**Quantitative, Qualitative, and PCR Analysis**

Nelson Tran

Master of Data Science, Merrimack College

DSE6111 Predictive Modeling

Dr. Fotios Kokkotos

December 15th, 2023

**Summary**

In this project, I embarked on a comprehensive exploration of data analysis on a data set about healthcare analytics. The project is aimed to provide insights into models that can help predict three distinct types of problems, quantitative response, qualitative response, and a principal components regression (PCR) problem. The qualitative response problem will predict the severity of an illness for an average patient in the hospital, and both the quantitative and PCR will predict the stay of an average patient. This data set was obtained from Kaggle and will be the data set that these models will be used on. The model or models chosen from the exploratory data analysis are based on the test mean squared error (MSE) obtained after training the model and testing the model on our test data. Qualified models that will be considered for our final predictions will be compared if they return an MSE for an output. Models that do not return an MSE or are not available will be omitted from the final models that we will choose for our prediction. Based on our quantitative exploratory data analysis (EDA), the models chosen are multiple linear regression and partial least squares because they have the lowest MSE out of all the models applied to our problem. The average stay for a patient is 17.06 days. The qualitative response uses quadratic discriminant analysis and linear discriminant analysis to predict how severe a patient's illness could be. Both models indicate that an average patient will have a moderate severity of illness. Lastly, the PCR problem was explored, and this model predicted a patient to stay an average of 16.68 days in a hospital.

**Data & Approach**

The analysis in this project is aimed at trying to tackle real-world problems. The data set used is healthcare analytics which includes how long a patient has stayed in the hospital and the severity of illness. Predicting the average stay for a patient will be used for quantitative and PCR analysis and predicting the severity of illness for a patient will be used for qualitative analysis. The variables in this data set are "case id," hospital code," "hospital type code," City Code Hospital," hospital region code," "Available extra rooms in hospital," "department," "ward type," "ward facility code," "bed grade," "patient id," "city code patient," "type of admission," "severity of illness," "visitors with patient," "age," "admission deposit," and "stay." The variable "stay" a range of variables and to make this variable easier to perform quantitative and PCR. analysis so I made a range of days correspond to a factor. 0 to 20 days of stay correspond to 0, 21 to 50 days of stay correspond to 1, and 51 to more than 100 days of stay correspond to 2. I did the same thing with the severity of illness in patients. Extreme illness corresponds to 1, minor illness corresponds to 2, and moderate illness responds to 3. This data set is split into a training set and a test set that we will train the models to and test with the test set. The split is 70 to 30 with the training set having 70% of the data and the test set having 30%.

**Quantitative Analysis**

After obtaining a summary of the data set in R, I used multiple linear regression on the numerical variables as predictors and "stay" as a response variable. Using the t-test and the p-values of the variables in the multiple linear regression will help determine which predictors are going to be included in our models. Best subset selection was used to see what the best model is with the number of predictors used in multiple linear regression. This is based on the lowest BIC value. Ridge regression, lasso regression, partial least squares, regression trees, bagging, random forest, and boosting models were also applied to the training set and then applied to the test set to determine the MSE for each model. The models that are valid for final selection will be models that return MSE and the model with the lowest MSE will be picked.

**Qualitative Analysis**

For qualitative analysis, we will use the same predictors for quantitative analysis but predict the severity of illness a patient may have. K-nearest-neighbors (KNN), logistic regression, linear discriminant analysis, quadratic discriminant analysis, classification trees, bagging approach, and random forest approach were used for qualitative analysis. The lowest MSE obtained from these models will be considered for the final model to predict the severity of illness for a patient. Again, models that do not return an MSE will not be included in this selection.

**Principal Components Regression**

In PCR, instead of using the numerical variables as predictors, the whole data set was considered in using PCR. Using the graph obtained from this regression, we can find out what the lowest cross-validation error is and use that for the number of components in our regression model. From this, we can use this and figure out how well the PCR model fits compared to the partial least squares model so we can see how it compares to other models used in quantitative analysis.

**Detailed Findings**

**Quantitative Analysis**

For all numerical variables in the multiple linear regression model, we applied a null hypothesis that would either eliminate or include each variable in any later models. For the t-test, we will reject the null hypothesis if the t-value does not equal 0. For p-values, if their values are greater than 0.05 then we will accept the null hypothesis and eliminate the variable from any further models used. All predictors used in this model rejected the null hypothesis and will be available for future use in models. The MSE for the multiple linear regression is 0.4420886. The best subset selection was used to determine how many predictors produced the best model. Based on the lowest BIC value, the best model uses all 7 predictors. When trying to find the MSE for this model, I kept getting a "NaN" value which disqualified this model for final selection in our prediction. These two models were used first to determine which variables would be used in future models to determine the best fit for our data set. The next two models are the ridge and lasso regression that produced .444018 and .4446472 MSE respectively. The following models produced the following MSEs, partial least squares: .4438446, regression tree: .559435, bagging approach: .5483819, random forest approach: .4902687, and boosting approach: .5514. The model with the lowest MSE is the multiple linear regression but also partial least squares model is not too far off so I will use both models to help with predicting the stay of a patient. I used the mean of the predicted values of both the multiple linear regression and partial least squares and multiplied it by 20 to get an average of about 16.688 days that a patient will stay in a hospital.

**Qualitative Analysis**

To help the data fit better for qualitative analysis I changed each variable into a factor so that they could fit better into the models. For the KNN model, I was able to run the model for the training set but was unable to return an MSE for the model. In the logistic regression, I chose to introduce a null hypothesis to try and remove any unnecessary predictors that we may not need. Predictors that have a p-value greater than 0.05 are not statistically significant and will be removed from any future use in qualitative analysis. Fortunately, we did not have to remove any variables as each p-value was below 0.05. Unfortunately, the logistic regression model returned an MSE of 0. Even though this model did return a mean squared error, we will not include it because if the MSE is 0 then it may mean that this model fits too well and should be excluded from the final decision. LDA and QDA produced .5435743 and .5415879 respectively. The last 3 models used in this analysis are the classification trees, bagging approach, and random forest. The classification trees yielded a 0.5617 MSE while I was not able to get a MSE for the bagging approach or random forest. I was able to run the models on the data sets but only got the output "N/A" for each MSE. I tried removing any "N/A" values from each prediction but still got the same output for each MSE. The only models that will be considered for prediction are the LDA, QDA, and classification trees. We will use both LDA and QDA for predictions as their MSEs are very close to each other. Each model indicates that on average a patient will have a moderate severity of illness.

**Principal Component Regression**

For PCR instead of using only the 7 predictors that we used for quantitative and qualitative analysis I used all variables to try and find the least number of components needed to get the best model. Here, we don't want to use all the components in the model, but we want to find the lowest mean squared error of prediction with the lowest number of components available to us in the model. First, we tried it on the whole data set, then the training data set, and finally the test data set to get an MSE of 0.4194666. The most components used in the model are 33. There must have been an error somewhere when I calculated this because there are not 33 predictors in the whole data set. I believe this has contributed to the way that the data set itself is set up using categorical variables. I tried turning each variable into a factor and/or a numerical value but it does not seem like it helps. I also compared this model to the least partial squares model which obtained a very close MSE at 0.4310673. Lastly, we used this model to determine the average stay of patients at a hospital. I took the mean of the predictions and used the same calculations that I did in the quantitative analysis to receive an average of 16.68 days.

**Validity & Reliability Assessment**

In this project, the careful selection of variables is driven by the aim of unraveling a compelling narrative through their interrelationships. Each variable chosen has been meticulously considered to contribute to a coherent story within our exploratory data analysis. The introduction of null hypotheses serves as a cornerstone in validating the significance of these relationships, adding a layer of statistical rigor to our methodology.

A pivotal step in bolstering the reliability of our analysis is the thoughtful categorization of predictors. Distinguishing between categorical and numerical predictors is paramount, particularly for models that demand numerical inputs. This categorization enhances the suitability of predictors for specific modeling techniques, contributing to the reliability of our chosen variables.

The robustness and credibility of our models are foundational to the project’s reproducibility and the meaningful interpretation of our findings. Beyond methodological necessities, the validation of our chosen models, predictors, and response variables is an indispensable facet of generalizing insights that transcend statistical outcomes.

**Appendix**

https://www.kaggle.com/datasets/nehaprabhavalkar/av-healthcare-analytics-ii/

Disclaimer: Many values that I used to make my calculations in code seemed to change after I knitted the code into a word document to use for the Appendix.

library(ISLR2)  
library(MASS)

library(class)  
library(e1071)

library(boot)

library(glmnet)

library(pls)

library(leaps)

library(tree)

library(randomForest)

library(BART)

library(dplyr)

library(gbm)

#Loading libraries and data needed  
#data used are already split into training and testing set  
#there is no need to split them  
hospital <- read.csv("./hospital\_data.csv")  
hospital\_dictionary <- read.csv("./hospital\_data\_dictionary.csv")  
#omitting any n/a values  
hospital <- na.omit(hospital)

hospital <- hospital %>%  
 mutate(Stay <- recode(Stay, '0-10' = 0, '11-20' = 0, '21-30' = 1,  
 '31-40' = 1, '41-50' = 1, '51-60' =2,'61-70' = 2,  
 '71-80' = 2, '81-90' = 2,'91-100' = 2,'>100' = 2))

colnames(hospital)[19] <- "stay"  
hospital <- hospital[,-18]  
#splitting data into a 70/30 split  
#70% train, 30% test  
train <- sample(1:dim(hospital)[1], size=0.7\*dim(hospital)[1])  
hos\_train <- hospital[train, ]  
hos\_test <- hospital[-train, ]  
  
table(hospital$stay)  
## 0 1 2   
## 100345 152169 54731

summary(hospital)

## case\_id Hospital\_code Hospital\_type\_code City\_Code\_Hospital  
## Min. : 1 Min. : 1.00 Length:313793 Min. : 1.000   
## 1st Qu.: 79271 1st Qu.:11.00 Class :character 1st Qu.: 2.000   
## Median :158950 Median :19.00 Mode :character Median : 5.000   
## Mean :158938 Mean :18.33 Mean : 4.778   
## 3rd Qu.:238399 3rd Qu.:26.00 3rd Qu.: 7.000   
## Max. :318438 Max. :32.00 Max. :13.000   
## Hospital\_region\_code Available.Extra.Rooms.in.Hospital Department   
## Length:313793 Min. : 0.000 Length:313793   
## Class :character 1st Qu.: 2.000 Class :character   
## Mode :character Median : 3.000 Mode :character   
## Mean : 3.196   
## 3rd Qu.: 4.000   
## Max. :24.000   
## Ward\_Type Ward\_Facility\_Code Bed.Grade patientid   
## Length:313793 Length:313793 Min. :1.000 Min. : 1   
## Class :character Class :character 1st Qu.:2.000 1st Qu.: 32833   
## Mode :character Mode :character Median :3.000 Median : 65735   
## Mean :2.623 Mean : 65743   
## 3rd Qu.:3.000 3rd Qu.: 98472   
## Max. :4.000 Max. :131624   
## City\_Code\_Patient Type.of.Admission Severity.of.Illness Visitors.with.Patient  
## Min. : 1.000 Length:313793 Length:313793 Min. : 0.000   
## 1st Qu.: 4.000 Class :character Class :character 1st Qu.: 2.000   
## Median : 8.000 Mode :character Mode :character Median : 3.000   
## Mean : 7.252 Mean : 3.281   
## 3rd Qu.: 8.000 3rd Qu.: 4.000   
## Max. :38.000 Max. :32.000  
## Age Admission\_Deposit stay   
## Length:313793 Min. : 1800 Min. :0.000   
## Class :character 1st Qu.: 4188 1st Qu.:0.000   
## Mode :character Median : 4742 Median :1.000   
## Mean : 4882 Mean :0.852   
## 3rd Qu.: 5410 3rd Qu.:1.000   
## Max. :11008 Max. :2.000   
## NA's :6548

Numerical columns: Hospital\_code, City\_Code\_Hospital, Available.Extra.Rooms.in.Hospital, Bed.Grade, City\_Code\_Patient, Visitors.with.Patient, and Admission\_Deposit, stay

Categorical Columns: Hospital\_type\_code, Hospital\_region\_code, Ward\_Type, Ward\_Facility\_Code, Type of Admission, Severity of Illness, and, Age

Quantitative Problem: - Predicting hospitalization duration for patients.

for multiple linear regression: we will assume for the t-test that H(o): B1 = 0 and H(A): B1 != 0 rejecting the null hypothesis will tell if there is a statistically significant difference between the means of two variables

we will assume for the p-value that H(o): p(value) > 0.05 and H(A): p-value < 0.05 this will determine if a variable is statistically significant

#Multiple Linear Regression  
hos\_lm <- lm(stay ~ Hospital\_code + City\_Code\_Hospital +  
 Available.Extra.Rooms.in.Hospital + Bed.Grade +  
 City\_Code\_Patient + Visitors.with.Patient +   
 Admission\_Deposit, data = hos\_train)  
summary(hos\_lm)  
## Call:  
## lm(formula = stay ~ Hospital\_code + City\_Code\_Hospital + Available.Extra.Rooms.in.Hospital +   
## Bed.Grade + City\_Code\_Patient + Visitors.with.Patient + Admission\_Deposit,   
## data = hos\_train)  
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.8710 -0.6156 0.1285 0.3933 2.4391

## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 5.764e-01 1.028e-02 56.092 < 2e-16 \*\*\*  
## Hospital\_code 3.750e-03 1.579e-04 23.742 < 2e-16 \*\*\*  
## City\_Code\_Hospital -4.897e-03 4.389e-04 -11.155 < 2e-16 \*\*\*  
## Available.Extra.Rooms.in.Hospital -9.166e-02 1.190e-03 -77.023 < 2e-16 \*\*\*  
## Bed.Grade -6.789e-02 1.569e-03 -43.281 < 2e-16 \*\*\*  
## City\_Code\_Patient -2.268e-03 2.829e-04 -8.017 1.09e-15 \*\*\*  
## Visitors.with.Patient 1.991e-01 9.189e-04 216.673 < 2e-16 \*\*\*  
## Admission\_Deposit 1.728e-05 1.286e-06 13.442 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
## Residual standard error: 0.6247 on 215083 degrees of freedom  
## (4564 observations deleted due to missingness)  
## Multiple R-squared: 0.1898, Adjusted R-squared: 0.1898   
## F-statistic: 7200 on 7 and 215083 DF, p-value: < 2.2e-16

lin\_pred <- predict(hos\_lm, hos\_test)  
lin\_MSE <- mean((lin\_pred - hos\_test$stay)^2)  
print(lin\_MSE)

## [1] NA

From the summary of the multiple linear regression, all variables used in this regression are statistically significant because their p-values are all below the null hypothesis which is 0.05. As for the t-test, each t-value does not equal 0 which means we can reject the null hypothesis for the t-value which means that there is a significant difference in size of the difference relative to the variation in your sample data. The test MSE for multiple linear regression is 0.4420886.

#Best Subset Selection  
best\_subset\_train <- regsubsets(stay ~ Hospital\_code + City\_Code\_Hospital +  
 Available.Extra.Rooms.in.Hospital + Bed.Grade +  
 City\_Code\_Patient + Visitors.with.Patient +   
 Admission\_Deposit,data = hos\_train, nvmax = 7)  
subset\_sum <- summary(best\_subset\_train)  
print(subset\_sum)

## Subset selection object  
## Call: regsubsets.formula(stay ~ Hospital\_code + City\_Code\_Hospital +   
## Available.Extra.Rooms.in.Hospital + Bed.Grade + City\_Code\_Patient +   
## Visitors.with.Patient + Admission\_Deposit, data = hos\_train,   
## nvmax = 7)  
## 7 Variables (and intercept)  
## Forced in Forced out  
## Hospital\_code FALSE FALSE  
## City\_Code\_Hospital FALSE FALSE  
## Available.Extra.Rooms.in.Hospital FALSE FALSE  
## Bed.Grade FALSE FALSE  
## City\_Code\_Patient FALSE FALSE  
## Visitors.with.Patient FALSE FALSE  
## Admission\_Deposit FALSE FALSE  
## 1 subsets of each size up to 7  
## Selection Algorithm: exhaustive  
## Hospital\_code City\_Code\_Hospital Available.Extra.Rooms.in.Hospital  
## 1 ( 1 ) " " " " " "   
## 2 ( 1 ) " " " " "\*"   
## 3 ( 1 ) " " " " "\*"   
## 4 ( 1 ) "\*" " " "\*"   
## 5 ( 1 ) "\*" " " "\*"   
## 6 ( 1 ) "\*" "\*" "\*"   
## 7 ( 1 ) "\*" "\*" "\*"   
## Bed.Grade City\_Code\_Patient Visitors.with.Patient Admission\_Deposit  
## 1 ( 1 ) " " " " "\*" " "   
## 2 ( 1 ) " " " " "\*" " "   
## 3 ( 1 ) "\*" " " "\*" " "   
## 4 ( 1 ) "\*" " " "\*" " "   
## 5 ( 1 ) "\*" " " "\*" "\*"   
## 6 ( 1 ) "\*" " " "\*" "\*"   
## 7 ( 1 ) "\*" "\*" "\*" "\*"

subset\_sum$rsq

## [1] 0.1583829 0.1797169 0.1864155 0.1884286 0.1891377 0.1895909 0.1898330

par(mfrow = c(2,2))  
plot(subset\_sum$adjr2, xlab = "Number of Variables",  
 ylab = "Adjr2", main = "Adjusted R-Squared")  
plot(subset\_sum$bic, xlab = "Number of Variables",  
 ylab = "Adjr2", main = "BIC")  
plot(subset\_sum$rss, xlab = "Number of Variables",  
 ylab = "Adjr2", main = "MSE")  
plot(subset\_sum$cp, xlab = "Number of Variables",  
 ylab = "Adjr2", main = "CP")

A group of graphs showing numbers and a number of variables

Description automatically generated

data.frame(  
 adj.r2 = which.max(subset\_sum$adjr2),  
 cp = which.min(subset\_sum$cp),  
 bic = which.min(subset\_sum$bic)  
)

## adj.r2 cp bic  
## 1 7 7 7

#based on the lowest BIC value, the best model uses 7 variables

#forward and backward stepwise selection  
regfit\_fwd <- regsubsets(stay ~ Hospital\_code + City\_Code\_Hospital +  
 Available.Extra.Rooms.in.Hospital + Bed.Grade +  
 City\_Code\_Patient + Visitors.with.Patient +   
 Admission\_Deposit, data = hos\_train, nvmax = 7,  
 method = "forward")  
summary(regfit\_fwd)

## Subset selection object  
## Call: regsubsets.formula(stay ~ Hospital\_code + City\_Code\_Hospital +   
## Available.Extra.Rooms.in.Hospital + Bed.Grade + City\_Code\_Patient +   
## Visitors.with.Patient + Admission\_Deposit, data = hos\_train,   
## nvmax = 7, method = "forward")  
## 7 Variables (and intercept)  
## Forced in Forced out  
## Hospital\_code FALSE FALSE  
## City\_Code\_Hospital FALSE FALSE  
## Available.Extra.Rooms.in.Hospital FALSE FALSE  
## Bed.Grade FALSE FALSE  
## City\_Code\_Patient FALSE FALSE  
## Visitors.with.Patient FALSE FALSE  
## Admission\_Deposit FALSE FALSE  
## 1 subsets of each size up to 7  
## Selection Algorithm: forward  
## Hospital\_code City\_Code\_Hospital Available.Extra.Rooms.in.Hospital  
## 1 ( 1 ) " " " " " "   
## 2 ( 1 ) " " " " "\*"   
## 3 ( 1 ) " " " " "\*"   
## 4 ( 1 ) "\*" " " "\*"   
## 5 ( 1 ) "\*" " " "\*"   
## 6 ( 1 ) "\*" "\*" "\*"   
## 7 ( 1 ) "\*" "\*" "\*"   
## Bed.Grade City\_Code\_Patient Visitors.with.Patient Admission\_Deposit  
## 1 ( 1 ) " " " " "\*" " "   
## 2 ( 1 ) " " " " "\*" " "   
## 3 ( 1 ) "\*" " " "\*" " "   
## 4 ( 1 ) "\*" " " "\*" " "   
## 5 ( 1 ) "\*" " " "\*" "\*"   
## 6 ( 1 ) "\*" " " "\*" "\*"   
## 7 ( 1 ) "\*" "\*" "\*" "\*"

regfit\_bwd <- regsubsets(stay ~ Hospital\_code + City\_Code\_Hospital +  
 Available.Extra.Rooms.in.Hospital + Bed.Grade +  
 City\_Code\_Patient + Visitors.with.Patient +   
 Admission\_Deposit, data = hos\_train, nvmax = 7,  
 method = "backward")  
summary(regfit\_bwd)

## Subset selection object  
## Call: regsubsets.formula(stay ~ Hospital\_code + City\_Code\_Hospital +   
## Available.Extra.Rooms.in.Hospital + Bed.Grade + City\_Code\_Patient +   
## Visitors.with.Patient + Admission\_Deposit, data = hos\_train,   
## nvmax = 7, method = "backward")  
## 7 Variables (and intercept)  
## Forced in Forced out  
## Hospital\_code FALSE FALSE  
## City\_Code\_Hospital FALSE FALSE  
## Available.Extra.Rooms.in.Hospital FALSE FALSE  
## Bed.Grade FALSE FALSE  
## City\_Code\_Patient FALSE FALSE  
## Visitors.with.Patient FALSE FALSE  
## Admission\_Deposit FALSE FALSE  
## 1 subsets of each size up to 7  
## Selection Algorithm: backward  
## Hospital\_code City\_Code\_Hospital Available.Extra.Rooms.in.Hospital  
## 1 ( 1 ) " " " " " "   
## 2 ( 1 ) " " " " "\*"   
## 3 ( 1 ) " " " " "\*"   
## 4 ( 1 ) "\*" " " "\*"   
## 5 ( 1 ) "\*" " " "\*"   
## 6 ( 1 ) "\*" "\*" "\*"   
## 7 ( 1 ) "\*" "\*" "\*"   
## Bed.Grade City\_Code\_Patient Visitors.with.Patient Admission\_Deposit  
## 1 ( 1 ) " " " " "\*" " "   
## 2 ( 1 ) " " " " "\*" " "   
## 3 ( 1 ) "\*" " " "\*" " "   
## 4 ( 1 ) "\*" " " "\*" " "   
## 5 ( 1 ) "\*" " " "\*" "\*"   
## 6 ( 1 ) "\*" " " "\*" "\*"   
## 7 ( 1 ) "\*" "\*" "\*" "\*"

coef(regfit\_fwd, 7)

## (Intercept) Hospital\_code   
## 0.5764228459 0.0037497810   
## City\_Code\_Hospital Available.Extra.Rooms.in.Hospital   
## -0.0048965026 -0.0916566635   
## Bed.Grade City\_Code\_Patient   
## -0.0678928049 -0.0022678283   
## Visitors.with.Patient Admission\_Deposit   
## 0.1991101692 0.0000172839

coef(regfit\_bwd, 7)

## (Intercept) Hospital\_code   
## 0.5764228459 0.0037497810   
## City\_Code\_Hospital Available.Extra.Rooms.in.Hospital   
## -0.0048965026 -0.0916566635   
## Bed.Grade City\_Code\_Patient   
## -0.0678928049 -0.0022678283   
## Visitors.with.Patient Admission\_Deposit   
## 0.1991101692 0.0000172839

set.seed(1)  
  
regfit.best <- regsubsets(stay ~ Hospital\_code + City\_Code\_Hospital +  
 Available.Extra.Rooms.in.Hospital + Bed.Grade +  
 City\_Code\_Patient + Visitors.with.Patient +   
 Admission\_Deposit, data = hos\_train, nvmax = 7)  
  
test.mat <- model.matrix(stay ~ Hospital\_code + City\_Code\_Hospital +  
 Available.Extra.Rooms.in.Hospital + Bed.Grade +  
 City\_Code\_Patient + Visitors.with.Patient +   
 Admission\_Deposit, data = hos\_train)  
  
val.errors <- rep(NA, 7)  
for (i in 1:7) {  
 coefi <- coef(regfit.best, id = i)  
 pred <- test.mat[, names(coef)] %\*% coefi  
 val.errors[i] <- mean((hos\_test$stay - pred)^2)  
}  
val.errors

## [1] NaN NaN NaN NaN NaN NaN NaN

coef(regfit.best, 7)

## (Intercept) Hospital\_code   
## 0.5764228459 0.0037497810   
## City\_Code\_Hospital Available.Extra.Rooms.in.Hospital   
## -0.0048965026 -0.0916566635   
## Bed.Grade City\_Code\_Patient   
## -0.0678928049 -0.0022678283   
## Visitors.with.Patient Admission\_Deposit   
## 0.1991101692 0.0000172839

#ridge regression  
#need to make a matrix first  
hos\_train[is.na(hos\_train)] <- 0  
hos\_test[is.na(hos\_test)] <- 0  
hospital[is.na(hospital)] <- 0  
set.seed(1)  
train\_matrix <- model.matrix(stay ~ Hospital\_code + City\_Code\_Hospital +  
 Available.Extra.Rooms.in.Hospital + Bed.Grade +  
 City\_Code\_Patient + Visitors.with.Patient +   
 Admission\_Deposit, data = hos\_train)  
test\_matrix <- model.matrix(stay ~ Hospital\_code + City\_Code\_Hospital +  
 Available.Extra.Rooms.in.Hospital + Bed.Grade +  
 City\_Code\_Patient + Visitors.with.Patient +   
 Admission\_Deposit, data = hos\_test)  
  
grid = 10^seq(10, -2, length = 100)  
  
ridge\_cv <- cv.glmnet(train\_matrix, hos\_train[,'stay'], alpha = 0,  
 lambda = grid)  
plot(ridge\_cv)

A graph of a function

Description automatically generated

r\_lambda\_min <- ridge\_cv$lambda.min  
print(r\_lambda\_min)

## [1] 0.01

coef(ridge\_cv)

## 9 x 1 sparse Matrix of class "dgCMatrix"  
## s1  
## (Intercept) 7.908928e-01  
## (Intercept) .   
## Hospital\_code 2.512776e-03  
## City\_Code\_Hospital -2.297725e-03  
## Available.Extra.Rooms.in.Hospital -5.329980e-02  
## Bed.Grade -5.114103e-02  
## City\_Code\_Patient -2.096454e-03  
## Visitors.with.Patient 9.433320e-02  
## Admission\_Deposit 3.738770e-06

ridge\_cv <- cv.glmnet(train\_matrix, hos\_train[,'stay'], alpha = 0,  
 lambda = grid)  
ridge\_pred <- predict(ridge\_cv, s = r\_lambda\_min, lambda = grid,  
 alpha = 0, newx = test\_matrix)  
ridge\_MSE <- mean((ridge\_pred - hos\_test[,"stay"])^2)  
print(ridge\_MSE)

## [1] 0.445358

Test MSE for Ridge Regression is .444018

#Lasso Regression  
set.seed(1)  
  
lasso\_cv <- cv.glmnet(train\_matrix, hos\_train[,"stay"],  
 alpha = 1, lambda = grid)  
plot(lasso\_cv)

A graph of a number and number

Description automatically generated with medium confidence

best\_lambda\_lasso <- lasso\_cv$lambda.min  
print(best\_lambda\_lasso)

## [1] 0.01

coef(lasso\_cv)

## 9 x 1 sparse Matrix of class "dgCMatrix"  
## s1  
## (Intercept) 0.7480349342  
## (Intercept) .   
## Hospital\_code 0.0009500399  
## City\_Code\_Hospital .   
## Available.Extra.Rooms.in.Hospital -0.0483227161  
## Bed.Grade -0.0396772631  
## City\_Code\_Patient .   
## Visitors.with.Patient 0.0996966267  
## Admission\_Deposit .

lasso\_pred <- predict(lasso\_cv, s = best\_lambda\_lasso,  
 lambda = grid, alpha = 1, newx = test\_matrix)  
lasso\_MSE <- mean((lasso\_pred - hos\_test[,"stay"])^2)  
print(lasso\_MSE)

## [1] 0.4463289

Lasso regression MSE is .4446472

#Partial Least Squares  
set.seed(1)  
pls.fit <- plsr(stay ~ Hospital\_code + City\_Code\_Hospital +  
 Available.Extra.Rooms.in.Hospital + Bed.Grade +  
 City\_Code\_Patient + Visitors.with.Patient +   
 Admission\_Deposit, data = hos\_train, scale = TRUE,  
 validation = "CV")  
summary(pls.fit)

## Data: X dimension: 219655 7   
## Y dimension: 219655 1  
## Fit method: kernelpls  
## Number of components considered: 7  
##   
## VALIDATION: RMSEP  
## Cross-validated using 10 random segments.  
## (Intercept) 1 comps 2 comps 3 comps 4 comps 5 comps 6 comps  
## CV 0.6974 0.667 0.6658 0.6658 0.6658 0.6658 0.6658  
## adjCV 0.6974 0.667 0.6658 0.6658 0.6658 0.6658 0.6658  
## 7 comps  
## CV 0.6658  
## adjCV 0.6658  
##   
## TRAINING: % variance explained  
## 1 comps 2 comps 3 comps 4 comps 5 comps 6 comps 7 comps  
## X 13.873 28.347 45.039 59.340 73.617 87.549 100.000  
## stay 8.529 8.863 8.868 8.868 8.868 8.868 8.868

validationplot(pls.fit, val.type = "MSEP")

A graph with numbers and a red line

Description automatically generated

#Lowest cross-validation error occurs when M = 1 when PLS is used.  
pls.pred <- predict(pls.fit, hos\_test, ncomp = 1)  
pls\_MSE <- mean((pls.pred - hos\_test[,"stay"])^2)  
print(pls\_MSE)

## [1] 0.4472809

#performing PLS using the full data set using M=1  
#we can use this to compare to PCR  
pls.fit2 <- plsr(stay ~ ., data = hospital, scale = TRUE,  
 ncomp = 1)  
summary(pls.fit2)

## Data: X dimension: 313793 44   
## Y dimension: 313793 1  
## Fit method: kernelpls  
## Number of components considered: 1  
## TRAINING: % variance explained  
## 1 comps  
## X 3.427  
## stay 12.024

Test MSE for PLS is 0.4438446

#regression tree  
reg\_tree <- tree(stay ~ Hospital\_code + City\_Code\_Hospital +  
 Available.Extra.Rooms.in.Hospital + Bed.Grade +  
 City\_Code\_Patient + Visitors.with.Patient +   
 Admission\_Deposit, data = hos\_train)  
summary(reg\_tree)

##   
## Regression tree:  
## tree(formula = stay ~ Hospital\_code + City\_Code\_Hospital + Available.Extra.Rooms.in.Hospital +   
## Bed.Grade + City\_Code\_Patient + Visitors.with.Patient + Admission\_Deposit,   
## data = hos\_train)  
## Variables actually used in tree construction:  
## [1] "Visitors.with.Patient" "Available.Extra.Rooms.in.Hospital"  
## Number of terminal nodes: 4   
## Residual mean deviance: 0.4146 = 91080 / 219700   
## Distribution of residuals:  
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -1.3030 -0.6626 0.3374 0.0000 0.3374 1.3370

plot(reg\_tree)  
text(reg\_tree, pretty = 0)

A diagram of patients with numbers

Description automatically generated

reg\_tree\_hat <- predict(reg\_tree, data = hos\_test)  
reg\_tree\_MSE <- mean((reg\_tree\_hat - hos\_test$stay)^2)

## Warning in reg\_tree\_hat - hos\_test$stay: longer object length is not a multiple  
## of shorter object length

print(reg\_tree\_MSE)

## [1] 0.5612125

The test MSE for the regression tree is 0.559435

#using cross validation to get the optimal level of tree complexity  
set.seed(1)  
cv\_tree <- cv.tree(reg\_tree)  
plot(cv\_tree$size, cv\_tree$dev, type = "b")

A graph with a line and a dotted line

Description automatically generated

#pruning tree to optimal level from graph  
#seems like optimal tree level is 4  
prune\_hos <- prune.tree(reg\_tree, best = 4)  
plot(prune\_hos)  
text(prune\_hos, pretty=0)

A diagram of patients with numbers

Description automatically generated

#The pruned tree is the same as the regression tree  
#MSE of the pruned tree  
prune\_tree\_hat <- predict(prune\_hos, data = hos\_test)  
prune\_tree\_MSE <- mean((prune\_tree\_hat - hos\_test$stay)^2)

## Warning in prune\_tree\_hat - hos\_test$stay: longer object length is not a  
## multiple of shorter object length

print(prune\_tree\_MSE)

## [1] 0.5612125

As expected the pruned tree model’s MSE is the same as the regression tree 0.559435.

#bagging approach  
set.seed(1)  
bag\_hos <- randomForest(stay ~ Visitors.with.Patient,  
 data = hos\_train, mtry = 1, ntree = 10,  
 importance = TRUE)

## Warning in randomForest.default(m, y, ...): The response has five or fewer  
## unique values. Are you sure you want to do regression?

#MSE of the bagging approach  
bag\_hos\_hat <- predict(bag\_hos, data = hos\_test)  
bag\_hos\_MSE <- mean((bag\_hos\_hat - hos\_test$stay)^2, na.rm = TRUE)

## Warning in bag\_hos\_hat - hos\_test$stay: longer object length is not a multiple  
## of shorter object length

print(bag\_hos\_MSE)

## [1] 0.551816

The bagging approach got a .5483819 as the test MSE

#Random Forest Approach  
set.seed(1)  
rf\_hos <- randomForest(stay ~ City\_Code\_Hospital,   
 data = hos\_train, mtry = 1,  
 importance = TRUE, ntree = 10)

## Warning in randomForest.default(m, y, ...): The response has five or fewer  
## unique values. Are you sure you want to do regression?

rf\_hos\_hat <- predict(rf\_hos, data = hos\_test)  
rf\_MSE <- mean((rf\_hos\_hat - hos\_test$stay)^2, na.rm = TRUE)

## Warning in rf\_hos\_hat - hos\_test$stay: longer object length is not a multiple  
## of shorter object length

print(rf\_MSE)

## [1] 0.4947005

The MSE for the random forest approach is 0.4902687

#Boosting Approach  
set.seed(1)  
pow <- seq(-2,0,0.1)  
lambdas = 10^pow  
train\_error <- rep(NA, length(lambdas))  
  
for (i in 1:length(lambdas)) {  
 boost\_hos <- gbm(stay ~ Visitors.with.Patient, data = hos\_train,  
 distribution = "gaussian", n.trees = 500,  
 shrinkage = lambdas[i])  
 #predicting the training error  
 boost\_pred <- predict(boost\_hos, hos\_train, n.trees = 500)  
 train\_error[i] <- mean((boost\_pred - hos\_train$stay)^2)  
}  
plot(lambdas, train\_error, type = 'b',  
 xlab = "Shrinkage Values",  
 ylab = "Training MSE")

A graph of a graph

Description automatically generated

test\_error <- rep(NA, length(lambdas))  
  
for (i in 1:length(lambdas)) {  
 boost\_hos <- gbm(stay ~ Visitors.with.Patient, data = hos\_test,  
 distribution = "gaussian", n.trees = 500,  
 shrinkage = lambdas[i])  
 #predicting the training error  
 boost\_pred <- predict(boost\_hos, hos\_test, n.trees = 500)  
 test\_error[i] <- mean((boost\_pred - hos\_test$stay)^2)  
}  
plot(lambdas, test\_error, type = 'b',  
 xlab = "Shrinkage Values",  
 ylab = "Training MSE")

A graph of a graph

Description automatically generated

min(test\_error)

## [1] 0.4261526

lambdas[which.min(test\_error)]

## [1] 0.1584893

boost\_hat <- predict(boost\_hos, data = hos\_test, n.trees=500)  
boost\_MSE <- mean((boost\_hat - hos\_test$stay)^2)  
print(boost\_MSE)

## [1] 0.42616

Using both methods to find the MSE, the test MSE for the Boosting method is about .5514

Qualitative Response: - Predicting the Severity of an illness

I will change each character variable to a factor to help with ease of applying the data set to models.

hospital2 <- hospital  
hos\_train2 <- hos\_train  
hos\_test2 <- hos\_test  
  
hospital2$Hospital\_type\_code <- as.factor(hospital2$Hospital\_type\_code)  
hospital2$Hospital\_region\_code <- as.factor(hospital2$Hospital\_region\_code)  
hospital2$Department <- as.factor(hospital2$Department)  
hospital2$Ward\_Type <- as.factor(hospital2$Ward\_Type)  
hospital2$Ward\_Facility\_Code <- as.factor(hospital2$Ward\_Facility\_Code)  
hospital2$Type.of.Admission <- as.factor(hospital2$Type.of.Admission)  
hospital2$Severity.of.Illness <- as.factor(hospital2$Severity.of.Illness)  
hospital2$Age <- as.factor(hospital2$Age)  
  
hos\_train2$Hospital\_type\_code <- as.factor(hos\_train2$Hospital\_type\_code)  
hos\_train2$Hospital\_region\_code <- as.factor(hos\_train2$Hospital\_region\_code)  
hos\_train2$Department <- as.factor(hos\_train2$Department)  
hos\_train2$Ward\_Type <- as.factor(hos\_train2$Ward\_Type)  
hos\_train2$Ward\_Facility\_Code <- as.factor(hos\_train2$Ward\_Facility\_Code)  
hos\_train2$Type.of.Admission <- as.factor(hos\_train2$Type.of.Admission)  
hos\_train2$Severity.of.Illness <- as.factor(hos\_train2$Severity.of.Illness)  
hos\_train2$Age <- as.factor(hos\_train2$Age)  
  
hos\_test2$Hospital\_type\_code <- as.factor(hos\_test2$Hospital\_type\_code)  
hos\_test2$Hospital\_region\_code <- as.factor(hos\_test2$Hospital\_region\_code)  
hos\_test2$Department <- as.factor(hos\_test2$Department)  
hos\_test2$Ward\_Type <- as.factor(hos\_test2$Ward\_Type)  
hos\_test2$Ward\_Facility\_Code <- as.factor(hos\_test2$Ward\_Facility\_Code)  
hos\_test2$Type.of.Admission <- as.factor(hos\_test2$Type.of.Admission)  
hos\_test2$Severity.of.Illness <- as.factor(hos\_test2$Severity.of.Illness)  
hos\_test2$Age <- as.factor(hos\_test2$Age)

#KNN  
set.seed(1)  
train.set <- data.frame(hospital2[train,])  
test.set <- data.frame(hospital2[-train,])  
train.direction <- hospital2[train,]$Severity.of.Illness  
  
#knn.pred <- knn(train.set, test.set, train.direction, k=1)  
#knn.MSE <- mean(knn.pred==hospital2[!train, ]$direction)  
#print(knn.MSE)

I don’t know why the K-nearest-neighbor did not work as I’m trying to get the Mean Squared error for this model. I had to change each variable from a character to factor then to a number. This might be why this model did not work but it was the only way for R to take my input.

null hypothesis for p-value: H(o) > 0.05, H(A): < 0.05

#Logistic Regression on training set  
hos\_glm <- glm(Severity.of.Illness ~ Hospital\_code + City\_Code\_Hospital +  
 Available.Extra.Rooms.in.Hospital + Bed.Grade +  
 City\_Code\_Patient + Visitors.with.Patient +   
 Admission\_Deposit, data = hos\_train2, family = binomial)  
summary(hos\_glm)

##   
## Call:  
## glm(formula = Severity.of.Illness ~ Hospital\_code + City\_Code\_Hospital +   
## Available.Extra.Rooms.in.Hospital + Bed.Grade + City\_Code\_Patient +   
## Visitors.with.Patient + Admission\_Deposit, family = binomial,   
## data = hos\_train2)  
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -5.534e-01 4.321e-02 -12.808 <2e-16 \*\*\*  
## Hospital\_code -6.302e-03 6.776e-04 -9.301 <2e-16 \*\*\*  
## City\_Code\_Hospital 2.930e-03 1.859e-03 1.576 0.1150   
## Available.Extra.Rooms.in.Hospital 1.056e-01 5.126e-03 20.592 <2e-16 \*\*\*  
## Bed.Grade 6.568e-01 7.075e-03 92.836 <2e-16 \*\*\*  
## City\_Code\_Patient 3.070e-03 1.225e-03 2.506 0.0122 \*   
## Visitors.with.Patient -8.724e-02 3.111e-03 -28.041 <2e-16 \*\*\*  
## Admission\_Deposit 1.016e-04 5.586e-06 18.192 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 206081 on 219654 degrees of freedom  
## Residual deviance: 195532 on 219647 degrees of freedom  
## AIC: 195548  
##   
## Number of Fisher Scoring iterations: 4

hos\_glm\_prob <- predict(hos\_glm, type = "response")  
hos\_glm\_pred <- rep("Wrong", length(hos\_glm\_prob))  
hos\_glm\_pred[hos\_glm\_prob>0.5] = "Correct"  
table(hos\_glm\_pred, hos\_train2$Severity.of.Illness)

##   
## hos\_glm\_pred Extreme Minor Moderate  
## Correct 39114 59099 121257  
## Wrong 93 6 86

hos\_glm\_test <- glm(Severity.of.Illness ~ Hospital\_code + City\_Code\_Hospital +  
 Available.Extra.Rooms.in.Hospital + Bed.Grade +  
 City\_Code\_Patient + Visitors.with.Patient +   
 Admission\_Deposit, data = hos\_test2, family = binomial)  
summary(hos\_glm)

##   
## Call:  
## glm(formula = Severity.of.Illness ~ Hospital\_code + City\_Code\_Hospital +   
## Available.Extra.Rooms.in.Hospital + Bed.Grade + City\_Code\_Patient +   
## Visitors.with.Patient + Admission\_Deposit, family = binomial,   
## data = hos\_train2)  
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -5.534e-01 4.321e-02 -12.808 <2e-16 \*\*\*  
## Hospital\_code -6.302e-03 6.776e-04 -9.301 <2e-16 \*\*\*  
## City\_Code\_Hospital 2.930e-03 1.859e-03 1.576 0.1150   
## Available.Extra.Rooms.in.Hospital 1.056e-01 5.126e-03 20.592 <2e-16 \*\*\*  
## Bed.Grade 6.568e-01 7.075e-03 92.836 <2e-16 \*\*\*  
## City\_Code\_Patient 3.070e-03 1.225e-03 2.506 0.0122 \*   
## Visitors.with.Patient -8.724e-02 3.111e-03 -28.041 <2e-16 \*\*\*  
## Admission\_Deposit 1.016e-04 5.586e-06 18.192 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 206081 on 219654 degrees of freedom  
## Residual deviance: 195532 on 219647 degrees of freedom  
## AIC: 195548  
##   
## Number of Fisher Scoring iterations: 4

hos\_glm\_prob\_test <- predict(hos\_glm\_test, type = "response")  
hos\_glm\_pred\_test <- rep("Wrong", length(hos\_glm\_prob\_test))  
hos\_glm\_pred\_test[hos\_glm\_prob\_test>0.5] = "Correct"  
table(hos\_glm\_pred\_test, hos\_test2$Severity.of.Illness)

##   
## hos\_glm\_pred\_test Extreme Minor Moderate  
## Correct 16793 25209 52056  
## Wrong 49 5 26

log\_MSE <- mean(hos\_glm\_pred\_test == hos\_test2$Severity.of.Illness)  
print(log\_MSE)

## [1] 0

#LDA  
hos\_lda <- lda(Severity.of.Illness ~ Hospital\_code + City\_Code\_Hospital +  
 Available.Extra.Rooms.in.Hospital + Bed.Grade +  
 City\_Code\_Patient + Visitors.with.Patient +   
 Admission\_Deposit, data = hos\_train2)  
hos\_lda

## Call:  
## lda(Severity.of.Illness ~ Hospital\_code + City\_Code\_Hospital +   
## Available.Extra.Rooms.in.Hospital + Bed.Grade + City\_Code\_Patient +   
## Visitors.with.Patient + Admission\_Deposit, data = hos\_train2)  
##   
## Prior probabilities of groups:  
## Extreme Minor Moderate   
## 0.1784935 0.2690811 0.5524254   
##   
## Group means:  
## Hospital\_code City\_Code\_Hospital Available.Extra.Rooms.in.Hospital  
## Extreme 18.71967 4.874614 3.169102  
## Minor 17.98743 4.645614 3.180848  
## Moderate 18.37694 4.817748 3.210708  
## Bed.Grade City\_Code\_Patient Visitors.with.Patient Admission\_Deposit  
## Extreme 2.249955 7.204581 3.438315 4747.521  
## Minor 2.988546 7.417460 3.173353 4982.150  
## Moderate 2.564565 7.192611 3.281087 4874.748  
##   
## Coefficients of linear discriminants:  
## LD1 LD2  
## Hospital\_code -0.010144751 -0.0092870977  
## City\_Code\_Hospital -0.003336410 0.0977381449  
## Available.Extra.Rooms.in.Hospital 0.138015269 0.5661360120  
## Bed.Grade 1.166656581 -0.1564011454  
## City\_Code\_Patient 0.012945336 -0.0853721322  
## Visitors.with.Patient -0.141601900 -0.2322672942  
## Admission\_Deposit 0.000141038 0.0004280099  
##   
## Proportion of trace:  
## LD1 LD2   
## 0.993 0.007

hos\_lda2 <- lda(Severity.of.Illness ~ Hospital\_code + City\_Code\_Hospital +  
 Available.Extra.Rooms.in.Hospital + Bed.Grade +  
 City\_Code\_Patient + Visitors.with.Patient +   
 Admission\_Deposit, data = hos\_test2)  
hos\_lda2

## Call:  
## lda(Severity.of.Illness ~ Hospital\_code + City\_Code\_Hospital +   
## Available.Extra.Rooms.in.Hospital + Bed.Grade + City\_Code\_Patient +   
## Visitors.with.Patient + Admission\_Deposit, data = hos\_test2)  
##   
## Prior probabilities of groups:  
## Extreme Minor Moderate   
## 0.1789076 0.2678408 0.5532516   
##   
## Group means:  
## Hospital\_code City\_Code\_Hospital Available.Extra.Rooms.in.Hospital  
## Extreme 18.84853 4.877152 3.163995  
## Minor 17.87118 4.659475 3.191203  
## Moderate 18.34893 4.789236 3.214373  
## Bed.Grade City\_Code\_Patient Visitors.with.Patient Admission\_Deposit  
## Extreme 2.261133 7.206983 3.438546 4750.757  
## Minor 2.988736 7.432617 3.180654 4990.874  
## Moderate 2.564802 7.168100 3.281306 4875.565  
##   
## Coefficients of linear discriminants:  
## LD1 LD2  
## Hospital\_code -0.0132398498 -0.0195635646  
## City\_Code\_Hospital -0.0015219399 0.0295951955  
## Available.Extra.Rooms.in.Hospital 0.1448044407 0.5400559099  
## Bed.Grade 1.1626880675 -0.1816053101  
## City\_Code\_Patient 0.0149987131 -0.1082751969  
## Visitors.with.Patient -0.1386745089 -0.2517206115  
## Admission\_Deposit 0.0001550798 0.0003547397  
##   
## Proportion of trace:  
## LD1 LD2   
## 0.9922 0.0078

lda\_pred <- predict(hos\_lda, hos\_test2)  
names(lda\_pred)

## [1] "class" "posterior" "x"

lda\_class <- lda\_pred$class  
table(lda\_class, hos\_test2$Severity.of.Illness)

##   
## lda\_class Extreme Minor Moderate  
## Extreme 59 7 39  
## Minor 1275 4200 4979  
## Moderate 15508 21007 47064

lda\_MSE <- mean(lda\_class == hos\_test2$Severity.of.Illness)  
print(lda\_MSE)

## [1] 0.545189

probability of extreme severity of illness: 0.1771867 probability of minor severity of illness: 0.2708046 probability of moderate severity of illness: 0.5520088 The mean for the LDA model is .5435743

#QDA  
qda\_fit <- qda(Severity.of.Illness ~ Hospital\_code + City\_Code\_Hospital +  
 Available.Extra.Rooms.in.Hospital + Bed.Grade +  
 City\_Code\_Patient + Visitors.with.Patient +   
 Admission\_Deposit, data = hos\_train2)  
qda\_fit

## Call:  
## qda(Severity.of.Illness ~ Hospital\_code + City\_Code\_Hospital +   
## Available.Extra.Rooms.in.Hospital + Bed.Grade + City\_Code\_Patient +   
## Visitors.with.Patient + Admission\_Deposit, data = hos\_train2)  
##   
## Prior probabilities of groups:  
## Extreme Minor Moderate   
## 0.1784935 0.2690811 0.5524254   
##   
## Group means:  
## Hospital\_code City\_Code\_Hospital Available.Extra.Rooms.in.Hospital  
## Extreme 18.71967 4.874614 3.169102  
## Minor 17.98743 4.645614 3.180848  
## Moderate 18.37694 4.817748 3.210708  
## Bed.Grade City\_Code\_Patient Visitors.with.Patient Admission\_Deposit  
## Extreme 2.249955 7.204581 3.438315 4747.521  
## Minor 2.988546 7.417460 3.173353 4982.150  
## Moderate 2.564565 7.192611 3.281087 4874.748

qda\_fit2 <- qda(Severity.of.Illness ~ Hospital\_code + City\_Code\_Hospital +  
 Available.Extra.Rooms.in.Hospital + Bed.Grade +  
 City\_Code\_Patient + Visitors.with.Patient +   
 Admission\_Deposit, data = hos\_test2)  
qda\_fit2

## Call:  
## qda(Severity.of.Illness ~ Hospital\_code + City\_Code\_Hospital +   
## Available.Extra.Rooms.in.Hospital + Bed.Grade + City\_Code\_Patient +   
## Visitors.with.Patient + Admission\_Deposit, data = hos\_test2)  
##   
## Prior probabilities of groups:  
## Extreme Minor Moderate   
## 0.1789076 0.2678408 0.5532516   
##   
## Group means:  
## Hospital\_code City\_Code\_Hospital Available.Extra.Rooms.in.Hospital  
## Extreme 18.84853 4.877152 3.163995  
## Minor 17.87118 4.659475 3.191203  
## Moderate 18.34893 4.789236 3.214373  
## Bed.Grade City\_Code\_Patient Visitors.with.Patient Admission\_Deposit  
## Extreme 2.261133 7.206983 3.438546 4750.757  
## Minor 2.988736 7.432617 3.180654 4990.874  
## Moderate 2.564802 7.168100 3.281306 4875.565

qda\_pred2 <- predict(qda\_fit2, hos\_test2)  
names(qda\_pred2)

## [1] "class" "posterior"

qda\_class2 <- qda\_pred2$class  
table(qda\_class2, hos\_test2$Severity.of.Illness)

##   
## qda\_class2 Extreme Minor Moderate  
## Extreme 700 218 819  
## Minor 1382 4795 5532  
## Moderate 14760 20201 45731

qda\_MSE <- mean(qda\_class2 == hos\_test2$Severity.of.Illness)  
print(qda\_MSE)

## [1] 0.5441586

probability of extreme severity of illness: 0.1771867 probability of minor severity of illness: 0.2708046 probability of moderate severity of illness: 0.5520088 The mean for the QDA model is 0.5415879

#Classification trees  
tree\_class <- tree(Severity.of.Illness ~ Hospital\_code + City\_Code\_Hospital +  
 Available.Extra.Rooms.in.Hospital + Bed.Grade +  
 City\_Code\_Patient + Visitors.with.Patient +   
 Admission\_Deposit, data = hos\_train2)  
summary(tree\_class)

##   
## Classification tree:  
## tree(formula = Severity.of.Illness ~ Hospital\_code + City\_Code\_Hospital +   
## Available.Extra.Rooms.in.Hospital + Bed.Grade + City\_Code\_Patient +   
## Visitors.with.Patient + Admission\_Deposit, data = hos\_train2)  
## Variables actually used in tree construction:  
## [1] "Bed.Grade"  
## Number of terminal nodes: 3   
## Residual mean deviance: 1.844 = 405000 / 219700   
## Misclassification error rate: 0.4362 = 95814 / 219655

plot(tree\_class)  
text(tree\_class, pretty = 0)

A diagram of a bed grade

Description automatically generated

tree\_class2 <- tree(Severity.of.Illness ~ Hospital\_code + City\_Code\_Hospital +  
 Available.Extra.Rooms.in.Hospital + Bed.Grade +  
 City\_Code\_Patient + Visitors.with.Patient +   
 Admission\_Deposit, data = hos\_test2)  
set.seed(2)  
tree\_class\_pred <- predict(tree\_class2, data = hos\_test2, type = 'class')  
table(tree\_class\_pred, hos\_test2$Severity.of.Illness)

##   
## tree\_class\_pred Extreme Minor Moderate  
## Extreme 4126 581 3117  
## Minor 0 0 0  
## Moderate 12716 24633 48965

class\_tree\_MSE <- .5617

(4105 + 0 + 48876)/(4105+594+3189+12575+24899+48776) = .5617 The mean for the classification tree is .5617.

#bagging approach qualitative  
  
set.seed(1)  
bag\_qual <- randomForest(Severity.of.Illness ~ Hospital\_code + City\_Code\_Hospital +  
 Available.Extra.Rooms.in.Hospital + Bed.Grade +  
 City\_Code\_Patient + Visitors.with.Patient +   
 Admission\_Deposit, ntree=10,  
 data = hos\_train2, mtry = 7, importance = TRUE)  
bag\_qual

##   
## Call:  
## randomForest(formula = Severity.of.Illness ~ Hospital\_code + City\_Code\_Hospital + Available.Extra.Rooms.in.Hospital + Bed.Grade + City\_Code\_Patient + Visitors.with.Patient + Admission\_Deposit, data = hos\_train2, ntree = 10, mtry = 7, importance = TRUE)   
## Type of random forest: classification  
## Number of trees: 10  
## No. of variables tried at each split: 7  
##   
## OOB estimate of error rate: 53.25%  
## Confusion matrix:  
## Extreme Minor Moderate class.error  
## Extreme 10614 7700 20502 0.7265561  
## Minor 7567 20842 30067 0.6435803  
## Moderate 20056 29880 70200 0.4156622

bag\_qual\_hat <- predict(bag\_qual, data = hos\_test2)  
bag\_qual\_hat <- na.omit(bag\_qual\_hat)  
bag\_qual\_MSE <- mean((bag\_qual\_hat - hos\_test2$Severity.of.Illness)^2)

## Warning in Ops.factor(bag\_qual\_hat, hos\_test2$Severity.of.Illness): '-' not  
## meaningful for factors

print(bag\_qual\_MSE)

## [1] NA

#random forest approach qualitative  
set.seed(1)  
rf\_qual <- randomForest(Severity.of.Illness ~ Bed.Grade + Visitors.with.Patient +   
 Admission\_Deposit, ntree=10,  
 data = hos\_train2, mtry = 3, importance = TRUE)  
rf\_qual

##   
## Call:  
## randomForest(formula = Severity.of.Illness ~ Bed.Grade + Visitors.with.Patient + Admission\_Deposit, data = hos\_train2, ntree = 10, mtry = 3, importance = TRUE)   
## Type of random forest: classification  
## Number of trees: 10  
## No. of variables tried at each split: 3  
##   
## OOB estimate of error rate: 51.33%  
## Confusion matrix:  
## Extreme Minor Moderate class.error  
## Extreme 8406 8185 22200 0.7833003  
## Minor 5315 21776 31383 0.6275952  
## Moderate 14200 30316 75613 0.3705683

rf\_qual\_hat <- predict(rf\_qual, data = hos\_test2)  
rf\_qual\_hat <- na.omit(rf\_qual\_hat)  
rf\_qual\_MSE <- mean((rf\_qual\_hat - hos\_test2$Severity.of.Illness)^2)

## Warning in Ops.factor(rf\_qual\_hat, hos\_test2$Severity.of.Illness): '-' not  
## meaningful for factors

print(rf\_qual\_MSE)

## [1] NA

any(is.na(hos\_test2$rf\_qual\_hat))

## [1] FALSE

any(is.na(hos\_test2$Severity.of.Illness))

## [1] FALSE

Principal Components Regression: - Predict the length of stay in a hospital

#ten-fold cross-validation error on the whole data set first  
set.seed(2)  
pcr\_fit\_whole <- pcr(stay ~ ., data = hospital2, scale = TRUE,  
 validation = "CV")  
summary(pcr\_fit\_whole)

## Data: X dimension: 313793 44   
## Y dimension: 313793 1  
## Fit method: svdpc  
## Number of components considered: 44  
##   
## VALIDATION: RMSEP  
## Cross-validated using 10 random segments.  
## (Intercept) 1 comps 2 comps 3 comps 4 comps 5 comps 6 comps  
## CV 0.6982 0.6981 0.6976 0.6973 0.6965 0.6965 0.692  
## adjCV 0.6982 0.6981 0.6976 0.6973 0.6965 0.6965 0.692  
## 7 comps 8 comps 9 comps 10 comps 11 comps 12 comps 13 comps  
## CV 0.6917 0.6868 0.6864 0.6841 0.6839 0.6833 0.6798  
## adjCV 0.6917 0.6868 0.6863 0.6841 0.6839 0.6833 0.6796  
## 14 comps 15 comps 16 comps 17 comps 18 comps 19 comps 20 comps  
## CV 0.6796 0.6788 0.6741 0.6708 0.6692 0.6672 0.667  
## adjCV 0.6796 0.6788 0.6741 0.6708 0.6692 0.6672 0.667  
## 21 comps 22 comps 23 comps 24 comps 25 comps 26 comps 27 comps  
## CV 0.6667 0.6667 0.6666 0.6658 0.6658 0.6658 0.6657  
## adjCV 0.6667 0.6667 0.6666 0.6658 0.6658 0.6658 0.6657  
## 28 comps 29 comps 30 comps 31 comps 32 comps 33 comps 34 comps  
## CV 0.6656 0.6654 0.6542 0.6537 0.6535 0.6482 0.648  
## adjCV 0.6656 0.6654 0.6542 0.6537 0.6535 0.6482 0.648  
## 35 comps 36 comps 37 comps 38 comps 39 comps 40 comps 41 comps  
## CV 0.648 0.648 0.6473 0.6471 0.6468 0.6468 0.6465  
## adjCV 0.648 0.648 0.6473 0.6470 0.6468 0.6468 0.6465  
## 42 comps 43 comps 44 comps  
## CV 0.6461 0.646 0.646  
## adjCV 0.6461 0.646 0.646  
##   
## TRAINING: % variance explained  
## 1 comps 2 comps 3 comps 4 comps 5 comps 6 comps 7 comps 8 comps  
## X 7.40385 13.7174 18.9026 23.5013 27.9111 32.156 35.945 39.494  
## stay 0.02794 0.1714 0.2479 0.4771 0.4793 1.773 1.856 3.247  
## 9 comps 10 comps 11 comps 12 comps 13 comps 14 comps 15 comps  
## X 42.768 45.682 48.554 51.352 53.980 56.603 59.185  
## stay 3.361 4.007 4.041 4.224 5.257 5.265 5.481  
## 16 comps 17 comps 18 comps 19 comps 20 comps 21 comps 22 comps  
## X 61.659 64.111 66.484 68.808 71.10 73.389 75.661  
## stay 6.788 7.681 8.139 8.685 8.74 8.814 8.817  
## 23 comps 24 comps 25 comps 26 comps 27 comps 28 comps 29 comps  
## X 77.93 80.159 82.370 84.513 86.578 88.575 90.478  
## stay 8.84 9.076 9.078 9.079 9.113 9.131 9.196  
## 30 comps 31 comps 32 comps 33 comps 34 comps 35 comps 36 comps  
## X 92.21 93.67 94.94 96.08 97.13 97.88 98.52  
## stay 12.20 12.34 12.41 13.81 13.88 13.89 13.89  
## 37 comps 38 comps 39 comps 40 comps 41 comps 42 comps 43 comps  
## X 99.11 99.59 99.76 99.88 99.93 99.98 100.00  
## stay 14.07 14.13 14.20 14.20 14.27 14.38 14.42  
## 44 comps  
## X 100.00  
## stay 14.42

validationplot(pcr\_fit\_whole, val.type = 'MSEP')

A graph with a line

Description automatically generated

#performing PCR on the training data set  
#based off of PCR on the whole data set, we will use M=33  
pcr\_fit\_train <- pcr(stay ~ ., data = hos\_train2, scale = TRUE,  
 validation = "CV")  
summary(pcr\_fit\_train)

## Data: X dimension: 219655 44   
## Y dimension: 219655 1  
## Fit method: svdpc  
## Number of components considered: 44  
##   
## VALIDATION: RMSEP  
## Cross-validated using 10 random segments.  
## (Intercept) 1 comps 2 comps 3 comps 4 comps 5 comps 6 comps  
## CV 0.6974 0.6973 0.6967 0.6965 0.6958 0.6958 0.6914  
## adjCV 0.6974 0.6973 0.6967 0.6965 0.6958 0.6958 0.6914  
## 7 comps 8 comps 9 comps 10 comps 11 comps 12 comps 13 comps  
## CV 0.691 0.6861 0.6857 0.6834 0.6833 0.6827 0.6801  
## adjCV 0.691 0.6861 0.6857 0.6834 0.6833 0.6827 0.6795  
## 14 comps 15 comps 16 comps 17 comps 18 comps 19 comps 20 comps  
## CV 0.6787 0.6784 0.6742 0.6702 0.6686 0.6668 0.6665  
## adjCV 0.6787 0.6784 0.6741 0.6701 0.6686 0.6667 0.6664  
## 21 comps 22 comps 23 comps 24 comps 25 comps 26 comps 27 comps  
## CV 0.6659 0.6659 0.6658 0.6649 0.6649 0.6648 0.6648  
## adjCV 0.6659 0.6659 0.6658 0.6649 0.6648 0.6648 0.6648  
## 28 comps 29 comps 30 comps 31 comps 32 comps 33 comps 34 comps  
## CV 0.6647 0.6645 0.6535 0.6531 0.6528 0.6477 0.6475  
## adjCV 0.6647 0.6645 0.6535 0.6531 0.6528 0.6477 0.6475  
## 35 comps 36 comps 37 comps 38 comps 39 comps 40 comps 41 comps  
## CV 0.6475 0.6475 0.6469 0.6466 0.6463 0.6463 0.6462  
## adjCV 0.6475 0.6475 0.6469 0.6466 0.6463 0.6463 0.6461  
## 42 comps 43 comps 44 comps  
## CV 0.6458 0.6456 0.6456  
## adjCV 0.6457 0.6456 0.6456  
##   
## TRAINING: % variance explained  
## 1 comps 2 comps 3 comps 4 comps 5 comps 6 comps 7 comps 8 comps  
## X 7.40379 13.7194 18.9095 23.5130 27.9236 32.170 35.955 39.510  
## stay 0.03956 0.1919 0.2671 0.4712 0.4722 1.735 1.835 3.226  
## 9 comps 10 comps 11 comps 12 comps 13 comps 14 comps 15 comps  
## X 42.785 45.700 48.570 51.372 54.000 56.62 59.201  
## stay 3.342 3.988 4.008 4.188 5.131 5.31 5.397  
## 16 comps 17 comps 18 comps 19 comps 20 comps 21 comps 22 comps  
## X 61.674 64.129 66.504 68.82 71.124 73.409 75.682  
## stay 6.585 7.686 8.109 8.63 8.706 8.864 8.865  
## 23 comps 24 comps 25 comps 26 comps 27 comps 28 comps 29 comps  
## X 77.942 80.175 82.385 84.528 86.589 88.584 90.487  
## stay 8.887 9.142 9.143 9.146 9.167 9.178 9.248  
## 30 comps 31 comps 32 comps 33 comps 34 comps 35 comps 36 comps  
## X 92.22 93.67 94.94 96.08 97.14 97.88 98.53  
## stay 12.22 12.33 12.41 13.78 13.84 13.84 13.85  
## 37 comps 38 comps 39 comps 40 comps 41 comps 42 comps 43 comps  
## X 99.11 99.59 99.76 99.88 99.93 99.98 100.00  
## stay 14.00 14.07 14.15 14.15 14.20 14.31 14.35  
## 44 comps  
## X 100.00  
## stay 14.35

validationplot(pcr\_fit\_train, val.type = 'MSEP')

A graph with a line

Description automatically generated from the graph, the lowest cross-validation error occurs at M = 33, without having to use all the components which have very little difference from using all components

pcr\_pred <- predict(pcr\_fit\_train, hos\_test2, ncomp = 33)  
pcr\_MSE <- mean((pcr\_pred - hos\_test2$stay)^2)  
print(pcr\_MSE)

## [1] 0.421935

The test MSE for Principal Component Regression is 0.4194666

I want to compare this regression to the partial least squares using the same method.

#Partial Least Squares  
pls\_comp\_fit <- plsr(stay ~ ., data = hospital2,  
 scale = TRUE, validation = "CV")  
summary(pls\_comp\_fit)

## Data: X dimension: 313793 44   
## Y dimension: 313793 1  
## Fit method: kernelpls  
## Number of components considered: 44  
##   
## VALIDATION: RMSEP  
## Cross-validated using 10 random segments.  
## (Intercept) 1 comps 2 comps 3 comps 4 comps 5 comps 6 comps  
## CV 0.6982 0.6549 0.6491 0.6476 0.6471 0.6469 0.6467  
## adjCV 0.6982 0.6549 0.6490 0.6476 0.6471 0.6469 0.6467  
## 7 comps 8 comps 9 comps 10 comps 11 comps 12 comps 13 comps  
## CV 0.6466 0.6465 0.6464 0.6463 0.6462 0.6461 0.6461  
## adjCV 0.6466 0.6465 0.6464 0.6463 0.6462 0.6461 0.6461  
## 14 comps 15 comps 16 comps 17 comps 18 comps 19 comps 20 comps  
## CV 0.646 0.646 0.646 0.646 0.646 0.646 0.646  
## adjCV 0.646 0.646 0.646 0.646 0.646 0.646 0.646  
## 21 comps 22 comps 23 comps 24 comps 25 comps 26 comps 27 comps  
## CV 0.646 0.646 0.646 0.646 0.646 0.646 0.646  
## adjCV 0.646 0.646 0.646 0.646 0.646 0.646 0.646  
## 28 comps 29 comps 30 comps 31 comps 32 comps 33 comps 34 comps  
## CV 0.646 0.646 0.646 0.646 0.646 0.646 0.646  
## adjCV 0.646 0.646 0.646 0.646 0.646 0.646 0.646  
## 35 comps 36 comps 37 comps 38 comps 39 comps 40 comps 41 comps  
## CV 0.646 0.646 0.646 0.646 0.646 0.646 0.646  
## adjCV 0.646 0.646 0.646 0.646 0.646 0.646 0.646  
## 42 comps 43 comps 44 comps  
## CV 0.646 0.646 3.203  
## adjCV 0.646 0.646 8.086  
##   
## TRAINING: % variance explained  
## 1 comps 2 comps 3 comps 4 comps 5 comps 6 comps 7 comps 8 comps  
## X 3.427 7.824 12.43 16.31 20.99 24.07 26.96 29.88  
## stay 12.024 13.597 13.99 14.12 14.17 14.21 14.25 14.28  
## 9 comps 10 comps 11 comps 12 comps 13 comps 14 comps 15 comps  
## X 32.44 34.81 37.19 39.97 41.71 43.52 45.76  
## stay 14.31 14.34 14.37 14.38 14.39 14.40 14.41  
## 16 comps 17 comps 18 comps 19 comps 20 comps 21 comps 22 comps  
## X 47.81 50.01 52.58 55.39 57.33 58.87 59.96  
## stay 14.41 14.41 14.42 14.42 14.42 14.42 14.42  
## 23 comps 24 comps 25 comps 26 comps 27 comps 28 comps 29 comps  
## X 61.64 63.18 65.14 66.34 67.56 69.03 71.31  
## stay 14.42 14.42 14.42 14.42 14.42 14.42 14.42  
## 30 comps 31 comps 32 comps 33 comps 34 comps 35 comps 36 comps  
## X 72.60 74.05 75.02 77.01 79.26 81.48 83.85  
## stay 14.42 14.42 14.42 14.42 14.42 14.42 14.42  
## 37 comps 38 comps 39 comps 40 comps 41 comps 42 comps 43 comps  
## X 86.10 88.49 90.95 93.19 95.45 97.73 100.00  
## stay 14.42 14.42 14.42 14.42 14.42 14.42 14.42  
## 44 comps  
## X 102.6  
## stay -13321.3

validationplot(pls\_comp\_fit, val.type = "MSEP")

A graph with a line

Description automatically generated

#finding the MSE of PLS  
pls\_comp\_pred <- predict(pls\_comp\_fit, hos\_test2, ncomp = 1)  
pls\_comp\_MSE <- mean((pls\_comp\_pred - hos\_test2$stay)^2)  
print(pls\_comp\_MSE)

## [1] 0.4310673

Comparing the PLS model and PCR model for this data set, we can see that the PCR model fits better with a lower test MSE than the PLS model.

#Comparing the Test MSE for Quantitative problem  
par(mfrow = c(1,2))  
barplot(c(lin\_MSE, ridge\_MSE, lasso\_MSE, pls\_MSE, reg\_tree\_MSE),  
 names.arg = c("OLS", "Ridge", "Lasso", "PLS", "Reg\_Tree"),  
 ylab = "Mean Squared Error", col = "blue")  
barplot(c(prune\_tree\_MSE, bag\_hos\_MSE, rf\_MSE, boost\_MSE),  
 names.arg = c("Pruned\_Reg\_Tree", "Bagging", "RandomForest", "Boosting"),  
 ylab = "Mean Squared Error", col = "blue")

A comparison of blue bars

Description automatically generated

Quant\_MSE <- data.frame(Model\_MSE = c(lin\_MSE, ridge\_MSE, lasso\_MSE, pls\_MSE, reg\_tree\_MSE,  
 prune\_tree\_MSE, bag\_hos\_MSE, rf\_MSE, boost\_MSE))  
min(Quant\_MSE)

## [1] NA

Comparing the MSE of models that qualified for this comparison showed that Multiple linear regression is the best model used for this data set. Partial least Squares is a close second in this list. I will used both of these models to predict the number of days a patient will stay in a hospital.

#Predicting the average amount of days a patient will stay in a hospital.  
mean(lin\_pred)

## [1] 0.8719085

mean(pls.pred)

## [1] 0.8339318

#Both models give you about 0.8344 so if we times that with 20(for the range of days for the factor 0), we should be able to predict the average amount of days a patient would stay in the hospital.  
avg\_stay <- 0.8344\*20  
print(avg\_stay)

## [1] 16.688

#A patient will stay on average 16.688 days in a hospital.

#Comparing the MSE for Qualitative Problem  
#The same concept will apply to the quantitative problem, I will only use models that I've gotten the MSE from.  
barplot(c(lda\_MSE,qda\_MSE,class\_tree\_MSE),  
 names.arg = c("lda\_MSE", "qda\_MSE","class\_tree\_MSE"),  
 ylab = "Mean Squared Error", col = "red")

A red rectangular bars with black text

Description automatically generated

Qual\_MSE <- data.frame(Model\_MSE = c(lda\_MSE,qda\_MSE,class\_tree\_MSE))  
min(Qual\_MSE)

## [1] 0.5441586

The lowest MSE in the qualified models used for our qualitative problem is the qda model, but the lda model is also really close so we will use both models to help without prediction.

#Predicting Qualitative problem  
lda\_pred\_num <- as.numeric(lda\_pred$class)  
qda\_pred2\_num <- as.numeric(qda\_pred2$class)  
mean(qda\_pred2\_num)

## [1] 2.838716

mean(lda\_pred\_num)

## [1] 2.886719

#From the data set extreme = 1, minor = 2, and moderate = 3.  
#A average patient at a hospital will have a moderate severity in illness.

#Predicting PCR problem  
mean(pcr\_pred)

## [1] 0.834302

#if we do the same math as we did for the quantitative problem would be able to get the average days a patient will stay in the hospital.  
pcr\_avg\_stay <- 0.834\*20  
print(pcr\_avg\_stay)

## [1] 16.68

#A patient will stay on average 16.68 days in a hospital.