

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 <code>recovery_time</code>	2
Summary of the dataset	4
Understand categorical variables	5
Understand continuous variables	6
Understand the correlation between continuous predictors	7
Understand the relationship with continuous predictors and the outcome	8
Understand the relationship between categorical predictors and continuous outcome	9
Considering variables based on the EDA	10
Primary analysis: continuous time to recovery	11
Model 1: Linear model	11
Model 2: Ridge	12
Model 3: Lasso	13
Model 4: Elastic net	14
Model 5: Partial least square	15
Model 6: GAM	16
Model 7: MARS	19
Model 8: Regression tree	22
Model 9: Random forest	24
Model 10: Boosting	24
Model comparison	25
Select final model	27
Secondary analysis: binary time to recovery	27
Set up	27
Understand the relationship between continuous predictors and the binary outcome	28
Understand the relationship between categorical predictors and the binary outcome	29
Make dummy variables	31
Model 1: Logistic regression	31
Model 2: MARS	32
Model 3: LDA	34
Model 4: Classification tree	35
Model 5: Random forest	36
Model 6: Boosting	37
Model comparison	38
Select final model	40

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)
library(pdp)
library(vip)

# 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

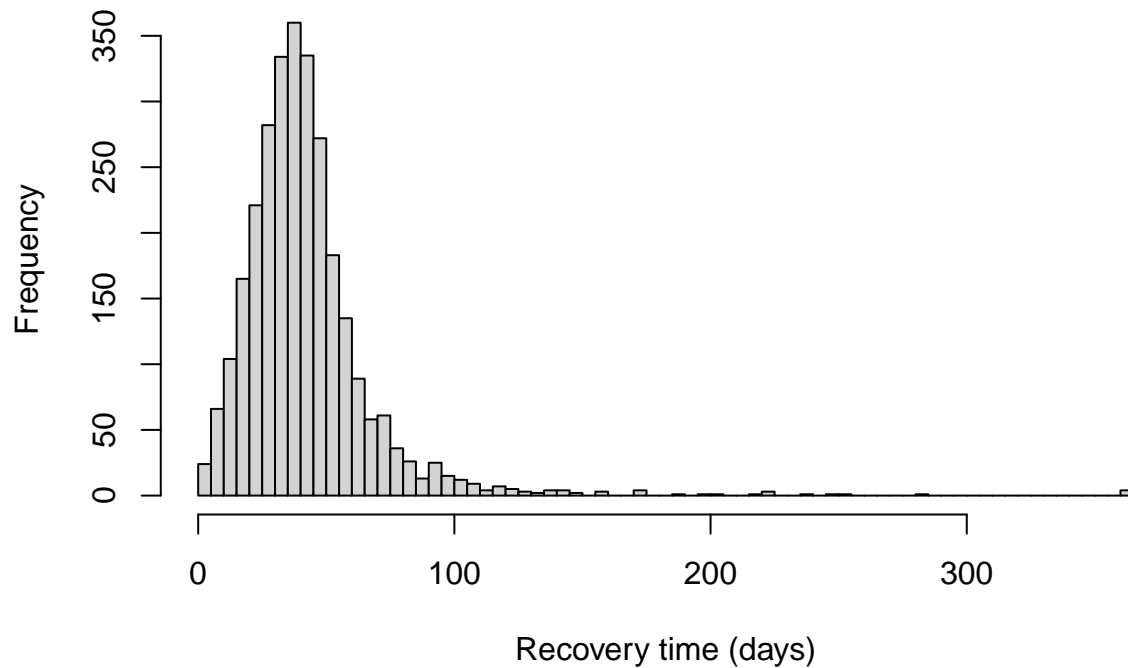
Here, we mainly want to investigate the EDA of the training dataset. Therefore, we will start with the data partition.

```
set.seed(2460)

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

Understanding the outcome variable recovery_time

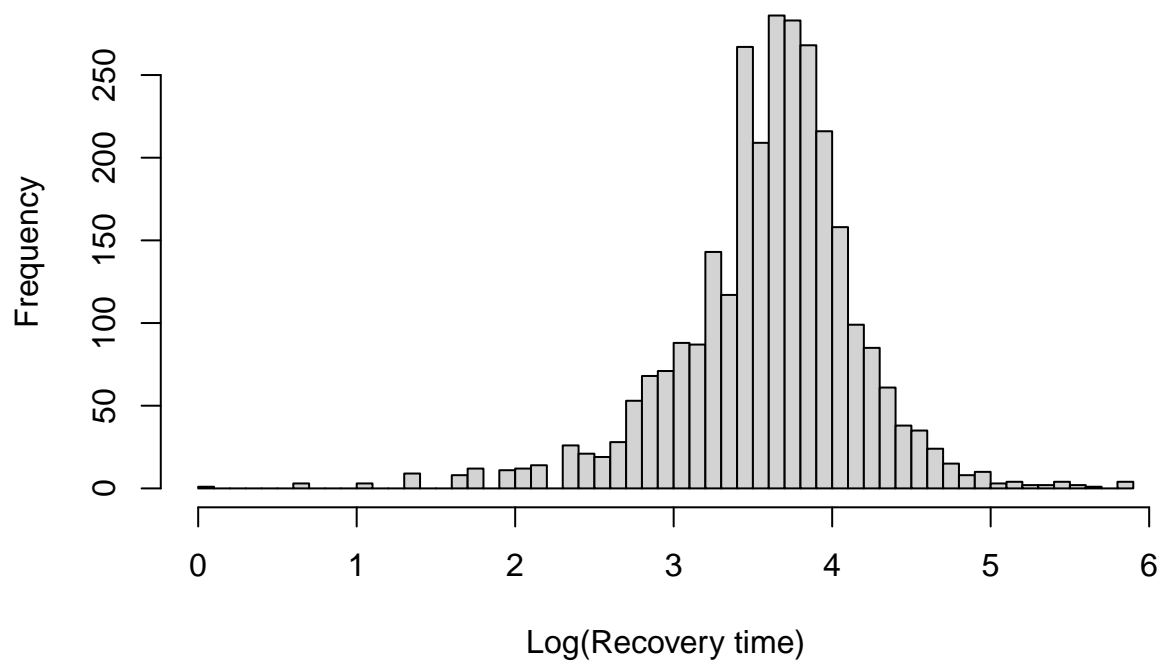
```
# check the outcome variable
hist(dat_bind$recovery_time[trainRows], breaks = 100, xlab = "Recovery time (days)", main = "")
```



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

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

# log-transformation helped with making it more normal
hist(dat_bind_primary$lrecovery_time[trainRows], breaks = 50, xlab = "Log(Recovery time)", main = "")
```



Summary of the dataset

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

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

```
## ### Data Frame Summary
## **dat_bind_primary**
## **Dimensions:** 2878 x 14
## **Duplicates:** 0
##
```

No	Variable	Stats / Values	Freqs (% of Valid)	Graph
1	gender\ [integer]	Min : 0\ Mean : 0.5\ Max : 1	0 : 1490 (51.8%)\ 1 : 1388 (48.2%)	IIIIIIIIII \ IIIIIIIIII
2	race\ [factor]	1\. 1\ 2\. 2\ 3\. 3\ 4\. 4	1863 (64.7%)\ 145 (5.0%)\ 569 (19.8%)\ 301 (10.5%)	IIIIIIIIIIII \ I \ III \ II
3	smoking\ [factor]	1\. 0\ 2\. 1\ 3\. 2	1753 (60.9%)\ 846 (29.4%)\ 279 (9.7%)	IIIIIIIIIIII \ IIIIII \ I
4	height\ [numeric]	Mean (sd) : 170.1 (5.9)\ min < med < max :\ 150.7 < 170.4 < 190.6\ IQR (CV) : 7.9 (0)	311 distinct values	\ \ \ \ \ \ \ . :\ \ \ \ \ \ \ \ : :\ \ \ \ \ \ \ \ : : :\ \ \ \ \ \ \ \ : : : :\ \ \ . : : : : .
5	weight\ [numeric]	Mean (sd) : 79.9 (7)\ min < med < max :\ 55.9 < 80.1 < 111.6\ IQR (CV) : 9.5 (0.1)	358 distinct values	\ \ \ \ \ \ \ \ \ :\ \ \ \ \ \ \ \ : : :\ \ \ \ \ \ \ \ : : : :\ \ \ \ \ \ \ \ : : : : :\ \ \ . : : : : .
6	bmi\ [numeric]	Mean (sd) : 27.7 (2.7)\ min < med < max :\ 19.7 < 27.5 < 38.1\ IQR (CV) : 3.6 (0.1)	162 distinct values	\ \ \ \ \ \ \ . :\ \ \ \ \ \ \ \ : : :\ \ \ \ \ \ \ \ : : : :\ \ \ \ \ \ \ \ : : : : :\ \ \ : : : : : .
7	hypertension\ [numeric]	Min : 0\ Mean : 0.5\ Max : 1	0 : 1499 (52.1%)\ 1 : 1379 (47.9%)	IIIIIIIIII \ IIIIIIIIII
8	diabetes\ [numeric]	Min : 0	0 : 2403 (83.5%)	IIIIIIIIIIIIIIIIIIII \ IIIIIIIIIIIIIIIIIIII

```

##      [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%)\    I I I I I I I I \
##      [integer]      Mean : 0.6\          1 : 1689 (58.7%)    I I I I I I I I I
##      Max : 1
##
## 12     severity\     Min : 0\          0 : 2589 (90.0%)\    I I I I I I I I I I I I I I \
##      [integer]      Mean : 0.1\          1 : 289 (10.0%)      I I
##      Max : 1
##
## 13     study\        1\ . A\          571 (19.8%)\      I I I \
##      [character]    2\ . B\          1741 (60.5%)\    I I I I I I I I I I \
##      3\ . C          566 (19.7%)      I I I
##
## 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_primary[trainRows, -1]) %>%
  ggplot(aes(x = factor(gender))) + geom_bar() + labs(x = "Gender", y = "Count")+
  scale_x_discrete(labels = c("Female", "Male")) +theme_classic()

race = (dat_bind_primary[trainRows, -1]) %>%
  ggplot(aes(x = race)) + geom_bar() + labs(x = "Race", y = "Count") +
  scale_x_discrete(labels = c("White", "Asian", "Black", "Hispanic"))+theme_classic()

smoking = (dat_bind_primary[trainRows, -1]) %>%
  ggplot(aes(x = smoking)) + geom_bar() + labs(x = "Smoking", y = "Count")+
  scale_x_discrete(labels = c("Never\nsmoked", "Former\nsmoker", "Current\nsmoker"))+theme_classic()

hypertension = (dat_bind_primary[trainRows, -1]) %>%
  ggplot(aes(x = factor(hypertension))) + geom_bar() + labs(x = "Hypertension",
                                                            y = "Count")+
  scale_x_discrete(labels = c("No hypertension", "Hypertension"))+theme_classic()

diabetes = (dat_bind_primary[trainRows, -1]) %>%

```

```

ggplot(aes(x = factor(diabetes))) + geom_bar() + labs(x = "Diabetes", y = "Count")+
scale_x_discrete(labels = c("No diabetes", "Diabetes"))+theme_classic()

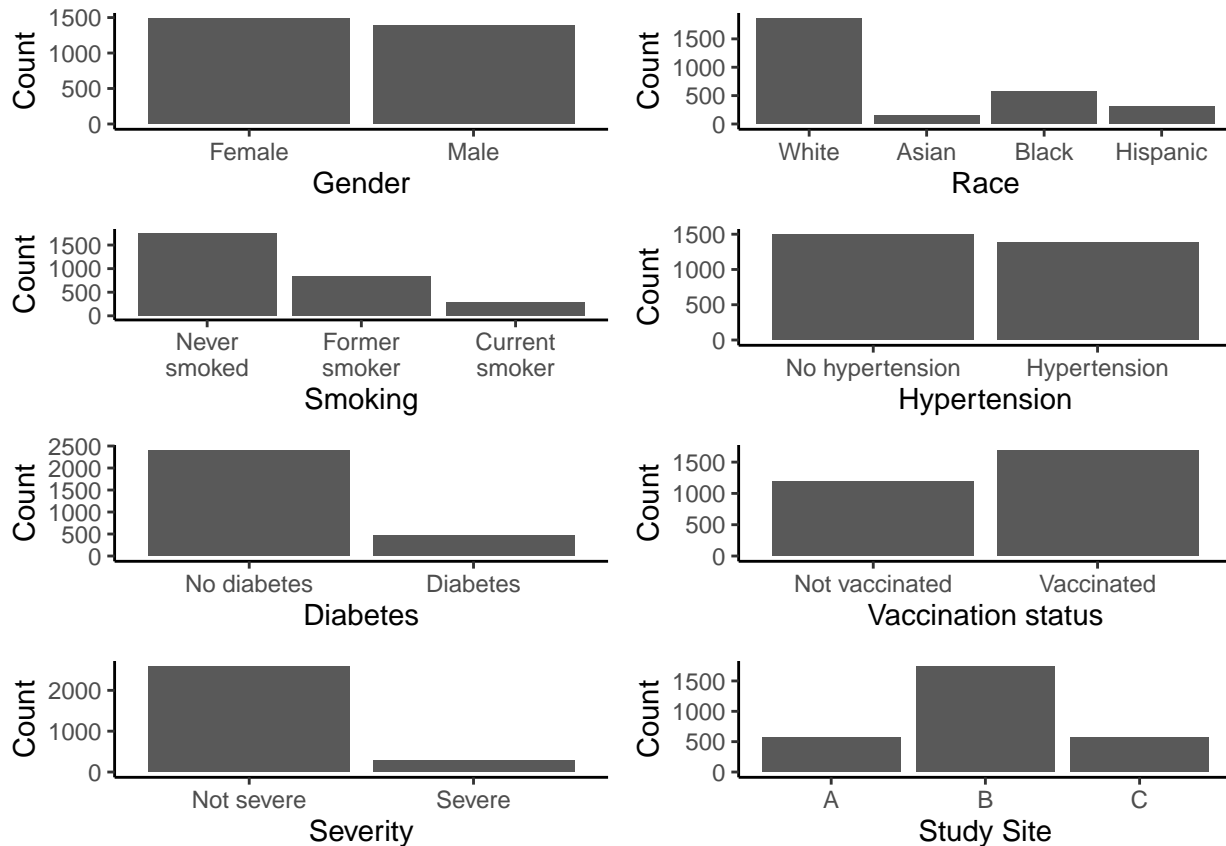
vaccine = (dat_bind_primary[trainRows, -1]) %>%
  ggplot(aes(x = factor(vaccine))) + geom_bar() + labs(x = "Vaccination status",
                                                    y = "Count")+
  scale_x_discrete(labels = c("Not vaccinated", "Vaccinated"))+theme_classic()
severity = (dat_bind_primary[trainRows, -1]) %>%
  ggplot(aes(x = factor(severity))) + geom_bar() + labs(x = "Severity", y = "Count")+
  scale_x_discrete(labels = c("Not severe", "Severe"))+theme_classic()

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

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))

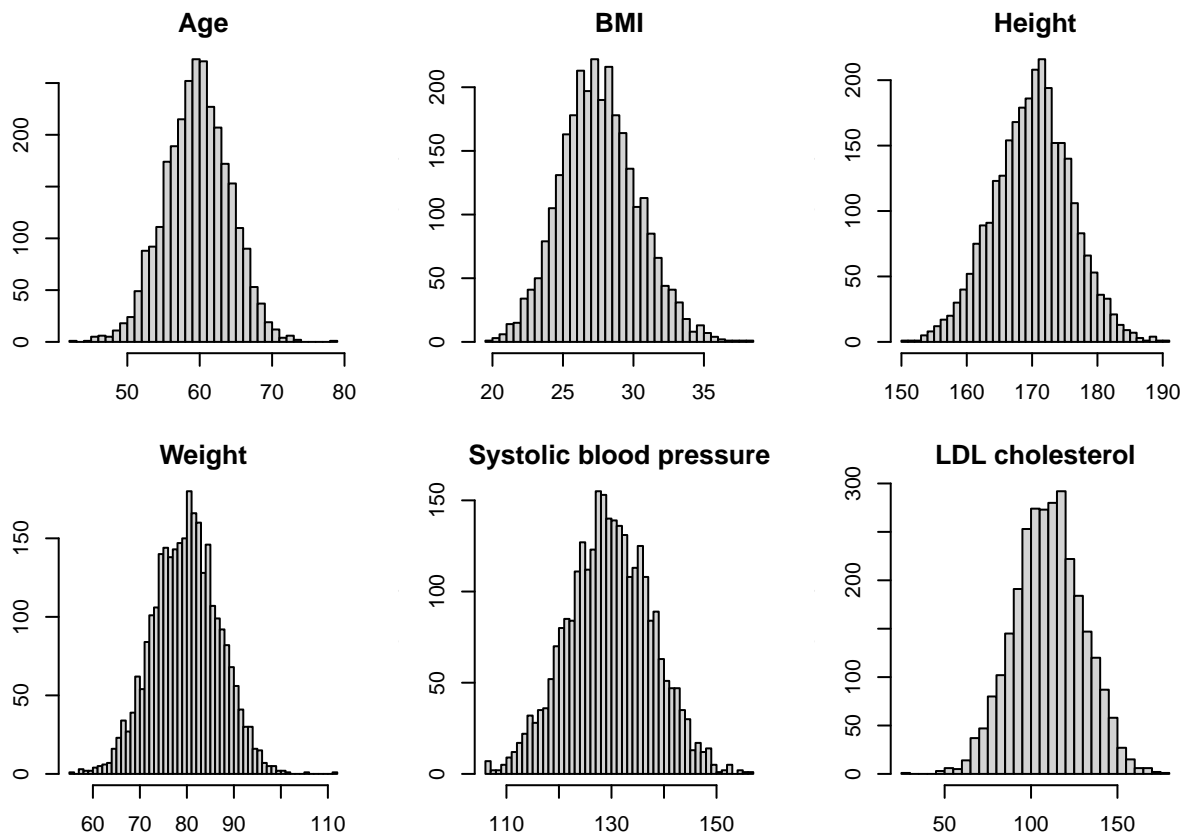
age = hist(dat_bind_primary$age[trainRows], breaks = 50, xlab = "Age (years)", main = "Age")

```

```

bmi = hist(dat_bind_primary$bmi[trainRows], breaks = 50, xlab = "BMI (kg/m^2)", main = "BMI")
height = hist(dat_bind_primary$height[trainRows], breaks = 50, xlab = "Height (cm)", main = "Height")
weight = hist(dat_bind_primary$weight[trainRows], breaks = 50, xlab = "Weight (kg)", main = "Weight")
SBP = hist(dat_bind_primary$SBP[trainRows], breaks = 50, xlab = "SBP (mm/Hg)", main = "Systolic blood pressure")
LDL = hist(dat_bind_primary$LDL[trainRows], breaks = 50, xlab = "LDL (mg/dL)", main = "LDL cholesterol")

```

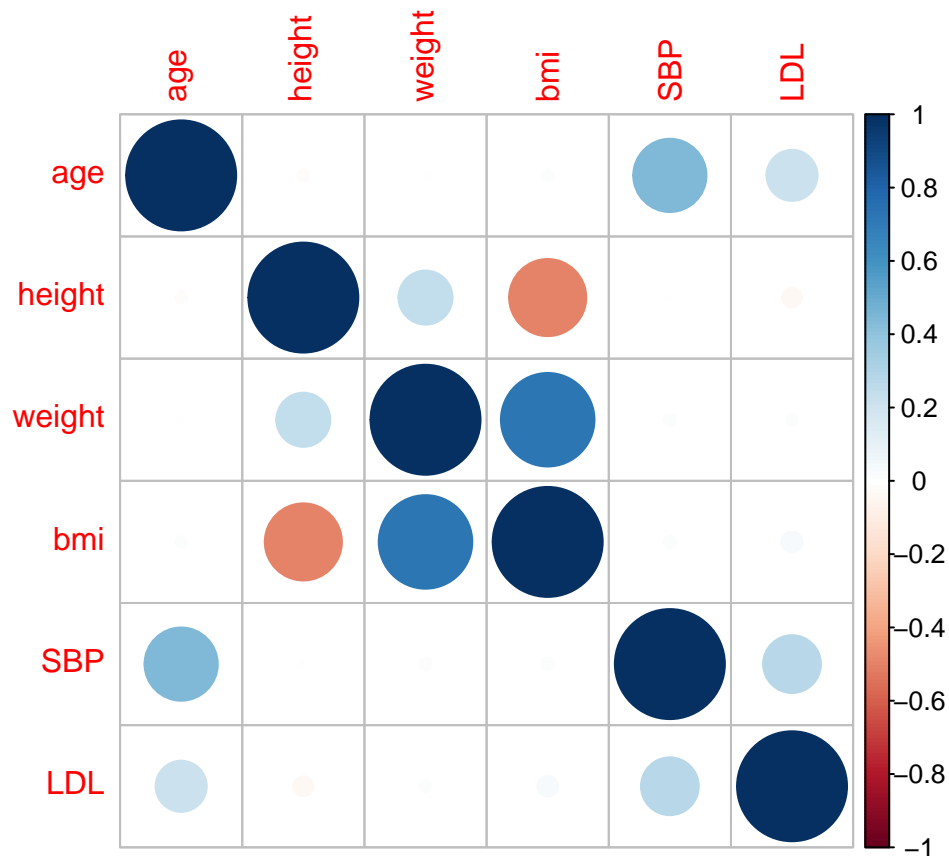


Understand the correlation between continuous predictors

```

correlation <- model.matrix(lrecovery_time ~ ., dat_bind_primary)[trainRows,-1]
corrplot(cor(dat_bind_primary[trainRows,c(1,5,6,7,10,11)]), method = "circle", type = "full")

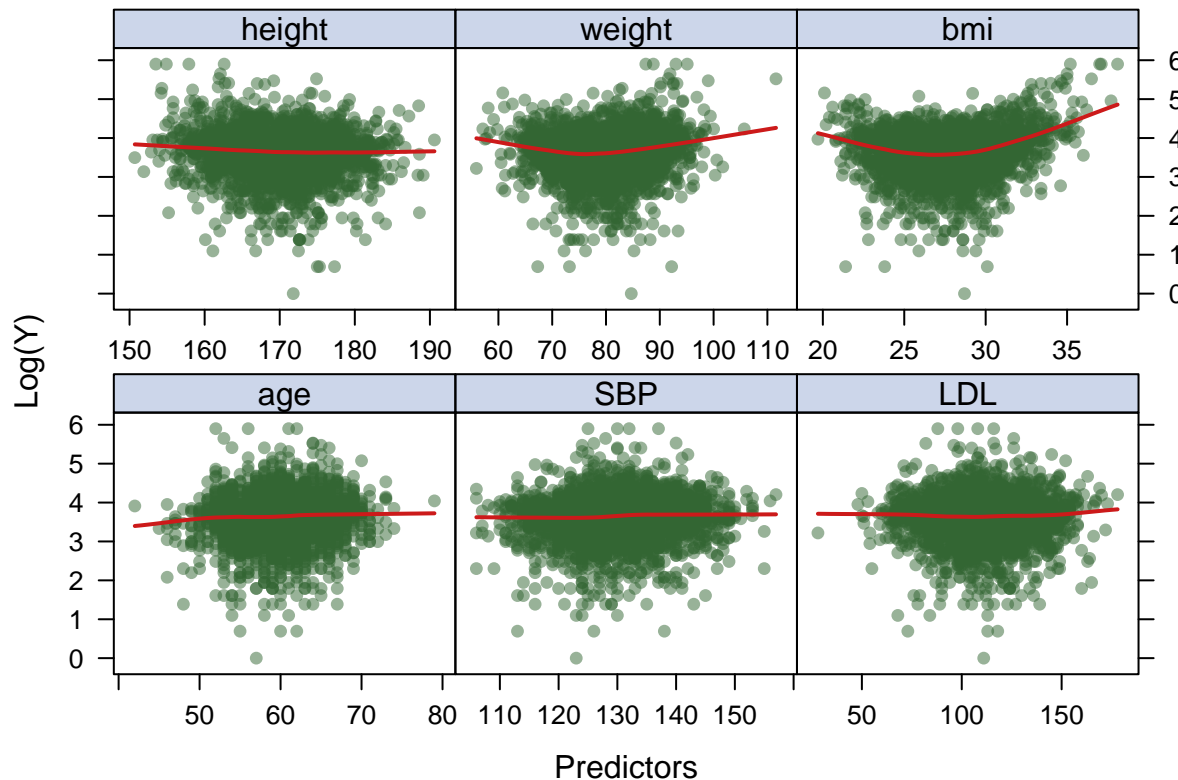
```



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_primary)[trainRows,c("age", "SBP", "LDL", "height")],
            y = dat_bind_primary$lrecovery_time[trainRows],
            plot = "scatter",
            span = .5,
            labels = c("Predictors", "Log(Y)"),
            type = c("p", "smooth"),
            layout = c(3,2))
```

Understand the relationship between categorical predictors and continuous outcome

```
gender3 = (dat_bind_primary[trainRows, -1]) %>%
  ggplot(aes(x = as.factor(gender), y = lrecovery_time)) + geom_boxplot() + labs(x = "Gender", y = "Log Recovery Time") +
  scale_x_discrete(labels = c("White", "Asian", "Black", "Hispanic")) + theme_classic()

race3 = (dat_bind_primary[trainRows, -1]) %>%
  ggplot(aes(x = as.factor(race), y = lrecovery_time)) + geom_boxplot() + labs(x = "Race", y = "Log Recovery Time") +
  scale_x_discrete(labels = c("White", "Asian", "Black", "Hispanic")) + theme_classic()

smoking3 = (dat_bind_primary[trainRows, -1]) %>%
  ggplot(aes(x = as.factor(smoking), y = lrecovery_time)) + geom_boxplot() + labs(x = "Smoking Status", y = "Log Recovery Time") +
  scale_x_discrete(labels = c("Never\nsmoked", "Former\nsmoker", "Current\nsmoker")) + theme_classic()

hypertension3 = (dat_bind_primary[trainRows, -1]) %>%
  ggplot(aes(x = as.factor(hypertension), y = lrecovery_time)) + geom_boxplot() + labs(x = "Hypertension Status", y = "Log Recovery Time") +
  scale_x_discrete(labels = c("No", "Yes")) + theme_classic()

diabetes3 = (dat_bind_primary[trainRows, -1]) %>%
  ggplot(aes(x = as.factor(diabetes), y = lrecovery_time)) + geom_boxplot() + labs(x = "Diabetes Status", y = "Log Recovery Time") +
  scale_x_discrete(labels = c("No", "Yes")) + theme_classic()

vaccine3 = (dat_bind_primary[trainRows, -1]) %>%
  ggplot(aes(x = as.factor(vaccine), y = lrecovery_time)) + geom_boxplot() + labs(x = "Vaccination Status", y = "Log Recovery Time") +
  scale_x_discrete(labels = c("No", "Yes")) + theme_classic()

severity3 = (dat_bind_primary[trainRows, -1]) %>%
  ggplot(aes(x = as.factor(severity), y = lrecovery_time)) + geom_boxplot() + labs(x = "Severity", y = "Log Recovery Time") +
  scale_x_discrete(labels = c("Mild", "Moderate", "Severe")) + theme_classic()
```

```

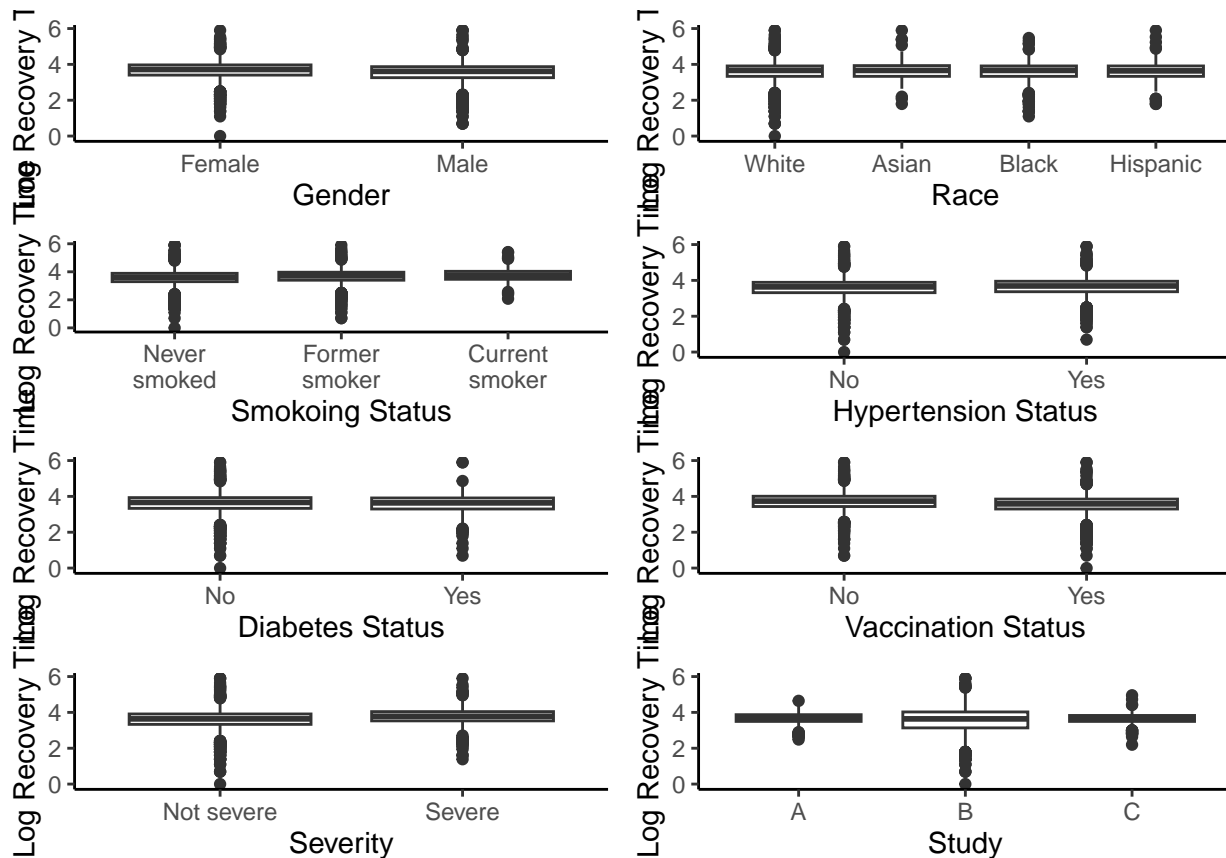
scale_x_discrete(labels = c("Not severe", "Severe"))+ theme_classic()

study3 = (dat_bind_primary[trainRows, -1]) %>%
  ggplot(aes(x = as.factor(study), y = lrecovery_time)) + geom_boxplot() + labs(x = "Study", y = "Log R

cat_combined_plot3 = ggarrange(gender3, race3, smoking3, hypertension3,
  diabetes3, vaccine3, severity3, study3,
  ncol = 2, nrow = 4)

cat_combined_plot3

```



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.

We 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, we will remove the **study** variable.

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

```

primary = dat_bind_primary %>%
  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
primary = primary %>%
  select(-study, -race, -smoking)

# partition again based on the new outcome variable
x <- model.matrix(lrecovery_time ~ ., primary)[trainRows,-1]

y <- primary$lrecovery_time[trainRows]

x2 <- model.matrix(lrecovery_time ~ ., primary)[-trainRows,-1]

y2 <- primary$lrecovery_time[-trainRows]

ctrl1 <- trainControl(method = "cv")

```

Primary analysis: continuous time to recovery

Model 1: Linear model

```

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
## -3.4481  -0.2424   0.0685   0.3126   2.0007
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.601182   0.010003  360.010 < 2e-16 ***
## age           0.036999   0.011244   3.291  0.00101 **
## gender        -0.061087   0.010035  -6.088 1.30e-09 ***
## height         1.002011   0.103072   9.722 < 2e-16 ***
## weight        -1.279007   0.129968  -9.841 < 2e-16 ***
## bmi            1.509986   0.144853  10.424 < 2e-16 ***
## hypertension  0.036686   0.016726   2.193  0.02836 *
## diabetes      -0.008605   0.010030  -0.858  0.39098

```

```
## SBP          -0.001969    0.017667   -0.111   0.91125
## LDL          -0.009753    0.010504   -0.929   0.35320
## vaccine      -0.089295    0.010018   -8.914   < 2e-16 ***
## severity      0.049651    0.010029    4.951   7.83e-07 ***
## race_2        0.009397    0.010144    0.926   0.35432
## race_3       -0.006895    0.010253   -0.672   0.50134
## race_4        0.001614    0.010243    0.158   0.87481
## smoking_1     0.046423    0.010264    4.523   6.35e-06 ***
## smoking_2     0.053878    0.010270    5.246   1.67e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.287973)
##
##      Null deviance: 937.87  on 2877  degrees of freedom
## Residual deviance: 823.89  on 2861  degrees of freedom
## AIC: 4603.6
##
## Number of Fisher Scoring iterations: 2

coef(lm.fit$finalModel) %>% round(2) %>%
  as.matrix() %>% as.data.frame() %>% View()
```

Model 2: Ridge

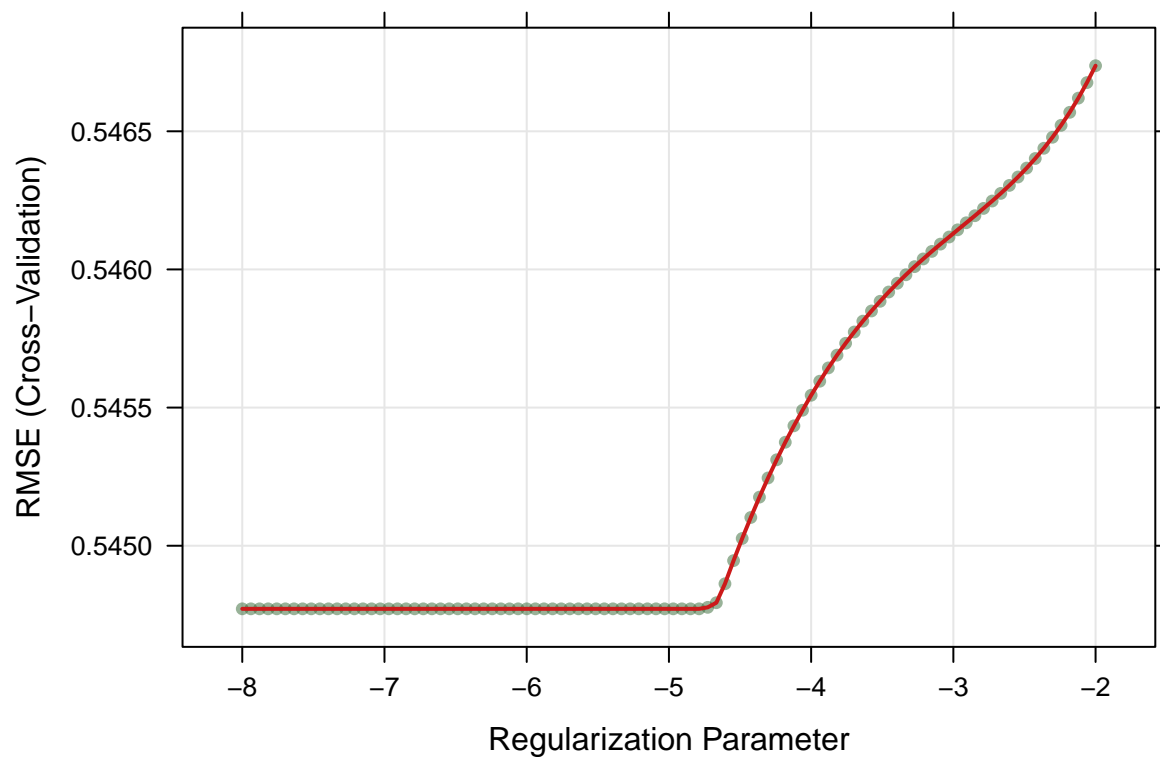
```
set.seed(2460)

ridge.fit <- train(x, y,
  method = "glmnet",
  tuneGrid = expand.grid(alpha = 0,
    lambda = exp(seq(-2, -8, length = 100))),
  preProc = c("center", "scale"),
  trControl = ctrl1)

ridge.fit$bestTune

##      alpha      lambda
## 54      0 0.008330109

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



```
coef(ridge.fit$finalModel, s = ridge.fit$bestTune$lambda) %>%
  round(2) %>%
  as.matrix() %>% as.data.frame() %>% View()
```

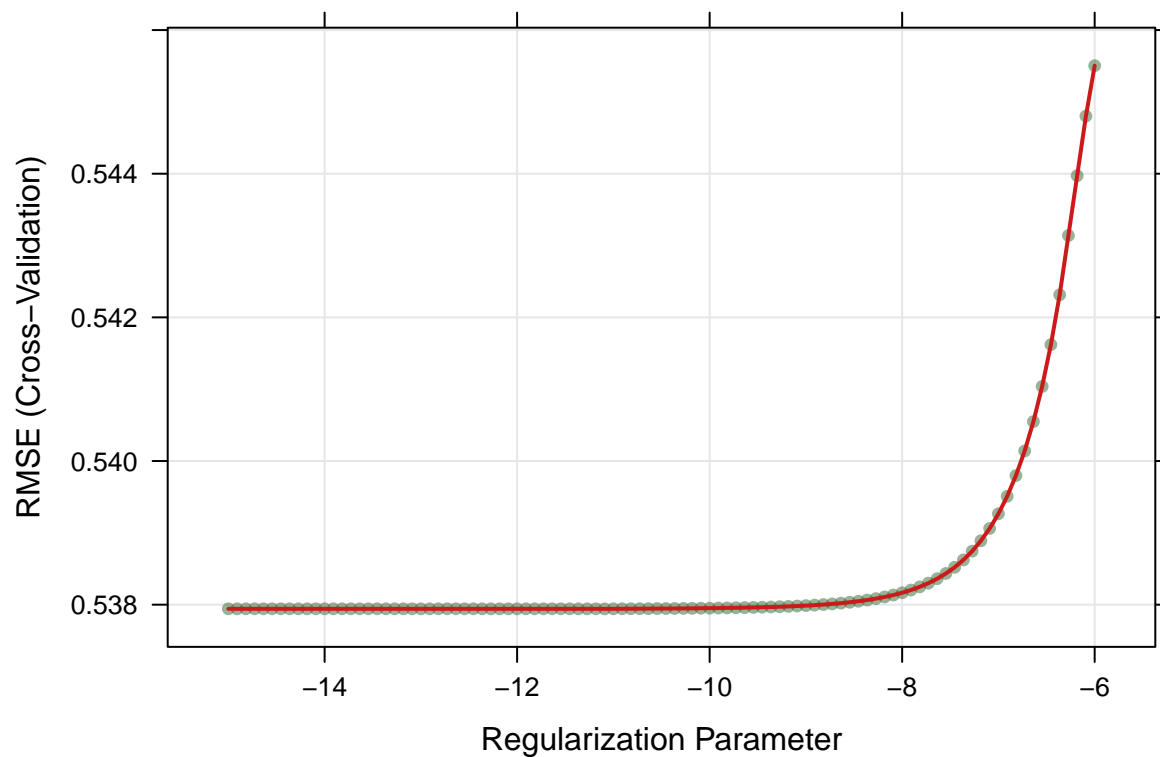
Model 3: Lasso

```
set.seed(2460)

lasso.fit <- train(x, y,
  method = "glmnet",
  tuneGrid = expand.grid(alpha = 1,
    lambda = exp(seq(-15, -6, length = 100))),
  preProcess = c("center", "scale"),
  trControl = ctrl1)

lasso.fit$bestTune

##      alpha      lambda
## 44      1 1.525033e-05
plot(lasso.fit, xTrans = log)
```



```
coef(lasso.fit$finalModel, s = lasso.fit$bestTune$lambda) %>%
  round(2) %>%
  as.matrix() %>% as.data.frame() %>% View()
```

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(-12, -2,
      length = 100))),
  preProcess = c("center", "scale"),
  trControl = ctrl1)

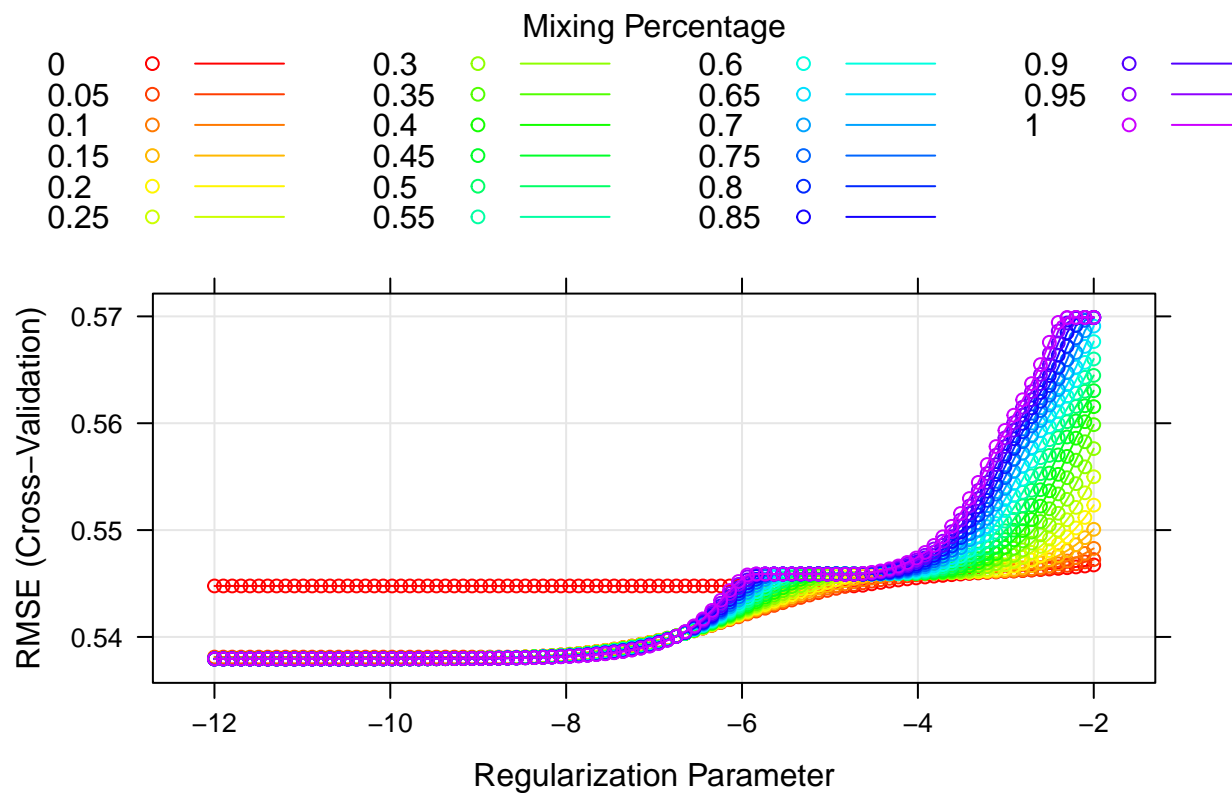
enet.fit$bestTune

##      alpha      lambda
## 1808    0.9 1.246072e-05

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) %>%
  round(2) %>%
  as.matrix() %>% as.data.frame() %>% View()
```

Model 5: Partial least square

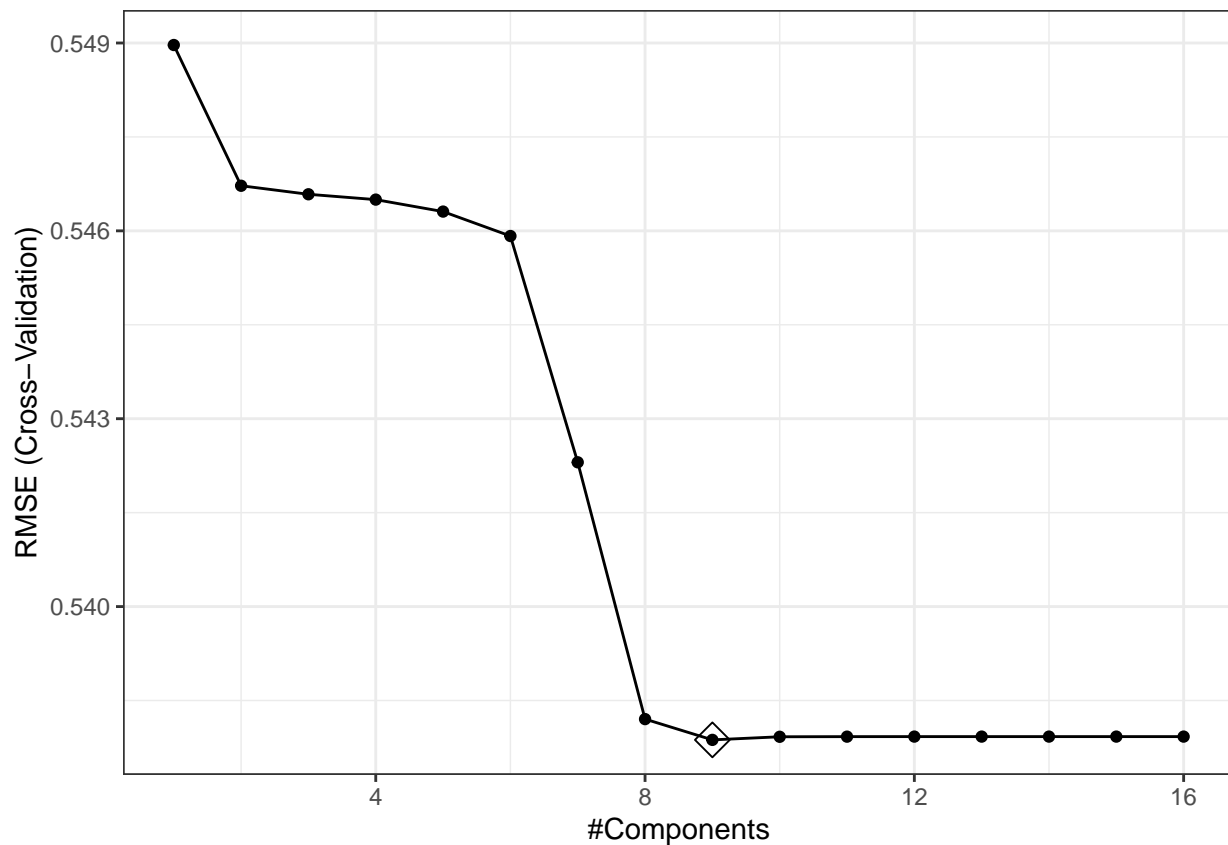
```
set.seed(2460)

pls.fit <- train(x, y,
  method = "pls",
  tuneGrid = data.frame(ncomp = 1:16),
  trControl = ctrl1,
  preProcess = c("center", "scale"))

pls.fit$bestTune

##   ncomp
## 9      9

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



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

```
## Data:      X dimension: 2878 16
## Y dimension: 2878 1
## Fit method: oscorespls
## Number of components considered: 9
## TRAINING: % variance explained
##           1 comps 2 comps 3 comps 4 comps 5 comps 6 comps 7 comps
## X           9.942 19.262 30.232 35.946 39.500 45.093 49.15
## .outcome     7.980  8.969  9.033  9.078  9.185  9.273 10.31
##           8 comps 9 comps
## X           51.69 57.20
## .outcome    12.06 12.15
```

```
coef(pls.fit$finalModel) %>% round(2) %>%
  as.matrix() %>% as.data.frame() %>% View()
```

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(bmi) + s(height) + s(weight)  
##  
## Estimated degrees of freedom:  
## 1.00 1.00 1.89 6.33 1.00 1.00 total = 23.22  
##  
## GCV score: 0.2698475
```

```
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(bmi) + s(height) + s(weight)  
##  
## Parametric coefficients:  
##              Estimate Std. Error t value Pr(>|t|)  
## (Intercept)  3.6011822  0.0096439 373.414 < 2e-16 ***  
## gender       -0.0596870  0.0096791  -6.167 7.96e-10 ***  
## hypertension  0.0452988  0.0161533   2.804 0.00508 **  
## diabetes     -0.0040637  0.0096880  -0.419 0.67491  
## vaccine      -0.0894786  0.0096702  -9.253 < 2e-16 ***  
## severity      0.0452155  0.0096825   4.670 3.15e-06 ***  
## race_2        0.0028659  0.0098071   0.292 0.77014  
## race_3       -0.0052124  0.0098904  -0.527 0.59822  
## race_4        0.0002138  0.0098871   0.022 0.98275  
## smoking_1     0.0496972  0.0099135   5.013 5.68e-07 ***  
## smoking_2     0.0572652  0.0099145   5.776 8.48e-09 ***  
## ---  
## 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 10.180 0.00143 **  
## s(SBP)        1.000  1.000  0.118 0.73090  
## s(LDL)        1.890  2.414  1.318 0.24710  
## s(bmi)        6.332  7.422 44.640 < 2e-16 ***  
## s(height)     1.000  1.000  1.631 0.20171
```

```
## s(weight) 1.000 1.000 1.752 0.18568
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) = 0.179 Deviance explained = 18.5%
## GCV = 0.26985 Scale est. = 0.26767 n = 2878
```

Plot continuous predictors in GAM

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

# 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)

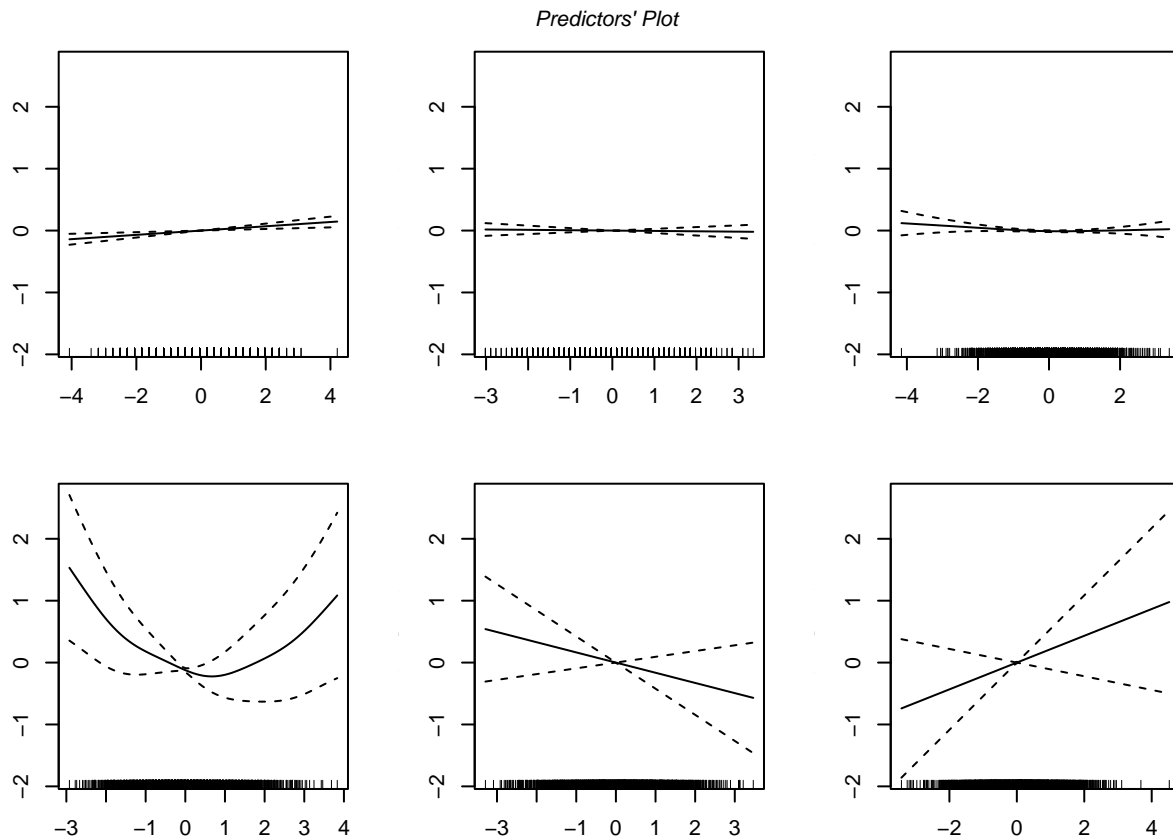
##      [,1] [,2] [,3]
## [1,] "age" "SBP" "LDL"
## [2,] "bmi" "height" "weight"

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

```

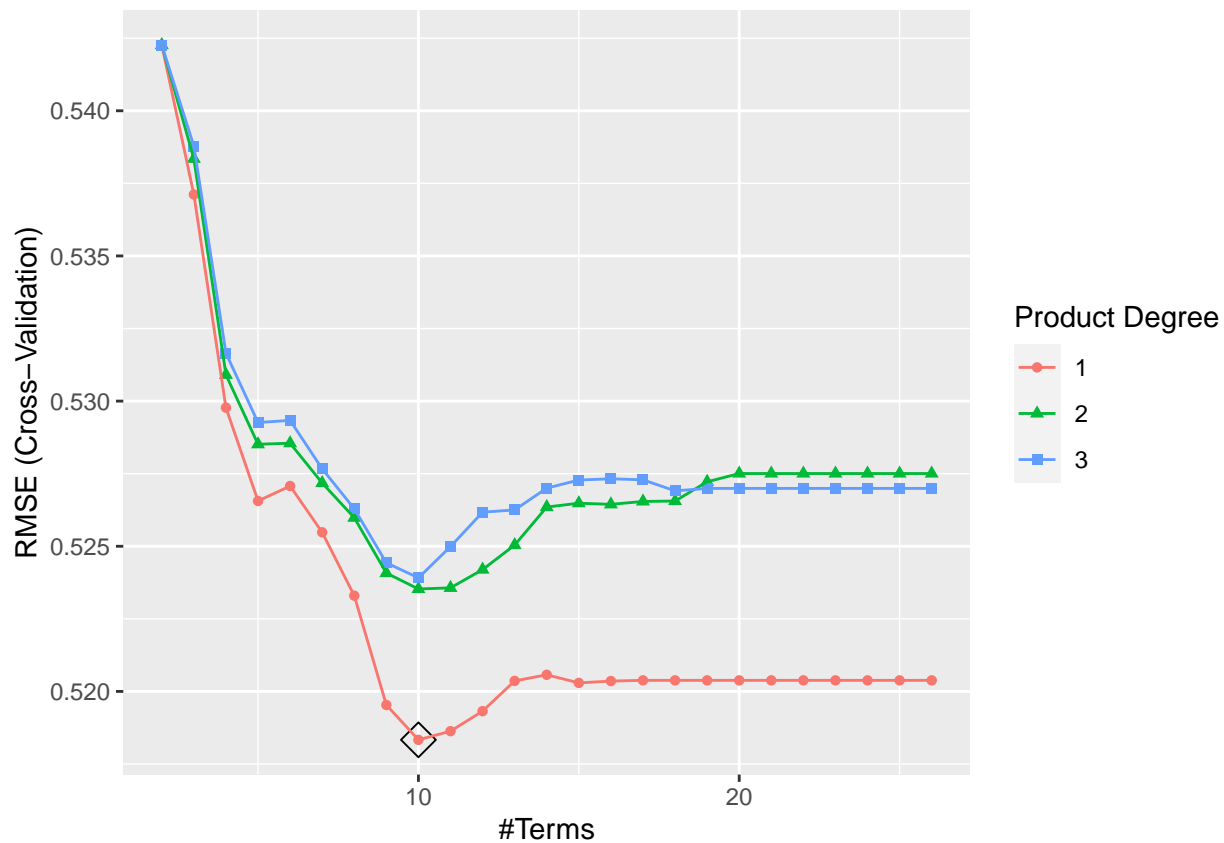
mars_grid <- expand.grid(degree = 1:3,
                        nprune = 2:26)

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
##  9      10      1
```

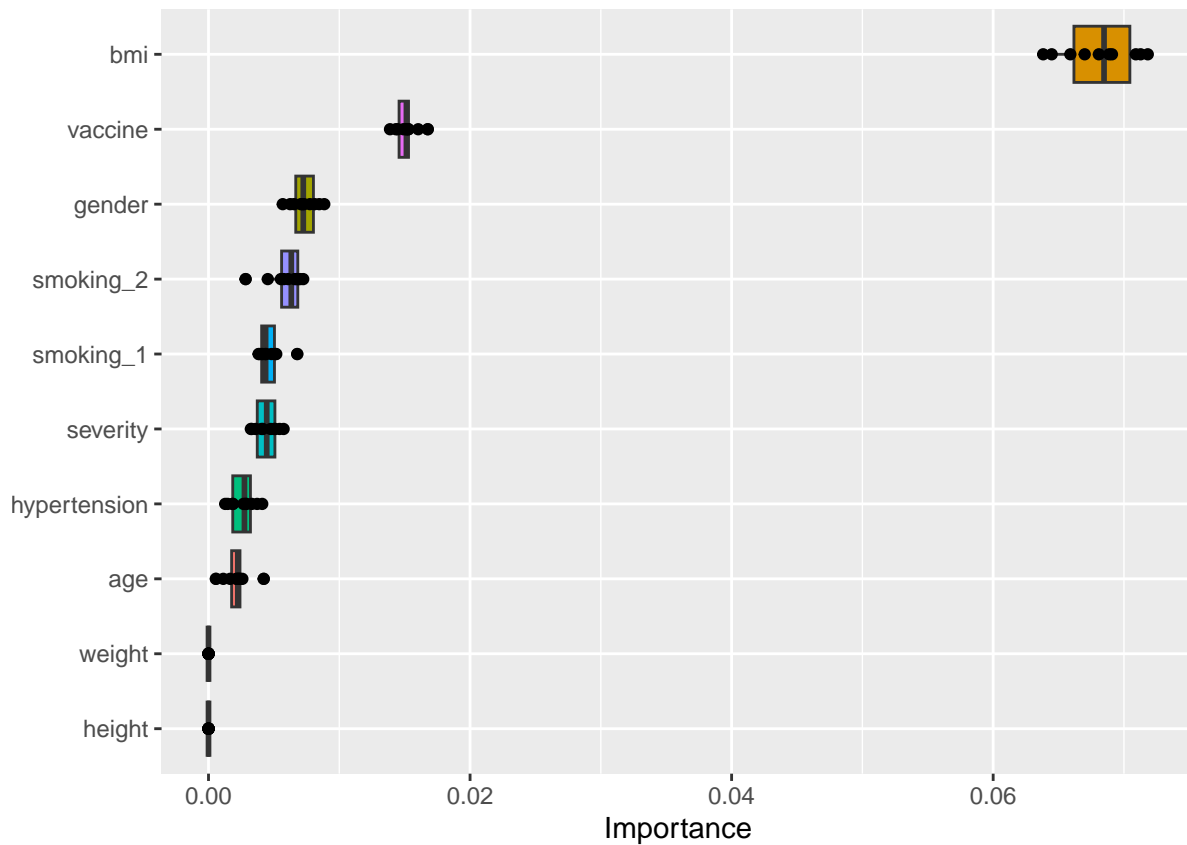
```
summary(mars.fit)
```

```
## Call: earth(x=matrix[2878,16], y=c(3.584,3.761,3...), keepxy=TRUE, degree=1,
##           nprune=10)
##
##               coefficients
## (Intercept)      2.32744078
## gender          -0.06033598
## hypertension     0.03804629
## vaccine          -0.08929877
## severity         0.04637756
## smoking_1        0.04943934
## smoking_2        0.05566747
## h(age- -2.7261)   0.03261911
## h(bmi- -1.4908)   0.49599535
## h(0.638255-bmi)  0.53847499
##
## Selected 10 of 19 terms, and 8 of 16 predictors (nprune=10)
## Termination condition: RSq changed by less than 0.001 at 19 terms
## Importance: bmi, vaccine, gender, hypertension, smoking_2, smoking_1, ...
## Number of terms at each degree of interaction: 1 9 (additive model)
## GCV 0.2705204    RSS 768.3118    GRSq 0.170441    RSq 0.1807888
```

```
coef(mars.fit$finalModel) %>% round(2) %>%
  as.matrix() %>% as.data.frame() %>% View()
```

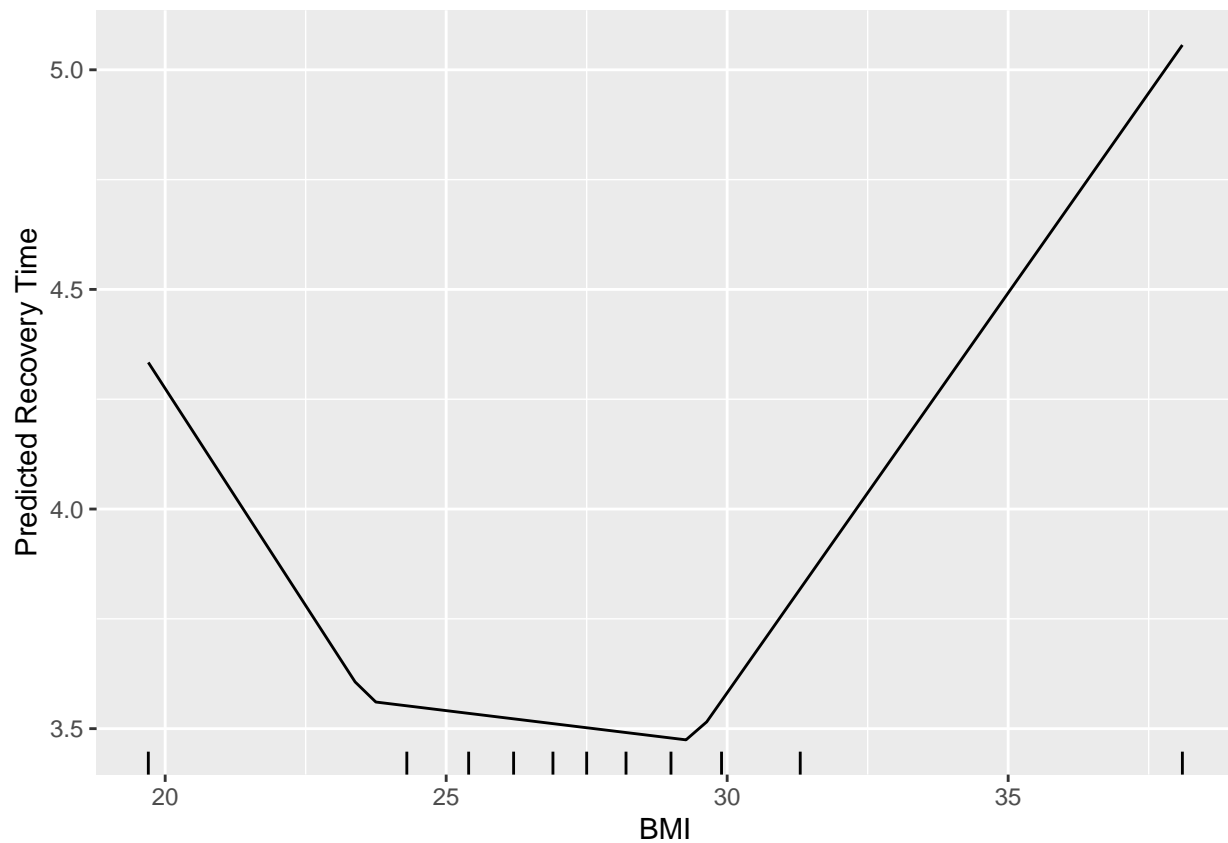
Variable importance

```
vip(mars.fit,
    method = "permute",
    train = primary[trainRows,],
    target = "lrecovery_time",
    metric = "RMSE",
    nsim = 10,
    pred_wrapper = predict,
    geom = "boxplot",
    all_permutations = TRUE,
    mapping = aes_string(fill = "Variable"))
```



Partial dependence

```
pdp1 <- mars.fit %>%
  partial(pred.var = c("bmi")) %>%
  autoplot(train = primary[trainRows,], rug = TRUE,
           xlab = "BMI", ylab = "Predicted Recovery Time")
pdp1
```



Model 8: Regression tree

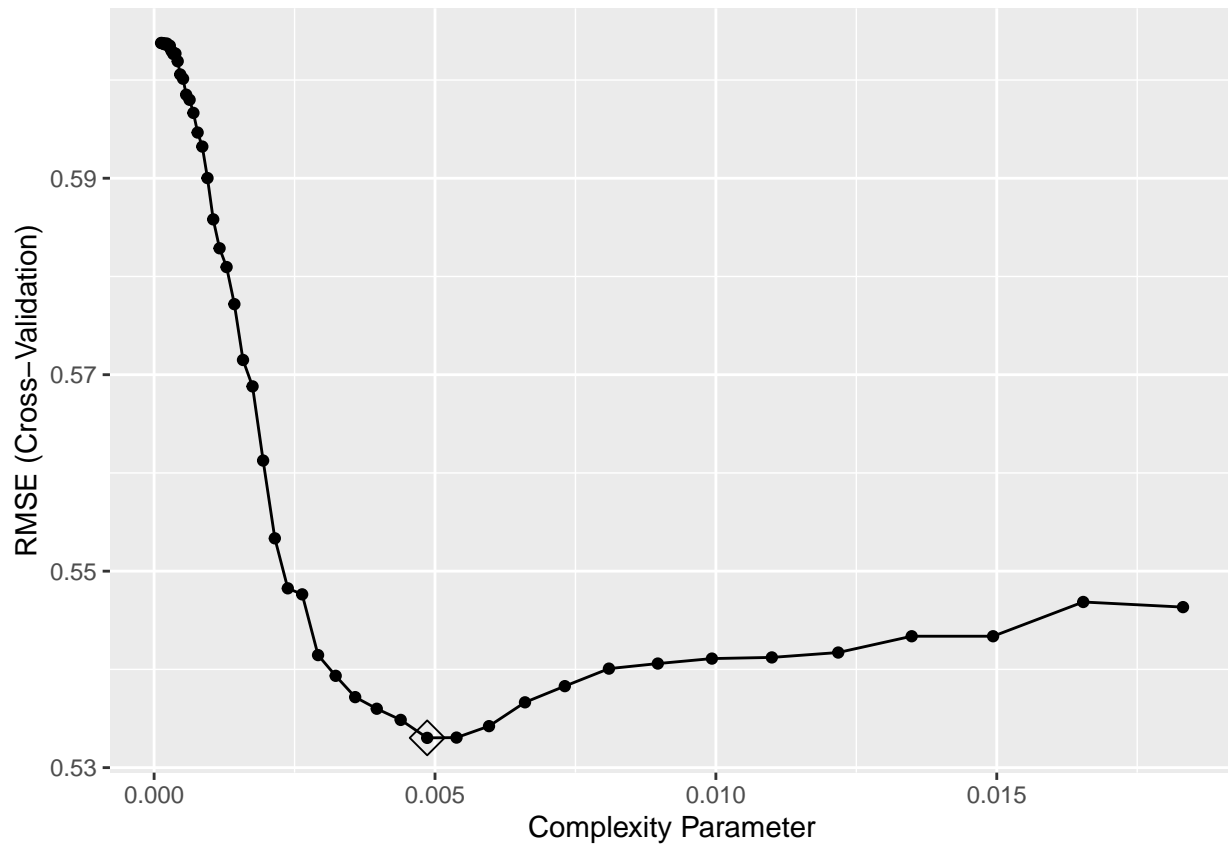
```
set.seed(2460)

rpart.fit <- train(lrecovery_time ~ . ,
  primary[trainRows,],
  method = "rpart",
  tuneGrid = data.frame(cp = exp(seq(-9,-4, length = 50))),
  trControl = ctrl1,
  preProcess = c("center", "scale"))

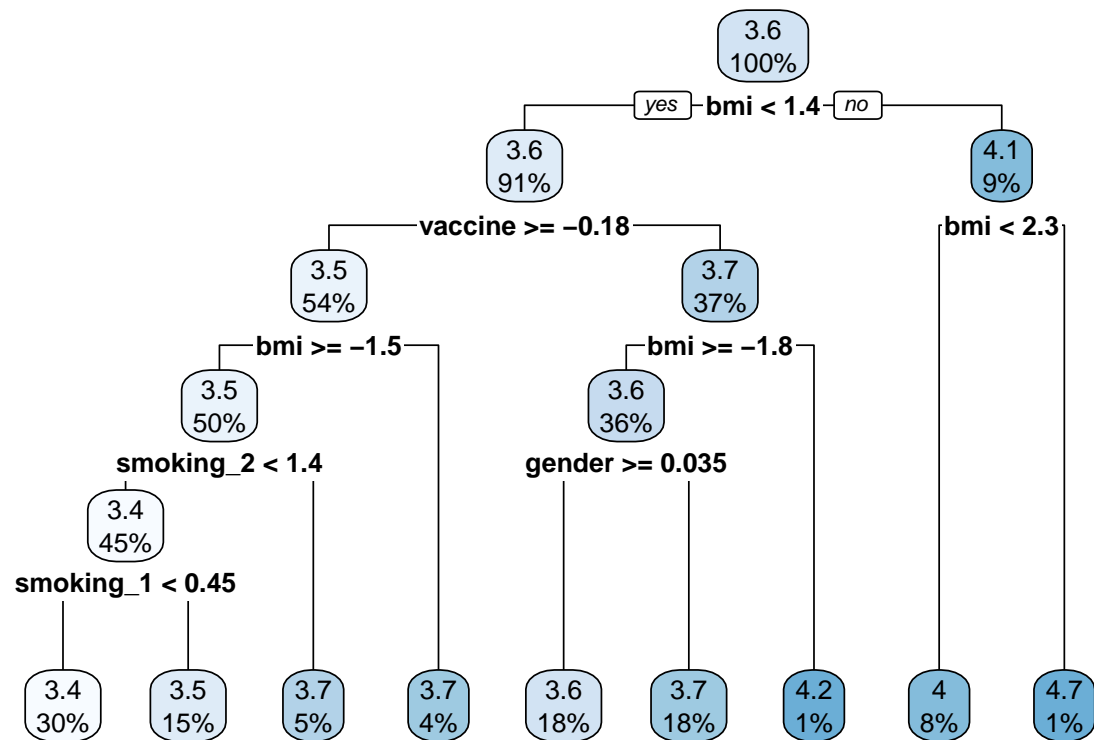
rpart.fit$bestTune

##           cp
## 37 0.004860905

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



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



Model 9: Random forest

```
rf.grid <- expand.grid(mtry = 1:16, #16 predictors
                      splitrule = "variance",
                      min.node.size = 1:6)
```

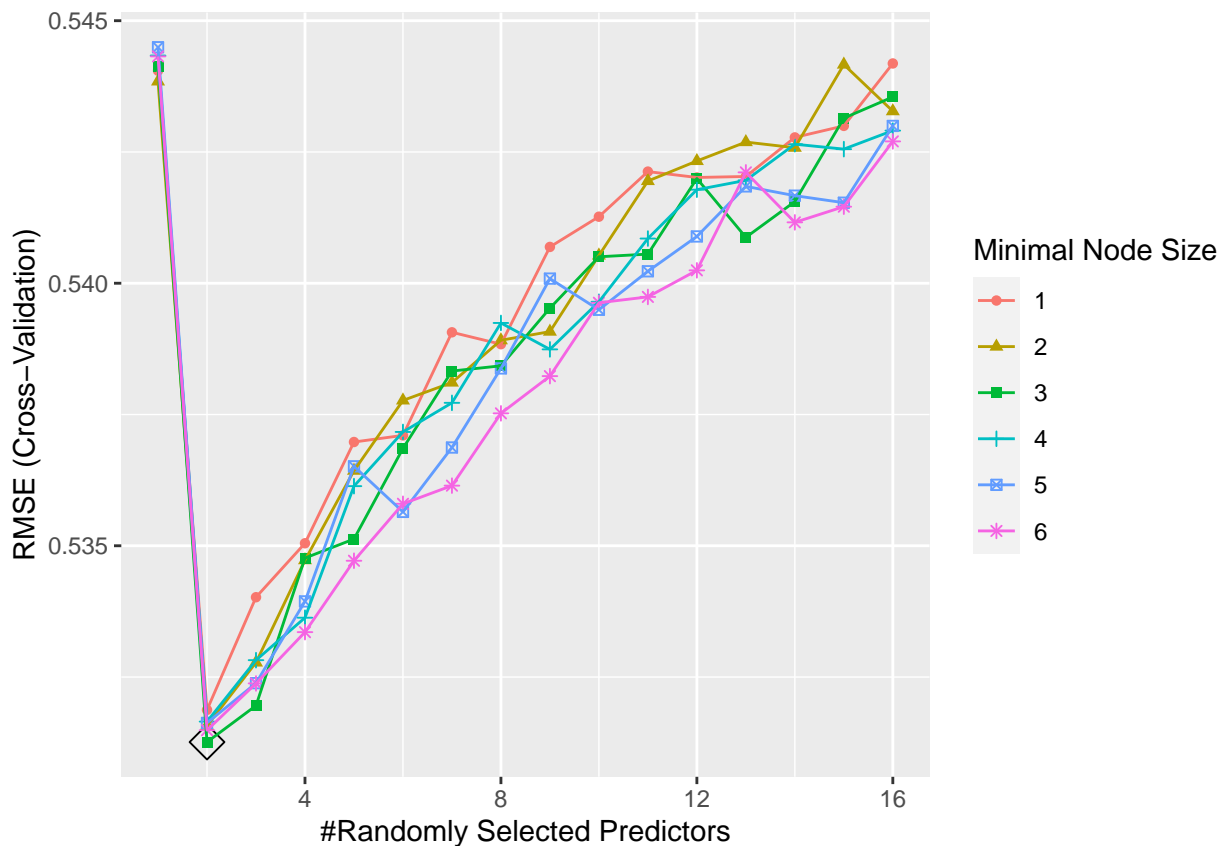
```
set.seed(2460)
```

```
rf.fit <- train(lrecovery_time ~ . ,
               primary[trainRows,],
               method = "ranger",
               tuneGrid = rf.grid,
               trControl = ctrl1,
               preProcess = c("center", "scale"))
```

```
rf.fit$bestTune
```

```
## mtry splitrule min.node.size
## 9      2 variance              3
```

```
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))

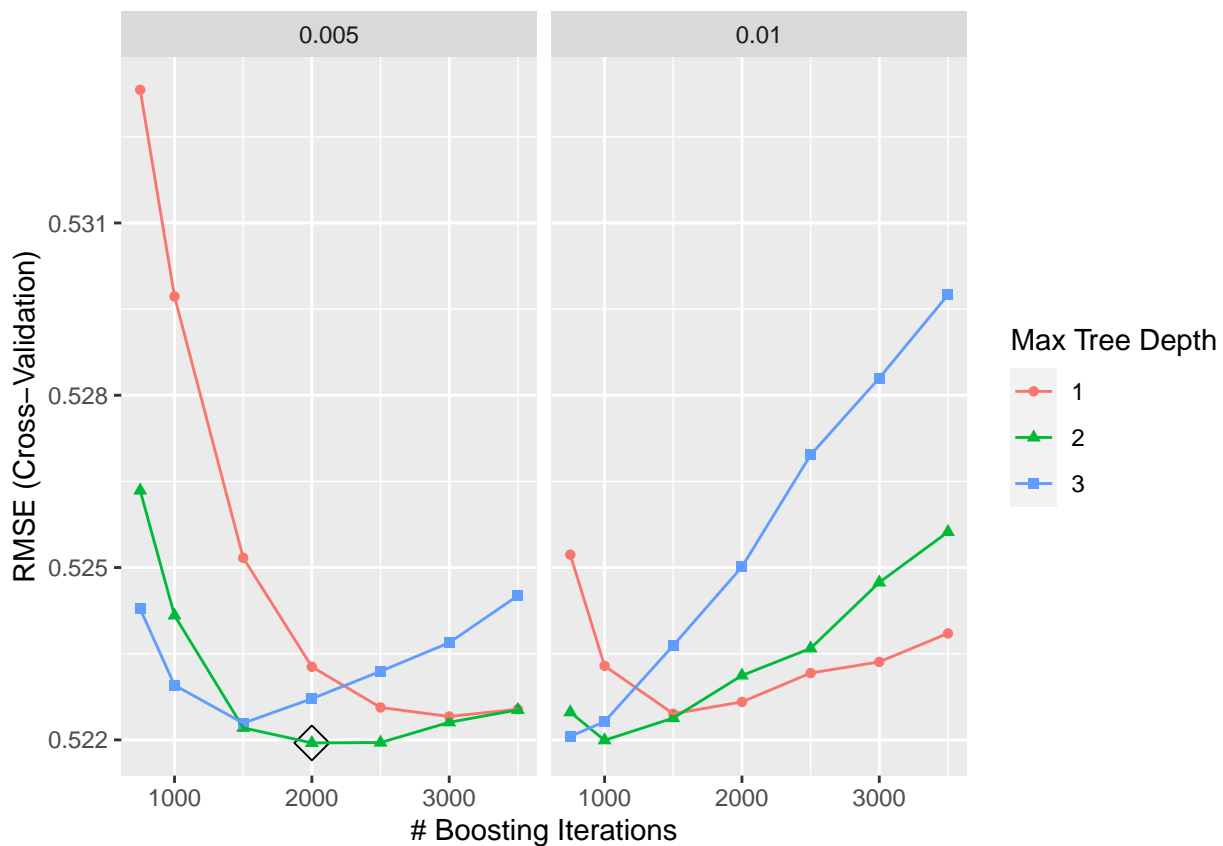
set.seed(2460)

gbm.fit <- train(lrecovery_time ~ . ,
  primary[trainRows,],
  method = "gbm",
  tuneGrid = gbm.grid,
  trControl = ctrl1,
  preProcess = c("center", "scale"),
  verbose = F)

gbm.fit$bestTune

##      n.trees interaction.depth shrinkage n.minobsinnode
## 11      2000                2      0.005              1
ggplot(gbm.fit, highlight = T)

```



Model comparison

```

res_pri <- resamples(list(lm = lm.fit,
  ridge = ridge.fit,
  lasso = lasso.fit,
  enet = enet.fit,
  pls = pls.fit,
  gam = gam.fit,

```

```

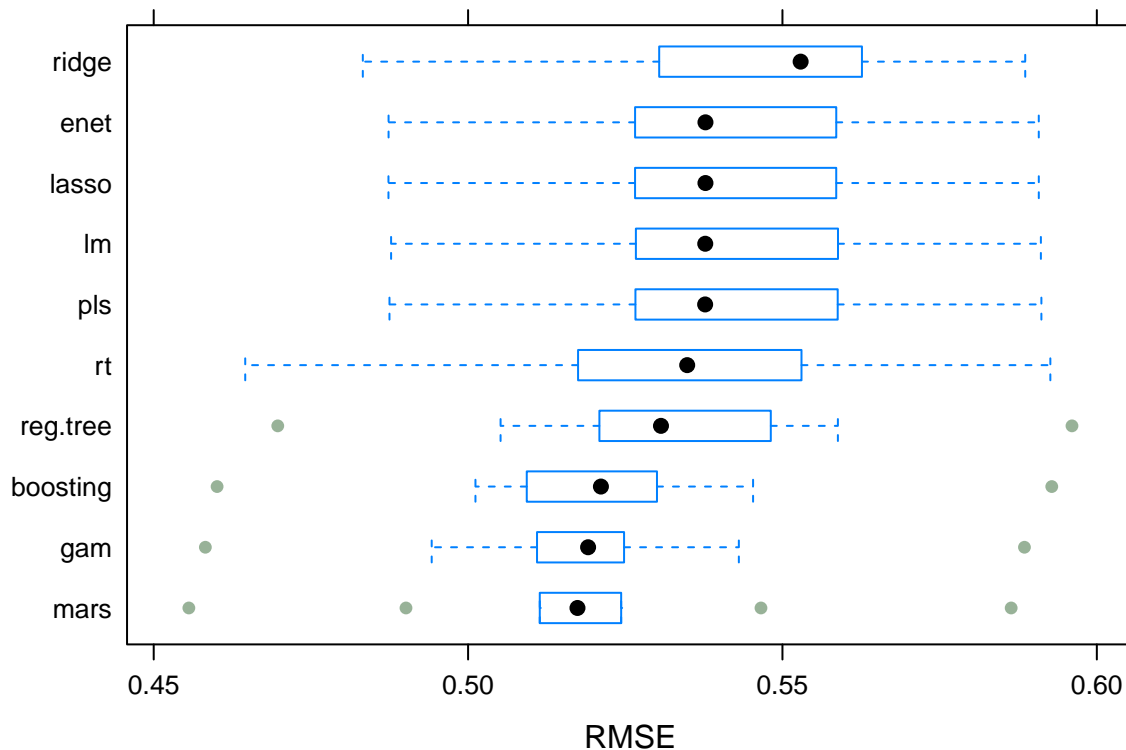
mars = mars.fit,
reg.tree = rpart.fit,
rt = rf.fit,
boosting = gbm.fit))

summary(res_pri)

##
## Call:
## summary.resamples(object = res_pri)
##
## 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          0.3715813 0.3801875 0.3891606 0.3905957 0.3974587 0.4168473    0
## ridge       0.3668470 0.3845314 0.3995422 0.3946062 0.4045918 0.4198322    0
## lasso       0.3711375 0.3801672 0.3895606 0.3905287 0.3973617 0.4166310    0
## enet        0.3711532 0.3801696 0.3895537 0.3905287 0.3973633 0.4166282    0
## pls         0.3714900 0.3799257 0.3891460 0.3905299 0.3973980 0.4169598    0
## gam         0.3563440 0.3714600 0.3789665 0.3798054 0.3835986 0.4138882    0
## mars        0.3544547 0.3736366 0.3756004 0.3794879 0.3851627 0.4114350    0
## reg.tree    0.3683618 0.3840758 0.3895214 0.3920198 0.3996331 0.4230047    0
## rt          0.3574498 0.3787421 0.3839666 0.3858224 0.3931670 0.4154785    0
## boosting    0.3547992 0.3734574 0.3788330 0.3804563 0.3846729 0.4107268    0
##
## RMSE
##           Min.    1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lm          0.4877568 0.5282710 0.5377199 0.5379234 0.5550979 0.5911010    0
## ridge       0.4832580 0.5336716 0.5528909 0.5447716 0.5625358 0.5886115    0
## lasso       0.4873377 0.5284287 0.5377526 0.5379422 0.5549924 0.5907703    0
## enet        0.4873523 0.5284252 0.5377535 0.5379415 0.5549939 0.5907646    0
## pls         0.4874998 0.5282229 0.5377011 0.5378712 0.5551181 0.5911807    0
## gam         0.4582162 0.5116572 0.5190892 0.5195132 0.5244718 0.5884876    0
## mars        0.4555906 0.5116768 0.5174001 0.5183330 0.5236598 0.5863775    0
## reg.tree    0.4697322 0.5212018 0.5306794 0.5330292 0.5481066 0.5960468    0
## rt          0.4645553 0.5195229 0.5348479 0.5312617 0.5502635 0.5926060    0
## boosting    0.4600815 0.5101843 0.5211270 0.5219467 0.5289549 0.5928327    0
##
## Rsquared
##           Min.    1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lm          0.07232202 0.08974674 0.10383362 0.11176919 0.1148894 0.1869123    0
## ridge       0.04866936 0.07546233 0.09240669 0.08803606 0.1037300 0.1154734    0
## lasso       0.07186500 0.09029708 0.10464976 0.11155582 0.1154413 0.1835720    0
## enet        0.07185371 0.09027505 0.10465331 0.11156003 0.1154374 0.1836312    0
## pls         0.07234132 0.09032163 0.10396846 0.11191446 0.1156243 0.1867030    0
## gam         0.10946228 0.11793949 0.17278765 0.17183125 0.2129779 0.2538031    0
## mars        0.11528609 0.12615126 0.17235247 0.17567557 0.2128120 0.2518032    0
## reg.tree    0.05815617 0.08599073 0.12448333 0.13064566 0.1460979 0.2345080    0
## rt          0.07965616 0.10426967 0.13226751 0.13957112 0.1801431 0.2003045    0
## boosting    0.09382376 0.10886935 0.16338729 0.16429301 0.2016713 0.2678397    0

```

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



Select final model

Since MARS model has the lowest mean RMSE, we selected it as our final model.

Test error

```
mars.pred <- predict(mars.fit, newdata = x2)

mars.testerror = mean((mars.pred - y2)^2)

mars.testerror

## [1] 0.2915917
```

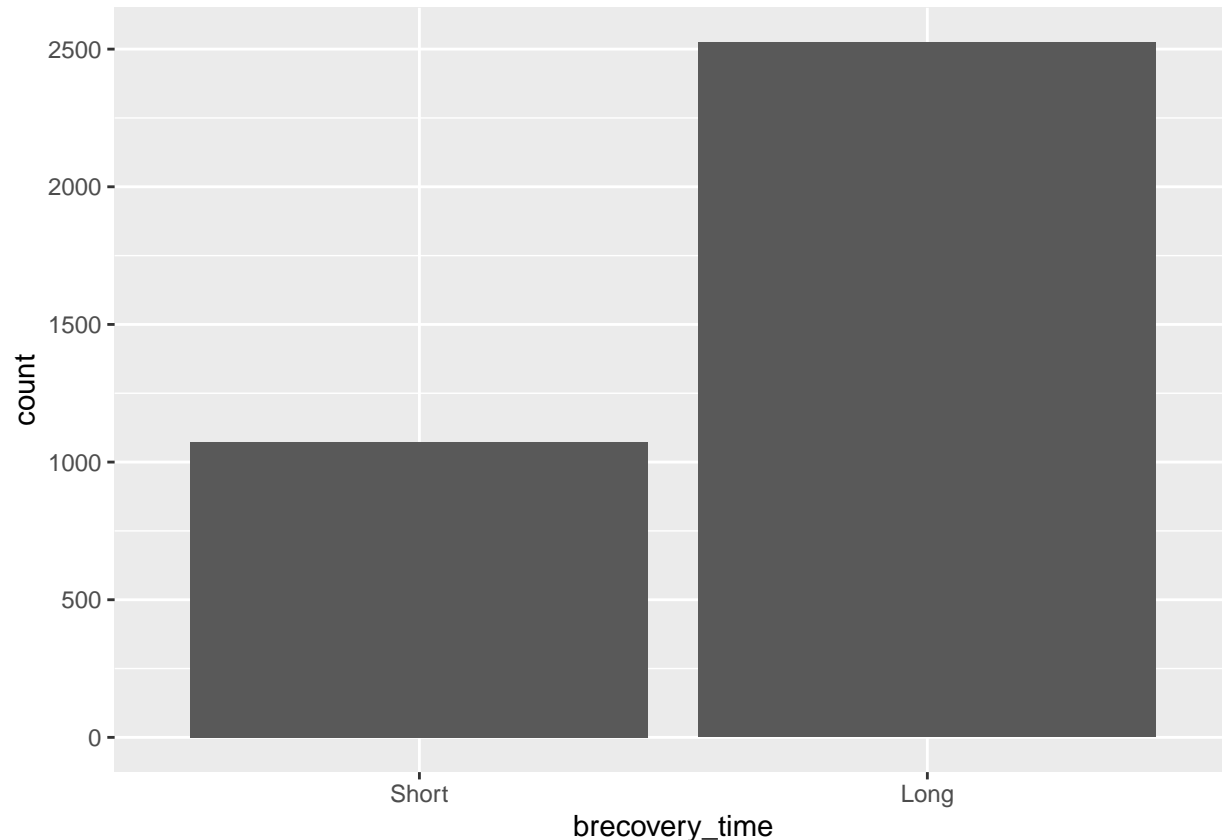
Secondary analysis: binary time to recovery

Set up

```
# make binary outcome
dat_bind_secondary = dat_bind %>%
  na.omit(dat_bind) %>%
  mutate(brecovery_time = ifelse(recovery_time > 30, 1, 0)) %>%
  mutate(brecovery_time = as.factor(brecovery_time)) %>%
  mutate(brecovery_time = dplyr::recode(brecovery_time,
    "1" = "Long", "0" = "Short")) %>%
  dplyr::select(-recovery_time, -id)
```

```
# lets check the distribution
binary = dat_bind_secondary %>%
  ggplot(aes(x = brecovery_time)) + geom_bar()
```

binary



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

trainRows_sec <- createDataPartition(y = dat_bind_secondary$brecovery_time, p = 0.8, list = FALSE)

ctrl12 <- trainControl(method = "cv",
  summaryFunction = twoClassSummary,
  classProbs = TRUE)

contrasts(dat_bind_secondary$brecovery_time)
```

```
##      Long
## Short  0
## Long   1
```

Understand the relationship between continuous predictors and the binary outcome

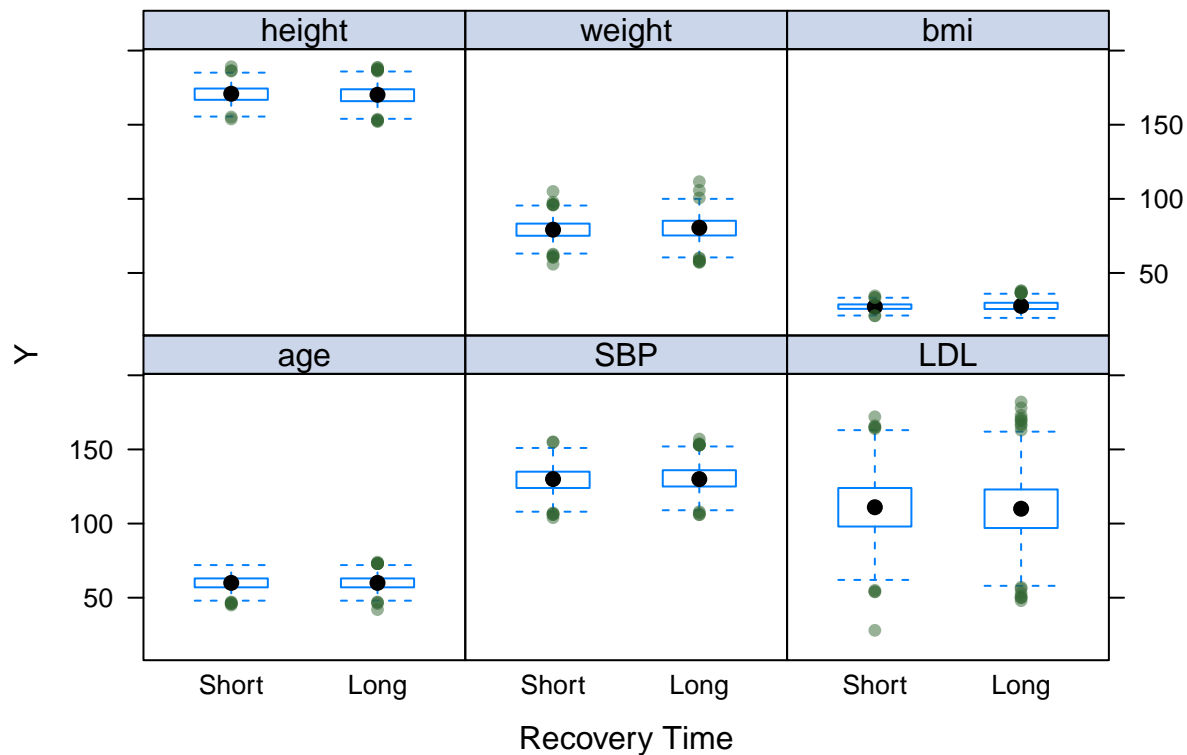
```
theme1 <- trellis.par.get()
theme1$plot.symbol$col <- rgb(.2, .4, .2, .5)
theme1$plot.symbol$pch <- 16
```

```

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(brecovery_time ~ ., dat_bind_secondary)[trainRows_sec,c("age", "SBP", "LDL", "height", "weight", "bmi")],
            y = dat_bind_secondary$brecovery_time[trainRows_sec],
            plot = "box",
            span = .5,
            labels = c("Recovery Time","Y"),
            layout = c(3,2))

```



Understand the relationship between categorical predictors and the binary outcome

```

gender2 = (dat_bind_secondary[trainRows_sec, -1]) %>%
  ggplot(aes(x = brecovery_time, fill = as.factor(gender), group = as.factor(gender))) + geom_bar(position = "dodge") +
  scale_fill_discrete(labels = c("Female", "Male"))

race2 = (dat_bind_secondary[trainRows_sec, -1]) %>%
  ggplot(aes(x = brecovery_time, fill = as.factor(race), group = as.factor(race))) + geom_bar(position = "dodge") +
  scale_fill_discrete(labels = c("White", "Asian", "Black", "Hispanic"))

smoking2 = (dat_bind_secondary[trainRows_sec, -1]) %>%
  ggplot(aes(x = brecovery_time, fill = as.factor(smoking), group = as.factor(smoking))) + geom_bar(position = "dodge") +
  scale_fill_discrete(labels = c("Never smoked", "Former Smoker", "Current smoker"))

```

```

hypertension2 = (dat_bind_secondary[trainRows_sec, -1]) %>%
  ggplot(aes(x = brecovery_time, fill = as.factor(hypertension), group = as.factor(hypertension))) + geom_bar(position = "dodge") +
  scale_fill_discrete(labels = c("No", "Yes"))

diabetes2 = (dat_bind_secondary[trainRows_sec, -1]) %>%
  ggplot(aes(x = brecovery_time, fill = as.factor(diabetes), group = as.factor(diabetes))) + geom_bar(position = "dodge") +
  scale_fill_discrete(labels = c("No", "Yes"))

vaccine2 = (dat_bind_secondary[trainRows_sec, -1]) %>%
  ggplot(aes(x = brecovery_time, fill = as.factor(vaccine), group = as.factor(vaccine))) + geom_bar(position = "dodge") +
  scale_fill_discrete(labels = c("No", "Yes"))

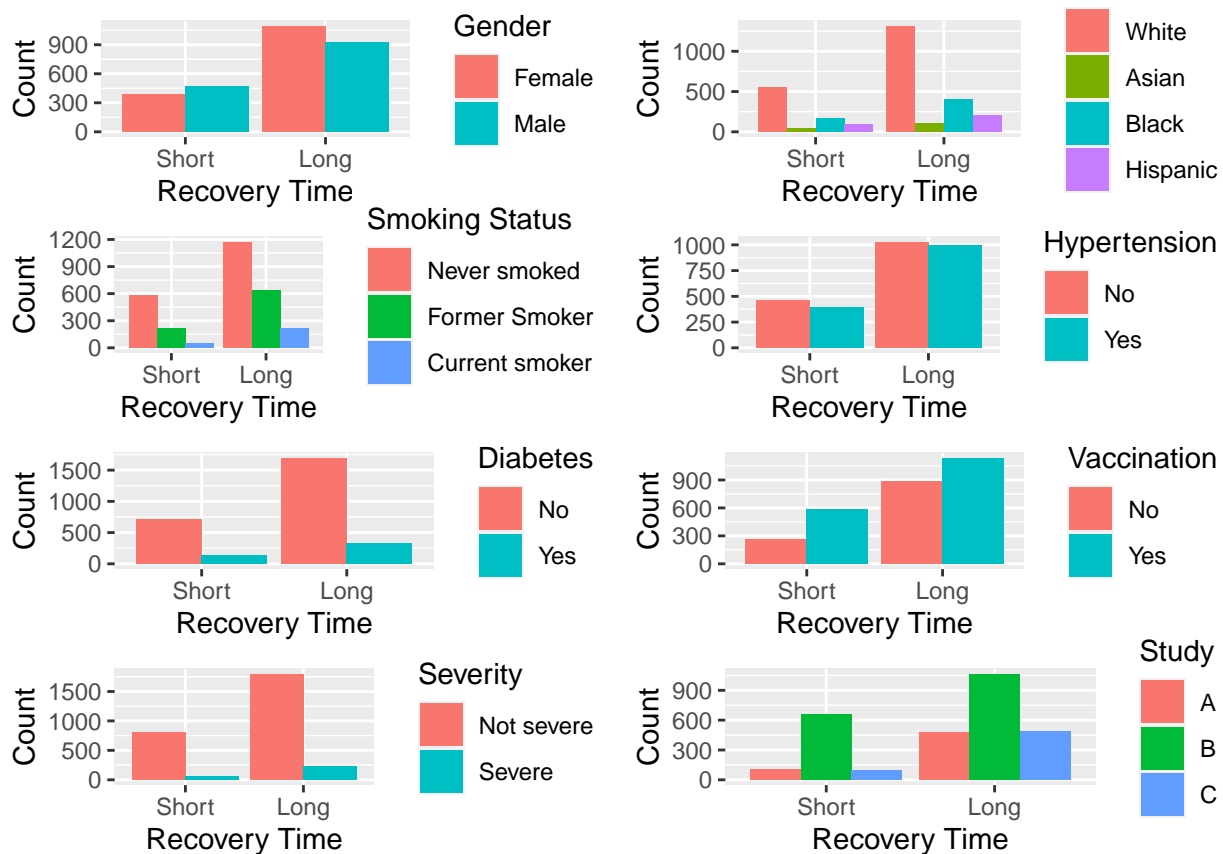
severity2 = (dat_bind_secondary[trainRows_sec, -1]) %>%
  ggplot(aes(x = brecovery_time, fill = as.factor(severity), group = as.factor(severity))) + geom_bar(position = "dodge") +
  scale_fill_discrete(labels = c("Not severe", "Severe"))

study2 = (dat_bind_secondary[trainRows_sec, -1]) %>%
  ggplot(aes(x = brecovery_time, fill = as.factor(study), group = as.factor(study))) + geom_bar(position = "dodge") +
  scale_fill_discrete(labels = c("A", "B", "C"))

cat_combined_plot2 = ggarrange(gender2, race2, smoking2, hypertension2,
                                diabetes2, vaccine2, severity2, study2,
                                ncol = 2, nrow = 4)

cat_combined_plot2

```



Make dummy variables

```
# dummy variable creation

secondary = dat_bind_secondary %>%
  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
secondary = secondary %>%
  dplyr::select(-study, -race, -smoking) %>%
  dplyr::select(brecovery_time, everything()) #arrange variable orders
```

Model 1: Logistic regression

```
set.seed(2460)

model.glm <- train(x = secondary[trainRows_sec,2:17],
  y = secondary$brecovery_time[trainRows_sec],
  method = "glm",
  metric = "ROC",
  trControl = ctrl2,
  preProcess = c("center", "scale"))

summary(model.glm)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -2.2831  -1.2567   0.6816   0.8716   1.4994
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.926893   0.043431  21.342 < 2e-16 ***
## age           0.168966   0.047345   3.569 0.000359 ***
## gender        -0.193910   0.042180  -4.597 4.28e-06 ***
## height         2.142077   0.468179   4.575 4.75e-06 ***
## weight        -2.740050   0.594517  -4.609 4.05e-06 ***
## bmi            3.330127   0.677136   4.918 8.74e-07 ***
## hypertension -0.009330   0.070639  -0.132 0.894924
## diabetes      -0.014487   0.041995  -0.345 0.730121
## SBP            0.051686   0.074248   0.696 0.486350
## LDL           -0.084622   0.043824  -1.931 0.053489 .
## vaccine       -0.277714   0.043300  -6.414 1.42e-10 ***
```

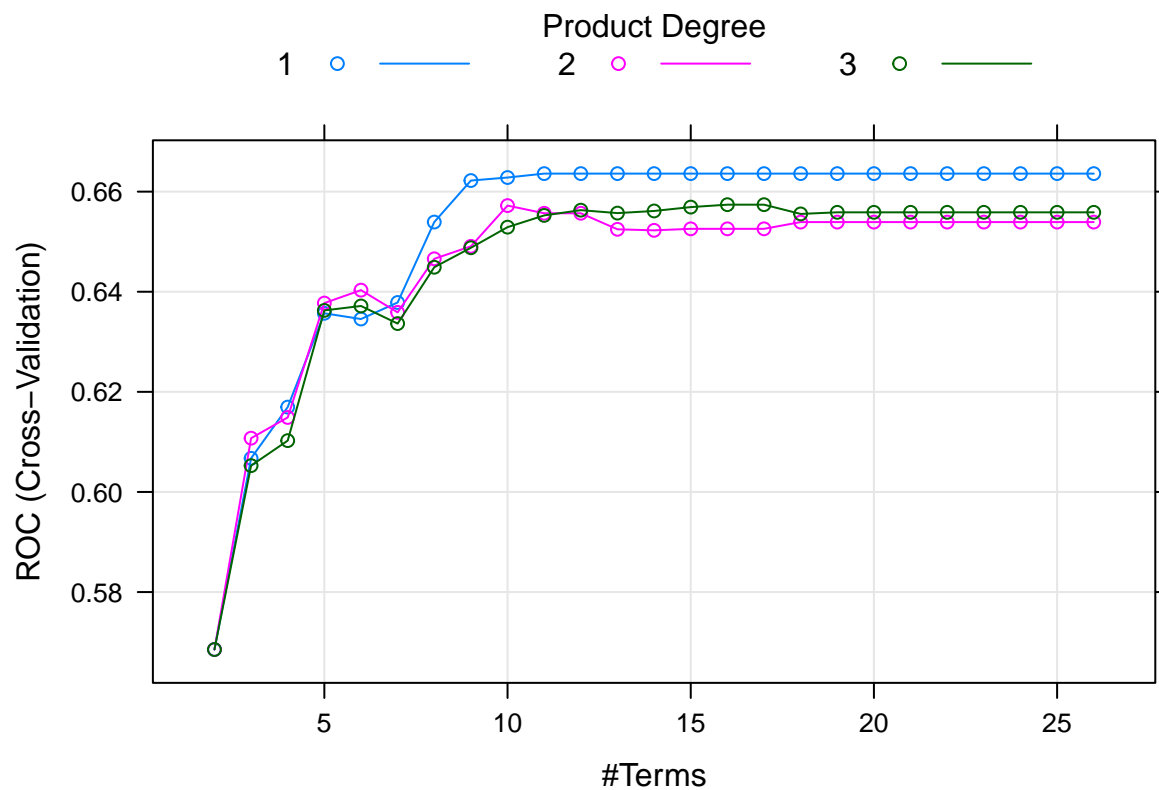
```
## severity      0.199213    0.047867    4.162 3.16e-05 ***
## race_2       -0.014534    0.042492   -0.342 0.732321
## race_3       -0.002099    0.043042   -0.049 0.961097
## race_4       -0.021811    0.042819   -0.509 0.610494
## smoking_1     0.174926    0.043862    3.988 6.66e-05 ***
## smoking_2     0.202077    0.047815    4.226 2.38e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 3506.9  on 2877  degrees of freedom
## Residual deviance: 3323.4  on 2861  degrees of freedom
## AIC: 3357.4
##
## Number of Fisher Scoring iterations: 4
```

Model 2: MARS

```
set.seed(2460)

model.mars <- train(x = secondary[trainRows_sec,2:17],
                    y = secondary$brecovery_time[trainRows_sec],
                    method = "earth",
                    tuneGrid = expand.grid(degree = 1:3,
                                           nprune = 2:26),
                    metric = "ROC",
                    trControl = ctrl2,
                    preProcess = c("center", "scale"))

plot(model.mars)
```

```
model.mars$bestTune
```

```
##      nprune degree
## 10      11      1
```

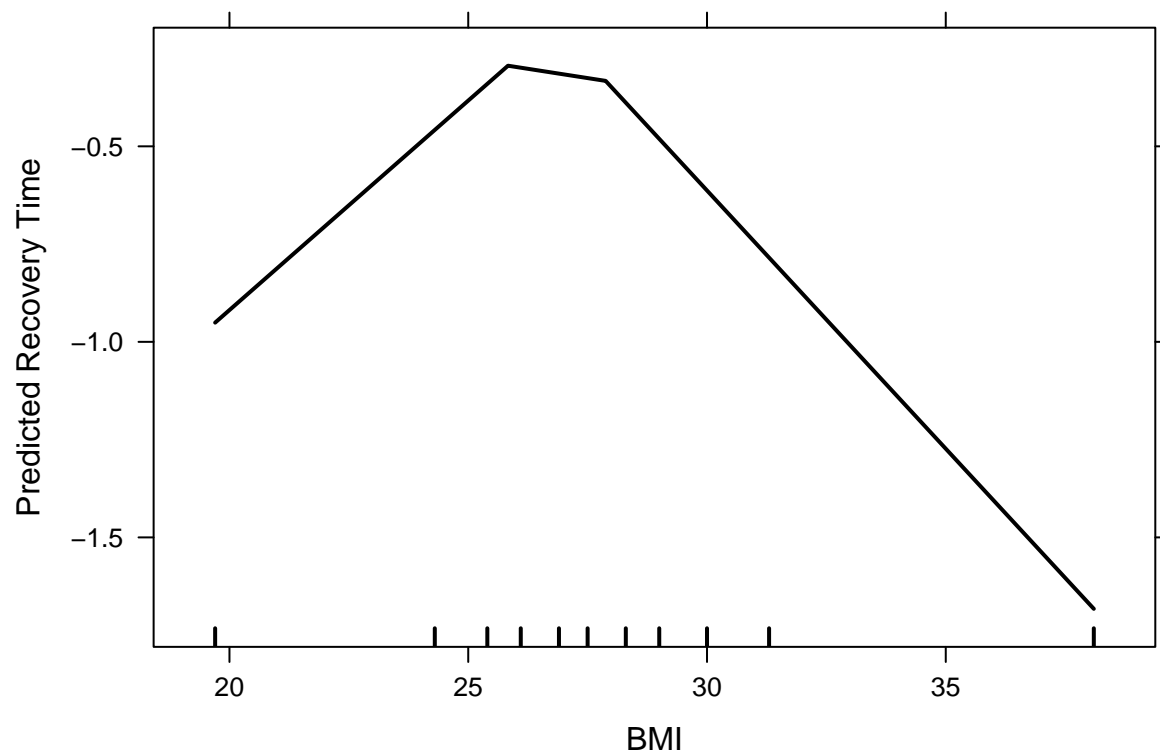
```
coef(model.mars$finalModel)
```

```
##      (Intercept) h(bmi- -0.320384) h(-0.320384-bmi)      vaccine
##      0.06779826      0.73070555      0.59261712      -0.28696690
##      gender      h(age- -1.83762) h(-1.83762-age)      severity
##      -0.20048727      0.15362456      -0.82201440      0.19998994
##      smoking_2      smoking_1 h(-0.675495-LDL)
##      0.20415317      0.18700246      0.27088201
```

Partial dependence

```
pdp2 <- pdp::partial(model.mars,
  pred.var = c("bmi"),
  grid.resolution = 10) %>%
  plotPartial(train = secondary[trainRows_sec,], rug = TRUE,
    xlab = "BMI", ylab = "Predicted Recovery Time")
```

```
pdp2
```



Model 3: LDA

```
set.seed(2460)

model.lda <- train(x = secondary[trainRows_sec,2:17],
  y = secondary$brecovery_time[trainRows_sec],
  method = "lda",
  metric = "ROC",
  trControl = ctrl2,
  preProcess = c("center", "scale"))

summary(model.lda)
```

##	Length	Class	Mode
## prior	2	-none-	numeric
## counts	2	-none-	numeric
## means	32	-none-	numeric
## scaling	16	-none-	numeric
## lev	2	-none-	character
## svd	1	-none-	numeric
## N	1	-none-	numeric
## call	3	-none-	call
## xNames	16	-none-	character
## problemType	1	-none-	character
## tuneValue	1	data.frame	list
## obsLevels	2	-none-	character
## param	0	-none-	list

Model 4: Classification tree

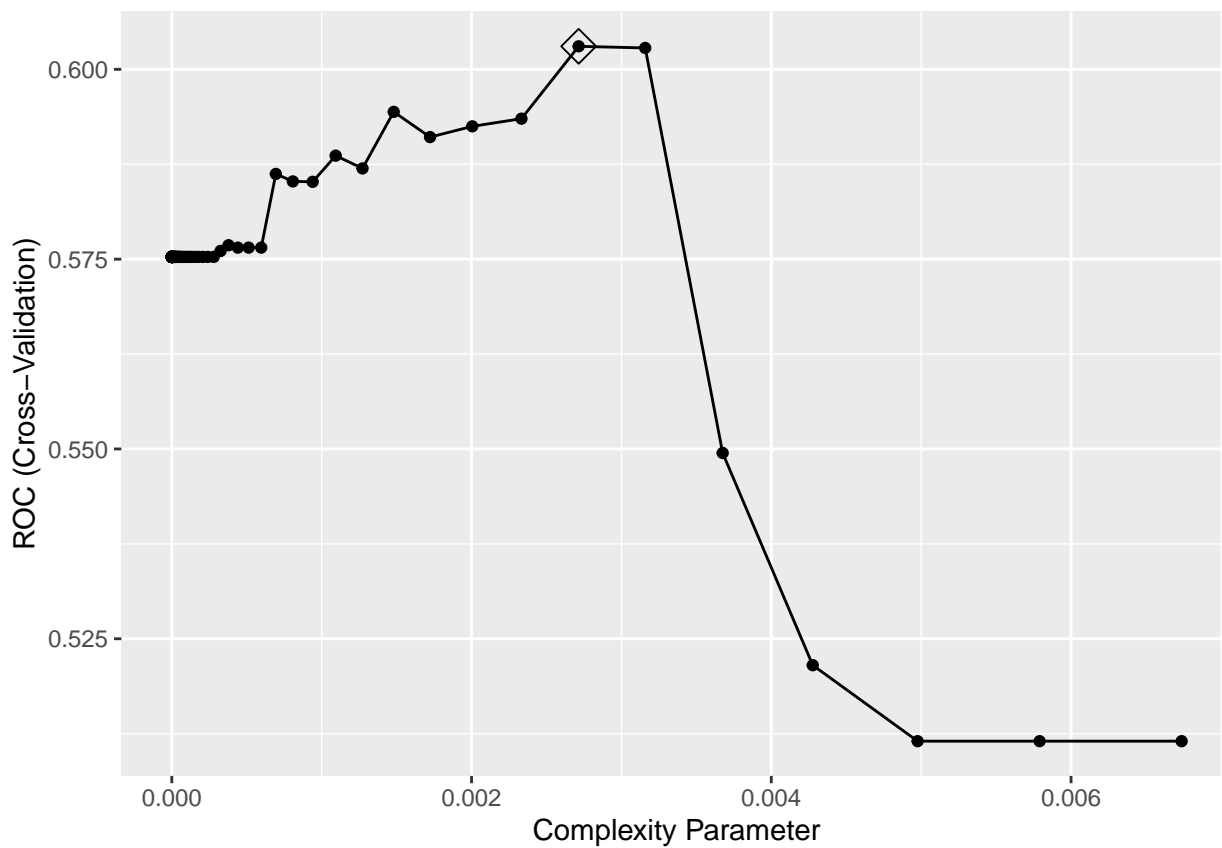
```
set.seed(2460)

model.rpart <- train(brecovery_time ~ . , secondary,
  subset = trainRows_sec,
  method = "rpart",
  tuneGrid = data.frame(cp = exp(seq(-20,-5, len = 100))),
  trControl = ctrl2,
  metric = "ROC",
  preProcess = c("center", "scale"))

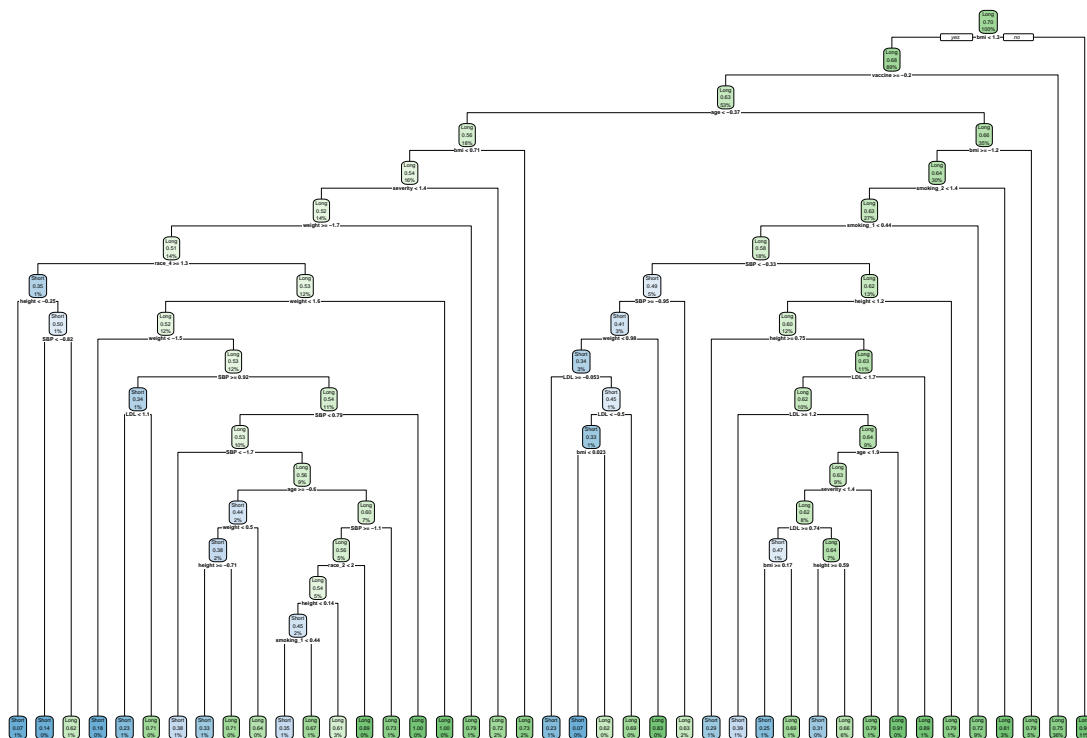
model.rpart$bestTune
```

```
##          cp
## 94 0.002714654
```

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



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



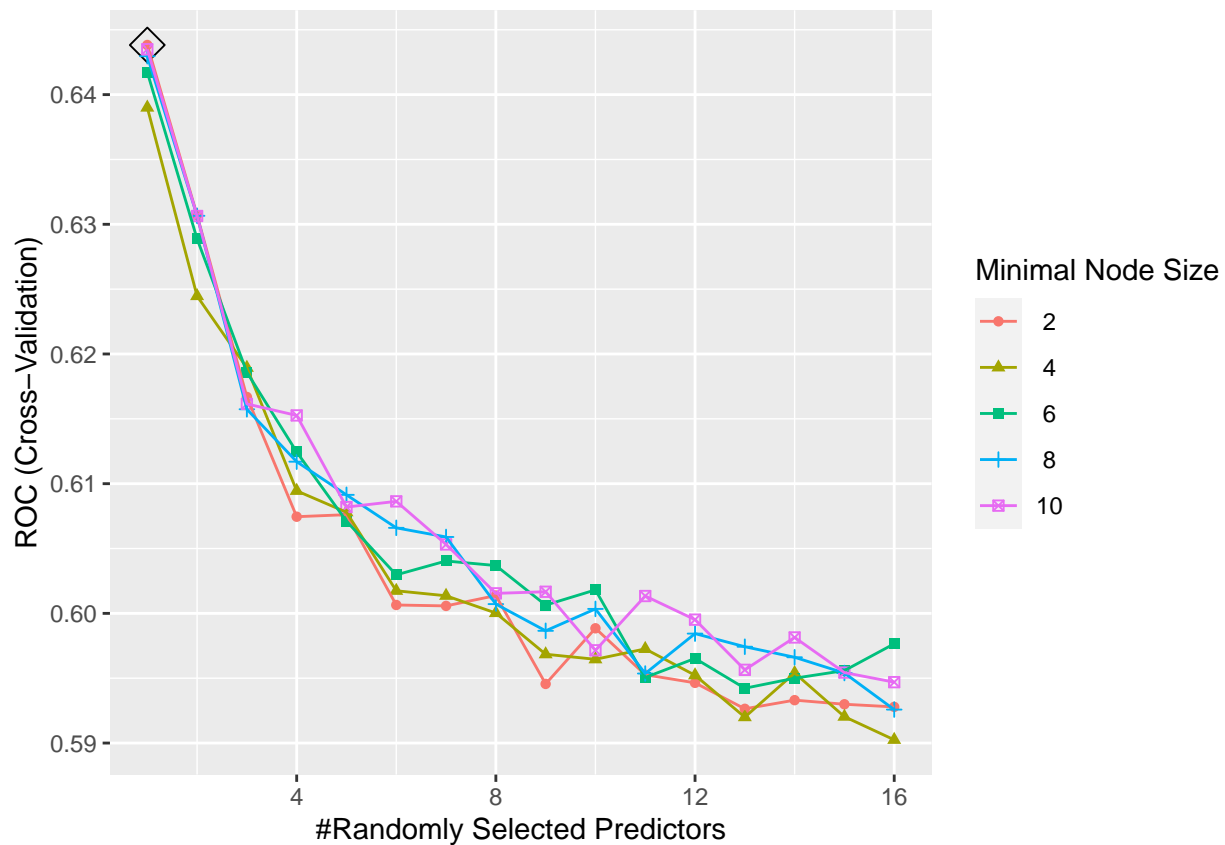
Model 5: Random forest

```
rf.grid.sec <- expand.grid(mtry = 1:16,
                           splitrule = "gini",
                           min.node.size = seq(from = 2, to = 10, by = 2))

set.seed(2460)

model.rf <- train(brecovery_time ~ . , secondary,
                  subset = trainRows_sec,
                  method = "ranger",
                  tuneGrid = rf.grid.sec,
                  metric = "ROC",
                  trControl = ctrl2,
                  preProcess = c("center", "scale"))

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



```
model.rf$bestTune
```

```
## mtry splitrule min.node.size
## 1 1 gini 2
```

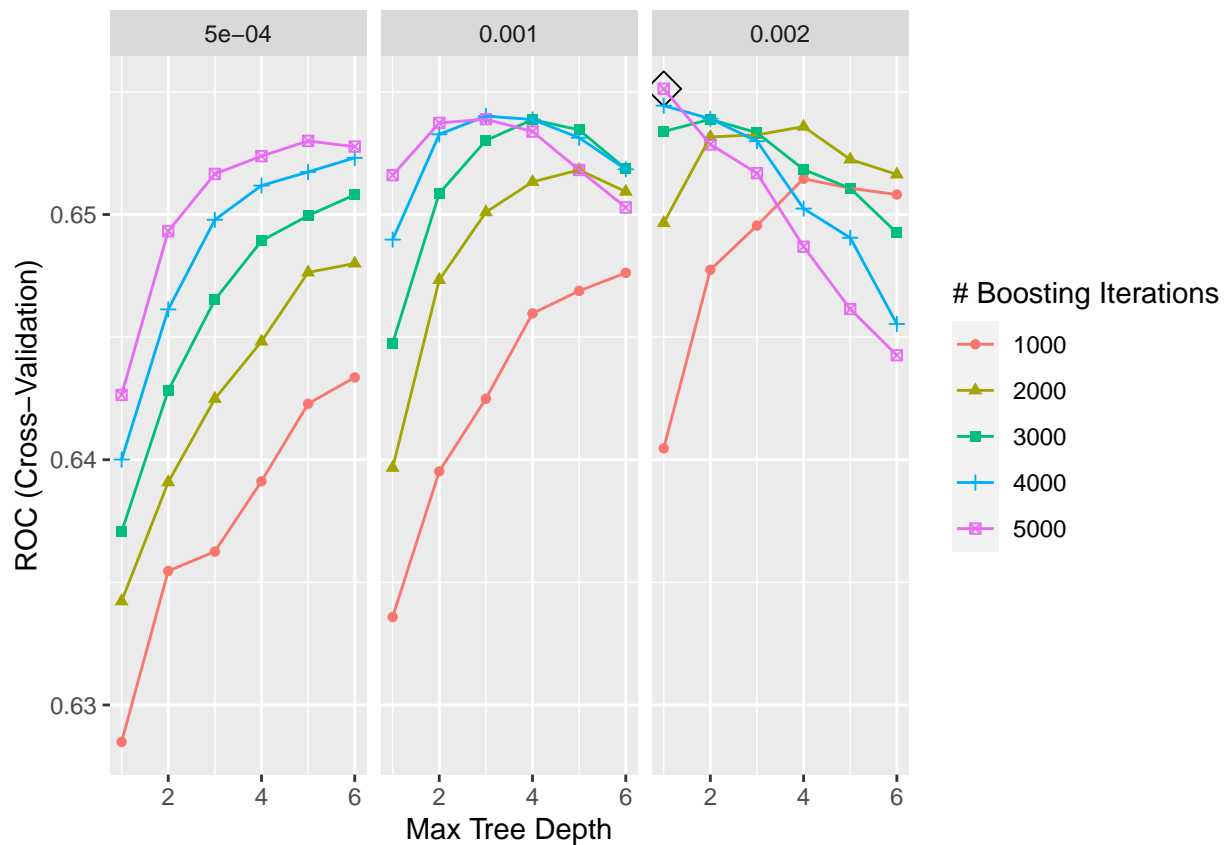
Model 6: Boosting

```
gbmA.grid <- expand.grid(n.trees = c(1000,2000,3000,4000,5000),
                        interaction.depth = 1:6,
                        shrinkage = c(0.0005,0.001,0.002),
                        n.minobsinnode = 1)
```

```
set.seed(2460)
```

```
model.gbmA <- train(brecovery_time ~ . , secondary,
                    subset = trainRows_sec,
                    tuneGrid = gbmA.grid,
                    trControl = ctrl2,
                    method = "gbm",
                    distribution = "adaboost",
                    metric = "ROC",
                    preProcess = c("center", "scale"),
                    verbose = FALSE)
```

```
ggplot(model.gbmA, highlight = TRUE)
```



```
model.gbmA$bestTune
```

```
##      n.trees interaction.depth shrinkage n.minobsinnode
## 65      5000                1      0.002                1
```

Model comparison

```
res_sec <- resamples(list(logistic = model.glm,
                          mars = model.mars,
                          lda = model.lda,
                          classification.tree = model.rpart,
                          rf = model.rf,
                          boosting = model.gbmA))
```

```
summary(res_sec)
```

```
##
## Call:
## summary.resamples(object = res_sec)
##
## Models: logistic, mars, lda, classification.tree, rf, boosting
## Number of resamples: 10
##
## ROC
##
```

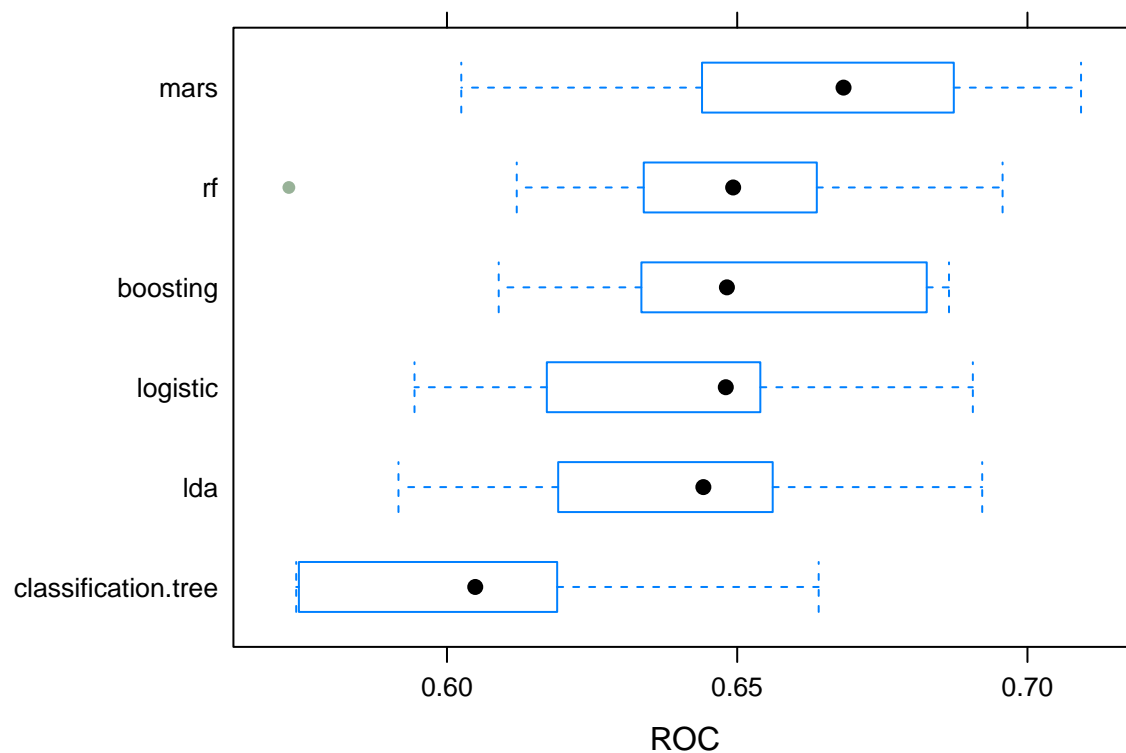
	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
## logistic	0.5944048	0.6244100	0.6480330	0.6437327	0.6534797	0.6905941
## mars	0.6024637	0.6468383	0.6683168	0.6636018	0.6851687	0.7092312

```

## lda 0.5916417 0.6252159 0.6441691 0.6434160 0.6559550 0.6922058
## classification.tree 0.5740245 0.5764305 0.6048670 0.6030341 0.6173728 0.6640361
## rf 0.5727608 0.6340663 0.6492998 0.6438254 0.6623590 0.6957172
## boosting 0.6089109 0.6359371 0.6482270 0.6551238 0.6825639 0.6864881
## NA's
## logistic 0
## mars 0
## lda 0
## classification.tree 0
## rf 0
## boosting 0
##
## Sens
## Min. 1st Qu. Median Mean 3rd Qu.
## logistic 0.03488372 0.04664843 0.06429549 0.07105335 0.08720930
## mars 0.06976744 0.08163475 0.09883721 0.12359781 0.15988372
## lda 0.03488372 0.04664843 0.05841313 0.06522572 0.07848837
## classification.tree 0.10465116 0.15697674 0.19186047 0.18777018 0.22093023
## rf 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## boosting 0.00000000 0.01463748 0.02325581 0.02326949 0.03197674
## Max. NA's
## logistic 0.12790698 0
## mars 0.21176471 0
## lda 0.11627907 0
## classification.tree 0.24705882 0
## rf 0.00000000 0
## boosting 0.04651163 0
##
## Spec
## Min. 1st Qu. Median Mean 3rd Qu. Max.
## logistic 0.9306931 0.9616337 0.9678218 0.9618812 0.9702970 0.9702970
## mars 0.9207921 0.9319307 0.9504950 0.9475248 0.9628713 0.9752475
## lda 0.9356436 0.9628713 0.9702970 0.9663366 0.9752475 0.9851485
## classification.tree 0.8613861 0.8873762 0.9059406 0.9014851 0.9195545 0.9257426
## rf 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
## boosting 0.9752475 0.9876238 0.9950495 0.9925743 0.9987624 1.0000000
## NA's
## logistic 0
## mars 0
## lda 0
## classification.tree 0
## rf 0
## boosting 0

```

```
bwplot(res_sec, metric = "ROC")
```



Select final model

Since MARS model has the highest mean AUC ROC, we selected it as our final model.

Test error

```
pred.mars <- predict(model.mars, newdata = secondary[-trainRows_sec,])

confusionMatrix(data = pred.mars, reference = secondary$brecovery_time[-trainRows_sec],
                 positive = "Long")

## Confusion Matrix and Statistics
##
##           Reference
## Prediction Short Long
##      Short    16   27
##      Long    198  477
##
##           Accuracy : 0.6866
##           95% CI : (0.6513, 0.7204)
##      No Information Rate : 0.7019
##      P-Value [Acc > NIR] : 0.8261
##
##           Kappa : 0.0275
##
##  Mcnemar's Test P-Value : <2e-16
##
##           Sensitivity : 0.94643
##           Specificity : 0.07477
##           Pos Pred Value : 0.70667
```



```
##          Neg Pred Value : 0.37209
##          Prevalence : 0.70195
##          Detection Rate : 0.66435
##    Detection Prevalence : 0.94011
##          Balanced Accuracy : 0.51060
##
##          'Positive' Class : Long
##
```

The accuracy of the MARS model was 0.6866. The misclassification (test error) is calculated as $1 - 0.6866 = 0.3134$.