Final project

Due April 30, 2021

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.1     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0

## -- Conflicts ------------------------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(data.table)
```

```
##
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':
##
##     between, first, last

## The following object is masked from 'package:purrr':
##
##     transpose
```

```
library(mlr)
```

```
## Loading required package: ParamHelpers

## 'mlr' is in maintenance mode since July 2019. Future development
## efforts will go into its successor 'mlr3' (<https://mlr3.mlr-org.com>).
```

```
library(dplyr)
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.0.5

## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':
##
##     combine

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(rpart)

data <- read.csv("NIS2012-200k.csv", header = TRUE, stringsAsFactors = TRUE)

data.dt <- data.frame(data)
```

The final project requires that you build a predictive model based on real data – your own or the provided National Impatient data– and a paper-style short report (2-3 of pages long) describing the problem, the approach(es) taken, and the results. Below is a *guideline* structure for the report. You should use the section breakdown into intro, methods, results, conclusions/discussion but don't have to necessarily include every element listed below within those sections. And you may want to include elements not listed below. Use your judgement.

Introduction

The National Impatient Sample (NIS) data, collected by the Healthcare Cost and Utilization Project (HCUP), is the largest publicly available dataset that contains information on inpatient healthcare in hospitals throughout the United States. The NIS is used by policymakers and health officials to make national estimates of healthcare utilization, and observe key features of inpatient care. The NIS was first started in 1998 by the Healthcare Cost and Utilization Project, and contains information such as patient demographics, classification of diseases, total hospital bill, length of stay, and many other features that characterize hospital care. The goal of this assignment will be to build a model to predict impatient mortality an determine what factors contribute a increased risk of death during hospitalization.

The data that will be used in this assignment consists of a random subset of 200,000 patients from the 2012 National Impatient Sample. The data was taken from the Healthcare Cost and Ultilization Project (HCUP), which is the largest collection of hospital care data in the United States. The data was taken from discharge records from all hospitals that are participating with the HCUP, and use state guidelines to help identify the hospitals that qualify for the data collection process. 47 states and the District of Columbia participate in the NIS, and data is available for hospitals in those states. The outcome of interest is the inpatient mortality, of whether the patient died during the period of hospitalization. Features such as patient demographic, severity of disease, risk of mortality, and comorbidities were incorpated to determine if a patient was likely to die during hospitalization. This can be used to identify features that increase the risk of patient mortality in hospitals and seek to prevent such deaths in the future.

1. Describe the problem explaining in particular why prediction is of primary interest (inference could also be of interest but there has to be a good reason for wanting to predict a particular outcome)
2. Describe the data (e.g. data source, data collection, outcome of interest, available features, sample size, missing data, etc.)

Methods

First the relevant features to the outcome of interest was sorted out from the 175 original features that were present.

Then then data was then reevaluated and factors were added when necessary.

1. Describe any data pre-processing steps (e.g. cleaning, recoding, variable transformation, dealing with missing data, selection of features to be included in your models, etc)

Out of the 175 possible features that were present in the original dataset, only 44 variables were selected to be included in analysis and model building. These 44 include data regarding patient demographics (age, race, gender), comorbidities (such as alcohol abuse and COPD), and the risks of patient mortality. Each variable was examined and was made into factor variables as was appropriate. A majority of the features were converted into dummy variables, however some remained as strings and integers. In examining the missing data, there was less than 1% of the total sample size that was missing from the target variable, whether the patient died. Because the sample was small compared to the dataset, the missing values of the target variable were removed before the analysis.

2. Briefly describe the Machine learning methods you will be using and why they are appropriate for your data (e.g. given the sample size and dimensionality of your training data, are you more concerned about bias or variance?) You should try and compare at least 3 distinct appropriate methods.
3. Describe how you are splitting the data into testing and training and any resampling strategy used for comparing methods, tuning parameters, and/or model/feature selection.

Logistic Regression

I will be comparing 3 different methods to build a predictive model for patient mortality. The first will be logistic regression model. The logistic regression model is one of the most commonly used and basic binary classifiers. Because the desired goal is to determine if a patient died during their hospitalization, the outcome is a binary outcome. Given the extremely large sample size of the data with around 200,000 observations, both the training and testing sets will be large enough to ensure an accurate prediction model.

Forward selection was used to determine the features that will be included in the logistic regression model. According to the forward selection process, only the **APRDRG_Risk_Mortality**, a factor variable that characterizes the risk of patient mortality, was determined to be significant in the data. However, the race variable was also included to determine the effect of patient demographics on mortality. There will only be a couple of features included in the actual logistic prediction model, therefore the model will be a simpler one indicating that the model will have a higher bias. However, the large sample size of the data and the use of cross validation will be used to determine the accuracy of the results.

```
library('pROC')
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
data_glm <- glm(DIED~APRDRG_Risk_Mortality + RACE + LOS, family = 'binomial',data = forward_log_data[log

pred_glm <- factor(predict(data_glm,newdata = forward_log_data[log_test, ],type = 'response') >0.5)


predict_prob_train <- predict(data_glm, newdata = forward_log_data[log_train, ])
predict_prob_test <- predict(data_glm,newdata = forward_log_data[log_test, ])

roc_glm_train <- roc(forward_log_data[log_train,]$DIED,predict_prob_train, ci = TRUE, of = 'auc')
```

```
## Setting levels: control = Alive, case = Died
```

```
## Setting direction: controls < cases
```

```
roc_glm_test <- roc(forward_log_data[log_test, ]$DIED,predict_prob_test, ci = TRUE, of = 'auc')
```

```
## Setting levels: control = Alive, case = Died
## Setting direction: controls < cases
```

Balanced Random Forests

The data itself is very unbalanced, with 184,598 patients that were successfully discharged compared to the 3,412 that died in the hospital. This could lead to an optimistically low misclassification error. Therefore, balanced random forests will be used to help rebalance the two binary outcomes.

```
library(randomForest)


data_rf <- randomForest(DIED~.,data = refine_data[log_train, ],
                        mtry = sqrt(44),
                        ntree = 500,
                        strata = refine_data$DIED,
                        sampsize = c(2274,2274))

data_rf
```

```
##
## Call:
##  randomForest(formula = DIED ~ ., data = refine_data[log_train,      ], mtry = sqrt(44), ntree = 500
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 1.69%
## Confusion matrix:
##        Alive Died  class.error
## Alive 123035   30 0.0002437736
## Died    2088  186 0.9182058047
```

```
rf_roc_train <- roc(refine_data[log_train, ]$DIED, data_rf$votes[,1])
```

```
## Setting levels: control = Alive, case = Died
```

```
## Setting direction: controls > cases
```

```
auc(rf_roc_train)
```

```
## Area under the curve: 0.9355
```

```
rf_predict_test <- predict(data_rf,
                           newdata = refine_data[log_test, ],
                           type = 'prob')

rf_roc_test <- roc(refine_data[log_test, ]$DIED,rf_predict_test[,1])
```

```
## Setting levels: control = Alive, case = Died
## Setting direction: controls > cases
```

```
auc(rf_roc_test)
```

```
## Area under the curve: 0.9304
```

```
ci(rf_roc_test)
```

```
## 95% CI: 0.9233-0.9374 (DeLong)
```

Boosting

```
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 4.0.4
```

```
## Loading required package: Matrix
```

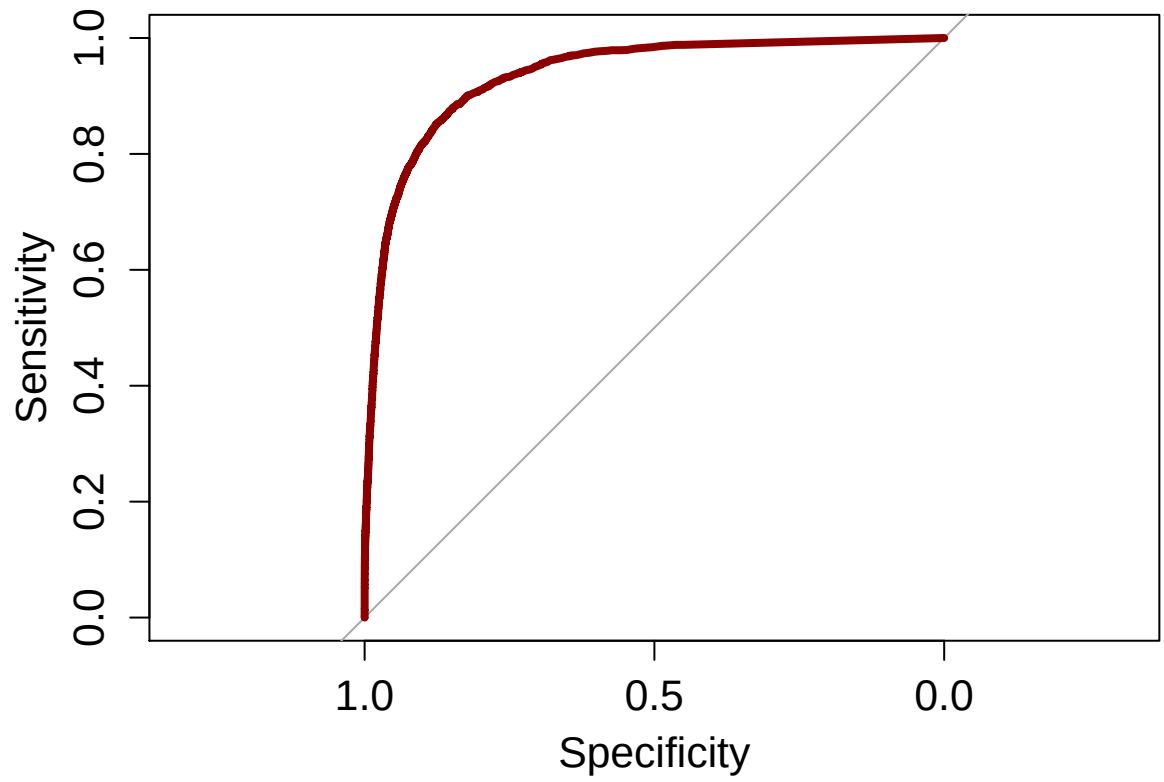
```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

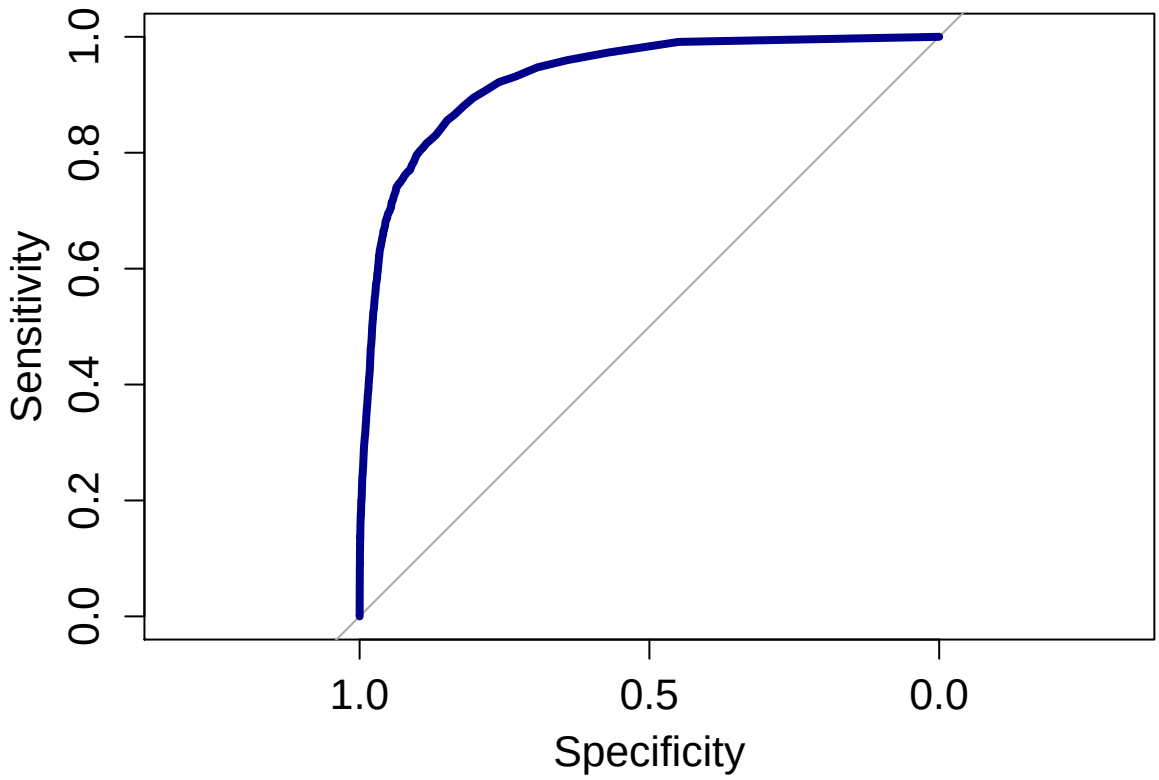
```
## Loaded glmnet 4.1-1
```

```
x <- refine_data$DIED
y <- model.matrix(refine_data$DIED~., data = refine_data[,-1])

learn_CV_lasso <- makeLearner("classif.cvglmnet",
                              fix.factors.prediction = TRUE,
                              predict.type ="prob",
                              alpha = 1,
                              type.measure = 'auc')

data_CV_lasso_train <- train(learn_CV_lasso, task = data_tsk,subset = log_train)
```

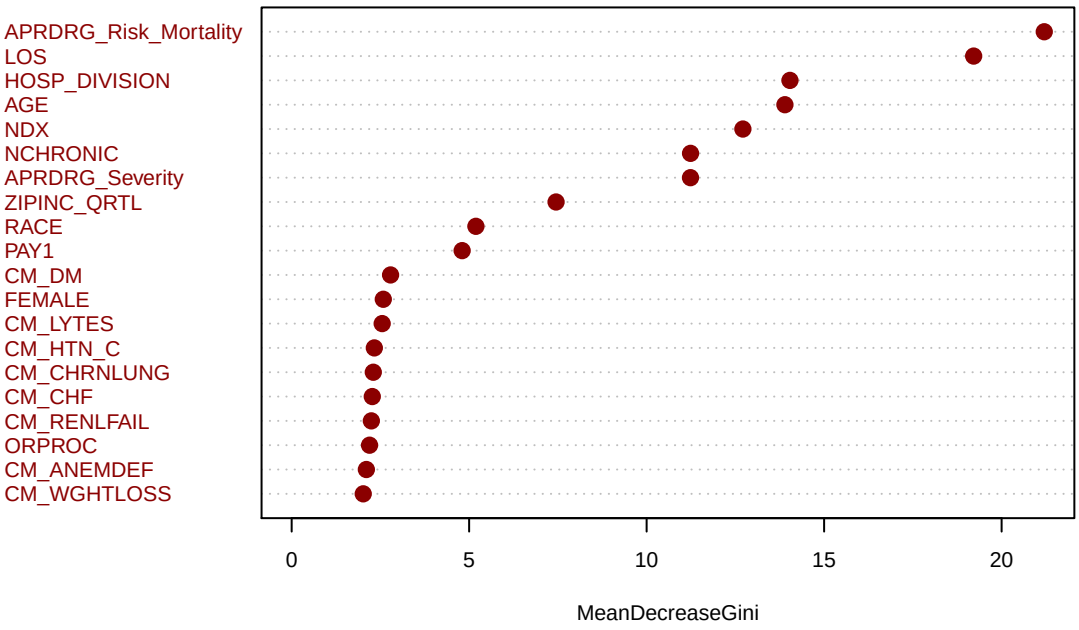
```
plot(rf_roc_train, lwd = 4, col = 'red4',cex.axis = 1.3, cex.lab = 1.3)
```




```
plot(rf_roc_test,lwd = 4, col = 'blue4', cex.axis = 1.3, cex.lab = 1.3)
```




```
varImpPlot(data_rf, cex = 0.7, pt.cex = 1.2, n.var = 20, main = "", pch = 16, col = 'red4')
```




```
library('pROC')
data_glm <- glm(DIED~APRDRG_Risk_Mortality + RACE, family = 'binomial',data = forward_log_data[log_train

pred_glm <- factor(predict(data_glm,newdata = forward_log_data[log_test, ],type = 'response') >0.5)


predict_prob_train <- predict(data_glm, newdata = forward_log_data[log_train, ])
predict_prob_test <- predict(data_glm,newdata = forward_log_data[log_test, ])

roc_glm_train <- roc(forward_log_data[log_train,]$DIED,predict_prob_train, ci = TRUE, of = 'auc')
```

```
## Setting levels: control = Alive, case = Died
```

```
## Setting direction: controls < cases
```

```
roc_glm_test <- roc(forward_log_data[log_test, ]$DIED,predict_prob_test, ci = TRUE, of = 'auc')
```

```
## Setting levels: control = Alive, case = Died
## Setting direction: controls < cases
```

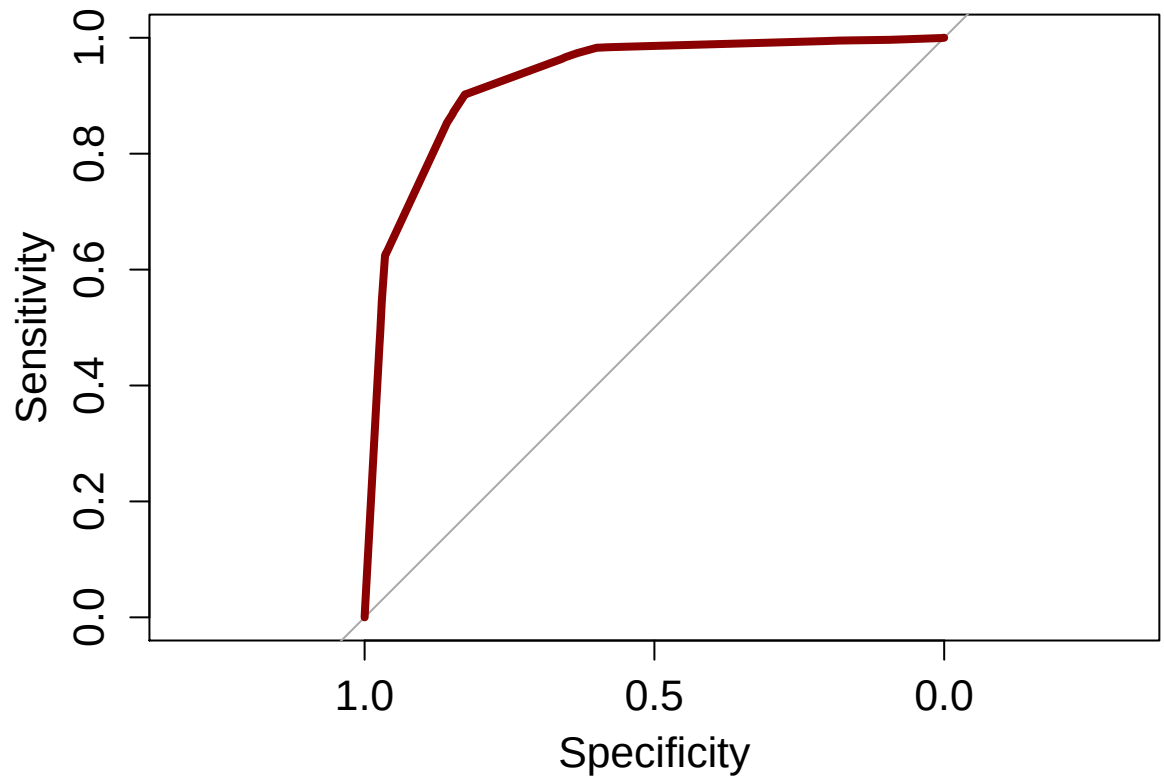
```
auc(roc_glm_train)
```

```
## Area under the curve: 0.9265
```

```
ci(roc_glm_train)
```

```
## 95% CI: 0.9216-0.9313 (DeLong)
```

```
plot(roc_glm_train, lwd = 4, col = 'red4',cex.axis = 1.3, cex.lab = 1.3)
```



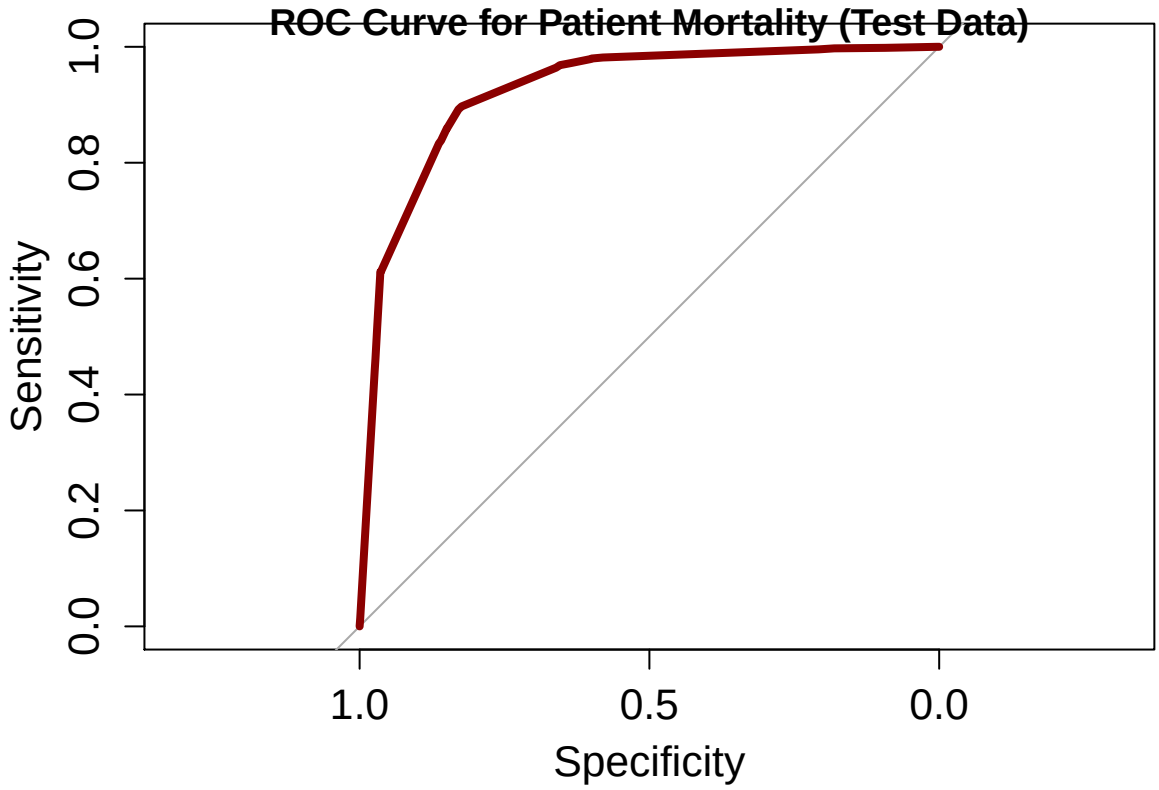

```
auc(roc_glm_test)
```

```
## Area under the curve: 0.9237
```

```
ci(roc_glm_test)
```

```
## 95% CI: 0.917-0.9305 (DeLong)
```

```
#have the auc of the test data
plot(roc_glm_test,lwd = 4, col = 'red4', cex.axis = 1.3, cex.lab = 1.3)
title(main = "ROC Curve for Patient Mortality (Test Data)")
```



4. If applicabble, describe any model/feature selection used.
5. If applicabble, describe any tuning parameters and how you will be tuning them.
6. Describe what performance metric(s) you will be using and why.

Results

1. Present key summaries (table and/or plots, but plots prefered when both available) of your data (e.g. class frequencies if a classification problem)
2. Report training, validation/cross-validation, and test errors. Present cross-validation plots for tuning parameters if available. Report variable importance (e.g. p-values, model coefficients, Random forest and boosting variable importance).

Conclusions/discussion

Discuss whether and why the prediction model(s) developed achieved sufficient high accuracy to be usefully deployed to predict new observations.

#Additional notes for those using the NIS data The data provided consists of a random subset of 200,000 patients from 2012 from the National Impatient Sample (NIS) data collected by the Healthcare Cost and Utilization Project (HCUP). You can find information on the HCUP database at https://www.hcup-us.ahrq.gov. You can choose to develop a model to predict death during hospitalization also known as impatient mortality (variable DIED in the dataset) or hospital length of stay (variable LOS in the dataset). For extra credit, you can also choose to predict both. The dataset has a relatively large number of variables. In the

provided data dictionary I preselected variables (highlighted) which are both availabe (not all variables in the dictionary are available for 2012) which might be relevant for predicting impatient mortality and/or hospital length of stay. Based on their description and additional info from the HCUP site you should choose which variables among the preselected ones you will consider as features/predictors. You don't have to use them all. There maybe variables that are redundant (capture pretty much the same info others already capture), variables that are too complex (e.g. categorical with way too many levels), or that based on your judgment are unlikely to be important. Be aware that the data is real and has not been pre-processed in any way and you will have to do some data cleaning. For example, you should carefully check the variables you consider as possible predictors for correctness of type (e.g. many numeric variables will be read in as factor variables when you use `read.csv`), outliers, missing observations, nonsensical values, etc.