Data Science 2 Final

Group 6

Contents

Set up **2**

Load libraries and data . . . 2

Subset 2 df and keep unique observations . . . 2

Exploratory analysis and data visualization **2**

Data Partition . . . 2

Understanding the outcome variable `recovery_time` . . . 3

Summary of the dataset . . . 4

Understand categorical variables . . . 6

Understand continuous variables . . . 7

Understand the correlation between continuous predictors **** . . . 8

Understand the relationship with continuous predictors and the outcome . . . 9

Considering variables based on the EDA . . . 10

Primary analysis: continuous time to recovery **11**

Mode 1: linear model . . . 11

Model 2: Ridge . . . 12

Model 3: Lasso . . . 13

Model 4: Elastic net . . . 15

Model 5: Partial least square . . . 17

Model 6: GAM . . . 18

Model 7: MARS . . . 20

Model 8: Regression tree . . . 22

Model 9: Random forest . . . 24

Model 10: Boosting . . . 25

Model comparison . . . 26

Secondary analysis: binary time to recovery **28**

Set up

Load libraries and data

```
library(caret)
library(mgcv)
library(earth)
library(tidyverse)
library(summarytools)
library(corrplot)
library(ggpubr)
library(rpart)
library(rpart.plot)
library(randomForest)
library(ranger)
library(gbm)

setwd("D:/CUMC/Y2S2/DS2/Final/ds2_final")

load("./recovery.RData")
```

Subset 2 df and keep unique observations

```
set.seed(2543)

dat1 <- dat[sample(1:10000, 2000),]

set.seed(4017)

dat2 <- dat[sample(1:10000, 2000),]

dat_bind <- unique(rbind(dat1, dat2))
```

Exploratory analysis and data visualization

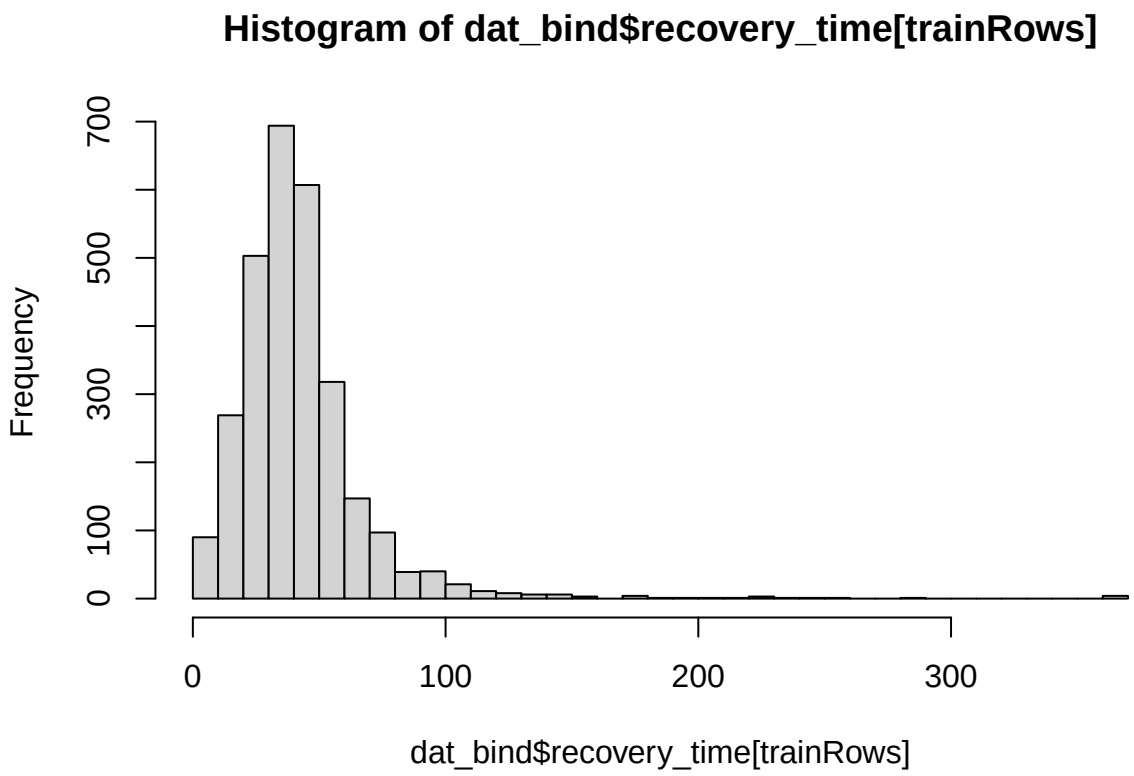
Data Partition

Here, we mainly want to investigate the EDA of the training dataset. Therefore, we will start with the data partition.

```
set.seed(2460)

trainRows <- createDataPartition(y = dat_bind$recovery_time,
                                 p = 0.8, list = FALSE)
```

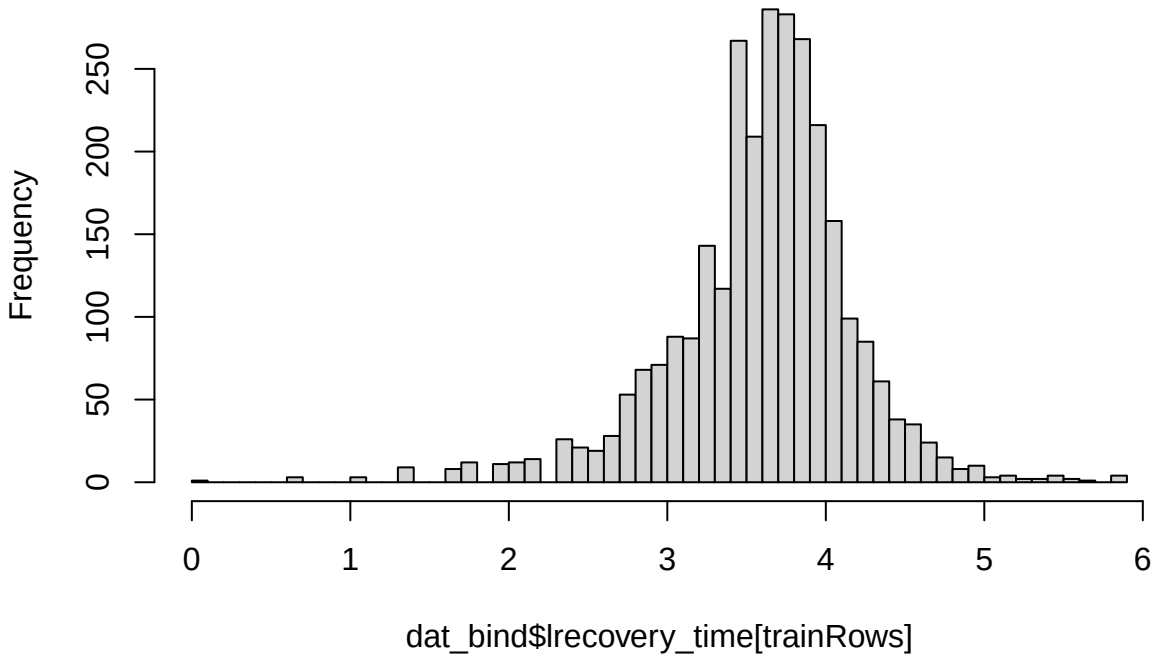
Understanding the outcome variable `recovery_time`

```
# check the outcome variable
hist(dat_bind$recovery_time[trainRows], breaks = 50)
```



The distribution of the outcome variable `recovery_time` is heavily right-skewed. To account for this, I will take the log-transformation of the outcome and use that variable for following analyses.

```
dat_bind = dat_bind %>%
  na.omit(dat_bind) %>%
  mutate(lrecovery_time = log(recovery_time)) %>%
  select(-recovery_time, -id)

# log-transformation helped with making it more normal
hist(dat_bind$lrecovery_time[trainRows], breaks = 50)
```



Summary of the dataset

```
st_options(plain.ascii = F,
           style = "rmarkdown",
           dfSummary.silent = T,
           footnote = NA,
           subtitle.emphasis = F)

dfSummary(dat_bind[trainRows, -1])
```

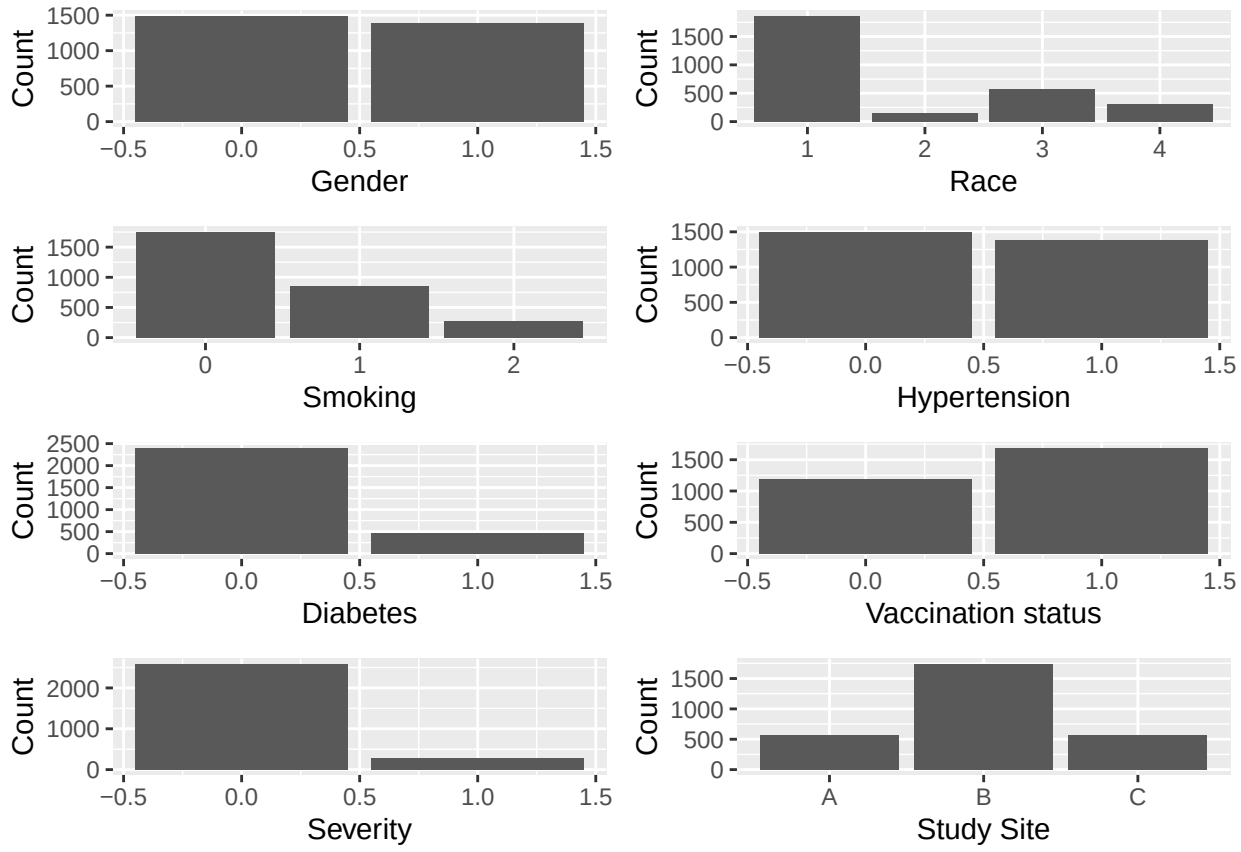
```
## ### Data Frame Summary
## **dat_bind**
## **Dimensions:** 2878 x 14
## **Duplicates:** 0
##
## ------------------------------------------------------------------------------------------------------
## No   Variable          Stats / Values              Freqs (% of Valid)    Graph
## ---- ----------------- --------------------------- --------------------- ----------------------------
## 1    gender\           Min  : 0\                   0 : 1490 (51.8%)\     IIIIIIIIII \
##      [integer]         Mean : 0.5\                 1 : 1388 (48.2%)      IIIIIIIII
##                        Max  : 1
##
## 2    race\             1\. 1\                      1863 (64.7%)\         IIIIIIIIIII \
```

```
##      [factor]          2\. 2\                    145 ( 5.0%)\          I \
##                        3\. 3\                    569 (19.8%)\          III \
##                        4\. 4                     301 (10.5%)           II
##
## 3    smoking\          1\. 0\                    1753 (60.9%)\         IIIIIIIIIII \
##      [factor]          2\. 1\                    846 (29.4%)\          IIIII \
##                        3\. 2                     279 ( 9.7%)           I
##
## 4    height\           Mean (sd) : 170.1 (5.9)\  311 distinct values   \ \ \ \ \ \ . :\
##      [numeric]         min < med < max:\                               \ \ \ \ \ \ : :\
##                        150.7 < 170.4 < 190.6\                          \ \ \ \ . : : .\
##                        IQR (CV) : 7.9 (0)                              \ \ \ \ : : : :\
##                                                                        \ \ . : : : : : .
##
## 5    weight\           Mean (sd) : 79.9 (7)\     358 distinct values   \ \ \ \ \ \ \ \ :\
##      [numeric]         min < med < max:\                               \ \ \ \ \ \ : : .\
##                        55.9 < 80.1 < 111.6\                            \ \ \ \ \ \ : : :\
##                        IQR (CV) : 9.5 (0.1)                            \ \ \ \ : : : :\
##                                                                        \ \ . : : : : : :
##
## 6    bmi\              Mean (sd) : 27.7 (2.7)\   162 distinct values   \ \ \ \ \ \ . :\
##      [numeric]         min < med < max:\                               \ \ \ \ \ \ : :\
##                        19.7 < 27.5 < 38.1\                             \ \ \ \ . : : :\
##                        IQR (CV) : 3.6 (0.1)                            \ \ \ \ : : : : .\
##                                                                        \ \ : : : : : : : .
##
## 7    hypertension\     Min  : 0\                 0 : 1499 (52.1%)\     IIIIIIIIII \
##      [numeric]         Mean : 0.5\               1 : 1379 (47.9%)      IIIIIIIII
##                        Max  : 1
##
## 8    diabetes\         Min  : 0\                 0 : 2403 (83.5%)\     IIIIIIIIIIIIIII \
##      [integer]         Mean : 0.2\               1 :  475 (16.5%)      III
##                        Max  : 1
##
## 9    SBP\              Mean (sd) : 130.2 (8)\    51 distinct values    \ \ \ \ \ \ \ \ : .\
##      [numeric]         min < med < max:\                               \ \ \ \ \ \ . : :\
##                        106 < 130 < 157\                                \ \ \ \ \ \ : : : :\
##                        IQR (CV) : 11 (0.1)                             \ \ \ \ : : : : :\
##                                                                        \ \ : : : : : : : .
##
## 10   LDL\              Mean (sd) : 110.6 (19.9)\ 121 distinct values   \ \ \ \ \ \ \ \ \ :\
##      [numeric]         min < med < max:\                               \ \ \ \ \ \ \ \ : : .\
##                        28 < 111 < 178\                                 \ \ \ \ \ \ \ \ : : :\
##                        IQR (CV) : 27 (0.2)                             \ \ \ \ \ \ . : : : :\
##                                                                        \ \ \ \ . : : : : : .
##
## 11   vaccine\          Min  : 0\                 0 : 1189 (41.3%)\     IIIIIIII \
##      [integer]         Mean : 0.6\               1 : 1689 (58.7%)      IIIIIIIIII
##                        Max  : 1
##
## 12   severity\         Min  : 0\                 0 : 2589 (90.0%)\     IIIIIIIIIIIIIIII \
##      [integer]         Mean : 0.1\               1 :  289 (10.0%)      II
##                        Max  : 1
##
```

```
## 13   study\            1\. A\                    571 (19.8%)\          III \
##      [character]       2\. B\                    1741 (60.5%)\         IIIIIIIIIII \
##                        3\. C                     566 (19.7%)           III
##
## 14   lrecovery_time\   Mean (sd) : 3.6 (0.6)\    147 distinct values   \ \ \ \ \ \ \ \ \ \ \ \ :\
##      [numeric]         min < med < max:\                               \ \ \ \ \ \ \ \ \ \ \ \ :\
##                        0 < 3.7 < 5.9\                                  \ \ \ \ \ \ \ \ \ \ : :\
##                        IQR (CV) : 0.6 (0.2)                            \ \ \ \ \ \ \ \ \ \ : :\
##                                                                        \ \ \ \ \ \ \ \ : : : :
## ---------------------------------------------------------------------------------------------------
```

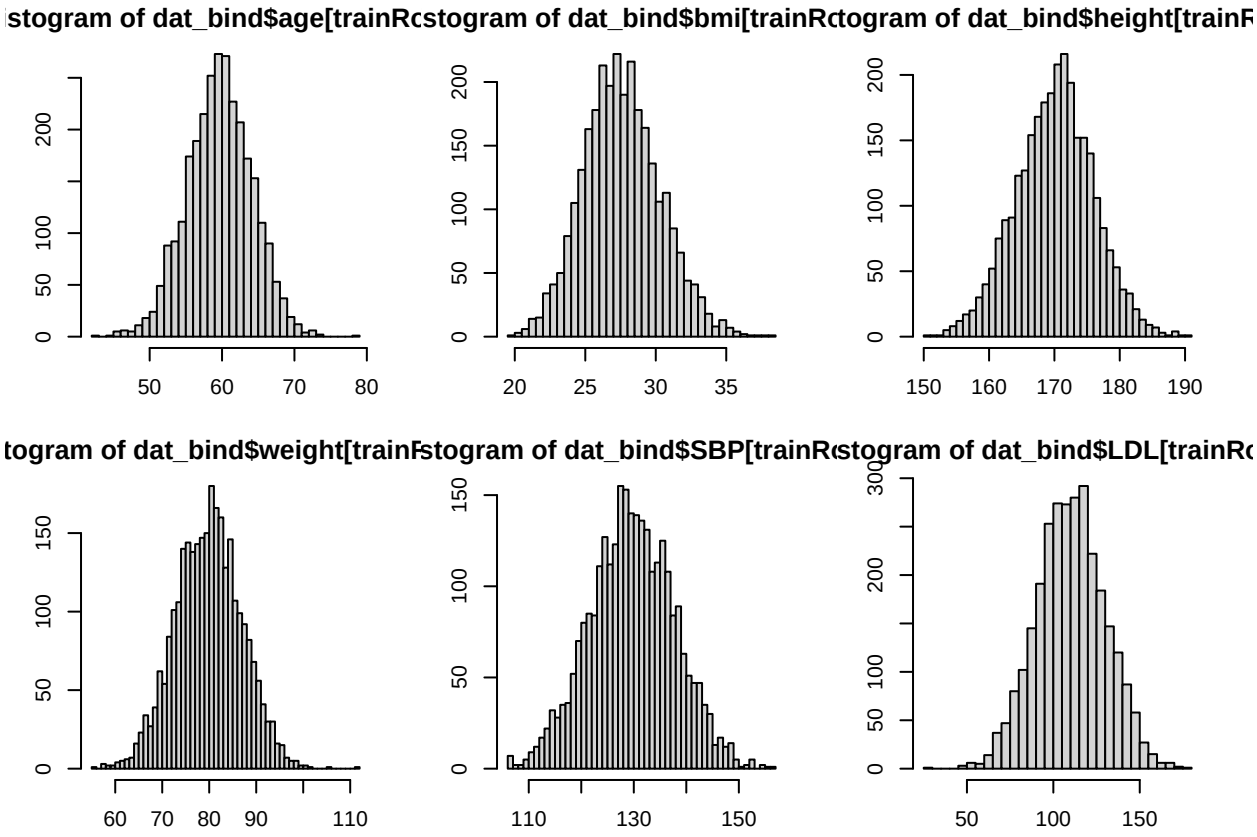
Understand categorical variables

```
gender = (dat_bind[trainRows, -1]) %>%
  ggplot(aes(x = gender)) +  geom_bar() +  labs(x = "Gender", y = "Count")

race = (dat_bind[trainRows, -1]) %>%
  ggplot(aes(x = race)) + geom_bar() + labs(x = "Race",y = "Count")

smoking = (dat_bind[trainRows, -1]) %>%
  ggplot(aes(x = smoking)) + geom_bar() + labs(x = "Smoking", y = "Count")

hypertension = (dat_bind[trainRows, -1]) %>%
  ggplot(aes(x = hypertension)) + geom_bar() + labs(x = "Hypertension",
                                                    y = "Count")
diabetes = (dat_bind[trainRows, -1]) %>%
  ggplot(aes(x = diabetes)) + geom_bar() + labs(x = "Diabetes",y = "Count")

vaccine = (dat_bind[trainRows, -1]) %>%
  ggplot(aes(x = vaccine)) + geom_bar() + labs(x = "Vaccination status",
                                               y = "Count")
severity = (dat_bind[trainRows, -1]) %>%
  ggplot(aes(x = severity)) + geom_bar() + labs(x = "Severity", y = "Count")

study = (dat_bind[trainRows, -1]) %>%
  ggplot(aes(x = study)) + geom_bar() + labs(x = "Study Site", y = "Count")

cat_combined_plot = ggarrange(gender, race, smoking, hypertension,
                              diabetes, vaccine,severity, study,
                          ncol = 2, nrow = 4)

cat_combined_plot
```



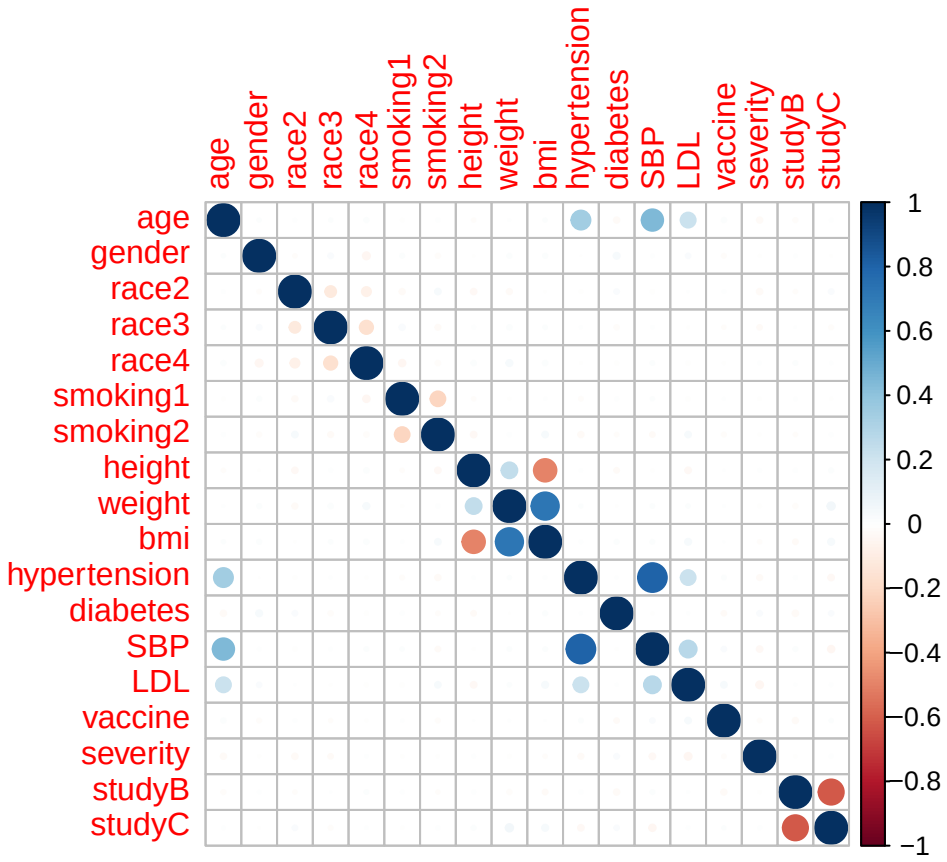
Understand continuous variables

```
par(mar = c(3, 3, 2, 2), mfrow = c(2, 3))

age = hist(dat_bind$age[trainRows], breaks = 50)

bmi = hist(dat_bind$bmi[trainRows], breaks = 50)

height = hist(dat_bind$height[trainRows], breaks = 50)

weight = hist(dat_bind$weight[trainRows], breaks = 50)

SBP = hist(dat_bind$SBP[trainRows], breaks = 50)

LDL = hist(dat_bind$LDL[trainRows], breaks = 50)
```



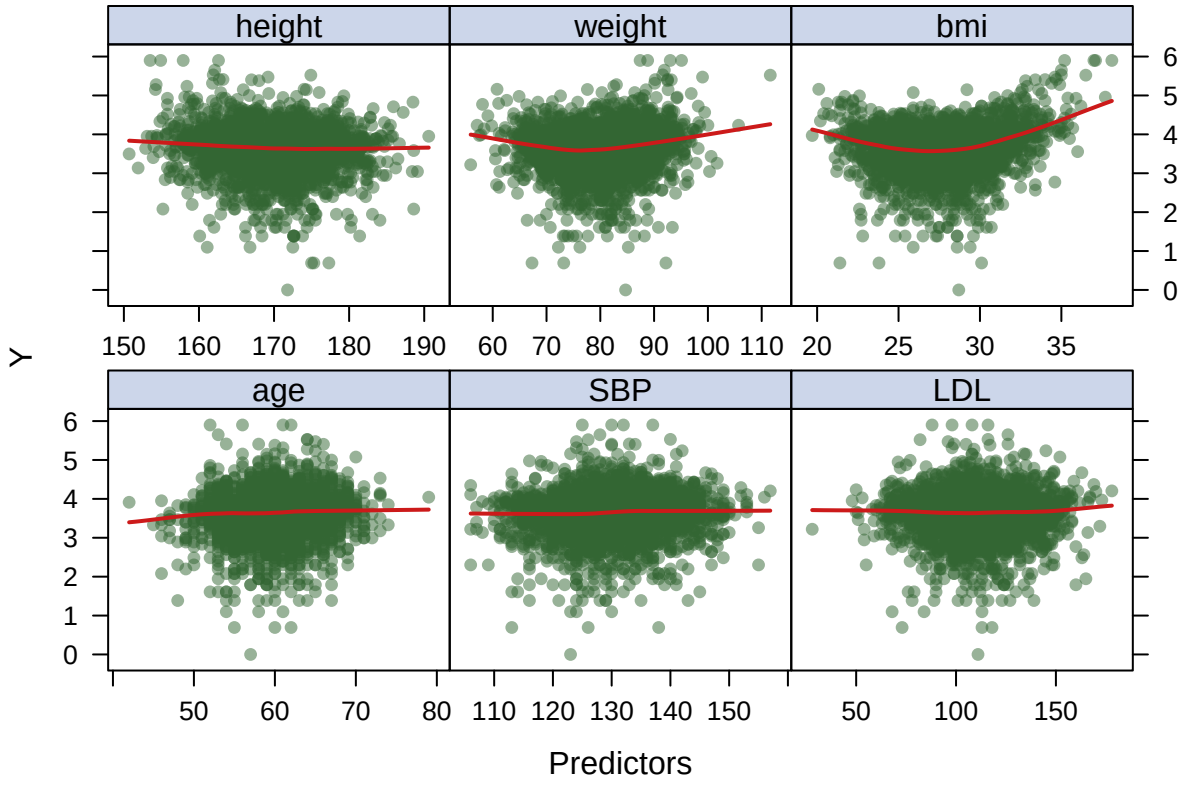
Understand the correlation between continuous predictors ****

```
correlation <- model.matrix(lrecovery_time ~ ., dat_bind)[trainRows,-1]

corrplot(cor(correlation), method = "circle", type = "full")
```



Understand the relationship with continuous predictors and the outcome

```
theme1 <- trellis.par.get()
theme1$plot.symbol$col <- rgb(.2, .4, .2, .5)
theme1$plot.symbol$pch <- 16
theme1$plot.line$col <- rgb(.8, .1, .1, 1)
theme1$plot.line$lwd <- 2
theme1$strip.background$col <- rgb(.0, .2, .6, .2)
trellis.par.set(theme1)

# plotting continuous predictors
featurePlot(x = model.matrix(lrecovery_time ~ ., dat_bind)[trainRows,c("age", "SBP", "LDL", "height", "
            y = dat_bind$lrecovery_time[trainRows],
            plot = "scatter",
            span = .5,
            labels = c("Predictors","Y"),
            type = c("p", "smooth"),
            layout = c(3,2))
```



Considering variables based on the EDA

From the correlation plot, we can observe that `bmi` is highly correlated with `weight` and `height`, which makes sense because BMI is calculated by weight divided by the square of height. This demonstrates collinearity between the variables, and to account for this, I will remove the `bmi` variable for the predictions.

Also, I believe that the `study` variable is more of a geographical indicator to distinguish different study sites, and it will not be critical in predicting recovery time. Therefore, I will also remove the `study` variable.

Lastly, I will remove variables `race` and `smoking` since I have created dummy variables for them and I will use the dummy variables in further analyses.

```
final = dat_bind %>%
  mutate(
  # create dummy variables for categorical variables
    # set up 3 dummy variables for `race`, reference = White:
         race_2 = ifelse(race == 2, 1, 0),
         race_3 = ifelse(race == 3, 1, 0),
         race_4 = ifelse(race == 4, 1, 0),
    # set up 2 dummy variables for `smoking`, reference = Never smoked:
         smoking_1 = ifelse(smoking == 1, 1, 0),
         smoking_2 = ifelse(smoking == 2, 1, 0))

# remove variables that will not be used
final = final %>%
  select(-bmi, -study, -race, -smoking)
```

```r
# partition again based on the new outcome variable
set.seed(2460)

trainRows_new <- createDataPartition(y = final$lrecovery_time, p = 0.8, list = FALSE)

x <- model.matrix(lrecovery_time ~ ., final)[trainRows_new,-1]

y <- final$lrecovery_time[trainRows_new]

x2 <- model.matrix(lrecovery_time ~ ., final)[-trainRows_new,-1]

y2 <- final$lrecovery_time[-trainRows_new]

#ctrl1 <- trainControl(method = "repeatedcv", number = 10, repeats = 5)

ctrl1 <- trainControl(method = "cv")
```

Primary analysis: continuous time to recovery

Mode 1: linear model

Train the model

```r
set.seed(2460)

lm.fit <- train(x, y,
                method = "glm",
                preProcess = c("center", "scale"),
                trControl = ctrl1)

summary(lm.fit)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.4903  -0.2520   0.0629   0.3101   2.1554
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.601182   0.010189 353.424  < 2e-16 ***
## age           0.035305   0.011452   3.083  0.00207 **
## gender       -0.058928   0.010219  -5.766 8.97e-09 ***
## height       -0.067022   0.010525  -6.368 2.22e-10 ***
## weight        0.071532   0.010519   6.800 1.27e-11 ***
## hypertension  0.037399   0.017038   2.195  0.02824 *
## diabetes     -0.007324   0.010216  -0.717  0.47350
## SBP          -0.001230   0.017996  -0.068  0.94552
```

```
## LDL          -0.007225   0.010697  -0.675  0.49944
## vaccine      -0.090306   0.010204  -8.850  < 2e-16 ***
## severity      0.050085   0.010216   4.903 9.98e-07 ***
## race_2        0.011106   0.010331   1.075  0.28246
## race_3       -0.006021   0.010444  -0.577  0.56428
## race_4        0.003255   0.010433   0.312  0.75506
## smoking_1     0.046371   0.010456   4.435 9.55e-06 ***
## smoking_2     0.054759   0.010461   5.234 1.78e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.2988063)
##
##     Null deviance: 937.87  on 2877  degrees of freedom
## Residual deviance: 855.18  on 2862  degrees of freedom
## AIC: 4708.9
##
## Number of Fisher Scoring iterations: 2
```

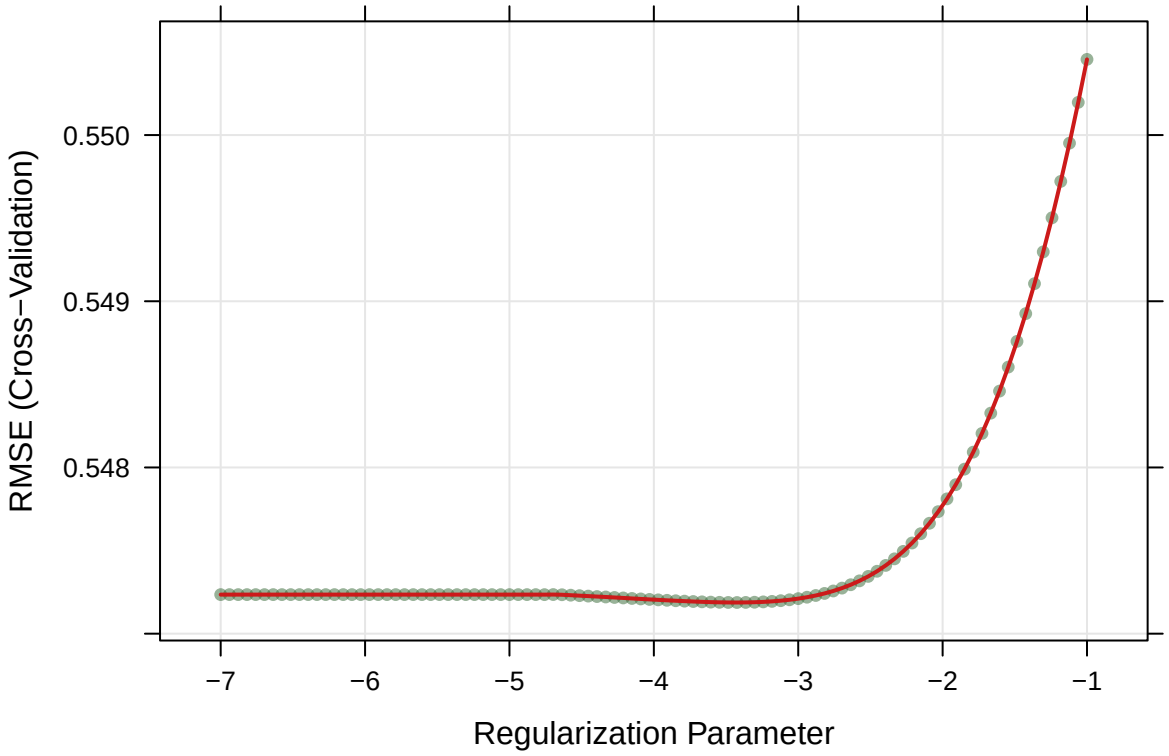
Model 2: Ridge

Train the model

```
set.seed(2460)

ridge.fit <- train(x, y,
                   method = "glmnet",
                   tuneGrid = expand.grid(alpha = 0,
                                          lambda = exp(seq(-1, -7, length=100))),
                   preProc = c("center", "scale"),
                   trControl = ctrl1)

ridge.fit$bestTune
```

```
##    alpha     lambda
## 60     0 0.03257395
```

```
plot(ridge.fit, xTrans = log)
```




```
coef(ridge.fit$finalModel, s = ridge.fit$bestTune$lambda)
```

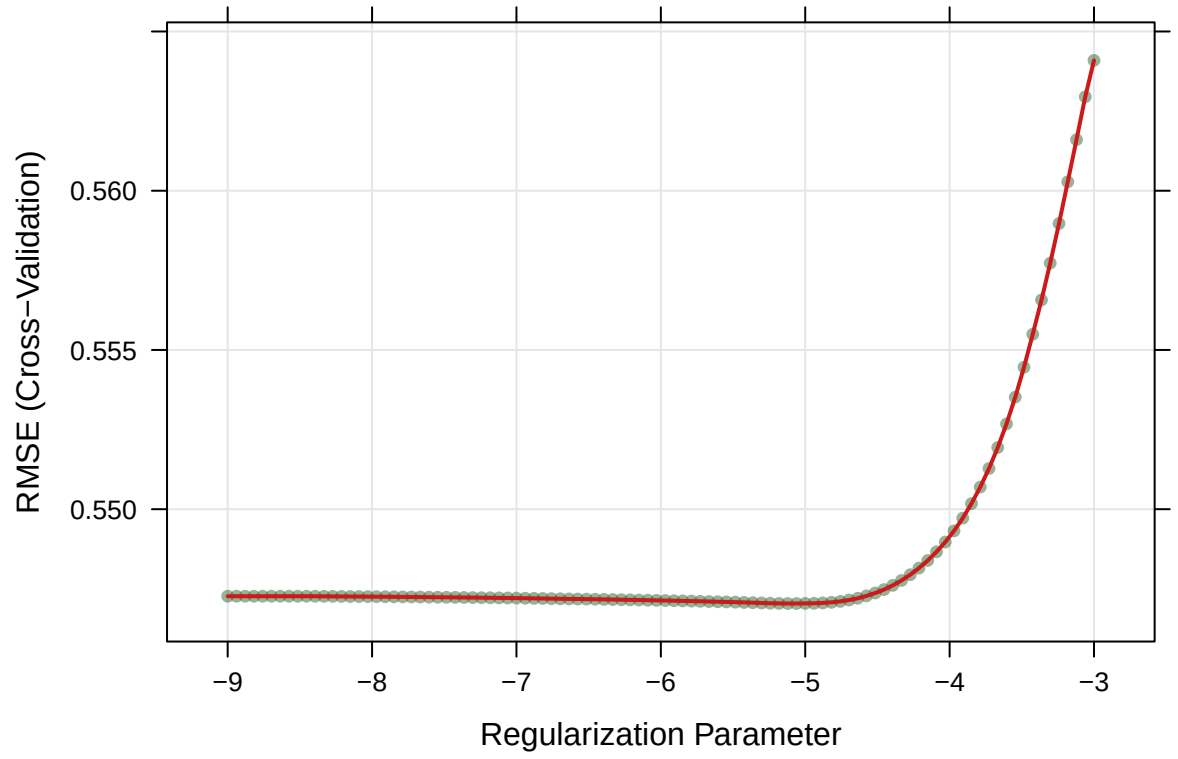
```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                       s1
## (Intercept)   3.601182153
## age           0.032920705
## gender       -0.055663811
## height       -0.062349580
## weight        0.066525616
## hypertension  0.032223709
## diabetes     -0.006817965
## SBP           0.003430640
## LDL          -0.006584234
## vaccine      -0.085437950
## severity      0.047319038
## race_2        0.010490406
## race_3       -0.005756413
## race_4        0.003127230
## smoking_1     0.042998547
## smoking_2     0.051421885
```

Model 3: Lasso

```
set.seed(2460)

lasso.fit <- train(x, y,
                   method = "glmnet",
                   tuneGrid = expand.grid(alpha = 1,
                                          lambda = exp(seq(-9, -3, length = 100))),
                   preProcess = c("center", "scale"),
                   trControl = ctrl1)

lasso.fit$bestTune
```

```
##    alpha      lambda
## 66     1 0.006341715
```

```
plot(lasso.fit, xTrans = log)
```




```
coef(lasso.fit$finalModel, s = lasso.fit$bestTune$lambda)
```

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                            s1
## (Intercept)   3.6011821535
## age           0.0294604496
## gender       -0.0530470560
```

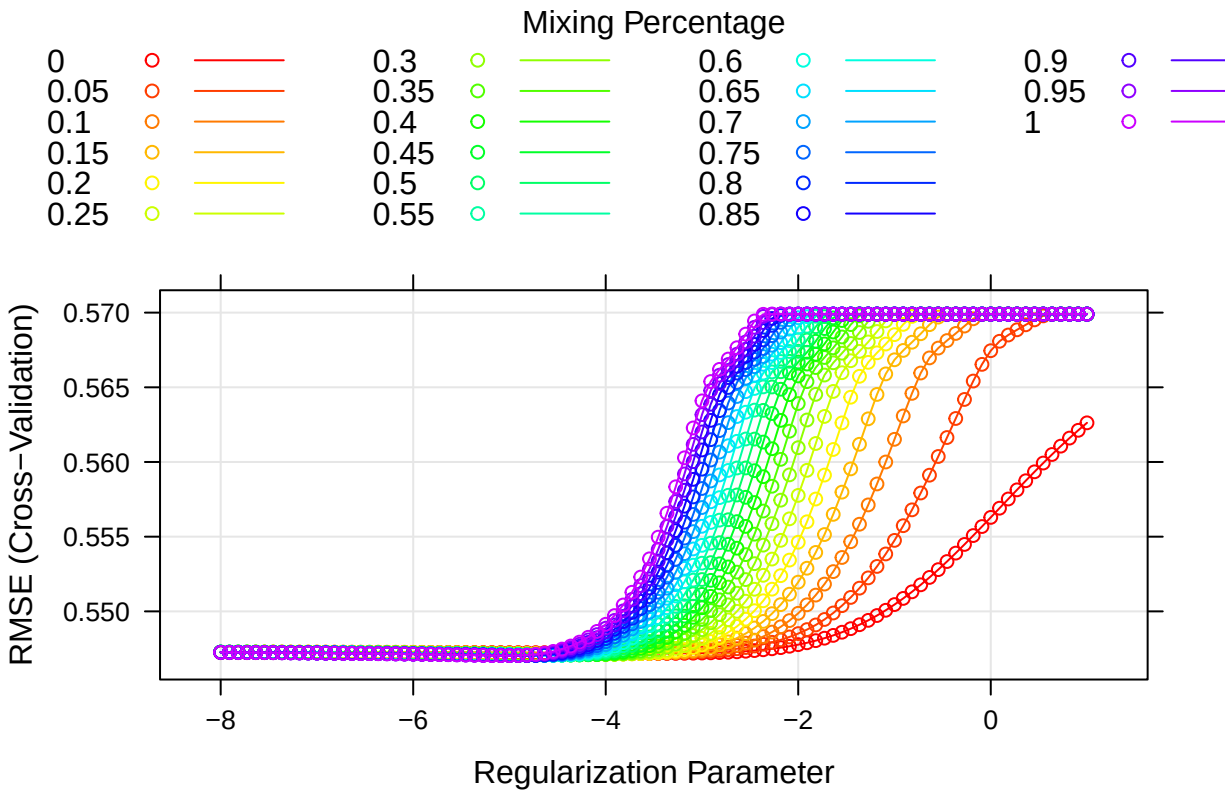
```
## height       -0.0586996415
## weight        0.0629186585
## hypertension  0.0299165942
## diabetes     -0.0007009403
## SBP           .
## LDL           .
## vaccine      -0.0838727387
## severity      0.0434688048
## race_2        0.0048256040
## race_3       -0.0007603145
## race_4        .
## smoking_1     0.0377315248
## smoking_2     0.0471105049
```

Model 4: Elastic net

```
set.seed(2460)

enet.fit <- train(x, y,
                  method = "glmnet",
                  tuneGrid = expand.grid(alpha = seq(0, 1, length = 21),
                                         lambda = exp(seq(-8, 1, length = 100))),
                  preProcess = c("center", "scale"),
                  trControl = ctrl1)

enet.fit$bestTune
```

```
##      alpha      lambda
## 2033     1 0.006152424
```

```
myCol <- rainbow(25)

myPar <- list(superpose.symbol = list(col = myCol),
              superpose.line = list(col = myCol))

plot(enet.fit, par.settings = myPar, xTrans = log)
```




```
coef(enet.fit$finalModel, s = enet.fit$bestTune$lambda)
```

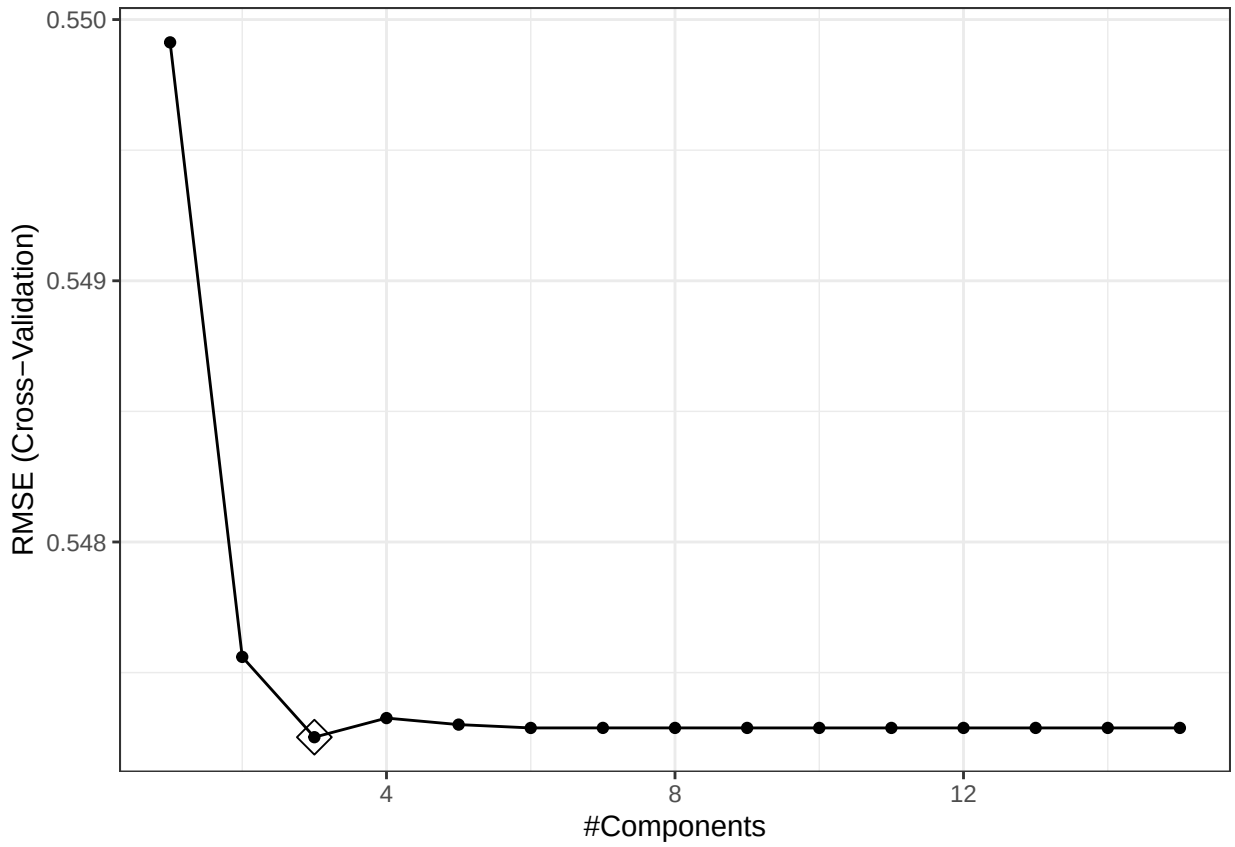
```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                        s1
## (Intercept)   3.601182153
## age           0.029597729
## gender       -0.053231523
## height       -0.058939769
## weight        0.063175514
## hypertension  0.030076866
## diabetes     -0.000901005
## SBP           .
## LDL           .
## vaccine      -0.084071644
## severity      0.043674572
## race_2        0.005004129
## race_3       -0.000933943
## race_4        .
## smoking_1     0.037983040
## smoking_2     0.047328010
```

Model 5: Partial least square

```
set.seed(2460)

pls.fit <- train(x, y,
                 method = "pls",
                 tuneGrid = data.frame(ncomp = 1:15),
                 trControl = ctrl1,
                 preProcess = c("center", "scale"))

pls.fit$bestTune
```

```
##   ncomp
## 3     3
```

```
ggplot(pls.fit, highlight = TRUE) + theme_bw()
```




```
summary(pls.fit)
```

```
## Data:    X dimension: 2878 15
##  Y dimension: 2878 1
## Fit method: oscorespls
## Number of components considered: 3
```

```
## TRAINING: % variance explained
##           1 comps  2 comps  3 comps
## X           8.767    20.78   27.176
## .outcome    7.824     8.74    8.798
```

```
coef(pls.fit$finalModel)
```

```
## , , 3 comps
##
##                   .outcome
## age           0.033477700
## gender       -0.059024179
## height       -0.066831383
## weight        0.071791734
## hypertension  0.025935728
## diabetes     -0.007453697
## SBP           0.011296191
## LDL          -0.007629102
## vaccine      -0.090455365
## severity      0.050006171
## race_2        0.011134505
## race_3       -0.005242574
## race_4        0.002406789
## smoking_1     0.047587597
## smoking_2     0.054282186
```

Model 6: GAM

```
set.seed(2460)

gam.fit <- train(x, y,
                 method = "gam",
                 tuneGrid = data.frame(method = "GCV.Cp", select = c(TRUE,FALSE)),
                 preProcess = c("center", "scale"),
                 trControl = ctrl1)

gam.fit$bestTune
```

```
##   select method
## 1  FALSE GCV.Cp
```

```
gam.fit$finalModel
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## .outcome ~ gender + hypertension + diabetes + vaccine + severity +
##     race_2 + race_3 + race_4 + smoking_1 + smoking_2 + s(age) +
```

```
##     s(SBP) + s(LDL) + s(height) + s(weight)
##
## Estimated degrees of freedom:
## 1.00 1.29 1.54 2.40 3.43  total = 20.66
##
## GCV score: 0.291134
```

```
summary(gam.fit)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## .outcome ~ gender + hypertension + diabetes + vaccine + severity +
##     race_2 + race_3 + race_4 + smoking_1 + smoking_2 + s(age) +
##     s(SBP) + s(LDL) + s(height) + s(weight)
##
## Parametric coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.601182   0.010022 359.342  < 2e-16 ***
## gender       -0.058501   0.010056  -5.817 6.65e-09 ***
## hypertension  0.041107   0.017011   2.417   0.0157 *
## diabetes     -0.006185   0.010066  -0.614   0.5390
## vaccine      -0.091605   0.010041  -9.123  < 2e-16 ***
## severity      0.048649   0.010055   4.838 1.38e-06 ***
## race_2        0.009961   0.010171   0.979   0.3275
## race_3       -0.004720   0.010277  -0.459   0.6461
## race_4        0.001637   0.010268   0.159   0.8734
## smoking_1     0.048308   0.010292   4.694 2.81e-06 ***
## smoking_2     0.056077   0.010300   5.444 5.65e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##             edf Ref.df      F p-value
## s(age)    1.000  1.000  8.724 0.00317 **
## s(SBP)    1.293  1.529  0.191 0.84218
## s(LDL)    1.540  1.925  0.921 0.46197
## s(height) 2.401  3.068 15.676 < 2e-16 ***
## s(weight) 3.426  4.363 28.478 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.113   Deviance explained = 11.9%
## GCV = 0.29113  Scale est. = 0.28904   n = 2878
```

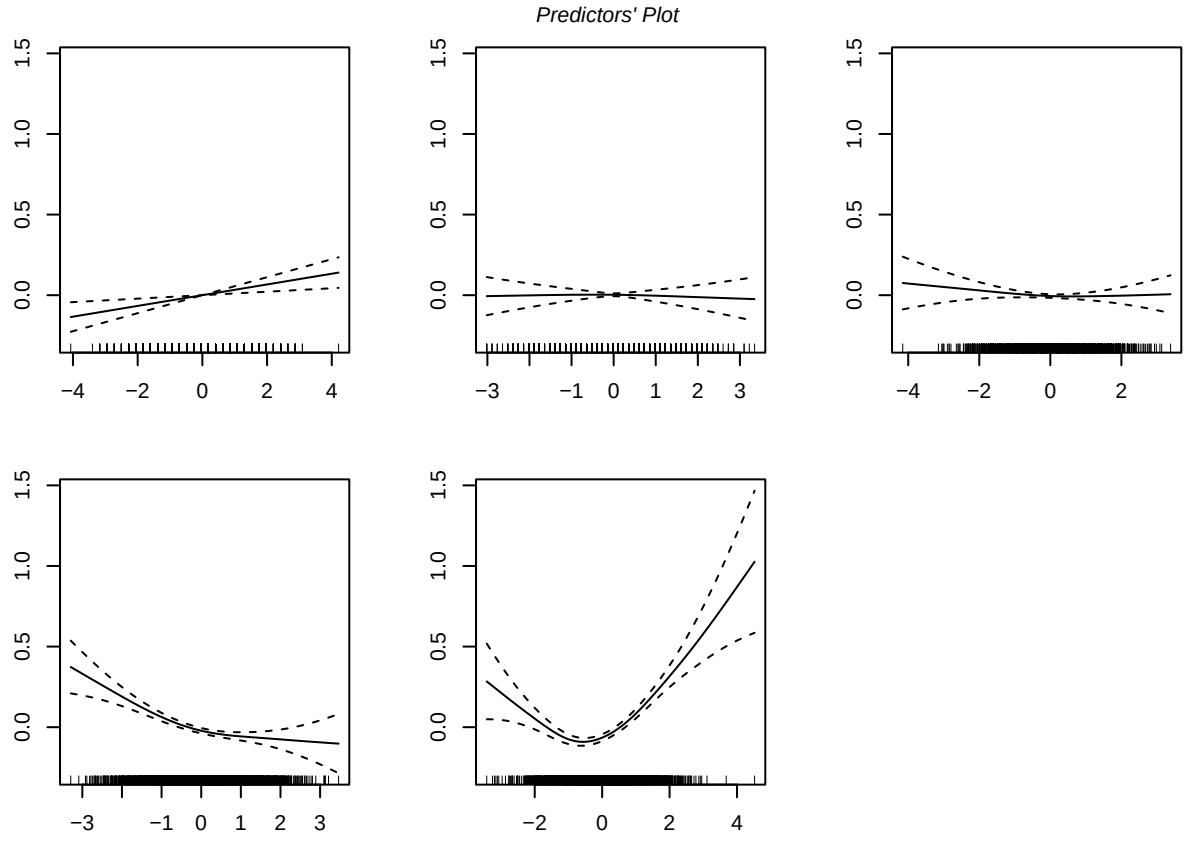
Plot continuous predictors in GAM

```
var.names <- c("age", "SBP", "LDL", "height", "weight", "-")
```

```r
# make a matrix for easier comprehension of the plot with 16 predictors
matrix <- matrix(var.names, nrow = 2, ncol = 3, byrow = TRUE)

# use the matrix to correspond each plot with each predictor
print(matrix)
```

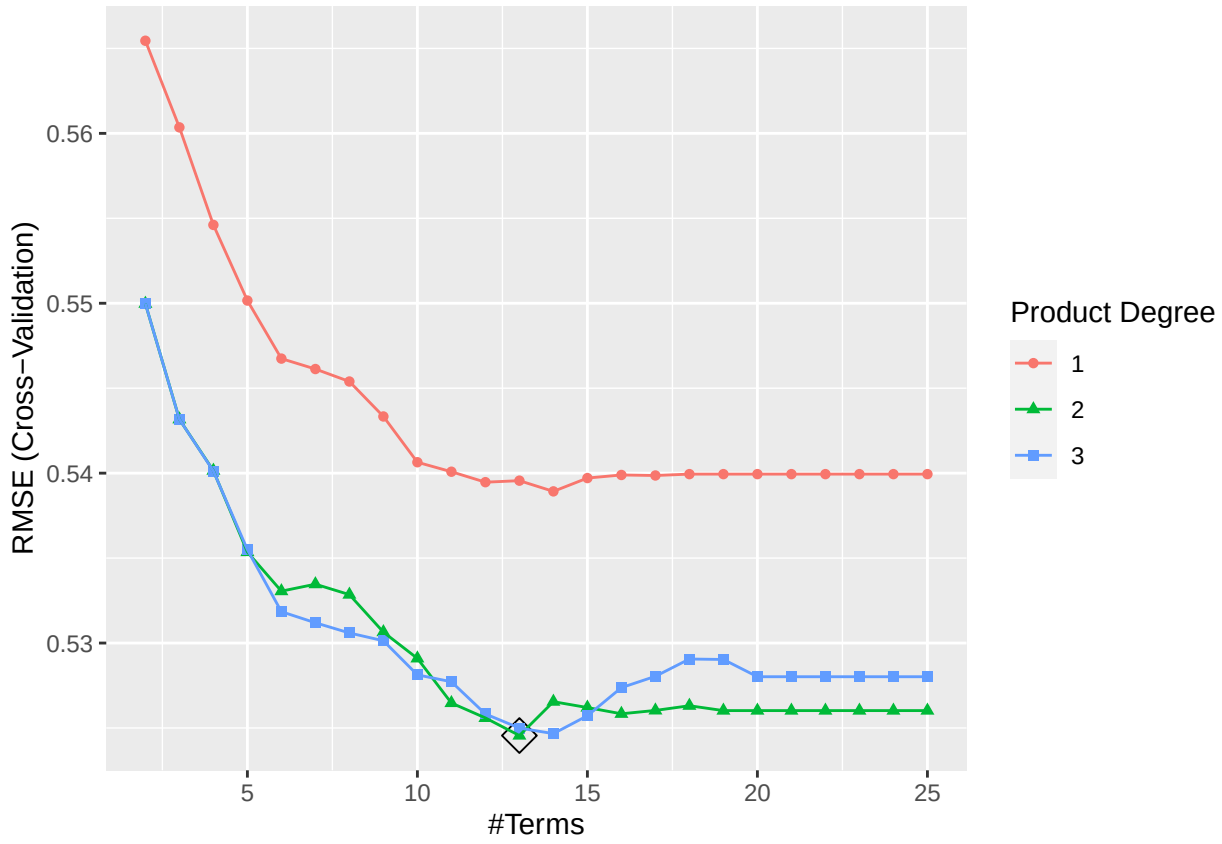
```
##      [,1]     [,2]     [,3]
## [1,] "age"    "SBP"    "LDL"
## [2,] "height" "weight" "-"
```

```r
gam.plot <- gam.fit$finalModel

# make 16 plots into one
par(mar = c(3, 3, 2, 2), mfrow = c(2, 3))

plot(gam.plot)

title(main = "Predictors' Plot", cex.main = 1, font.main = 3, outer = TRUE, line = -1)
```



Model 7: MARS

```
mars_grid <- expand.grid(degree = 1:3,
                         nprune = 2:25)

set.seed(2460)

mars.fit <- train(x, y,
                  method = "earth",
                  tuneGrid = mars_grid,
                  preProcess = c("center", "scale"),
                  trControl = ctrl1)

ggplot(mars.fit, highlight = TRUE)
```




```
mars.fit$bestTune
```

```
##    nprune degree
## 36     13      2
```

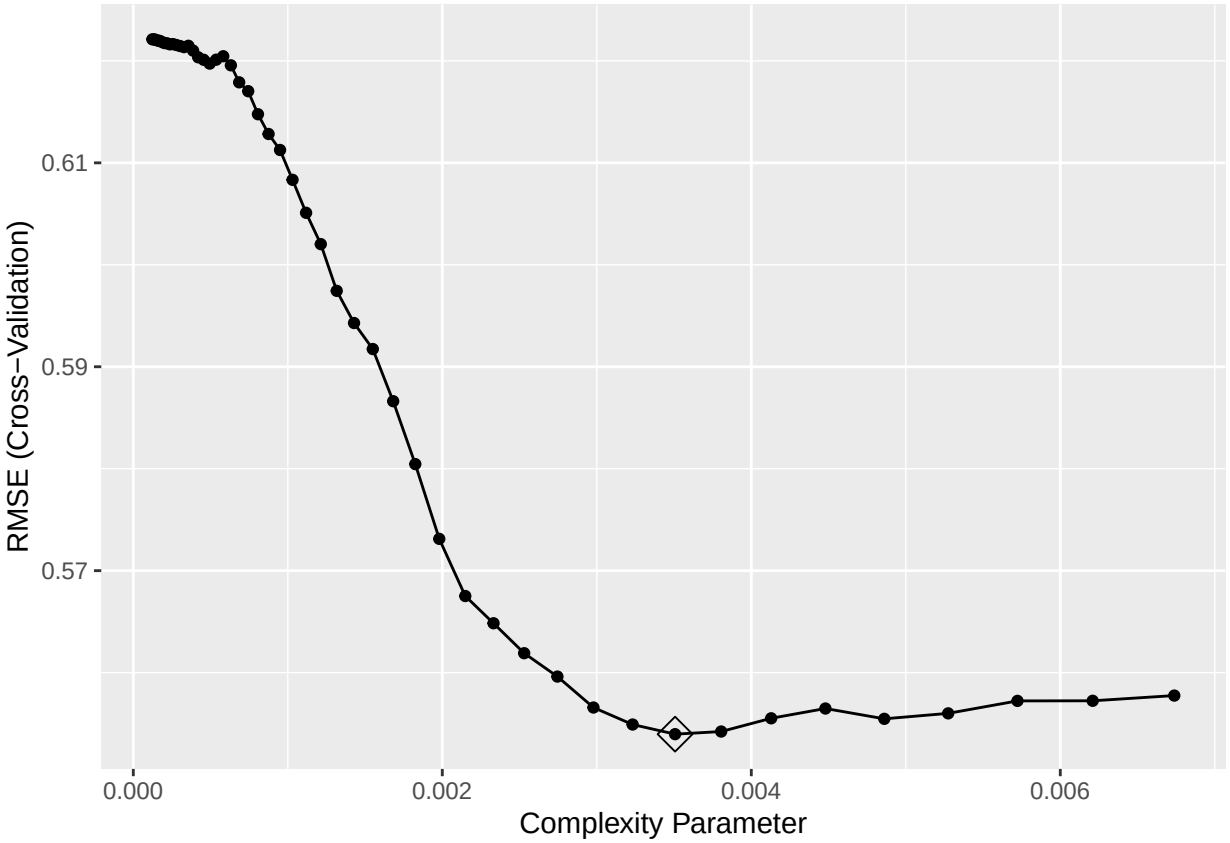
```
coef(mars.fit$finalModel)
```

```
##                                (Intercept)
##                                 3.49876550
## h(-0.260011-height) * h(weight- -0.827164)
```

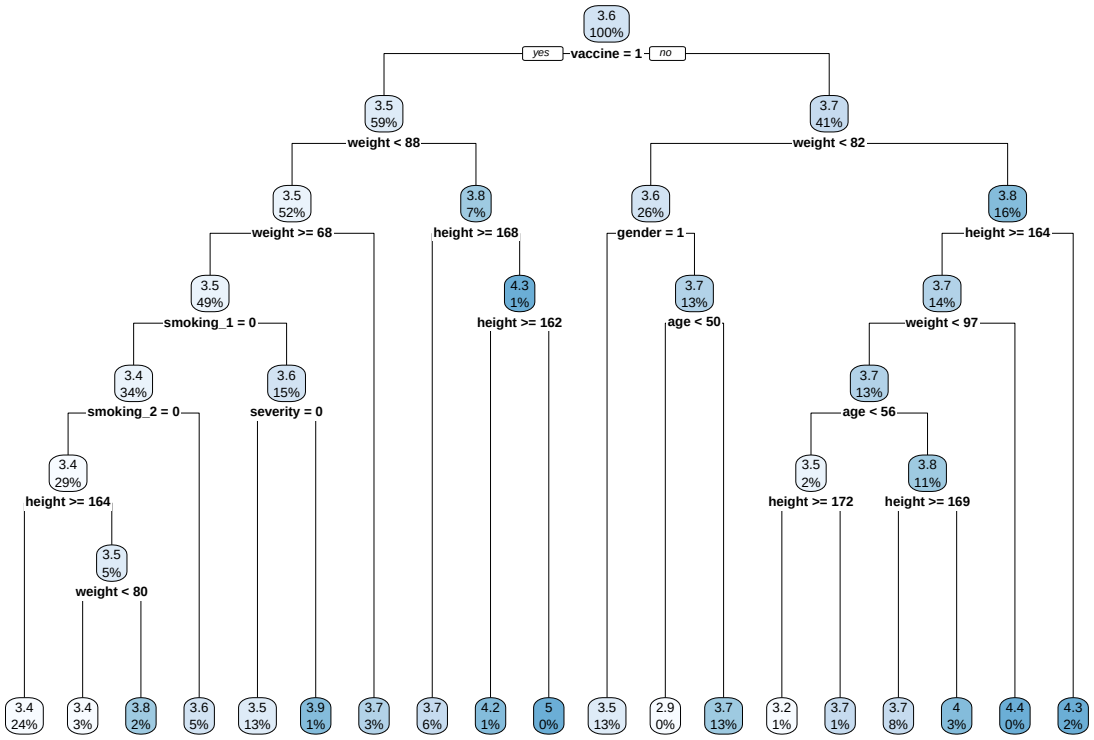
```
##                                0.25089199
##                                   vaccine
##                               -0.08986125
##                                    gender
##                               -0.06078746
##  h(height- -1.59822) * h(-0.827164-weight)
##                                0.18077865
##                          h(weight-0.798914)
##                                0.48334562
##   h(height- -0.53104) * h(weight-0.798914)
##                               -0.21549862
##                                  severity
##                                0.04599274
##                                 smoking_2
##                                0.05691174
##                                 smoking_1
##                                0.04753692
##                              hypertension
##                                0.04828695
##       h(age- -1.83095) * h(weight-2.16824)
##                                0.19508041
##         h(-1.83095-age) * h(SBP- -1.01857)
##                               -0.84560948
```

Model 8: Regression tree

```
set.seed(2460)

rpart.fit <- train(lrecovery_time ~ . ,
                   final[trainRows_new,],
                   method = "rpart",
                   tuneGrid = data.frame(cp = exp(seq(-9,-5, length = 50))),
                   trControl = ctrl1)

ggplot(rpart.fit, highlight = TRUE)
```

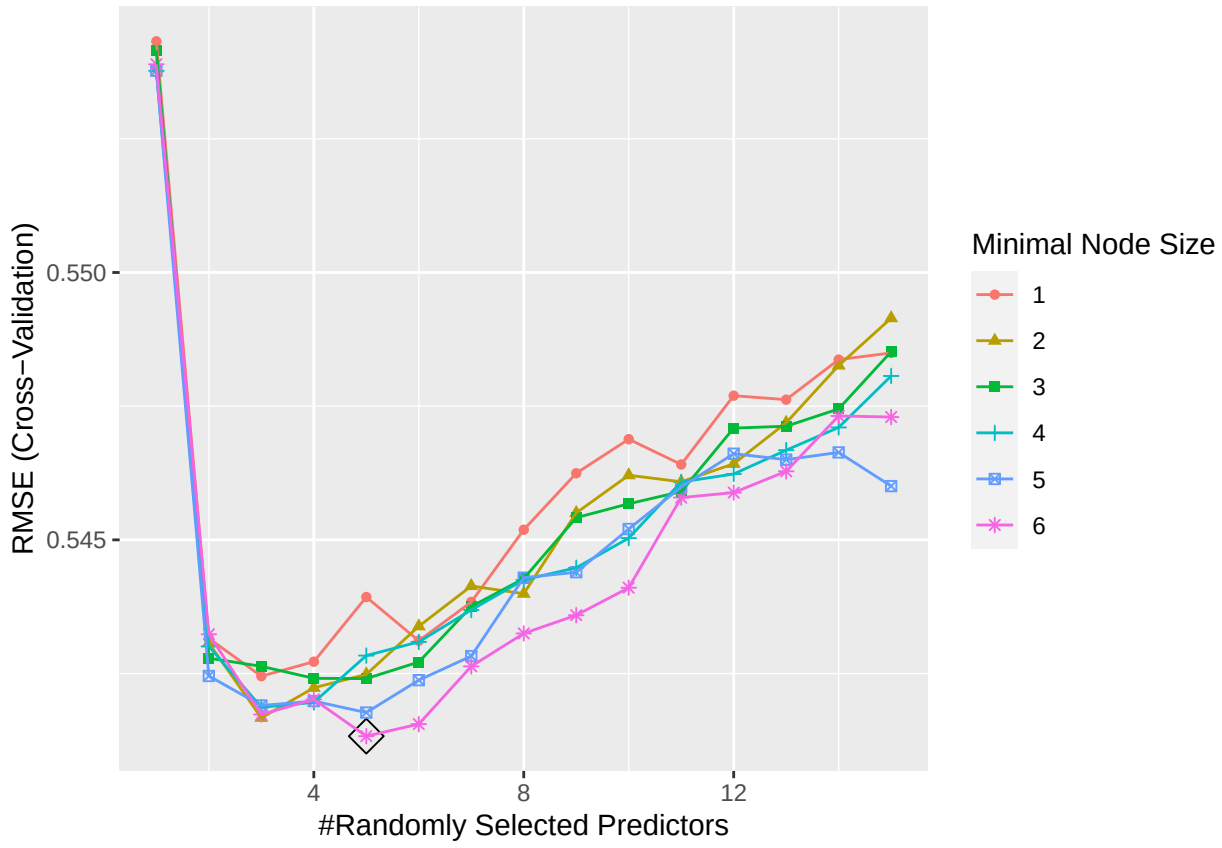



```
rpart.plot(rpart.fit$finalModel)
```



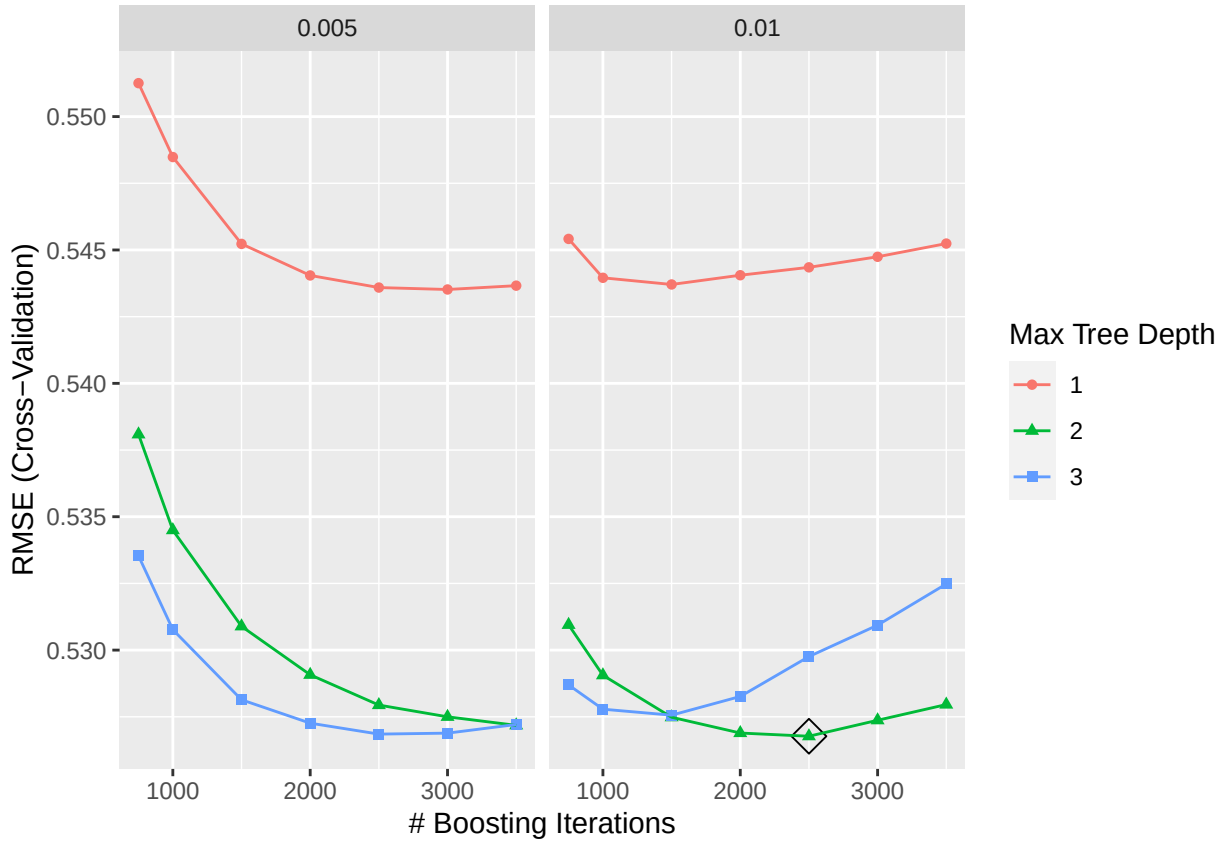
Model 9: Random forest

```
rf.grid <- expand.grid(mtry = 1:15, #15 predictors
                       splitrule = "variance",
                       min.node.size = 1:6)

set.seed(2460)

rf.fit <- train(lrecovery_time ~ . ,
                final[trainRows_new,],
                method = "ranger",
                tuneGrid = rf.grid,
                trControl = ctrl1)

ggplot(rf.fit, highlight = TRUE)
```



Model 10: Boosting

```
gbm.grid <- expand.grid(n.trees = c(750,1000,1500,2000,2500,3000,3500),
                        interaction.depth = 1:3,
                        shrinkage = c(0.005,0.01),
                        n.minobsinnode = c(1))

set.seed(2460)

gbm.fit <- train(lrecovery_time ~ . ,
                 final[trainRows_new,],
                 method = "gbm",
                 tuneGrid = gbm.grid,
                 trControl = ctrl1,
                 verbose = F)

ggplot(gbm.fit, highlight = T)
```



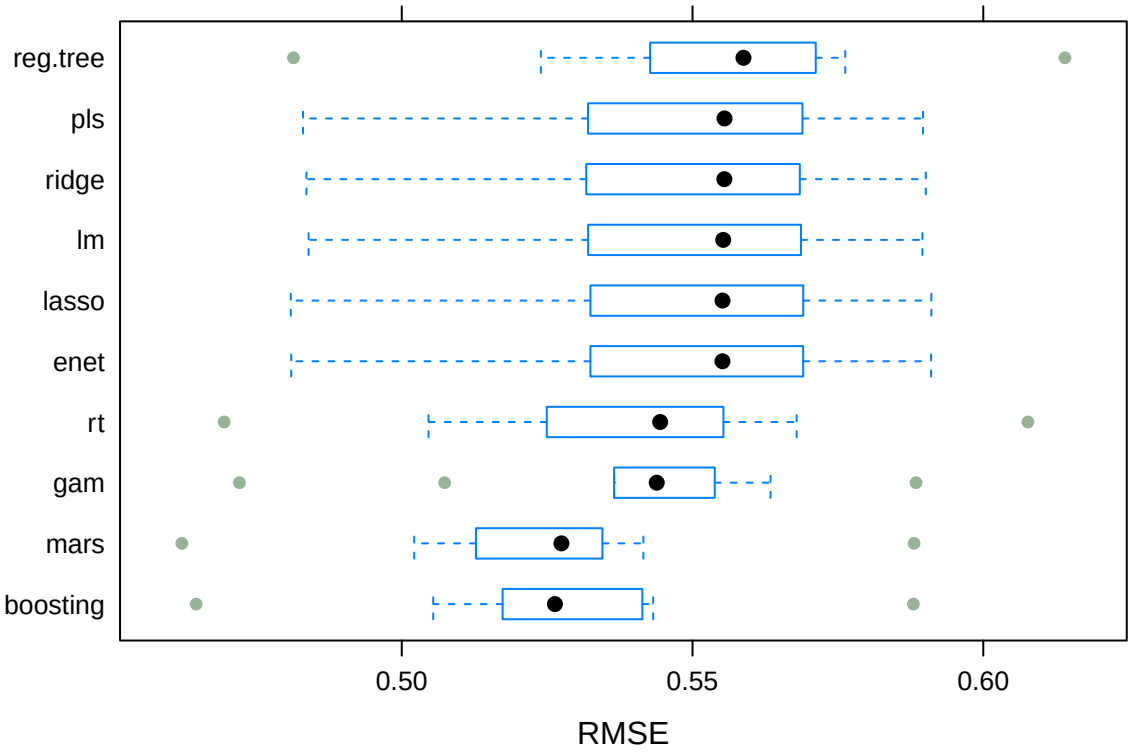
Model comparison

```
res <- resamples(list(lm = lm.fit,
                      ridge = ridge.fit,
                      lasso = lasso.fit,
                      enet = enet.fit,
                      pls = pls.fit,
                      gam = gam.fit,
                      mars = mars.fit,
                      reg.tree = rpart.fit,
                      rt = rf.fit,
                      boosting = gbm.fit))

summary(res)
```

```
##
## Call:
## summary.resamples(object = res)
##
## Models: lm, ridge, lasso, enet, pls, gam, mars, reg.tree, rt, boosting
## Number of resamples: 10
##
## MAE
```

```
##                Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lm       0.3672370 0.3865806 0.4016759 0.3963931 0.4057979 0.4212521    0
## ridge    0.3663893 0.3858184 0.4007157 0.3957724 0.4057746 0.4205702    0
## lasso    0.3641447 0.3844048 0.3997274 0.3948164 0.4053024 0.4205352    0
## enet     0.3642115 0.3844272 0.3997751 0.3948488 0.4053092 0.4205531    0
## pls      0.3664275 0.3861188 0.4020634 0.3964451 0.4061394 0.4207914    0
## gam      0.3665705 0.3883064 0.3941720 0.3934222 0.3976350 0.4183411    0
## mars     0.3605023 0.3759073 0.3827130 0.3828179 0.3896047 0.4111283    0
## reg.tree 0.3698327 0.3973105 0.4060173 0.4041754 0.4160490 0.4349119    0
## rt       0.3655662 0.3879478 0.3962364 0.3967996 0.4034208 0.4299948    0
## boosting 0.3606087 0.3800204 0.3816756 0.3836988 0.3888841 0.4114478    0
##
## RMSE
##                Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lm       0.4839741 0.5354972 0.5552816 0.5472882 0.5681003 0.5895310    0
## ridge    0.4836118 0.5350047 0.5554747 0.5471876 0.5679490 0.5901098    0
## lasso    0.4809255 0.5353835 0.5551609 0.5470383 0.5686550 0.5910677    0
## enet     0.4809878 0.5353887 0.5551527 0.5470381 0.5686367 0.5910205    0
## pls      0.4830191 0.5356004 0.5555022 0.5472527 0.5682639 0.5896238    0
## gam      0.4721006 0.5366099 0.5438440 0.5396259 0.5528714 0.5884379    0
## mars     0.4621855 0.5133002 0.5274561 0.5245544 0.5344957 0.5880896    0
## reg.tree 0.4813814 0.5436530 0.5587578 0.5539620 0.5699192 0.6140393    0
## rt       0.4694629 0.5291217 0.5444377 0.5413292 0.5546981 0.6076705    0
## boosting 0.4646449 0.5192539 0.5263374 0.5267712 0.5385465 0.5879888    0
##
## Rsquared
##                 Min.    1st Qu.     Median       Mean    3rd Qu.      Max. NA's
## lm       0.04286670 0.06516457 0.08215628 0.08007858 0.09627282 0.1082650    0
## ridge    0.04284037 0.06517825 0.08206608 0.08009576 0.09622654 0.1086690    0
## lasso    0.04243978 0.06563023 0.08182797 0.08126462 0.10081870 0.1125835    0
## enet     0.04244385 0.06569254 0.08180434 0.08121464 0.10080075 0.1124389    0
## pls      0.04165485 0.06640106 0.08173479 0.08022472 0.09734494 0.1076085    0
## gam      0.05484980 0.07377435 0.10031228 0.10766035 0.13754613 0.1703333    0
## mars     0.06938156 0.11875170 0.16991301 0.15737021 0.18480431 0.2434841    0
## reg.tree 0.01247323 0.04761855 0.06651702 0.07630246 0.11216803 0.1330730    0
## rt       0.04055203 0.07556116 0.09890249 0.10401770 0.14406515 0.1638617    0
## boosting 0.08201504 0.11297991 0.15033425 0.14831998 0.19115410 0.2076651    0
```

```
bwplot(res, metric = "RMSE")
```



Secondary analysis: binary time to recovery