosis") + coord_flip()

p4 <- ggplot(train_pr, aes(x = apache_2_bodysystem, fill = hospital_death)) +
    geom_bar(position = "fill") +
    ggtitle("Proportion of Death by Apache II System") +
    xlab("Apache II Diagnosis") + coord_flip()

p3 / p4
```

```
p5 <- ggplot(train_pr, aes(x = age)) +
  geom_histogram(aes(fill = hospital_death), color = "blue") +
  ggtitle("Distribution of Age")
p6 <- ggplot(train_pr, aes(x = age)) + geom_boxplot() +
  ggtitle("Boxplot of Age")
p5 / p6
```

```
p7 <- ggplot(train_pr, aes(x = ethnicity)) +
  geom_bar(fill = "blue") + coord_flip()
p8 <- ggplot(train_pr, aes(x = ethnicity, fill = hospital_death)) +
  geom_bar(position = "fill") + coord_flip()

p7 + p8
```

```
p9 <- ggplot(train_pr, aes(x = height)) + geom_boxplot() +
  ggtitle("Boxplot of Height (cm)")
p10 <- ggplot(train_pr, aes(x = weight)) + geom_boxplot() +
  ggtitle("Boxplot of Weight (kg)")

p11 <- ggplot(train_pr, aes(x = height)) +
  geom_histogram(aes(fill = hospital_death), color = "blue") +
  ggtitle("Distribution of Height (cm)")
p12 <- ggplot(train_pr, aes(x = weight)) +
  geom_histogram(aes(fill = hospital_death), color = "black", position = "fill") +
  ggtitle("Distribution of Weight (kg)")
p13 <- ggplot(train_pr, aes(x = weight)) +
  geom_histogram(aes(fill = hospital_death), color = "black") +
  ggtitle("Distribution of Weight (kg)")

p9 + p10
```

Boxplot of Height (cm)
0.4
0.2
0.0
−0.2
−0.4
140
160
180
height
Boxplot of Weight (kg)
0.4
0.2
0.0
−0.2
−0.4
50
100
150
weight

```
p11 / (p12 + p13)
```

```
ggplot(train_pr, aes(x = height, y = weight, color = hospital_death)) +
  geom_point() +
  ggtitle("Scatterplot of Height and Weight by Hospital Death Factor")
```

# Feature Handling - Downsampling and Dummy Variables

```
set.seed(100)
#Due to our severely class imbalance, we've chosen to downsample the minority class
downsampledTrain <- downSample(x = subset(train_pr, select = -hospital_death),
                               y = train_pr$hospital_death,
                               yname = "hospital_death")
dim(train_pr)
```

```
## [1] 71195    44
```

```
dim(downsampledTrain)
```

```
## [1] 12332    44
```

```
table(downsampledTrain$hospital_death)
```

```
##
##    0    1
## 6166 6166
```

```
X_train <- subset(downsampledTrain, select = -hospital_death)
y_train <- downsampledTrain$hospital_death
X_test <- subset(test_pr, select = -hospital_death)
```

```r
y_test <- test_pr$hospital_death
#Dummy Variables
character_features_train <- X_train[, sapply(X_train, class) == "character"]
dummy_variables_train <- fastDummies::dummy_cols(character_features_train, remove_first_dummy = TRUE)
character_features_test <- X_test[, sapply(X_test, class) == "character"]
dummy_variables_test <- fastDummies::dummy_cols(character_features_test, remove_first_dummy = TRUE)
```

## Feature Extraction

```r
# Drop the original 7 features #
dummy_variables_train <- dummy_variables_train[, -c(1:6)]
dummy_variables_test <- dummy_variables_test[, -c(1:6)]

# Combine Dummy Variables with DataFrame #
## Drop y-variable and merge with dummy variables ##
train_withdummies <- data.frame(X_train, dummy_variables_train)
test_withdummies <- data.frame(X_test, dummy_variables_test)

## Drop Non-Dummy Encoded Variables ##
train_withdummies <- train_withdummies[, sapply(train_withdummies, class) != "character"]
test_withdummies <- test_withdummies[, sapply(test_withdummies, class) != "character"]
```

```r
linear = findLinearCombos(train_withdummies)
#linear # suggests removing 66 67 68 69 71 72
print("Columns to be removed:")
```

```
## [1] "Columns to be removed:"
```

```r
colnames(train_withdummies[, c(66, 67, 68, 69, 71, 72)])
```

```
## [1] "apache_2_bodysystem_Gastrointestinal"
## [2] "apache_2_bodysystem_Haematologic"
## [3] "apache_2_bodysystem_Metabolic"
## [4] "apache_2_bodysystem_Neurologic"
## [5] "apache_2_bodysystem_Respiratory"
## [6] "apache_2_bodysystem_Trauma"
```

```r
cat("  \n")
```

```r
print("Columns to be retained")
```

```
## [1] "Columns to be retained"
```

```r
colnames(train_withdummies[, c(56, 59, 60, 62, 63, 65)])
```

```
## [1] "apache_3j_bodysystem_Gastrointestinal"
## [2] "apache_3j_bodysystem_Hematological"
```

```
## [3] "apache_3j_bodysystem_Metabolic"
## [4] "apache_3j_bodysystem_Neurological"
## [5] "apache_3j_bodysystem_Respiratory"
## [6] "apache_3j_bodysystem_Trauma"
```

```
train_withdummies_r <- train_withdummies[,-linear$remove]
test_withdummies_r <- test_withdummies[,-linear$remove]
```

```
levels(y_train) <- c("no_death", "death")
levels(y_test) <- c("no_death", "death")

y_train <- factor(y_train, levels=rev(levels(y_train)))
y_test <- factor(y_test, levels=rev(levels(y_test)))

# Visualize Weight of Classes #
table(y_train)
```

```
## y_train
##    death no_death
##     6166     6166
```

```
table(y_test)
```

```
## y_test
##    death no_death
##     1536    16260
```

# Models

## Logistic Regression Model

```
# Train Control Settings #
ctrl <- trainControl(method = "cv", #Cross-validation
                     summaryFunction = twoClassSummary, #for binary classification
                     classProbs = TRUE, #note class probabilities
                     savePredictions = TRUE)

## Logistic Regression Model #
set.seed(100)
lrFit <- train(x = train_withdummies_r,
            y = y_train,
            method = "glm",
            preProc = c("center","scale"),
            metric = "ROC", #best metric for classification
            trControl = ctrl)
lrFit
```

```
## Generalized Linear Model
##
```

```
## 12332 samples
##    68 predictor
##     2 classes: 'death', 'no_death'
##
## Pre-processing: centered (68), scaled (68)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 11099, 11098, 11099, 11100, 11099, 11100, ...
## Resampling results:
##
##   ROC        Sens      Spec
##   0.8579017  0.768256  0.7829987
```

```
#lrFit$finalModel
## Save ROC curve for the hold-out set
lrRoc <- roc(response = lrFit$pred$obs,
            predictor = lrFit$pred$death,
            levels = rev(levels(lrFit$pred$obs)))
plot(lrRoc, type = "s", col = 'red',
     legacy.axes = TRUE, print.auc = TRUE, print.auc.y = .6)
```

```
testResults <- data.frame(obs = y_test,
                          LR = predict(lrFit, test_withdummies_r))

confusionMatrix(testResults$LR, testResults$obs, positive = "death")
```

```
## Confusion Matrix and Statistics
```

```
##
##           Reference
## Prediction death no_death
##   death     1169     3297
##   no_death   367    12963
##
##                Accuracy : 0.7941
##                  95% CI : (0.7881, 0.8)
##     No Information Rate : 0.9137
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.2996
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.76107
##             Specificity : 0.79723
##          Pos Pred Value : 0.26176
##          Neg Pred Value : 0.97247
##              Prevalence : 0.08631
##          Detection Rate : 0.06569
##    Detection Prevalence : 0.25096
##       Balanced Accuracy : 0.77915
##
##        'Positive' Class : death
##
```

### Penalized Logistic Regression Model

```r
glmnGrid <- expand.grid(alpha = c(0, .2, .4, .6, 1),
                        lambda = seq(.01, .2, length = 2))

set.seed(100)
glmnFit <- train(x = train_withdummies_r,
                 y = y_train,
                 method = "glmnet",   #method is glmnet for penalized logistic regression
                 tuneGrid = glmnGrid,
                 metric = "ROC",
                 trControl = ctrl)
glmnFit
```

```
## glmnet
##
## 12332 samples
##    68 predictor
##     2 classes: 'death', 'no_death'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 11099, 11098, 11099, 11100, 11099, 11100, ...
## Resampling results across tuning parameters:
##
```

```
##   alpha  lambda  ROC        Sens       Spec
##   0.0    0.01    0.8575896  0.7677703  0.7805682
##   0.0    0.20    0.8529863  0.7533354  0.7812159
##   0.2    0.01    0.8576383  0.7612844  0.7834860
##   0.2    0.20    0.8393270  0.7202485  0.7893286
##   0.4    0.01    0.8571936  0.7611229  0.7834865
##   0.4    0.20    0.7986893  0.6711099  0.7617601
##   0.6    0.01    0.8566126  0.7616094  0.7810544
##   0.6    0.20    0.7831474  0.6782436  0.7293223
##   1.0    0.01    0.8547255  0.7577159  0.7833232
##   1.0    0.20    0.5000000  0.8000000  0.2000000
##
## ROC was used to select the optimal model using the largest value.
## The final values used for the model were alpha = 0.2 and lambda = 0.01.
```

```
glmnFit$results   #View results for each hyperparamater combination
```

```
##    alpha lambda       ROC      Sens      Spec       ROCSD     SensSD
## 1    0.0   0.01 0.8575896 0.7677703 0.7805682 0.006129892 0.02004755
## 2    0.0   0.20 0.8529863 0.7533354 0.7812159 0.006735173 0.02083116
## 3    0.2   0.01 0.8576383 0.7612844 0.7834860 0.006144798 0.02211895
## 4    0.2   0.20 0.8393270 0.7202485 0.7893286 0.010314350 0.01657460
## 5    0.4   0.01 0.8571936 0.7611229 0.7834865 0.006194961 0.02245604
## 6    0.4   0.20 0.7986893 0.6711099 0.7617601 0.015437975 0.01660620
## 7    0.6   0.01 0.8566126 0.7616094 0.7810544 0.006337836 0.02143141
## 8    0.6   0.20 0.7831474 0.6782436 0.7293223 0.016649144 0.01348821
## 9    1.0   0.01 0.8547255 0.7577159 0.7833232 0.006529745 0.02154028
## 10   1.0   0.20 0.5000000 0.8000000 0.2000000 0.000000000 0.42163702
##         SpecSD
## 1  0.010092702
## 2  0.009740332
## 3  0.010159039
## 4  0.013340922
## 5  0.009882179
## 6  0.016104997
## 7  0.008492177
## 8  0.015066643
## 9  0.009395132
## 10 0.421637021
```

```
glmnetCM <- confusionMatrix(glmnFit, norm = "none")
glmnetCM
```

```
## Cross-Validated (10 fold) Confusion Matrix
##
## (entries are un-normalized aggregated counts)
##
##           Reference
## Prediction death no_death
##   death     4694     1335
##   no_death  1472     4831
##
##  Accuracy (average) : 0.7724
```

```r
## Plot the ROC curve for the hold-out set
glmRoc <- roc(response = glmnFit$pred$obs,
              predictor = glmnFit$pred$death,
              levels = rev(levels(glmnFit$pred$obs)))

plot(glmRoc, legacy.axes = TRUE)
```

```r
testResults <- data.frame(obs = y_test,
                          GLM = predict(glmnFit, test_withdummies_r))

confusionMatrix(testResults$GLM, testResults$obs, positive = "death")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction death no_death
##   death     1164     3282
##   no_death   372    12978
##
##                Accuracy : 0.7947
##                  95% CI : (0.7887, 0.8006)
##     No Information Rate : 0.9137
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.2993
```

```
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.75781
##             Specificity : 0.79815
##          Pos Pred Value : 0.26181
##          Neg Pred Value : 0.97213
##              Prevalence : 0.08631
##          Detection Rate : 0.06541
##    Detection Prevalence : 0.24983
##       Balanced Accuracy : 0.77798
##
##        'Positive' Class : death
##
```

## LDA Model

```r
set.seed(100)
ldaFit <- train(x = train_withdummies_r,
                y = y_train,
                method = "lda",
                trControl = ctrl,
                preProcess=c("center", "scale","BoxCox"),
                metric="ROC")
ldaFit
```

```
## Linear Discriminant Analysis
##
## 12332 samples
##    68 predictor
##     2 classes: 'death', 'no_death'
##
## Pre-processing: centered (68), scaled (68), Box-Cox transformation (25)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 11099, 11098, 11099, 11100, 11099, 11100, ...
## Resampling results:
##
##   ROC        Sens       Spec
##   0.8574246  0.7624198  0.7847834
```

```r
#Training performance: ROC =0.861, Sens = 0.793, Spec = 0.757

## Save ROC curve for the hold-out set
ldaRoc <- roc(response = ldaFit$pred$obs,
            predictor = ldaFit$pred$death,
            levels = rev(levels(ldaFit$pred$obs)))

plot(ldaRoc, type = "s", col = 'red', legacy.axes = TRUE, print.auc = TRUE, print.auc.y = .6)
```

```
testResults$LDA = predict(ldaFit, test_withdummies_r)
confusionMatrix(testResults$LDA, testResults$obs, positive = "death")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction death no_death
##   death     1157     3266
##   no_death   379    12994
##
##                Accuracy : 0.7952
##                  95% CI : (0.7892, 0.8011)
##     No Information Rate : 0.9137
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.2984
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.75326
##             Specificity : 0.79914
##          Pos Pred Value : 0.26159
##          Neg Pred Value : 0.97166
##              Prevalence : 0.08631
##          Detection Rate : 0.06501
```

```
##    Detection Prevalence : 0.24854
##       Balanced Accuracy : 0.77620
##
##        'Positive' Class : death
##
```

```
#Test performance: Sens = 0.766, Spec = 0.794 (3370 no deaths classified as deaths)
varImp(ldaFit)
```

```
## ROC curve variable importance
##
##   only 20 most important variables shown (out of 68)
##
##                      Importance
## d1_sysbp_min             100.00
## gcs_verbal_apache         97.35
## ventilated_apache         97.12
## gcs_motor_apache          90.87
## gcs_eyes_apache           90.18
## d1_diasbp_min             83.62
## h1_sysbp_min              71.16
## d1_spo2_min               69.47
## d1_resprate_max           64.41
## h1_diasbp_min             63.10
## heart_rate_apache         60.22
## h1_resprate_max           59.10
## intubated_apache          56.53
## d1_temp_min               56.50
## apache_3j_diagnosis       53.67
## apache_2_diagnosis        53.26
## age                       53.13
## h1_resprate_min           52.75
## temp_apache               51.09
## map_apache                45.30
```

## PLSDA Model

```
set.seed(100)
plsGrid = expand.grid(.ncomp = 1:10)

plsdaFit <- train(x = train_withdummies_r,
                y = y_train,
                method = "pls",
                tuneGrid = plsGrid,
                preProc = c("center","scale"),
                metric = "ROC",
                trControl = ctrl)
plsdaFit
```

```
## Partial Least Squares
##
```

```
## 12332 samples
##    68 predictor
##     2 classes: 'death', 'no_death'
##
## Pre-processing: centered (68), scaled (68)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 11099, 11098, 11099, 11100, 11099, 11100, ...
## Resampling results across tuning parameters:
##
##   ncomp  ROC        Sens       Spec
##    1     0.8382861  0.7265718  0.7834858
##    2     0.8504667  0.7363073  0.7948394
##    3     0.8544708  0.7580408  0.7799183
##    4     0.8557215  0.7562549  0.7829998
##    5     0.8558596  0.7601468  0.7849452
##    6     0.8561790  0.7578767  0.7834844
##    7     0.8564196  0.7585263  0.7849452
##    8     0.8564527  0.7590141  0.7846203
##    9     0.8564918  0.7606356  0.7842967
##   10     0.8564199  0.7588520  0.7833237
##
## ROC was used to select the optimal model using the largest value.
## The final value used for the model was ncomp = 9.
```

```r
#Training performance: ROC =0.861, Sens = 0.793, Spec = 0.758

## Save ROC curve for the hold-out set
plsdaRoc <- roc(response = plsdaFit$pred$obs,
            predictor = plsdaFit$pred$death,
            levels = rev(levels(plsdaFit$pred$obs)))

plot(plsdaRoc, type = "s", col = 'red', legacy.axes = TRUE, print.auc = TRUE, print.auc.y = .6)
```

```
testResults$plsLDA = predict(plsdaFit, test_withdummies_r)
confusionMatrix(testResults$plsLDA, testResults$obs, positive = "death")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction death no_death
##   death     1153     3276
##   no_death   383    12984
##
##                Accuracy : 0.7944
##                  95% CI : (0.7884, 0.8003)
##     No Information Rate : 0.9137
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.2964
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.75065
##             Specificity : 0.79852
##          Pos Pred Value : 0.26033
##          Neg Pred Value : 0.97135
##              Prevalence : 0.08631
##          Detection Rate : 0.06479
```

```
##    Detection Prevalence : 0.24888
##       Balanced Accuracy : 0.77459
##
##        'Positive' Class : death
##
```

```
#Test performance: Sens = 0.764, Spec = 0.794 (3385 no deaths classified as deaths)
varImp(plsdaFit)
```

```
## pls variable importance
##
##   only 20 most important variables shown (out of 68)
##
##                                            Overall
## ventilated_apache                           100.00
## gcs_verbal_apache                            91.24
## gcs_eyes_apache                              90.91
## gcs_motor_apache                             88.72
## d1_sysbp_min                                 83.99
## d1_diasbp_min                                72.74
## intubated_apache                             65.87
## d1_spo2_min                                  61.32
## d1_temp_min                                  61.22
## h1_sysbp_min                                 60.99
## h1_diasbp_min                                53.78
## age                                          53.53
## h1_resprate_max                              48.42
## temp_apache                                  48.39
## apache_3j_bodysystem_Metabolic               48.01
## h1_resprate_min                              44.02
## heart_rate_apache                            43.89
## d1_resprate_max                              43.84
## apache_3j_diagnosis                          43.83
## icu_admit_source_Operating.Room...Recovery   43.23
```

## MDA Model

```
set.seed(100)

mdaFit <- train(x = train_withdummies_r,
               y = y_train,
               method = "mda",
               tuneGrid = expand.grid(subclasses=1:3),
               preProc = c("center","scale"),
               metric = "ROC",
               trControl = ctrl)

mdaFit
```

```
## Mixture Discriminant Analysis
##
```

```
## 12332 samples
##    68 predictor
##     2 classes: 'death', 'no_death'
##
## Pre-processing: centered (68), scaled (68)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 11099, 11098, 11099, 11100, 11099, 11100, ...
## Resampling results across tuning parameters:
##
##   subclasses  ROC        Sens       Spec
##   1           0.8558741  0.7587521  0.7835600
##   2           0.8259244  0.7317263  0.7822951
##   3           0.8254666  0.7286608  0.7858990
##
## ROC was used to select the optimal model using the largest value.
## The final value used for the model was subclasses = 1.
```

```r
## Save ROC curve for the hold-out set
mdaRoc <- roc(response = mdaFit$pred$obs,
              predictor = mdaFit$pred$death,
              levels = rev(levels(mdaFit$pred$obs)))

plot(mdaRoc, type = "s", col = 'red', legacy.axes = TRUE, print.auc = TRUE, print.auc.y = .6)
```

```r
testResults$MDA = predict(mdaFit, test_withdummies_r)
confusionMatrix(testResults$MDA, testResults$obs, positive = "death")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction death no_death
##   death     1154     3270
##   no_death   382    12990
##
##                Accuracy : 0.7948
##                  95% CI : (0.7888, 0.8007)
##     No Information Rate : 0.9137
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.2972
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.75130
##             Specificity : 0.79889
##          Pos Pred Value : 0.26085
##          Neg Pred Value : 0.97143
##              Prevalence : 0.08631
##          Detection Rate : 0.06485
##    Detection Prevalence : 0.24860
##       Balanced Accuracy : 0.77510
##
##        'Positive' Class : death
##
```

```r
varImp(mdaFit)
```

```
## ROC curve variable importance
##
##   only 20 most important variables shown (out of 68)
##
##                      Importance
## d1_sysbp_min            100.00
## gcs_verbal_apache        97.35
## ventilated_apache        97.12
## gcs_motor_apache         90.87
## gcs_eyes_apache          90.18
## d1_diasbp_min            83.62
## h1_sysbp_min             71.16
## d1_spo2_min              69.47
## d1_resprate_max          64.41
## h1_diasbp_min            63.10
## heart_rate_apache        60.22
## h1_resprate_max          59.10
## intubated_apache         56.53
## d1_temp_min              56.50
```

```
## apache_3j_diagnosis      53.67
## apache_2_diagnosis       53.26
## age                      53.13
## h1_resprate_min          52.75
## temp_apache              51.09
## map_apache               45.30
```

## (CART) Decision Tree Model

```r
# Change Character Variables to Factors #
x_tree_train <- X_train
x_tree_test <- X_test
x_tree_train[sapply(x_tree_train, is.character)] <- lapply(
  x_tree_train[sapply(x_tree_train, is.character)], as.factor)
x_tree_test[sapply(x_tree_test, is.character)] <- lapply(
  x_tree_test[sapply(x_tree_test, is.character)], as.factor)


set.seed(100)
dtree <- train(x = x_tree_train, y = y_train,
               method = "rpart", metric = "ROC",
               trControl = ctrl, tuneLength = 30)

dtree #Optimal cp = 0.0008108985
```

```
## CART
##
## 12332 samples
##    43 predictor
##     2 classes: 'death', 'no_death'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 11099, 11098, 11099, 11100, 11099, 11100, ...
## Resampling results across tuning parameters:
##
##   cp            ROC        Sens       Spec
##   0.0004865391  0.8069706  0.7531731  0.7388895
##   0.0005405990  0.8076738  0.7554442  0.7374298
##   0.0005676289  0.8114755  0.7586894  0.7440801
##   0.0005946589  0.8104424  0.7614491  0.7429466
##   0.0006487188  0.8134832  0.7614494  0.7461915
##   0.0007298086  0.8160897  0.7616128  0.7504044
##   0.0008108985  0.8167127  0.7633954  0.7499187
##   0.0008919883  0.8154879  0.7630723  0.7478123
##   0.0009730782  0.8153405  0.7658202  0.7476539
##   0.0010271381  0.8149172  0.7633869  0.7508961
##   0.0010541680  0.8140427  0.7641973  0.7478146
##   0.0011352579  0.8136765  0.7638766  0.7461878
##   0.0012163477  0.8102481  0.7659864  0.7453740
##   0.0012974376  0.8110857  0.7637174  0.7487821
##   0.0013785274  0.8111028  0.7633930  0.7494306
```

```
##   0.0016217970  0.8094707  0.7671178  0.7442416
##   0.0016623419  0.8019248  0.7609545  0.7491039
##   0.0019461563  0.7851467  0.7651674  0.7372614
##   0.0020434642  0.7830323  0.7632228  0.7395312
##   0.0021083360  0.7792465  0.7672812  0.7346637
##   0.0022705157  0.7774414  0.7667931  0.7345011
##   0.0028381447  0.7689627  0.7643657  0.7330469
##   0.0030814142  0.7686326  0.7682592  0.7268852
##   0.0037301330  0.7680229  0.7593343  0.7314311
##   0.0068115472  0.7681263  0.7543110  0.7275374
##   0.0129743756  0.7688547  0.7419813  0.7366199
##   0.0201913720  0.7517048  0.7048362  0.7534796
##   0.0277327279  0.7295409  0.6154721  0.8122008
##   0.0317872202  0.7030948  0.6355780  0.7525253
##   0.3848524165  0.5905218  0.7357638  0.4452798
##
## ROC was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.0008108985.
```

```
#ROC = 0.8167127, Sens. = 0.7633954, Spec. = 0.7499187
#dtree$finalModel
dtreeRoc <- roc(response = dtree$pred$obs,
            predictor = dtree$pred$death,
            levels = rev(levels(dtree$pred$obs)))
plot(dtreeRoc, type = "s", col = 'blue',
     legacy.axes = TRUE, print.auc = TRUE, print.auc.y = .6)
```

```r
testResults$DecisionTree <- c(predict(dtree, x_tree_test))

confusionMatrix(testResults$DecisionTree, testResults$obs, positive = "death")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction death no_death
##   death     1157     3988
##   no_death   379    12272
##
##                Accuracy : 0.7546
##                  95% CI : (0.7482, 0.7609)
##     No Information Rate : 0.9137
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.2461
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.75326
##             Specificity : 0.75474
##          Pos Pred Value : 0.22488
##          Neg Pred Value : 0.97004
##              Prevalence : 0.08631
##          Detection Rate : 0.06501
##    Detection Prevalence : 0.28911
##       Balanced Accuracy : 0.75400
##
##        'Positive' Class : death
##
```

## Random Forest Model

```r
# Recommended values of mtry: sqrt(p) = sqrt(43) ~ 6 or 7
mtryvalues <- c(7)

set.seed(100)
randforest <- train(x = x_tree_train, y = y_train,
               method = "rf", metric = "ROC",
               trControl = ctrl,
               tuneGrid = data.frame(mtry = mtryvalues),
               ntree = 50)

randforest
```

```
## Random Forest
##
## 12332 samples
##    43 predictor
##     2 classes: 'death', 'no_death'
```

```
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 11099, 11098, 11099, 11100, 11099, 11100, ...
## Resampling results:
##
##   ROC        Sens       Spec
##   0.8633535  0.7955027  0.7698633
##
## Tuning parameter 'mtry' was held constant at a value of 7
```

```
#ROC = 0.8633535, Sens. = 0.7955027, Spec. = 0.7698633
## Compared to Baseline Decision Tree, ROC and Sens. increased, Spec. decreased slightly
randforest$finalModel #OOB estimate of error rate: 22.64%
```

```
##
## Call:
##  randomForest(x = x, y = y, ntree = 50, mtry = min(param$mtry,      ncol(x)))
##                Type of random forest: classification
##                      Number of trees: 50
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 22.64%
## Confusion matrix:
##          death no_death class.error
## death     4818     1348   0.2186182
## no_death  1444     4722   0.2341875
```

```
randforestROC <- roc(response = randforest$pred$obs,
            predictor = randforest$pred$death,
            levels = rev(levels(randforest$pred$obs)))
plot(randforestROC, type = "s", col = 'purple',
     legacy.axes = TRUE, print.auc = TRUE, print.auc.y = .6)
```

```
testResults$RandomForest <- c(predict(randforest, x_tree_test))
#testResults$RandomForest <- as.factor(ifelse(testResults$RandomForest == 2, "no_death", "death"))

confusionMatrix(testResults$RandomForest, testResults$obs, positive = "death")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction death no_death
##   death     1227     3614
##   no_death   309    12646
##
##                Accuracy : 0.7796
##                  95% CI : (0.7734, 0.7856)
##     No Information Rate : 0.9137
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.292
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.79883
##             Specificity : 0.77774
##          Pos Pred Value : 0.25346
##          Neg Pred Value : 0.97615
```

```
##              Prevalence : 0.08631
##          Detection Rate : 0.06895
##    Detection Prevalence : 0.27203
##       Balanced Accuracy : 0.78828
##
##        'Positive' Class : death
##
```

## Boosted Tree Model

```r
gbmGrid <- expand.grid(interaction.depth = c(5),
                       n.trees = (1:20)*5,
                       shrinkage = c(.01, .1),
                       n.minobsinnode = 5)

set.seed(100)
boost_tree <- train(x = x_tree_train, y = y_train,
                method = "gbm", tuneGrid = gbmGrid,
                verbose = FALSE, metric = "ROC",
                trControl = ctrl)
boost_tree
```

```
## Stochastic Gradient Boosting
##
## 12332 samples
##    43 predictor
##     2 classes: 'death', 'no_death'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 11099, 11098, 11099, 11100, 11099, 11100, ...
## Resampling results across tuning parameters:
##
##   shrinkage  n.trees  ROC        Sens       Spec
##   0.01         5      0.7969763  0.7528495  0.7246153
##   0.01        10      0.8061954  0.7479849  0.7359732
##   0.01        15      0.8115177  0.7377744  0.7484598
##   0.01        20      0.8151833  0.7280355  0.7620843
##   0.01        25      0.8169332  0.7333918  0.7633819
##   0.01        30      0.8196914  0.7286864  0.7667850
##   0.01        35      0.8217036  0.7322555  0.7697044
##   0.01        40      0.8236841  0.7322542  0.7706753
##   0.01        45      0.8250887  0.7350115  0.7718114
##   0.01        50      0.8264713  0.7356603  0.7713244
##   0.01        55      0.8276595  0.7377689  0.7719729
##   0.01        60      0.8292226  0.7384185  0.7716491
##   0.01        65      0.8307137  0.7395543  0.7716491
##   0.01        70      0.8320329  0.7411761  0.7706745
##   0.01        75      0.8335434  0.7445800  0.7682410
##   0.01        80      0.8346083  0.7484748  0.7666197
##   0.01        85      0.8357148  0.7491233  0.7666192
##   0.01        90      0.8367413  0.7528537  0.7643489
```

```
##   0.01        95      0.8378904  0.7556100  0.7625658
##   0.01       100      0.8389130  0.7575562  0.7614313
##   0.10         5      0.8181994  0.7465272  0.7552745
##   0.10        10      0.8351199  0.7539898  0.7585137
##   0.10        15      0.8457670  0.7672849  0.7632183
##   0.10        20      0.8525161  0.7745819  0.7661367
##   0.10        25      0.8572740  0.7791242  0.7714891
##   0.10        30      0.8608624  0.7825291  0.7732722
##   0.10        35      0.8637478  0.7847997  0.7765163
##   0.10        40      0.8657817  0.7878815  0.7757041
##   0.10        45      0.8672010  0.7873953  0.7773248
##   0.10        50      0.8689191  0.7886945  0.7795952
##   0.10        55      0.8702337  0.7896659  0.7797559
##   0.10        60      0.8710704  0.7916126  0.7826772
##   0.10        65      0.8718519  0.7914506  0.7825146
##   0.10        70      0.8726666  0.7933960  0.7831621
##   0.10        75      0.8733420  0.7946942  0.7820263
##   0.10        80      0.8734793  0.7942064  0.7815401
##   0.10        85      0.8738232  0.7948557  0.7820260
##   0.10        90      0.8741476  0.7951796  0.7828359
##   0.10        95      0.8745928  0.7955037  0.7851062
##   0.10       100      0.8745278  0.7933957  0.7852699
##
## Tuning parameter 'interaction.depth' was held constant at a value of 5
##
## Tuning parameter 'n.minobsinnode' was held constant at a value of 5
## ROC was used to select the optimal model using the largest value.
## The final values used for the model were n.trees = 95, interaction.depth =
##  5, shrinkage = 0.1 and n.minobsinnode = 5.
```

```r
# Shrinkage = 0.1, n.tree = 5
# ROC = 0.8745928, Sens. = 0.7955037, Spec. = 0.7851062

#boost_tree$finalModel

confusionMatrix(boost_tree, norm = "none")
```

```
## Cross-Validated (10 fold) Confusion Matrix
##
## (entries are un-normalized aggregated counts)
##
##           Reference
## Prediction death no_death
##   death     4905     1325
##   no_death  1261     4841
##
##  Accuracy (average) : 0.7903
```

```r
## Plot the ROC curve for the hold-out set
boost_treeRoc <- roc(response = boost_tree$pred$obs,
              predictor = boost_tree$pred$death,
              levels = rev(levels(boost_tree$pred$obs)))
```

```r
plot(boost_treeRoc, type = "s", col = 'green',
     legacy.axes = TRUE, print.auc = TRUE, print.auc.y = .6)
```

```r
testResults$BoostedTree <- c(predict(boost_tree, x_tree_test))
#testResults$BoostedTree <- as.factor(ifelse(testResults$BoostedTree == 2, "no_death", "death"))

confusionMatrix(testResults$BoostedTree, testResults$obs, positive = "death",
                mode = "everything")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction death no_death
##   death     1216     3252
##   no_death   320    13008
##
##                Accuracy : 0.7993
##                  95% CI : (0.7933, 0.8051)
##     No Information Rate : 0.9137
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.3174
##
##  Mcnemar's Test P-Value : <2e-16
##
```

```
##             Sensitivity : 0.79167
##             Specificity : 0.80000
##          Pos Pred Value : 0.27216
##          Neg Pred Value : 0.97599
##               Precision : 0.27216
##                  Recall : 0.79167
##                      F1 : 0.40506
##              Prevalence : 0.08631
##          Detection Rate : 0.06833
##    Detection Prevalence : 0.25107
##       Balanced Accuracy : 0.79583
##
##        'Positive' Class : death
##
```

## KNN Model

```
set.seed(100)
knnTune <- train(x = train_withdummies_r, y = y_train,
                 method = "knn", metric = "ROC",
                 preProc = c("center", "scale"),  #measuring distances, keep on the same scale and cent
                 tuneGrid = data.frame(k = seq(111, 151, by=2)),   #see which k-value performms the bes
                 trControl = ctrl)
knnTune
```

```
## k-Nearest Neighbors
##
## 12332 samples
##    68 predictor
##     2 classes: 'death', 'no_death'
##
## Pre-processing: centered (68), scaled (68)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 11099, 11098, 11099, 11100, 11099, 11100, ...
## Resampling results across tuning parameters:
##
##   k    ROC        Sens       Spec
##   111  0.8392338  0.5973145  0.8743057
##   113  0.8394531  0.5976394  0.8741436
##   115  0.8395760  0.5995851  0.8743057
##   117  0.8400274  0.5997487  0.8765784
##   119  0.8402472  0.5989378  0.8764150
##   121  0.8404548  0.5974789  0.8777129
##   123  0.8406540  0.5969935  0.8773880
##   125  0.8408012  0.5978030  0.8772262
##   127  0.8407288  0.5960197  0.8775503
##   129  0.8409849  0.5958582  0.8781989
##   131  0.8411691  0.5950457  0.8785230
##   133  0.8411024  0.5965049  0.8780371
##   135  0.8414073  0.5960179  0.8786856
##   137  0.8414942  0.5947192  0.8798217
##   139  0.8413985  0.5943953  0.8804703
```

```
##   141  0.8414334  0.5929345  0.8804692
##   143  0.8414963  0.5921228  0.8796591
##   145  0.8417672  0.5917984  0.8803077
##   147  0.8417383  0.5908249  0.8824162
##   149  0.8418843  0.5919602  0.8820923
##   151  0.8418216  0.5916361  0.8817692
##
## ROC was used to select the optimal model using the largest value.
## The final value used for the model was k = 149.
```

```
plot(knnTune)
```

```
## Plot the ROC curve for the hold-out set
knnRoc <- roc(response = knnTune$pred$obs,
              predictor = knnTune$pred$death,
              levels = rev(levels(knnTune$pred$obs)))

plot(knnRoc, type = "s", col = 'green',
     legacy.axes = TRUE, print.auc = TRUE, print.auc.y = .6)
```

```
testResults$Knn <- predict(knnTune, test_withdummies_r[, names(train_withdummies_r)])
confusionMatrix(testResults$Knn, testResults$obs, positive = "death",
                mode = "everything")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction death no_death
##   death      915     1856
##   no_death   621    14404
##
##                Accuracy : 0.8608
##                  95% CI : (0.8556, 0.8659)
##     No Information Rate : 0.9137
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.353
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.59570
##             Specificity : 0.88585
##          Pos Pred Value : 0.33021
##          Neg Pred Value : 0.95867
##               Precision : 0.33021
```

```
##                  Recall : 0.59570
##                      F1 : 0.42489
##              Prevalence : 0.08631
##          Detection Rate : 0.05142
##    Detection Prevalence : 0.15571
##       Balanced Accuracy : 0.74078
##
##        'Positive' Class : death
##
```

## Neural Net Model

```
set.seed(100)
nnetGrid <- expand.grid(size=1:2, decay=c(0,0.1,1))
nnetFit <- train(x = train_withdummies_r,
                y = y_train,
                method = "nnet",
                metric = "ROC",
                tuneGrid = nnetGrid,
                preProc = c("center","scale"),
                trControl = ctrl,
                maxit = 100,
                trace = FALSE)

nnetFit
```

```
## Neural Network
##
## 12332 samples
##    68 predictor
##     2 classes: 'death', 'no_death'
##
## Pre-processing: centered (68), scaled (68)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 11099, 11098, 11099, 11100, 11099, 11100, ...
## Resampling results across tuning parameters:
##
##   size  decay  ROC        Sens       Spec
##   1     0.0    0.8563365  0.7799354  0.7687286
##   1     0.1    0.8566597  0.7825265  0.7656465
##   1     1.0    0.8570569  0.7833416  0.7680774
##   2     0.0    0.8479542  0.7943619  0.7439117
##   2     0.1    0.8512042  0.7674656  0.7692043
##   2     1.0    0.8560697  0.7904755  0.7607824
##
## ROC was used to select the optimal model using the largest value.
## The final values used for the model were size = 1 and decay = 1.
```

```
nnetRoc <- roc(response = nnetFit$pred$obs,
            predictor = nnetFit$pred$death,
            levels = rev(levels(nnetFit$pred$obs)))
```

```r
plot(nnetRoc, legacy.axes = TRUE)
```

```r
nnetRoc$auc
```

```
## Area under the curve: 0.8522
```

```r
nnetImp <- varImp(nnetFit, scale = FALSE)
plot(nnetImp)
```

```
testResults$NN <- predict(nnetFit, test_withdummies_r)

confusionMatrix(testResults$NN, testResults$obs, positive = "death")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction death no_death
##   death     1215     3723
##   no_death   321    12537
##
##                Accuracy : 0.7728
##                  95% CI : (0.7665, 0.7789)
##     No Information Rate : 0.9137
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.2806
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.79102
##             Specificity : 0.77103
##          Pos Pred Value : 0.24605
##          Neg Pred Value : 0.97503
##              Prevalence : 0.08631
```

```
##          Detection Rate : 0.06827
##    Detection Prevalence : 0.27748
##       Balanced Accuracy : 0.78102
##
##        'Positive' Class : death
##
```

## Final Model Selection

```
par(oma=c(0,0,2,0))
## Plot ROC Curves ##
plot(lrRoc, type = "s", col = 'red', legacy.axes = TRUE)
plot(ldaRoc, type = "s", add = TRUE, col = 'green', legacy.axes = TRUE)
plot(plsdaRoc, type = "s", add = TRUE, col = "blue", legacy.axes = TRUE)
plot(mdaRoc, type = "s", add = TRUE, col = 'yellow', legacy.axes = TRUE)
plot(glmRoc, type = "s", add = TRUE, col = "black",legacy.axes = TRUE)
plot(dtreeRoc, type = "s", add = TRUE, col = "cyan",legacy.axes = TRUE)
plot(randforestROC, type = "s", add = TRUE, col = "coral",legacy.axes = TRUE)
plot(boost_treeRoc, type = "s", add = TRUE, col = "darkred",legacy.axes = TRUE)
plot(knnRoc, type = "s", add = TRUE, col = "purple",legacy.axes = TRUE)
plot(nnetRoc, type = "s", add = TRUE, col = "cyan",legacy.axes = TRUE)

legend("bottomright", legend = c("Logistic Regression", "LDA", "PLSDA", "MDA",
                                 "GLMNET", "Decision Tree", "Random Forest", "Boosted Tree", "KNN",
                                 "Neural Net"),
       col = c("red", "green", "blue", "yellow", "black", "cyan", "coral", "darkred", "purple", "cyan")
       lwd = 2, cex = 0.5)
title(main = "ROC Curves By Model", outer = TRUE)
```

```r
# Performance Metric Table #
final_metric_table <- data.frame(
  "Models" = c("Logistic Regression", "Penalized Logistic Regression", "LDA", "PLSDA",
               "MDA", "Decision Tree", "Random Forest", "Boosted Tree", "KNN", "Neural Net"),
  "ROC- Train" = c(0.8579, 0.8576, 0.8574, 0.8565, 0.8559,
                           0.8167, 0.8634, 0.8746, 0.8401, 0.8571),
  "Sens. - Train" = c(0.7683, 0.7613, 0.7624, 0.7606, 0.7588,
                                   0.7634, 0.7955, 0.7955, 0.5981, 0.7833),
  "Spec. - Train" = c(0.7830, 0.7835, 0.7848, 0.7843, 0.7836,
                                   0.7499, 0.7699, 0.7851, 0.8766, 0.7681),
  "Sens. - Test" = c(0.7611, 0.7578, 0.7533, 0.7507, 0.7513,
                                 0.7533, 0.7988, 0.7917, 0.6030, 0.7910),
  "Spec. - Test" = c(0.7972, 0.7982, 0.7991, 0.7985, 0.7989,
                                 0.7547, 0.7777, 0.8000, 0.8795, 0.7710))
kable(final_metric_table)
```

| Models | ROC..Train | Sens....Train | Spec....Train | Sens....Test | Spec....Test |
|---|---|---|---|---|---|
| Logistic Regression | 0.8579 | 0.7683 | 0.7830 | 0.7611 | 0.7972 |
| Penalized Logistic Regression | 0.8576 | 0.7613 | 0.7835 | 0.7578 | 0.7982 |
| LDA | 0.8574 | 0.7624 | 0.7848 | 0.7533 | 0.7991 |
| PLSDA | 0.8565 | 0.7606 | 0.7843 | 0.7507 | 0.7985 |
| MDA | 0.8559 | 0.7588 | 0.7836 | 0.7513 | 0.7989 |
| Decision Tree | 0.8167 | 0.7634 | 0.7499 | 0.7533 | 0.7547 |
| Random Forest | 0.8634 | 0.7955 | 0.7699 | 0.7988 | 0.7777 |

| Models | ROC..Train | Sens....Train | Spec....Train | Sens....Test | Spec....Test |
|---|---|---|---|---|---|
| Boosted Tree | 0.8746 | 0.7955 | 0.7851 | 0.7917 | 0.8000 |
| KNN | 0.8401 | 0.5981 | 0.8766 | 0.6030 | 0.8795 |
| Neural Net | 0.8571 | 0.7833 | 0.7681 | 0.7910 | 0.7710 |