Midterm Project Analysis

Zhezheng Jin

Contents

```r
library(tidyverse)
library(patchwork)
library(caret)
library(mgcv)
library(AppliedPredictiveModeling)
library(pdp)
library(corrplot)
library(plotmo)
library(ggrepel)
library(vip)
```

Data Wrangling

```r
load("recovery.RData")
dat = dat %>%
  select(-id) %>%
  mutate(
    gender = as.factor(gender),
    race = as.factor(race),
    smoking = as.factor(smoking),
    hypertension = as.factor(hypertension),
    diabetes = as.factor(diabetes),
    vaccine = as.factor(vaccine),
    severity = as.factor(severity),
    study = as.factor(study))
```

Model training Preparation

```r
# data partition
set.seed(2358)
indexTrain <- createDataPartition(y = dat$recovery_time, p = 0.8, list = FALSE)
train <- dat[indexTrain, ]
test <- dat[-indexTrain, ]

# matrix of predictors
x <- model.matrix(recovery_time~.,train)[,-1]
x2 <- model.matrix(recovery_time~.,test)[,-1]

# vector of response
y <- train$recovery_time
y2 <- test$recovery_time

# 10-fold cv on best
ctrl1 <- trainControl(method = "cv", number = 10)
```

Expoloratory Data Analysis

```
# Summary Statistics
summary(dat)
```

```
##       age       gender   race     smoking      height          weight      
##   Min.   :42.0   0:1544   1:1967   0:1822   Min.   :147.8   Min.   : 55.90  
##   1st Qu.:57.0   1:1456   2: 158   1: 859   1st Qu.:166.0   1st Qu.: 75.20  
##   Median :60.0            3: 604   2: 319   Median :169.9   Median : 79.80  
##   Mean   :60.2            4: 271            Mean   :169.9   Mean   : 79.96  
##   3rd Qu.:63.0                              3rd Qu.:173.9   3rd Qu.: 84.80  
##   Max.   :79.0                              Max.   :188.6   Max.   :103.70  
##       bmi        hypertension diabetes      SBP             LDL           vaccine
##   Min.   :18.80   0:1508       0:2537   Min.   :105.0   Min.   : 28.0   0:1212  
##   1st Qu.:25.80   1:1492       1: 463   1st Qu.:125.0   1st Qu.: 97.0   1:1788  
##   Median :27.65                         Median :130.0   Median :110.0           
##   Mean   :27.76                         Mean   :130.5   Mean   :110.5           
##   3rd Qu.:29.50                         3rd Qu.:136.0   3rd Qu.:124.0           
##   Max.   :38.90                         Max.   :156.0   Max.   :178.0           
##   severity study    recovery_time   
##   0:2679   A:2000   Min.   :  2.00  
##   1: 321   B:1000   1st Qu.: 31.00  
##                     Median : 39.00  
##                     Mean   : 42.17  
##                     3rd Qu.: 49.00  
##                     Max.   :365.00  
```

```
sum(is.na(dat))
```

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

```
skimr::skim(dat)
```

Table 1: Data summary

Name	dat
Number of rows	3000
Number of columns	15
Column type frequency:	
factor	8
numeric	7
Group variables	None

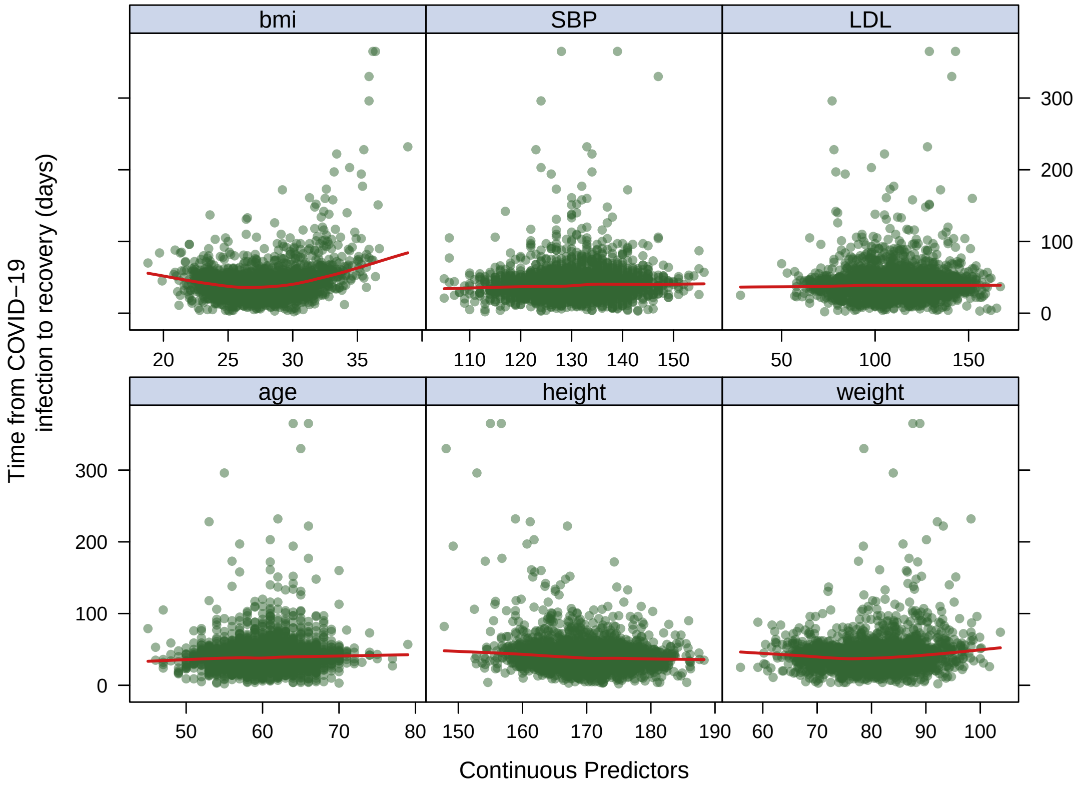
Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
gender	0	1	FALSE	2	0: 1544, 1: 1456
race	0	1	FALSE	4	1: 1967, 3: 604, 4: 271, 2: 158
smoking	0	1	FALSE	3	0: 1822, 1: 859, 2: 319
hypertension	0	1	FALSE	2	0: 1508, 1: 1492
diabetes	0	1	FALSE	2	0: 2537, 1: 463
vaccine	0	1	FALSE	2	1: 1788, 0: 1212
severity	0	1	FALSE	2	0: 2679, 1: 321
study	0	1	FALSE	2	A: 2000, B: 1000

Variable type: numeric

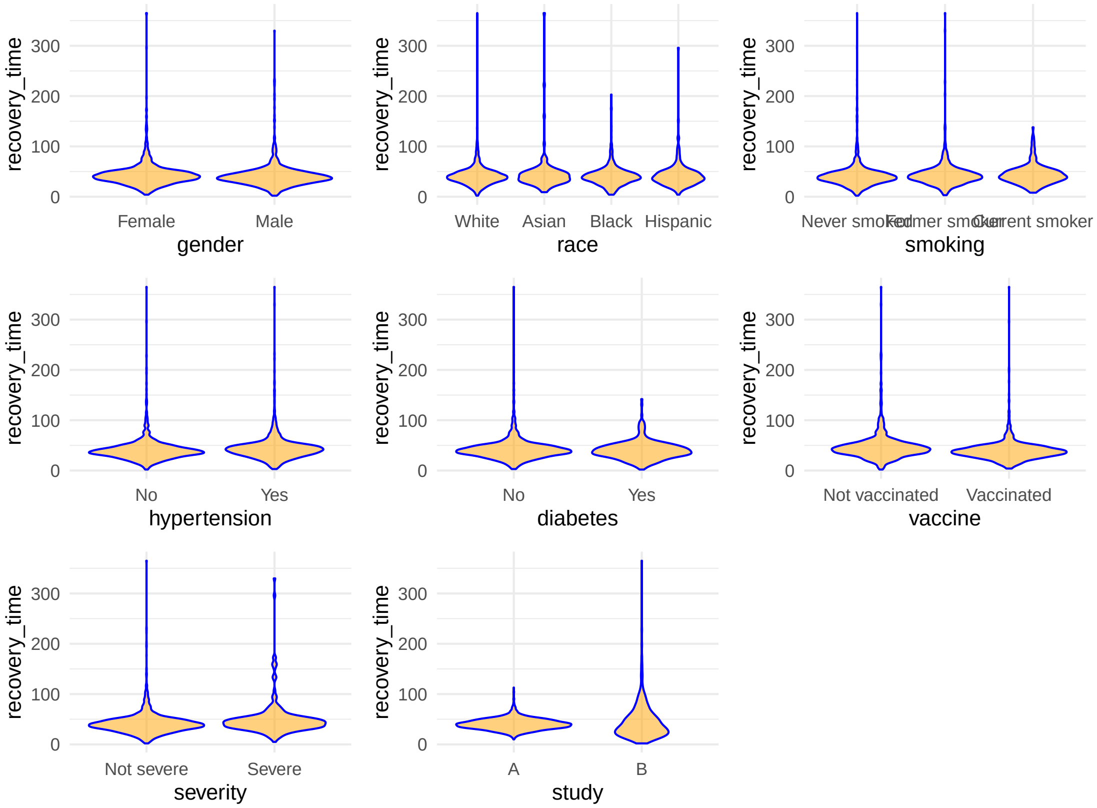
skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
age	0	1	60.20	4.48	42.0	57.0	60.00	63.0	79.0	
height	0	1	169.90	5.97	147.8	166.0	169.90	173.9	188.6	
weight	0	1	79.96	7.14	55.9	75.2	79.80	84.8	103.7	
bmi	0	1	27.76	2.79	18.8	25.8	27.65	29.5	38.9	
SBP	0	1	130.47	7.97	105.0	125.0	130.00	136.0	156.0	
LDL	0	1	110.45	19.76	28.0	97.0	110.00	124.0	178.0	
recovery_time	0	1	42.17	23.15	2.0	31.0	39.00	49.0	365.0	

```r
# Remove all the categorical predictors out from x before plotting
x_continuous <-
  x[, !(colnames(x) %in%
          c("gender1", "race2","race3", "race4",
            "smoking1","smoking2", "hypertension1",
            "diabetes1", "vaccine1", "severity1", "studyB"))]

# For Continuous Predictors
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)
featurePlot(x_continuous,
            y,
            plot = "scatter",
            span = .5,
            labels = c("Continuous Predictors","Time from COVID-19
                       infection to recovery (days)"),
            type = c("p", "smooth"),
            layout = c(3, 2))
```

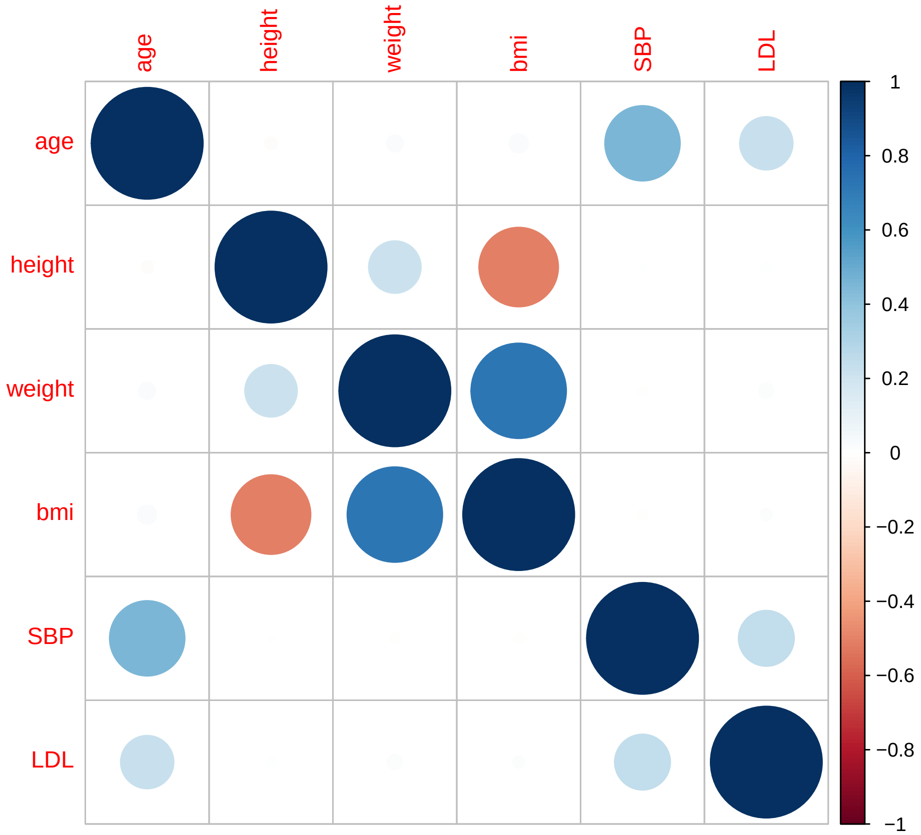


```
# For Categorical Predictors
dis_gender = train %>%
  ggplot(aes(x = gender, y = recovery_time)) +
  geom_violin(fill = "orange", color = "blue", alpha = .5) +
  scale_x_discrete(labels = c('Female','Male'))

dis_race = train %>%
  ggplot(aes(x = race, y = recovery_time)) +
  geom_violin(fill = "orange", color = "blue", alpha = .5) +
  scale_x_discrete(labels = c('White','Asian','Black', 'Hispanic'))

dis_smoking = train %>%
  ggplot(aes(x = smoking, y = recovery_time)) +
  geom_violin(fill = "orange", color = "blue", alpha = .5) +
  scale_x_discrete(labels = c('Never smoked','Former smoker','Current smoker'))

dis_hyper = train %>%
  ggplot(aes(x = hypertension, y = recovery_time)) +
  geom_violin(fill = "orange", color = "blue", alpha = .5) +
  scale_x_discrete(labels = c('No','Yes'))

dis_diabetes = train %>%
  ggplot(aes(x = diabetes, y = recovery_time)) +
  geom_violin(fill = "orange", color = "blue", alpha = .5) +
  scale_x_discrete(labels = c('No','Yes'))
```

```
dis_vac = train %>%
  ggplot(aes(x = vaccine, y = recovery_time)) +
  geom_violin(fill = "orange", color = "blue", alpha = .5) +
  scale_x_discrete(labels = c('Not vaccinated','Vaccinated'))

dis_serverity = train %>%
  ggplot(aes(x = severity, y = recovery_time)) +
  geom_violin(fill = "orange", color = "blue", alpha = .5) +
  scale_x_discrete(labels = c('Not severe','Severe'))

dis_study = train %>%
  ggplot(aes(x = study, y = recovery_time)) +
  geom_violin(fill = "orange", color = "blue", alpha = .5)

dis_gender + dis_race + dis_smoking + dis_hyper + dis_diabetes +
  dis_vac + dis_serverity + dis_study + plot_layout(ncol=3)
```



```
# Correlation plots
corrplot(cor(x_continuous), method = "circle", type = "full")
```



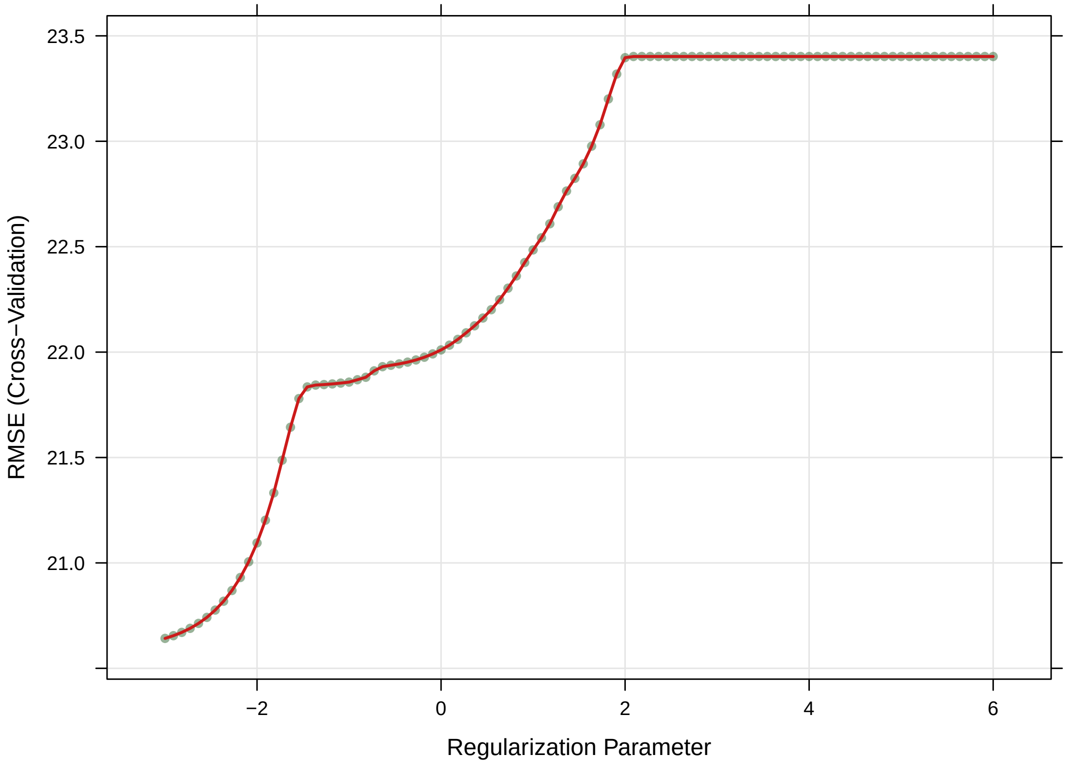
The "recovery" dataset contains 15 columns and 3000 observations without any missing values after omitting the `id` variable. We have 14 predictors (6 numeric and 8 factor(character) variables) in the dataset. Then we partition the dataset into two parts: training data (80%) and test data (20%).

From the correlation matrix, some multicollinearities are exist in the continuous predictors of training data, cross-validation will be applied in the next steps.

Next, we will fit 5 linear models(Lasso, ridge, elastic net, PCR, PLS) and 2 non-linear models(GAM, MARS) using caret, and conduct the model comparison to choose the best fitted one.

Lasso

```
# Fitting the model with Cross-validation
set.seed(2358)
lasso.fit <- train(x, y, method = "glmnet",
                   tuneGrid = expand.grid(alpha = 1,
                                          lambda = exp(seq(6, -3, length=100))),
                   trControl = ctrl1)

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



```
# Tuning parameters
lasso.fit$bestTune
```

```
##   alpha     lambda
## 1     1 0.04978707
```

```
# coefficients in the final model
coef(lasso.fit$finalModel, lasso.fit$bestTune$lambda)
```

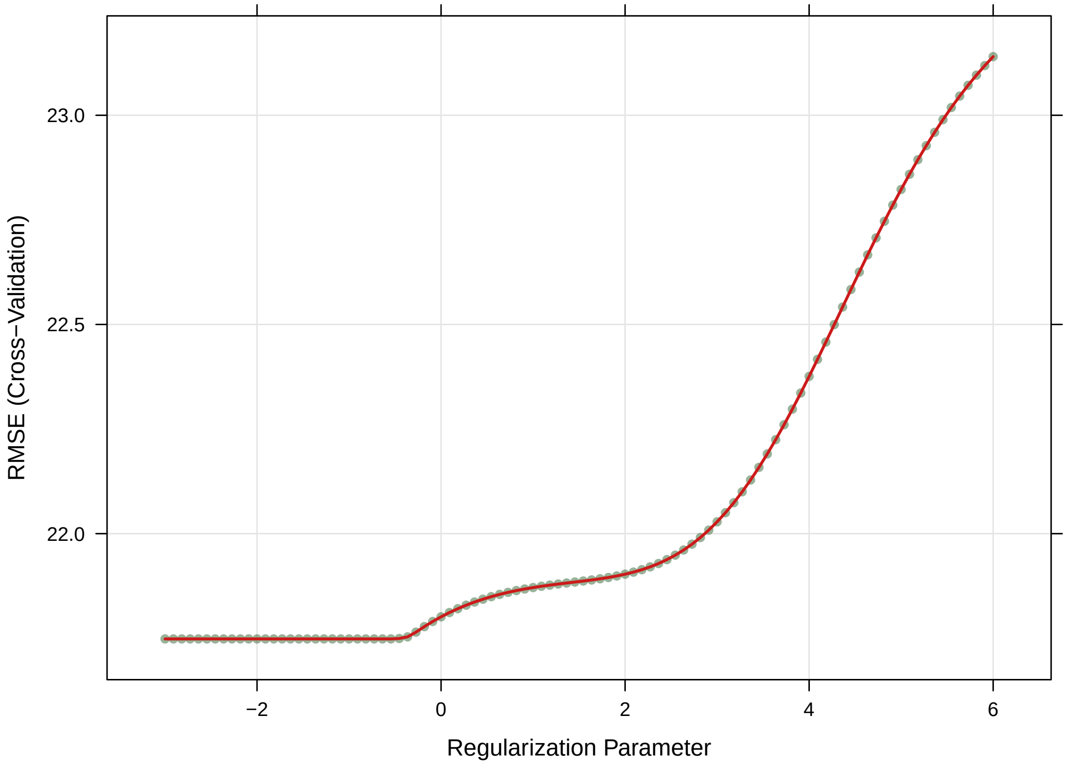
```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                           s1
## (Intercept)   -1.718204e+03
## age            2.032997e-01
## gender1       -2.590566e+00
## race2          2.025447e+00
## race3         -1.007652e+00
## race4         -2.366163e-01
## smoking1       2.126965e+00
## smoking2       2.808260e+00
## height         9.945716e+00
## weight        -1.092301e+01
## bmi            3.328983e+01
## hypertension1  1.686340e+00
## diabetes1     -1.962006e+00
## SBP            9.459282e-02
## LDL           -2.925425e-02
## vaccine1      -7.213945e+00
## severity1      7.737021e+00
## studyB         5.580648e+00
```

```
# Make prediction on test data
lasso.pred <- predict(lasso.fit, newdata = x2)

# Test error
lasso.mse <- mean((lasso.pred - y2)^2)
lasso.mse
```

```
## [1] 313.7238
```

Ridge

```
# Fitting the model with Cross-validation
set.seed(2358)
ridge.fit <- train(x, y, method = "glmnet",
                   tuneGrid = expand.grid(alpha = 0,
                                          lambda = exp(seq(6, -3, length=100))),
                   trControl = ctrl1)

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



```r
# Tuning parameters
ridge.fit$bestTune
```

```
##    alpha    lambda
## 28     0 0.5795783
```

```r
# coefficients in the final model
coef(ridge.fit$finalModel, s = ridge.fit$bestTune$lambda)
```

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                         s1
## (Intercept)   -96.79851044
## age             0.22200050
## gender1        -2.51703872
## race2           2.47088040
## race3          -1.10000451
## race4          -0.91634368
## smoking1        2.14929795
## smoking2        2.40954379
## height          0.37727295
## weight         -0.79747070
## bmi             4.23361543
## hypertension1   1.64368581
## diabetes1      -2.17067303
## SBP             0.09852299
## LDL            -0.03040921
## vaccine1       -7.33424314
## severity1       7.53259172
## studyB          5.73092774
```

```r
# Make prediction on test data
ridge.pred <- predict(ridge.fit, newdata = x2)

# Test error
ridge.mse <- mean((ridge.pred - y2)^2)
ridge.mse
```

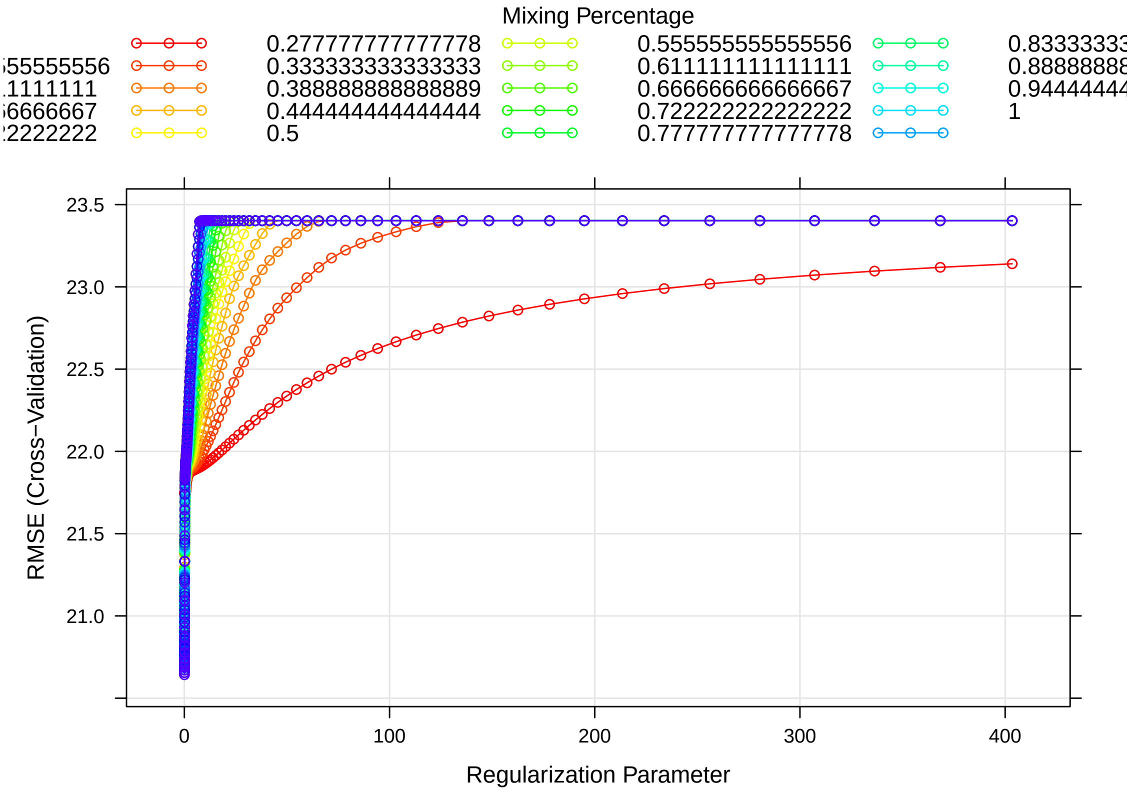
```
## [1] 352.1972
```

Elastic net

```r
# Fitting the model with Cross-validation
set.seed(2358)
enet.fit <- train(x, y,method = "glmnet",
                  tuneGrid = expand.grid(alpha = seq(0 , 1, length = 19),
                                         lambda = exp(seq(6, -3, length = 100))),
                  trControl = ctrl1)

# Tuning parameters
enet.fit$bestTune
```

```
##      alpha     lambda
## 1801     1 0.04978707
```

```
# 25 kinds of Plot colors
myCol<- rainbow(25)
myPar <- list(superpose.symbol = list(col = myCol),
superpose.line = list(col = myCol))

# Plot CV RMSE-alpha&lambda
plot(enet.fit, par.settings = myPar)
```



```
# Coefficients
coef(enet.fit$finalModel, enet.fit$bestTune$lambda)
```

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                        s1
## (Intercept)   -1.718204e+03
## age            2.032997e-01
## gender1       -2.590566e+00
## race2          2.025447e+00
## race3         -1.007652e+00
## race4         -2.366163e-01
## smoking1       2.126965e+00
## smoking2       2.808260e+00
## height         9.945716e+00
## weight        -1.092301e+01
## bmi              3.328983e+01
## hypertension1  1.686340e+00
## diabetes1     -1.962006e+00
## SBP            9.459282e-02
## LDL           -2.925425e-02
## vaccine1      -7.213945e+00
## severity1      7.737021e+00
## studyB         5.580648e+00
```

```
# Make prediction on test data
enet.pred <- predict(enet.fit, newdata = x2)

# Test error
enet.mse <- mean((enet.pred - y2)^2)
enet.mse
```

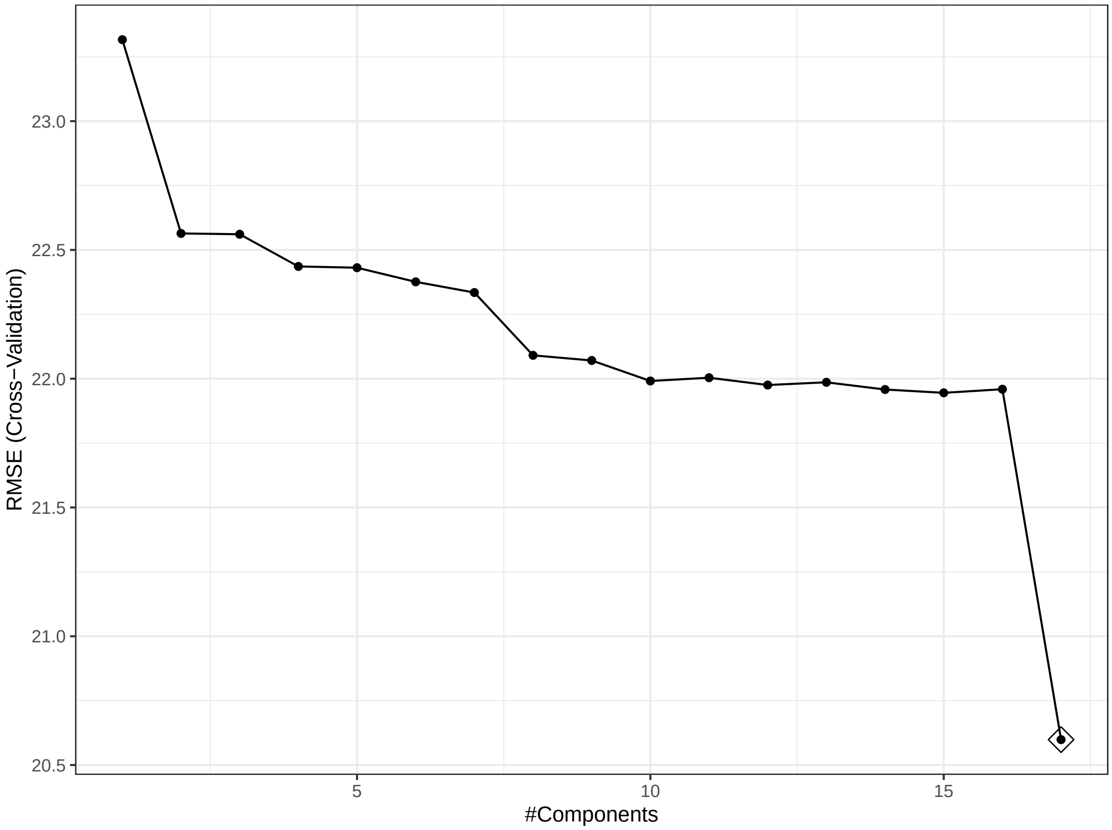
```
## [1] 313.7238
```

Principal Component Regression(PCR)

```
# Fitting the model with Cross-validation
set.seed(2358)
pcr.fit <-  train(x, y,
                  method = "pcr",
                  tuneGrid = data.frame(ncomp = 1:17),
                  trControl = ctrl1,
                  preProcess = c("center", "scale"))

# Make prediction on test data
pcr.pred <- predict(pcr.fit, newdata = x2)

# Test error
pcr.mse <- mean((pcr.pred - y2)^2)
pcr.mse
```

```
## [1] 314.7791
```

```
# Plot cv RMSE-components
ggplot(pcr.fit, highlight = TRUE) + theme_bw()
```

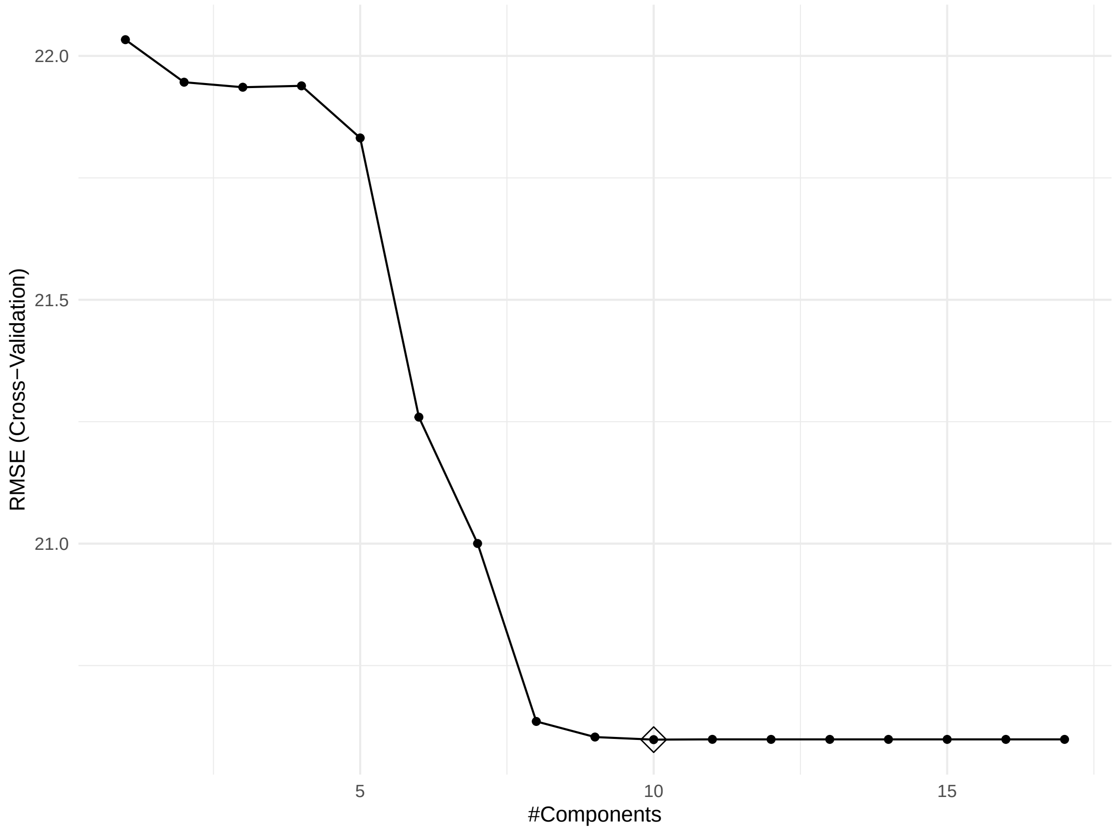


Partial least squares(PLS)

```
# Fitting the model with Cross-validation
set.seed(2358)
pls.fit <-  train(x, y,
                  method = "pls",
                  tuneGrid = data.frame(ncomp = 1:17),
                  trControl = ctrl1,
                  preProcess = c("center", "scale"))

# Make prediction on test data
pls.pred <- predict(pls.fit, newdata = x2)

# Test error
pls.mse <- mean((pls.pred - y2)^2)
pls.mse
```

```
## [1] 314.7983
```

```
# Plot cv RMSE-components
ggplot(pls.fit, highlight = TRUE)
```



Generalized additive model (GAM)

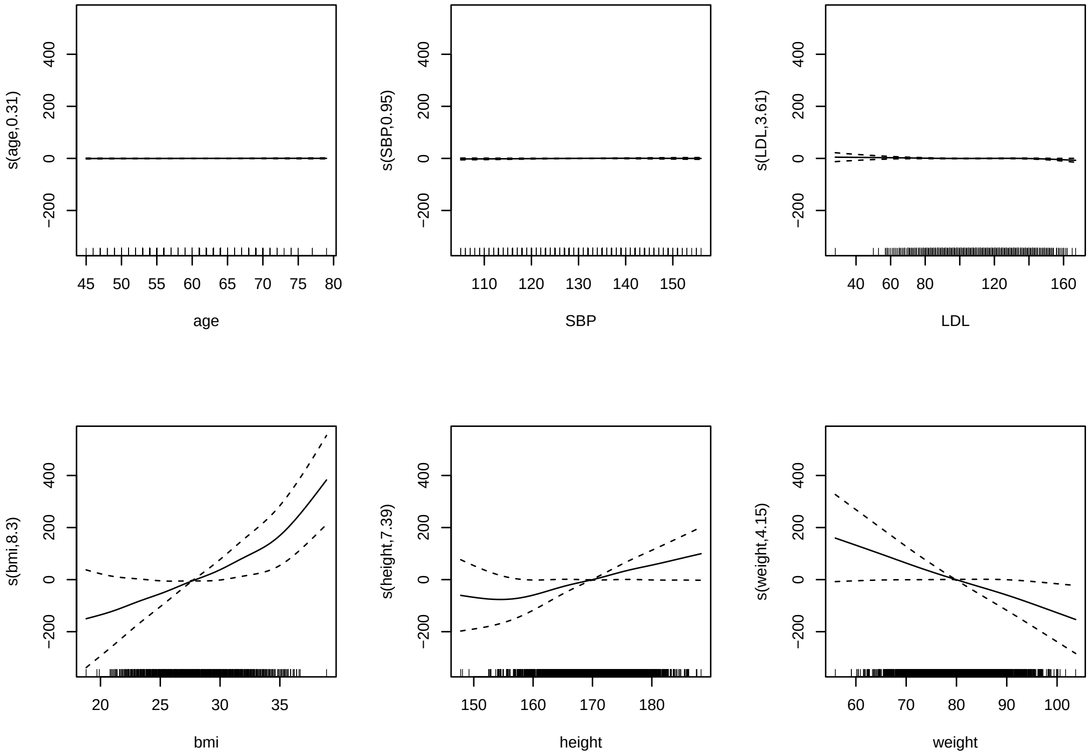
```
# Fit a generalized additive model (GAM) using all the predictors
set.seed(2358)
gam.fit <- train(x, y,
                 method = "gam",
                 tuneGrid = data.frame(method = "GCV.Cp", select = TRUE),
                 trControl = ctrl1)

gam.fit$bestTune
```

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

```
gam.fit$finalModel
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## .outcome ~ gender1 + race2 + race3 + race4 + smoking1 + smoking2 +
##     hypertension1 + diabetes1 + vaccine1 + severity1 + studyB +
##     s(age) + s(SBP) + s(LDL) + s(bmi) + s(height) + s(weight)
##
## Estimated degrees of freedom:
## 0.311 0.948 3.615 8.298 7.395 4.146  total = 36.71
##
## GCV score: 380.8098
```

```
# Plot the results
par(mfrow=c(2,3))
plot(gam.fit$finalModel)
```



```
# Make prediction on test data
gam.pred <- predict(gam.fit, newdata = x2)

# test error
gam.test.error <- mean((gam.pred - y2)^2)
gam.test.error
```

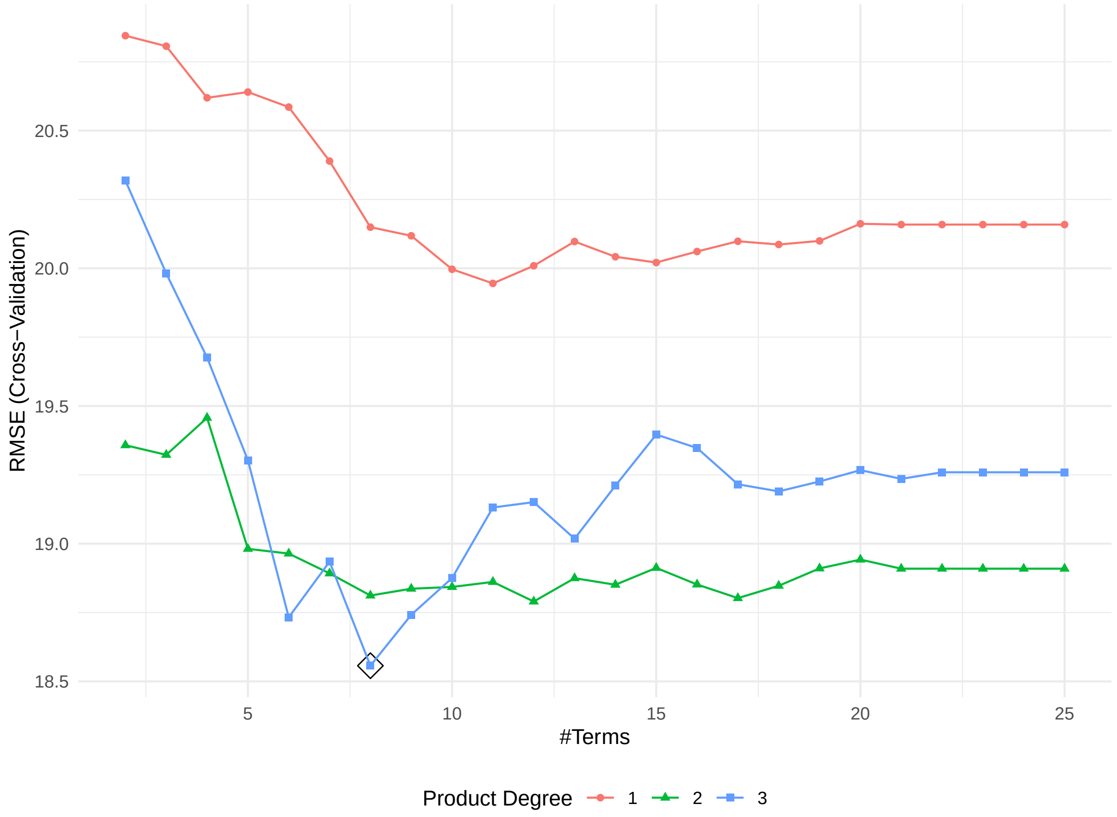
```
## [1] 291.3596
```

Multivariate Adaptive Regression Splines (MARS)

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

set.seed(2358)
mars.fit <- train(x, y,
                  method = "earth",
                  tuneGrid = mars_grid,
                  trControl = ctrl1)
```

```
## Loading required package: earth
```

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



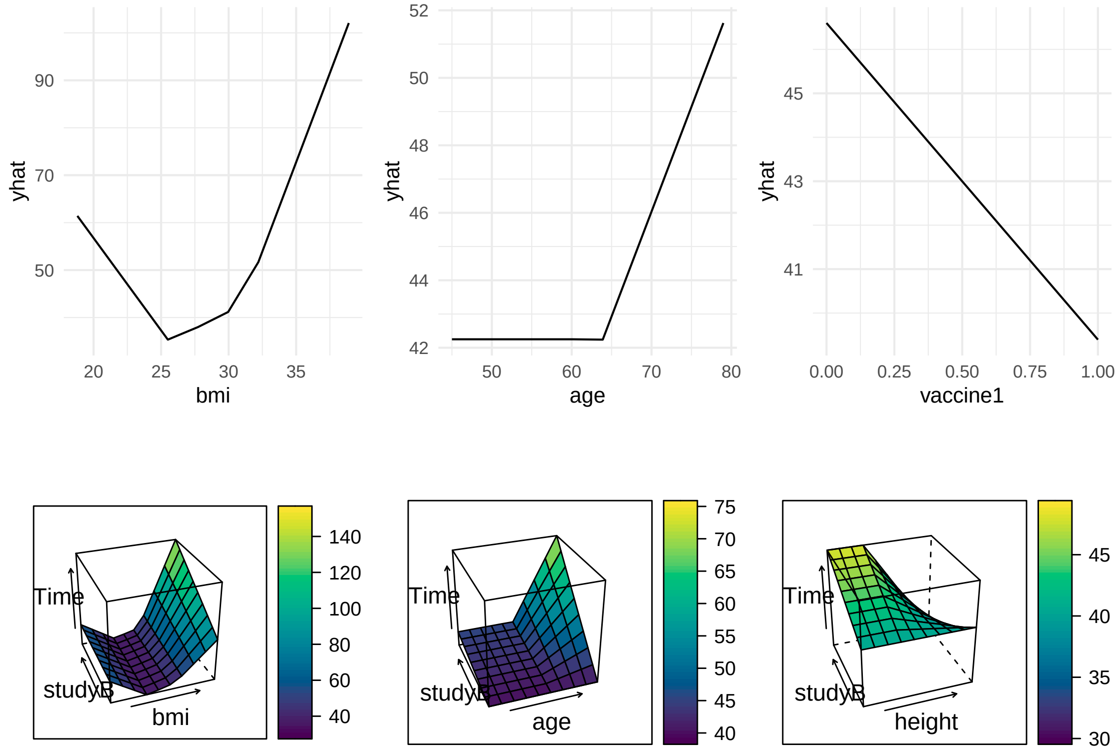
```
mars.fit$bestTune
```

```
##    nprune degree
## 55      8      3
```

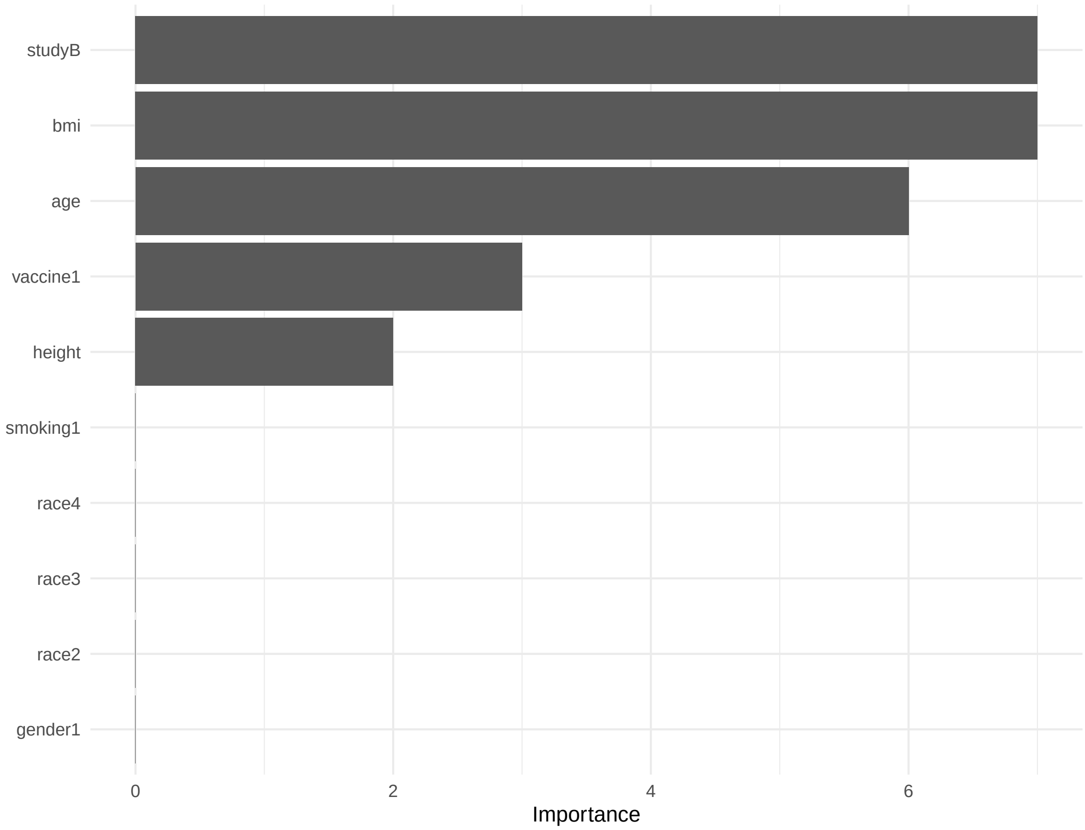
```
coef(mars.fit$finalModel)
```

```
##                           (Intercept)                                  h(31-bmi)
##                             18.222464                                   3.892250
##                   h(bmi-31) * studyB       h(age-62) * h(bmi-31) * studyB
##                          32.871468                               26.143670
##                           vaccine1 h(height-161.4) * h(bmi-31) * studyB
##                          -7.204188                               -3.139983
##                        h(bmi-25.6)        h(age-61) * h(bmi-31) * studyB
##                           5.316962                              -17.146877
```

```
# Present the partial dependence plot of an arbitrary predictor
p1 <- pdp::partial(mars.fit, pred.var = c("bmi"), grid.resolution = 10) %>%
  autoplot()

p2 <- pdp::partial(mars.fit, pred.var = c("age"), grid.resolution = 10) %>%
  autoplot()

p3 <- pdp::partial(mars.fit, pred.var = c("vaccine1"), grid.resolution = 10) %>%
  autoplot()

p4 <- pdp::partial(mars.fit, pred.var = c("bmi", "studyB"),
                   grid.resolution = 10) %>%
      pdp::plotPartial(levelplot = FALSE, zlab = "Time", drape = TRUE,
                       screen = list(z = 20, x = -60))

p5 <- pdp::partial(mars.fit, pred.var = c("age", "studyB"),
                   grid.resolution = 10) %>%
      pdp::plotPartial(levelplot = FALSE, zlab = "Time", drape = TRUE,
                       screen = list(z = 20, x = -60))

p6 <- pdp::partial(mars.fit, pred.var = c("height", "studyB"),
                   grid.resolution = 10) %>%
      pdp::plotPartial(levelplot = FALSE, zlab = "Time", drape = TRUE,
                       screen = list(z = 20, x = -60))

gridExtra::grid.arrange(p1, p2,p3, p4, p5, p6, ncol = 3)
```



```r
# Variable importance plot
p7 <- vip(mars.fit$finalModel)

p7
```



```
# Make prediction on test data
mars.pred <- predict(mars.fit, newdata = x2)

# test error
mars.test.error <- mean((mars.pred - y2)^2)
mars.test.error
```

```
## [1] 405.4464
```

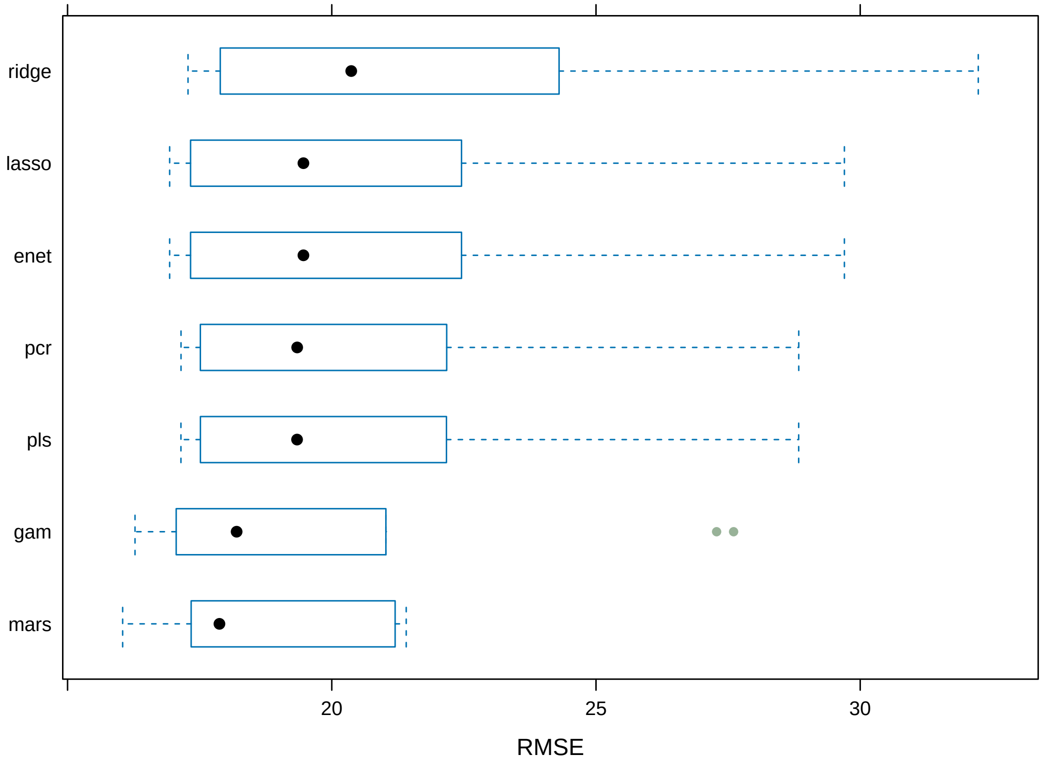
Model Comparison

```
# re-samples
set.seed(2358)
resamp <- resamples(list(lasso = lasso.fit,
                         ridge = ridge.fit,
                         enet = enet.fit,
                         pcr = pcr.fit,
                         pls = pls.fit,
                         gam = gam.fit,
                         mars = mars.fit))
summary(resamp)
```

```
##
## Call:
```

```
## summary.resamples(object = resamp)
##
## Models: lasso, ridge, enet, pcr, pls, gam, mars
## Number of resamples: 10
##
## MAE
##           Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## lasso 12.33294 12.72143 13.12135 13.29926 13.68387 14.61046    0
## ridge 12.41965 12.89082 12.97258 13.54382 14.04781 15.44988    0
## enet  12.33294 12.72143 13.12135 13.29926 13.68387 14.61046    0
## pcr   12.56665 13.01623 13.50877 13.55186 13.86917 14.80595    0
## pls   12.56717 13.01642 13.50925 13.55162 13.86545 14.80890    0
## gam   11.67916 12.35691 12.82913 12.92176 13.03443 15.00257    0
## mars  11.22777 11.51274 11.78872 12.08433 12.73739 13.29310    0
##
## RMSE
##           Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## lasso 16.93186 17.38482 19.46325 20.64202 21.79498 29.70053    0
## ridge 17.27977 17.89149 20.36833 21.74843 23.53011 32.23424    0
## enet  16.93186 17.38482 19.46325 20.64202 21.79498 29.70053    0
## pcr   17.14718 17.58935 19.34523 20.59863 21.68019 28.83700    0
## pls   17.14588 17.59106 19.34304 20.59811 21.67704 28.83590    0
## gam   16.27655 17.07963 18.19943 19.99675 20.85941 27.60386    0
## mars  16.04255 17.41683 17.87424 18.55684 20.44732 21.40862    0
##
## Rsquared
##             Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lasso 0.06401091 0.1740920 0.2260939 0.2365954 0.2965213 0.4616211    0
## ridge 0.02947992 0.1286964 0.1434451 0.1522536 0.1837512 0.2598344    0
## enet  0.06401091 0.1740920 0.2260939 0.2365954 0.2965213 0.4616211    0
## pcr   0.06848279 0.1670135 0.2323908 0.2405915 0.3099305 0.4833811    0
## pls   0.06853575 0.1669119 0.2323691 0.2406330 0.3101671 0.4835539    0
## gam   0.11995974 0.1673279 0.2158768 0.2909863 0.3792122 0.6079131    0
## mars  0.07183623 0.2437275 0.3843057 0.3954056 0.5643194 0.7308590    0
```

```
# RMSE box-plot between models
bwplot(resamp, metric = "RMSE")
```



Based on the summary and plot, we would likely use the Multivariate Adaptive Regression Splines (MARS) model to predict the recovery time from COVID-19 illness because it minimizes the mean RMSE over resamples.