ds2_miterm_yl5508

Yifei Liu

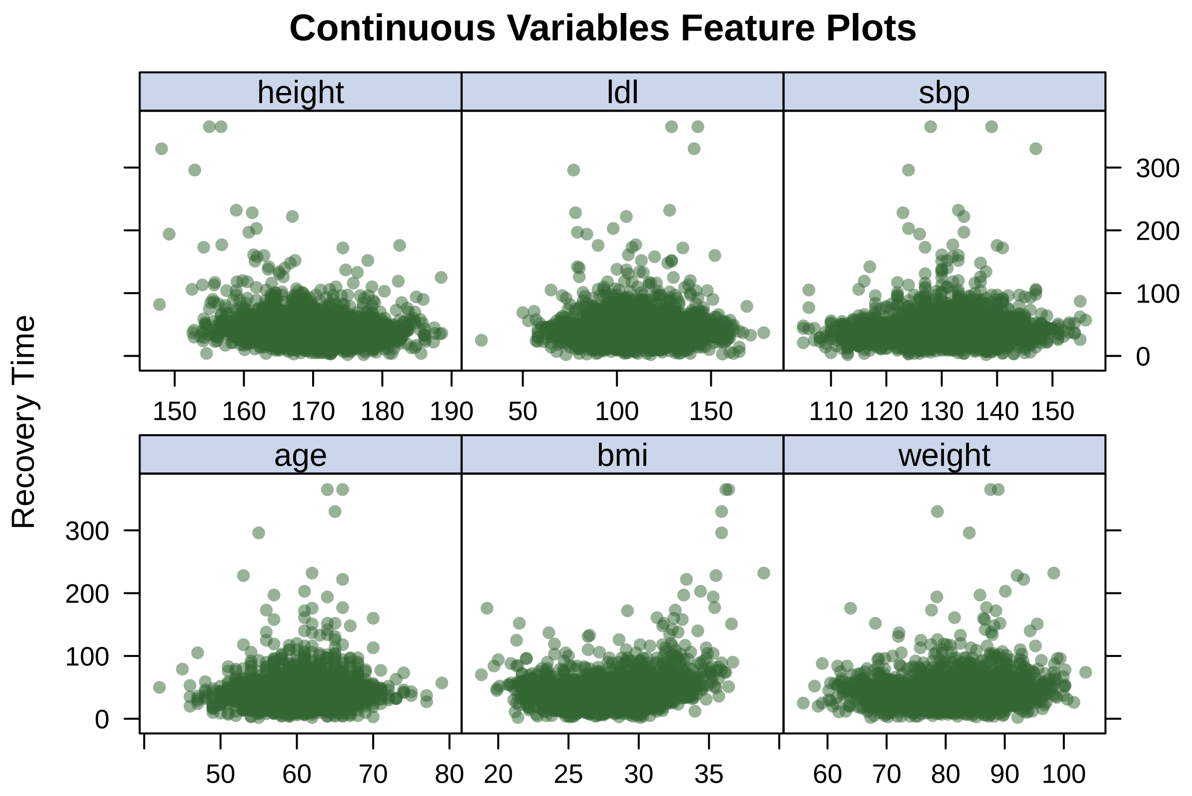
2023/03/22

```r
library(tidyverse)
library(ggridges)
library(corrplot)
library(ggcorrplot)
library(pheatmap)
library(rsample)
library(lattice)
library(caret)
library(pls)
library(rpart)
library(rpart.plot)
```

Data Wrangling

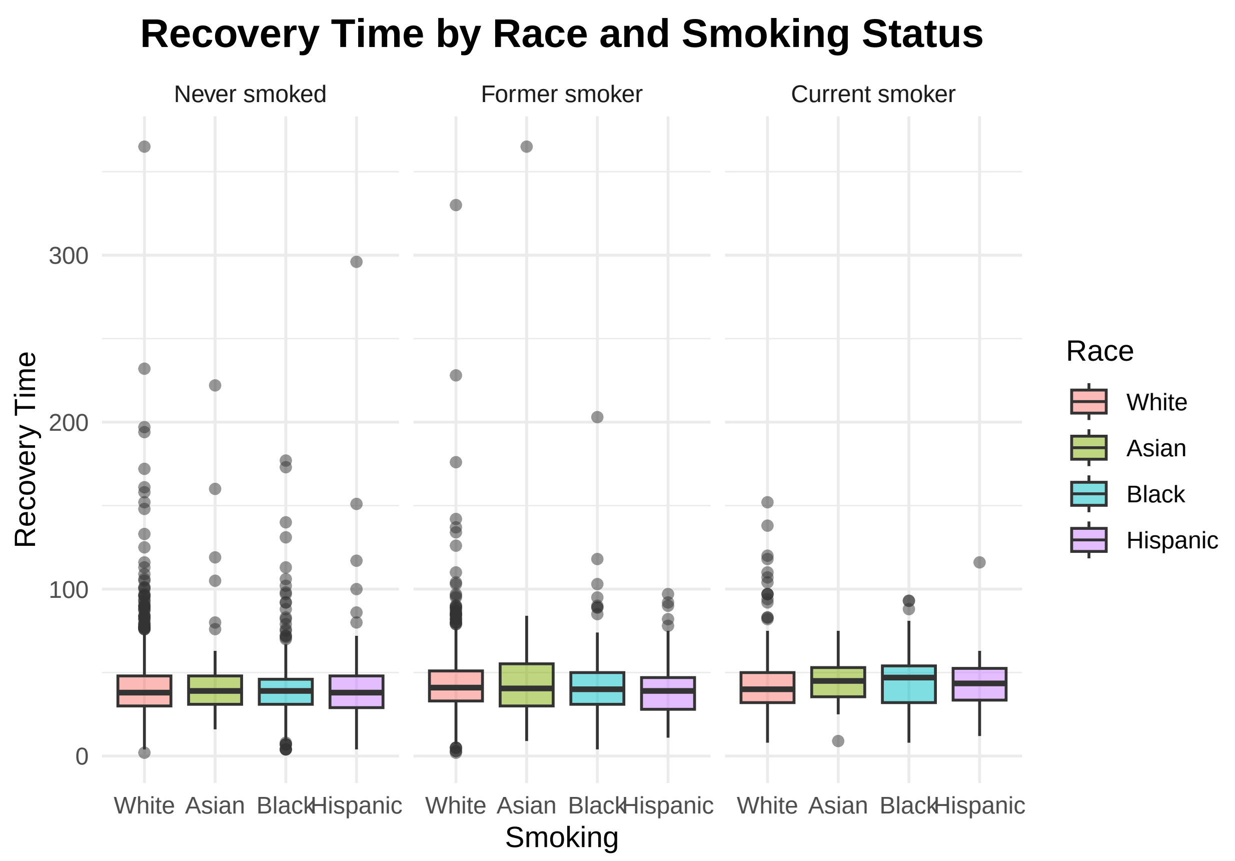
```r
load("recovery.Rdata")

covid = as_tibble(dat) |>
  na.omit() |>
  janitor::clean_names() |>
  mutate(gender = factor(gender),
         hypertension = factor(hypertension),
         diabetes = factor(diabetes),
         vaccine = factor(vaccine),
         severity = factor(severity),
         race = factor(race),
         smoking = factor(smoking)) |>
  select(- id) |>
  relocate(recovery_time)

set.seed(11)
covid_split = initial_split(covid, prop = 0.8)
training = training(covid_split)
testing = testing(covid_split)
xtrain = model.matrix(recovery_time ~ ., training)[,-1]
ytrain = training$recovery_time
xtest = model.matrix(recovery_time ~ ., testing)[,-1]
ytest = testing$recovery_time

# showing connection between the response and other variables
```

```r
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)

par(mar = c(4, 2, 1, 1), mfrow = c(4, 4))
x = model.matrix(recovery_time ~ age + bmi + weight + height + ldl + sbp, covid)[,-1]
y = covid$recovery_time
caret::featurePlot(x, y, plot = "scatter", labels = c("", "Recovery Time"), type = c("p"), layout = c(3
```



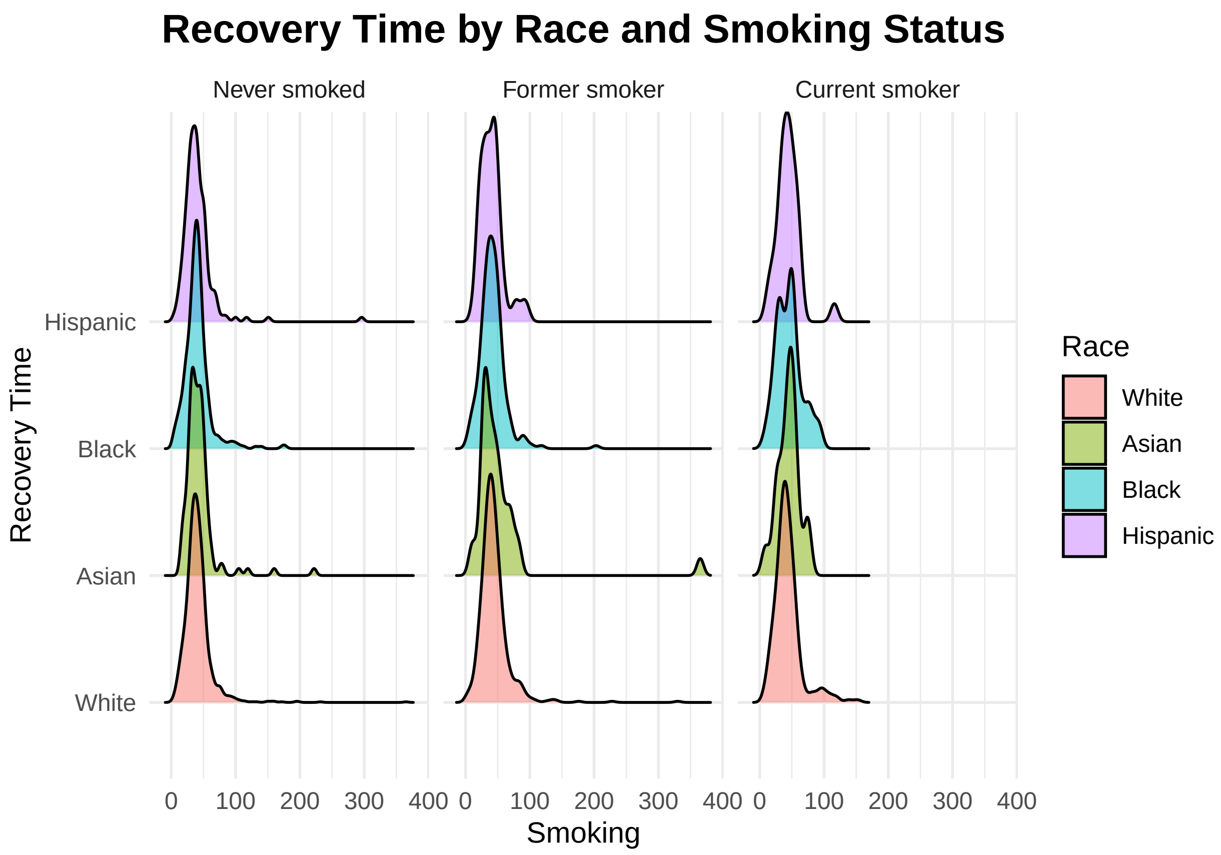
Boxplot of Recovery Time by Smoking Status and Gender

```r
# Boxplot
covid |>
  mutate(race = factor(race, levels = c(1, 2, 3, 4), labels = c("White", "Asian", "Black", "Hispanic"))
         smoking = factor(smoking, levels = c(0, 1, 2), labels = c("Never smoked", "Former smoker", "Cu
  ggplot(aes(x = race, y = recovery_time, fill = race)) +
  geom_boxplot(alpha = 0.5) +
  labs(
    title = "Recovery Time by Race and Smoking Status",
    x = "Smoking",
    y = "Recovery Time"
  ) +
  guides(fill = guide_legend("Race")) +
  theme_minimal() +
  facet_grid(~ smoking) +
  theme(plot.title = element_text(size = 15, face = "bold", hjust = 0.5))
```

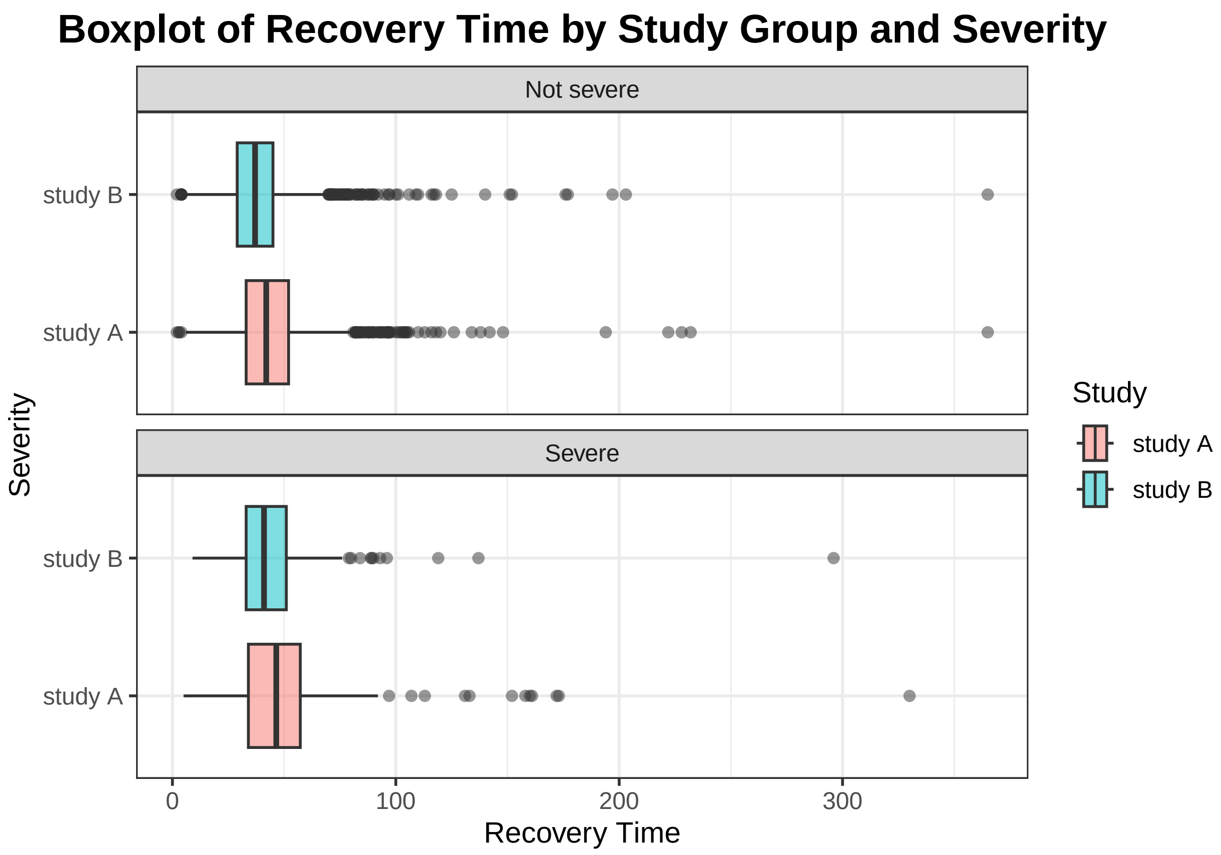


```
# Ridge plots
covid |>
  mutate(race = factor(race, levels = c(1, 2, 3, 4), labels = c("White", "Asian", "Black", "Hispanic")),
         smoking = factor(smoking, levels = c(0, 1, 2), labels = c("Never smoked", "Former smoker", "Cu
  ggplot(aes(y = race, x = recovery_time, fill = race)) +
  geom_density_ridges(alpha = 0.5) +
  labs(
    title = "Recovery Time by Race and Smoking Status",
    x = "Smoking",
    y = "Recovery Time"
  ) +
  guides(fill = guide_legend("Race")) +
  theme_minimal() +
  facet_grid(~ smoking) +
  theme(plot.title = element_text(size = 15, face = "bold", hjust = 0.5))
```

```
## Picking joint bandwidth of 3.79

## Picking joint bandwidth of 5.37

## Picking joint bandwidth of 5.91
```

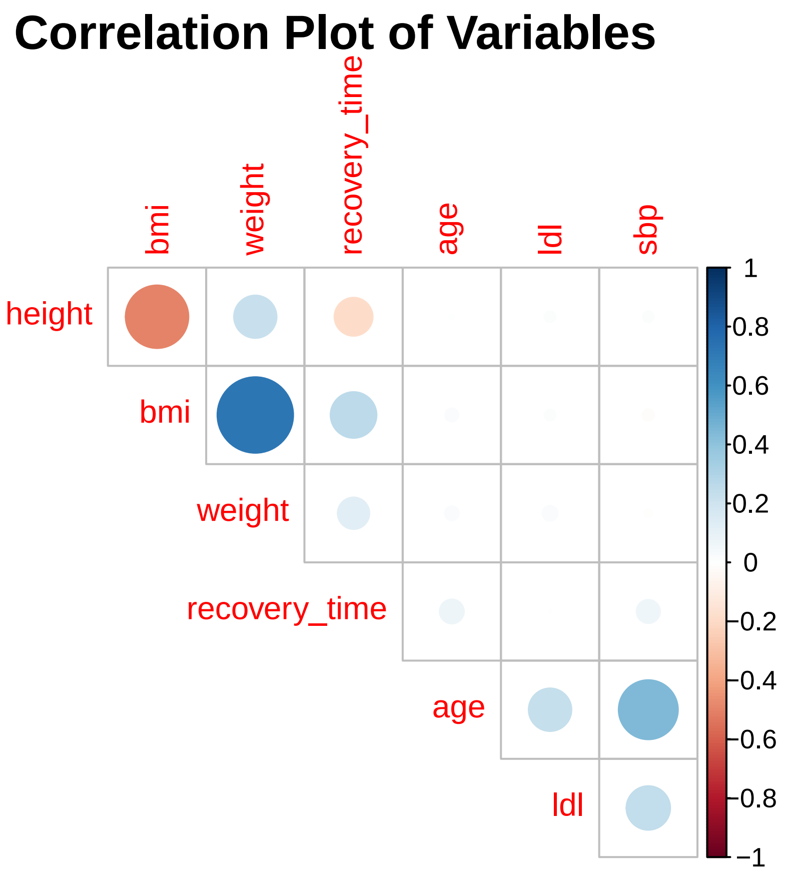


```r
# Boxplot
covid |>
  mutate(study = factor(vaccine, labels = c("study A", "study B")),
         severity = factor(severity, levels = c(0, 1), labels = c("Not severe", "Severe"))) |>
  ggplot(aes(y = study, x = recovery_time, fill = study)) +
  geom_boxplot(alpha = 0.5) +
  labs(
    title = "Boxplot of Recovery Time by Study Group and Severity",
    x = "Recovery Time",
    y = "Severity",
    fill = "Study"
  ) +
  theme_bw() +
  facet_wrap(~ severity, ncol = 1) +
  theme(plot.title = element_text(size = 15, face = "bold", hjust = 0.5))
```



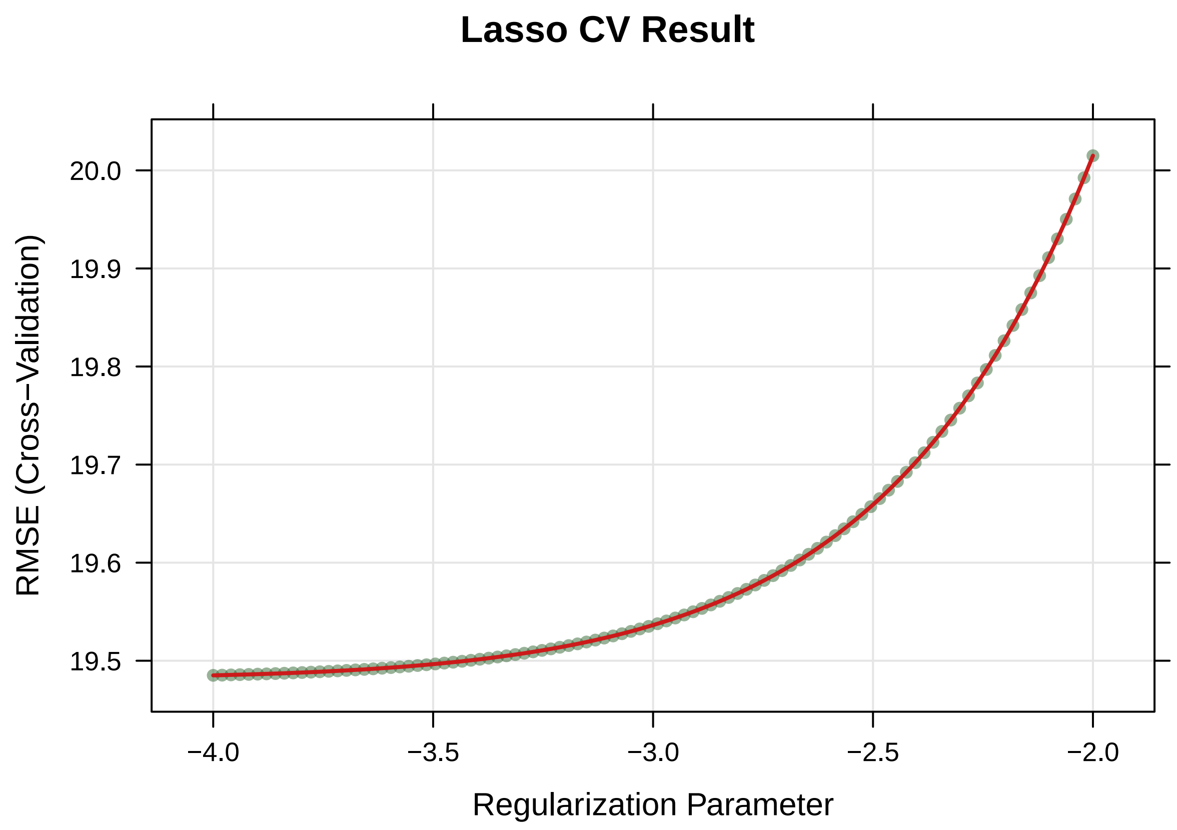
Correlation Plot

```
par(mar = c(1, 1, 1, 1), mfrow=c(1,1))
corrplot::corrplot(cor(covid |> select(recovery_time, age, bmi, weight, height, ldl, sbp)), type = "upp
```



Lasso

```r
set.seed(11)

ctrl = trainControl(method = 'cv', number = 10)
ctrl_1se = trainControl(method = 'cv', number = 10, selectionFunction =  'oneSE')

lasso.fit = train(recovery_time ~ ., data = training,
             method = 'glmnet',
             tuneGrid = expand.grid(alpha = 1,
                                    lambda = exp(seq(-4, -2, length = 100))),
             trControl = ctrl)

plot(lasso.fit, xTrans = log, main = "Lasso CV Result")
```



```
# selected lambda
lasso.fit$bestTune$lambda
```

```
## [1] 0.01831564
```

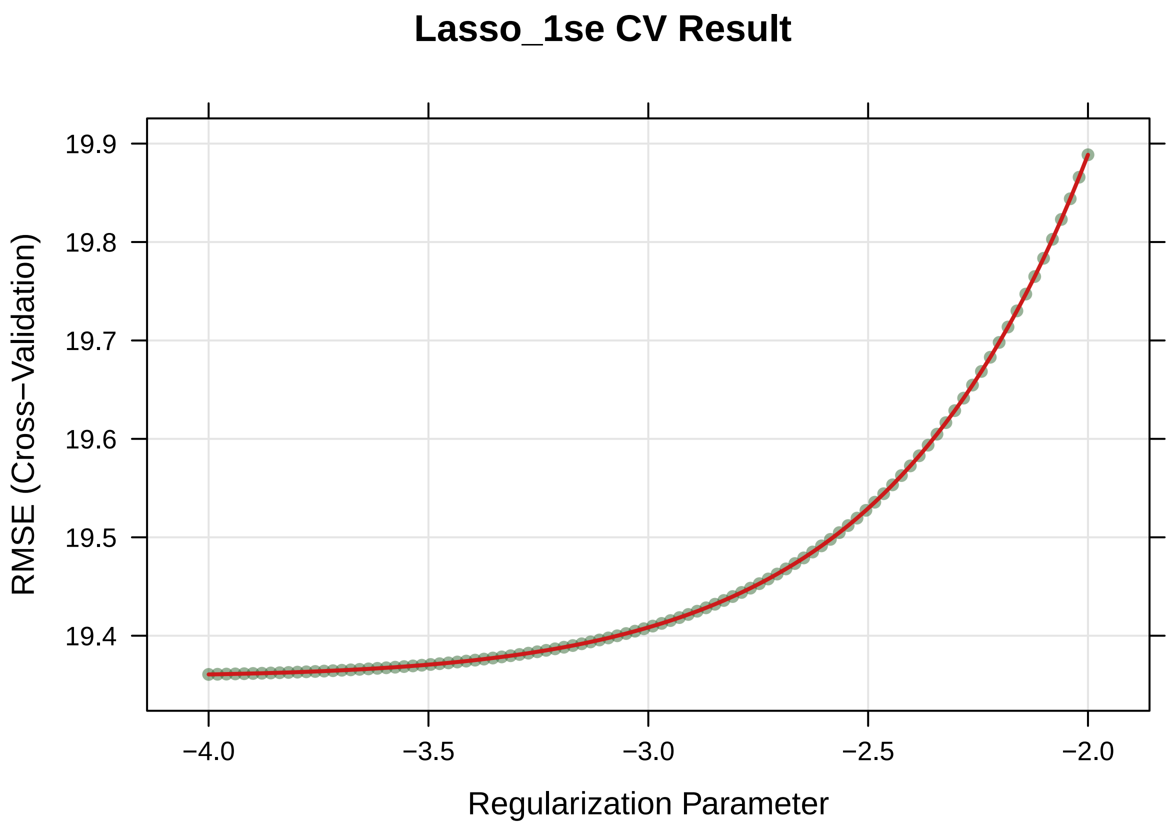
```
# coefficients
coef(lasso.fit$finalModel, s = lasso.fit$bestTune$lambda)
```

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                            s1
## (Intercept)   -1.865676e+03
## age            1.905529e-01
## gender1       -2.272167e+00
## race2          4.069257e+00
## race3         -5.713863e-01
## race4          4.209219e-01
## smoking1       2.232083e+00
## smoking2       4.229749e+00
## height         1.092898e+01
## weight        -1.186704e+01
## bmi            3.571072e+01
## hypertension1  3.389890e+00
## diabetes1     -1.766480e+00
## sbp           -3.790833e-03
## ldl           -2.479261e-02
## vaccine1      -6.318510e+00
## severity1      9.121546e+00
## studyB         4.617454e+00
```

```
# num of predictors
sum(lasso.fit$coefname != 0)
```

```
## [1] 17
```

```
# applying 1se rule
lasso.fit.1se = train(recovery_time ~ ., data = training,
             method = 'glmnet',
             tuneGrid = expand.grid(alpha = 1,
                                    lambda = exp(seq(-4, -2, length = 100))),
             trControl = ctrl_1se)

plot(lasso.fit.1se, xTrans = log, main = "Lasso_1se CV Result")
```



```
# selected alpha and lambda
lasso.fit.1se$bestTune$lambda
```

```
## [1] 0.1353353
```

```
# coefficients
coef(lasso.fit.1se$finalModel, s = lasso.fit.1se$bestTune$lambda)
```

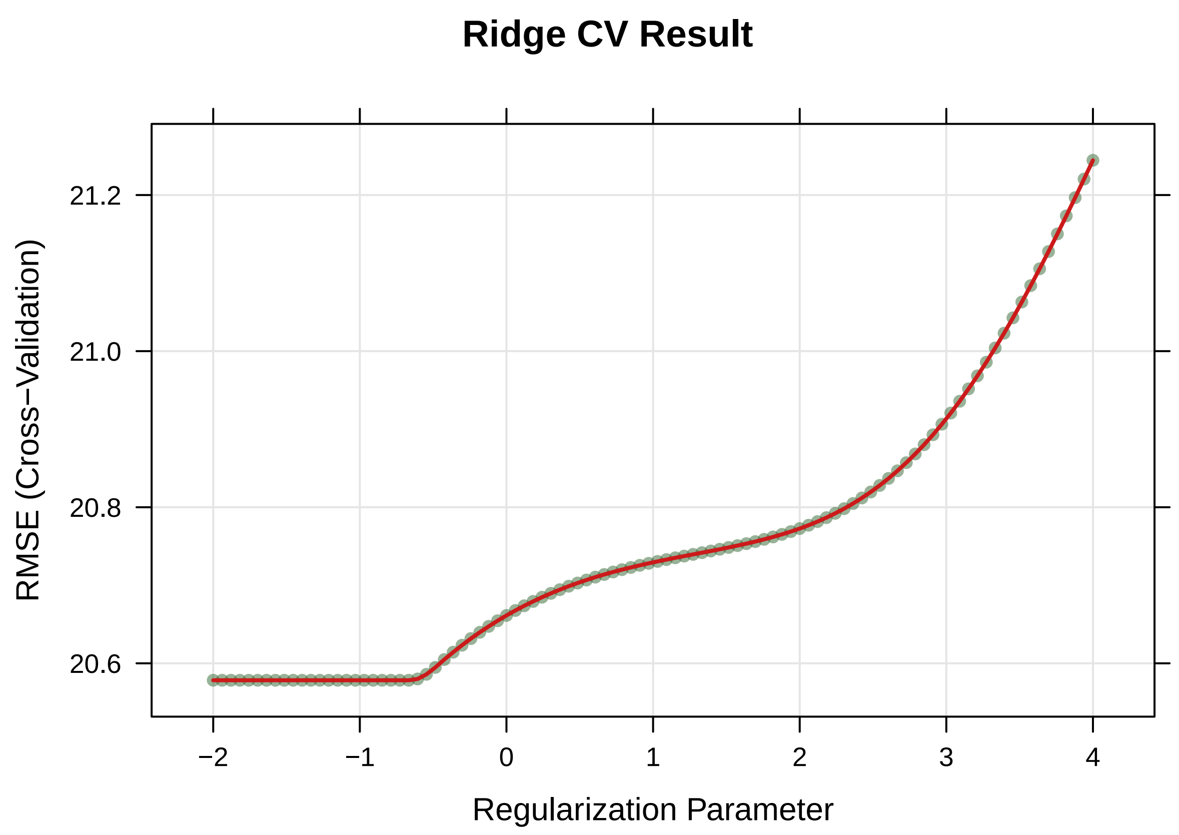
```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                          s1
## (Intercept)   -701.59097625
## age              0.17526842
## gender1         -1.97737691
## race2            3.66429805
## race3           -0.30954075
## race4            .
## smoking1         1.78688961
## smoking2         3.66858314
## height           4.06778665
## weight          -4.60404541
## bmi             14.82843804
## hypertension1    3.20654064
## diabetes1       -1.58783220
## sbp              .
## ldl             -0.01269455
## vaccine1        -6.41275572
## severity1        8.67268705
## studyB           4.55986643
```

```
# num of predictors
sum(lasso.fit.1se$coefname != 0)
```

```
## [1] 17
```

Ridge

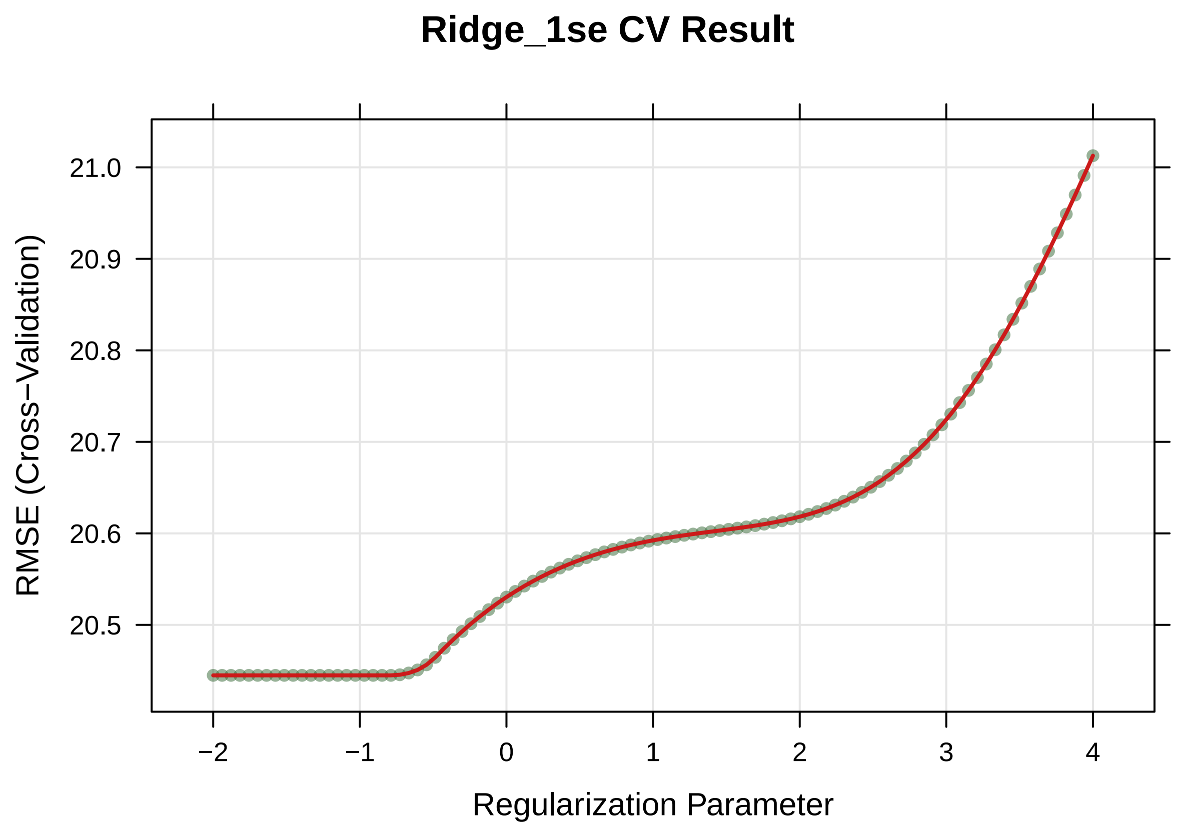
```
set.seed(11)

ctrl = trainControl(method = 'cv', number = 10)
ctrl_1se = trainControl(method = 'cv', number = 10, selectionFunction =  'oneSE')

ridge.fit = train(recovery_time ~ ., data = training,
             method = 'glmnet',
             tuneGrid = expand.grid(alpha = 0,
                                    lambda = exp(seq(-2, 4, length = 100))),
             trControl = ctrl)

plot(ridge.fit, xTrans = log, main = "Ridge CV Result")
```



```
# selected lambda
ridge.fit$bestTune$lambda
```

```
## [1] 0.5134171
```

```
# applying 1se rule
ridge.fit.1se = train(recovery_time ~ ., data = training,
             method = 'glmnet',
             tuneGrid = expand.grid(alpha = 0,
                                    lambda = exp(seq(-2, 4, length = 100))),
             trControl = ctrl_1se)

plot(ridge.fit.1se, xTrans = log, main = "Ridge_1se CV Result")
```

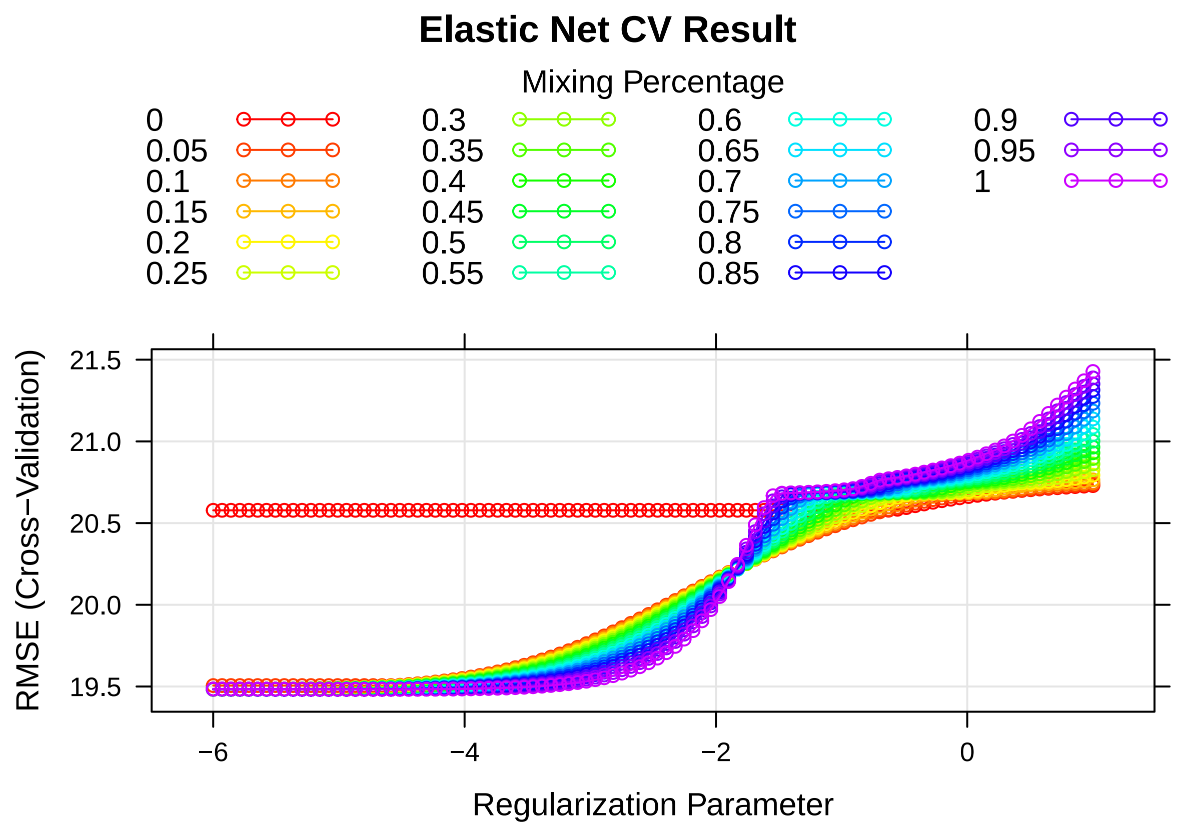


```r
# selected alpha and lambda
ridge.fit.1se$bestTune$lambda
```

```
## [1] 54.59815
```

Elastic Net

```r
set.seed(11)
ctrl = trainControl(method = 'cv', number = 10)
ctrl_1se = trainControl(method = 'cv', number = 10, selectionFunction =  'oneSE')

enet.fit = train(recovery_time ~ ., data = training,
             method = 'glmnet',
             tuneGrid = expand.grid(alpha = seq(0, 1, length = 21),
                                    lambda = exp(seq(-6, 1, length = 100))),
             trControl = ctrl)

myCol = rainbow(25)
myPar = list(superpose.symbol = list(col = myCol), superpose.line = list(col = myCol))

plot(enet.fit, par.settings = myPar, xTrans = log, main = "Elastic Net CV Result")
```



```
# selected alpha and lambda
enet.fit$bestTune
```

```
##     alpha      lambda
## 401   0.2 0.002478752
```

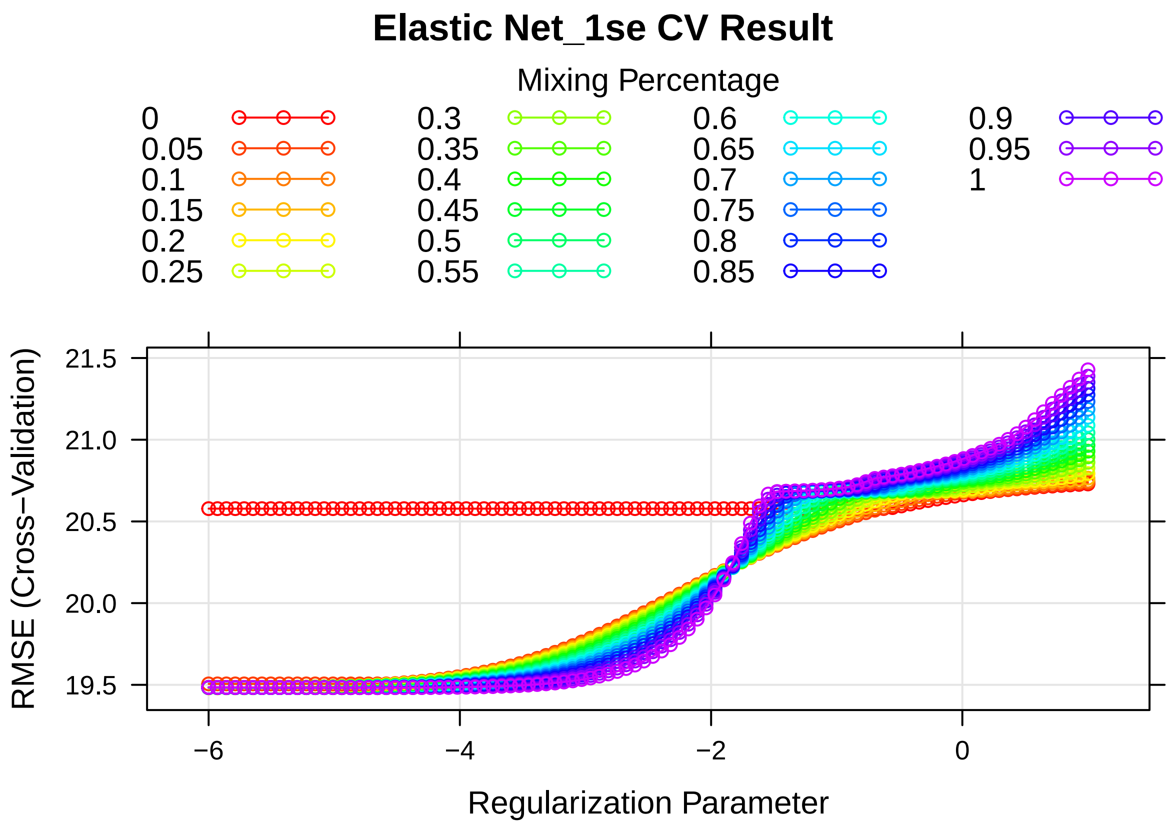
```
# coefficients
coef(enet.fit$finalModel, s = enet.fit$bestTune$lambda)
```

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                          s1
## (Intercept)   -1.952199e+03
## age            1.972764e-01
## gender1       -2.306583e+00
## race2          4.139347e+00
## race3         -6.070652e-01
## race4          5.040683e-01
## smoking1       2.287723e+00
## smoking2       4.307013e+00
## height         1.144672e+01
## weight        -1.241459e+01
## bmi            3.728410e+01
## hypertension1  3.568559e+00
## diabetes1     -1.802445e+00
## sbp           -1.688603e-02
## ldl           -2.603621e-02
## vaccine1       -6.333775e+00
## severity1       9.177936e+00
## studyB          4.637170e+00
```

```
# num of predictors
sum(enet.fit$coefname != 0)
```

```
## [1] 17
```

```
# applying 1se rule
set.seed(11)
enet.fit.1se = train(recovery_time ~ ., data = training,
             method = 'glmnet',
             tuneGrid = expand.grid(alpha = seq(0, 1, length = 21),
                                    lambda = exp(seq(-6, 1, length = 100))),
             trControl = ctrl_1se)

plot(enet.fit.1se, par.settings = myPar, xTrans = log, main = "Elastic Net_1se CV Result")
```

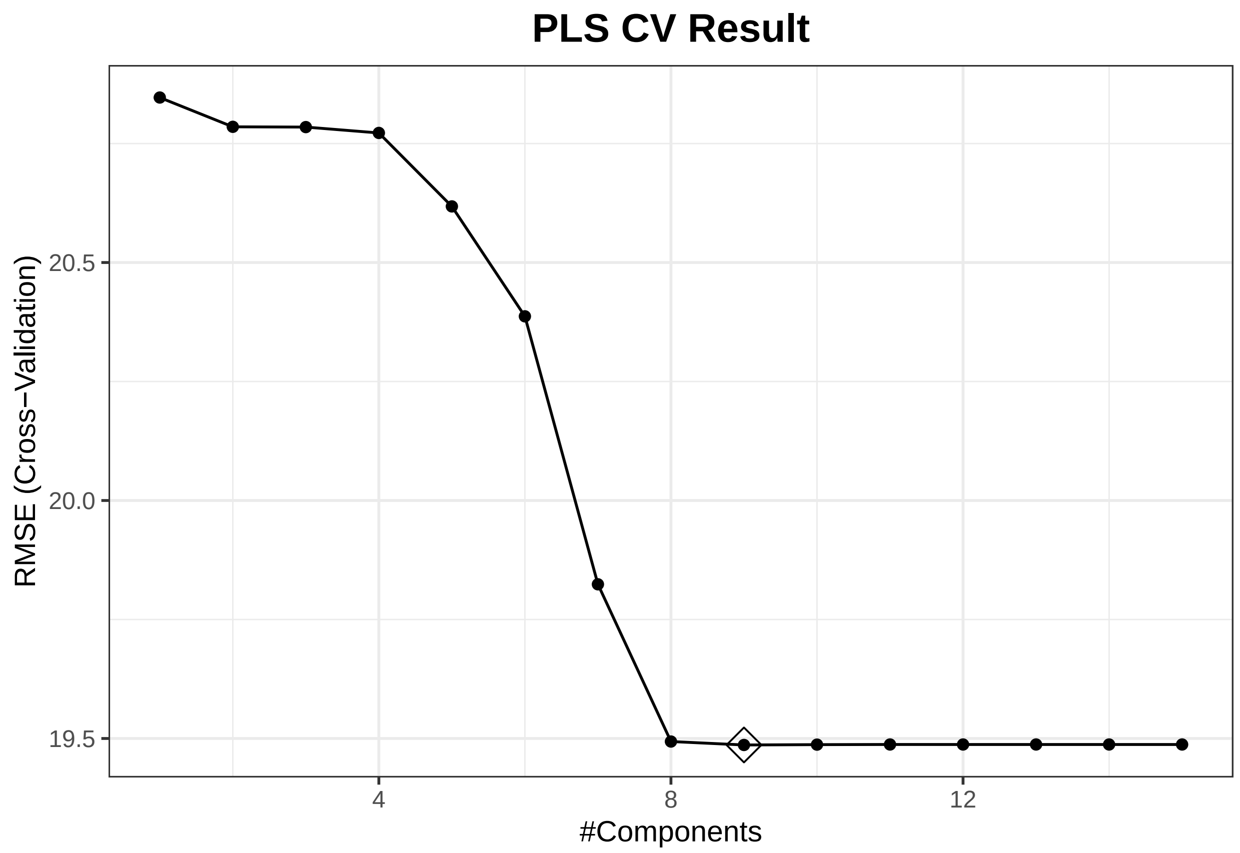


```
# selected alpha and lambda
enet.fit.1se$bestTune
```

```
##     alpha    lambda
## 170  0.05 0.3258845
```

PLS

```r
set.seed(11)

#pls = plsr(recovery_time ~ ., data = training, scale = TRUE, validation = 'CV')
#summary(pls)
#validationplot(pls, val.type = 'MSEP', legendpos = 'topright')
#cv.mse = RMSEP(pls)
#ncomp.cv = which.min(cv.mse$val[1,,]) - 1

pls.fit = train(recovery_time ~ ., data = training,
                method = "pls",
                tuneGrid = data.frame(ncomp = 1:15),
                trControl = ctrl,
                preProcess = c("center", "scale"))

ggplot(pls.fit, highlight = TRUE) +
  theme_bw() +
  labs(title = "PLS CV Result") +
  theme(plot.title = element_text(size = 15, face = "bold", hjust = 0.5))
```



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

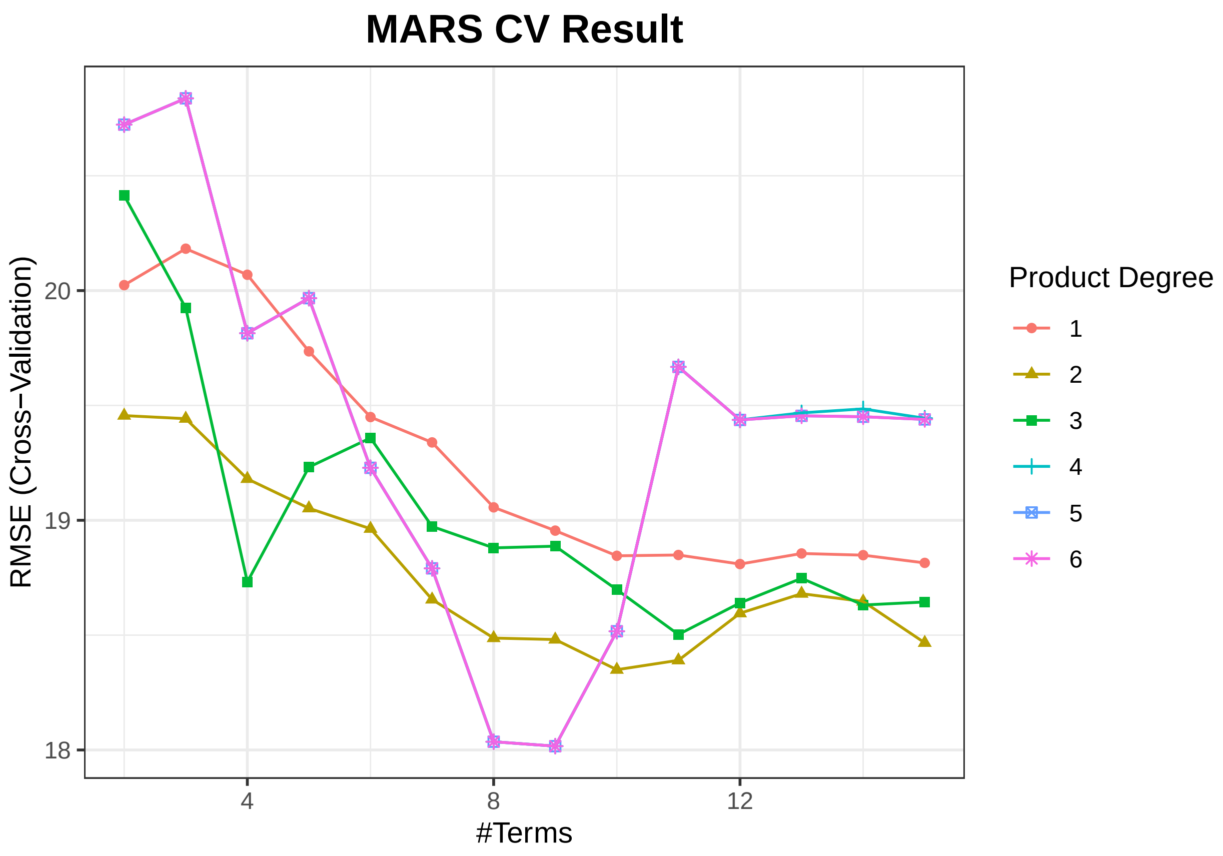
```
## Data:     X dimension: 2400 17
##   Y dimension: 2400 1
## Fit method: oscorespls
## Number of components considered: 9
## TRAINING: % variance explained
##           1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps
## X           9.704    17.88    28.92    34.88    38.00    42.19    44.12
## .outcome   12.363    13.29    13.38    13.62    14.58    15.86    22.82
##           8 comps  9 comps
## X           48.71    54.05
## .outcome    25.05    25.10
```

```
pls.fit$bestTune
```

```
##   ncomp
## 9     9
```

MARS

```
set.seed(11)
mars_grid = expand.grid(degree = 1:6, nprune = 2:15)
ctrl = trainControl(method = 'cv', number = 10)

mars.fit = train(xtrain, ytrain,
                 method = "earth",
                 tuneGrid = mars_grid,
                 trControl = ctrl)

ggplot(mars.fit) +
  theme_bw() +
  labs(title = "MARS CV Result") +
  theme(plot.title = element_text(size = 15, face = "bold", hjust = 0.5))
```



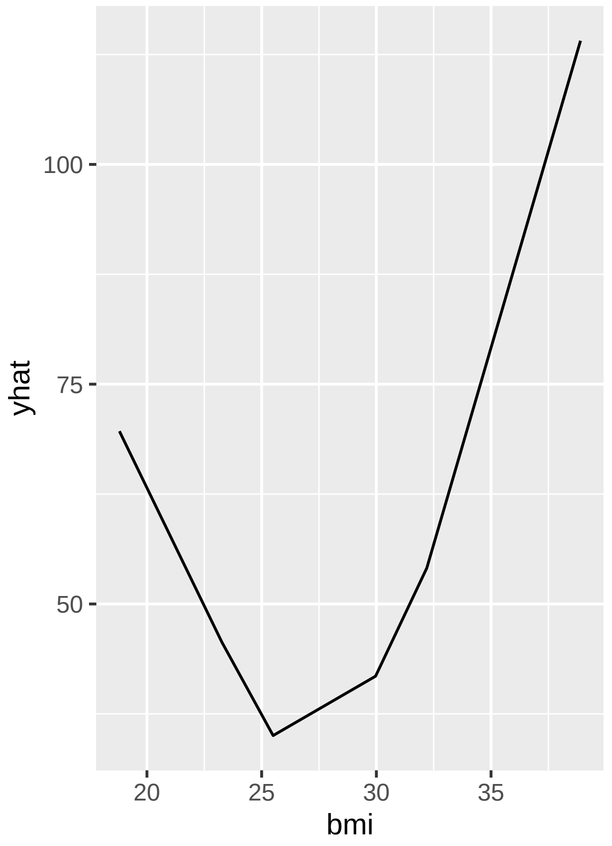
```
# fit of the model
mars.fit$bestTune
```

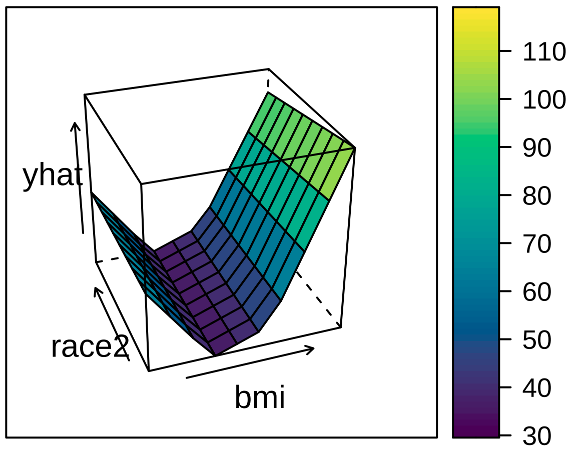
```
##    nprune degree
## 50      9      4
```

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

```
##                          (Intercept)                            h(31-bmi)
##                            6.9166504                            5.3725588
## h(161.6-height) * h(bmi-31) * studyB                          h(bmi-25.3)
##                            2.9896206                            6.8844160
##                             vaccine1             race2 * h(bmi-31) * studyB
##                           -5.7338813                         -523.1860845
##       h(bmi-31) * h(ldl-88) * studyB      age * race2 * h(bmi-31) * studyB
##                            0.2238751                            8.6160130
##                    severity1 * studyB
##                           18.1026072
```

```
# partial dependence plot (PDP)
p1 = pdp::partial(mars.fit, pred.var = c("bmi"), grid.resolution = 10) |> autoplot()

p2 = pdp::partial(mars.fit, pred.var = c("bmi", "race2"),
grid.resolution = 10) |>
pdp::plotPartial(levelplot = FALSE, zlab = "yhat", drape = TRUE,
screen = list(z = 20, x = -60))

gridExtra::grid.arrange(p1, p2, ncol = 2)
```





GAM

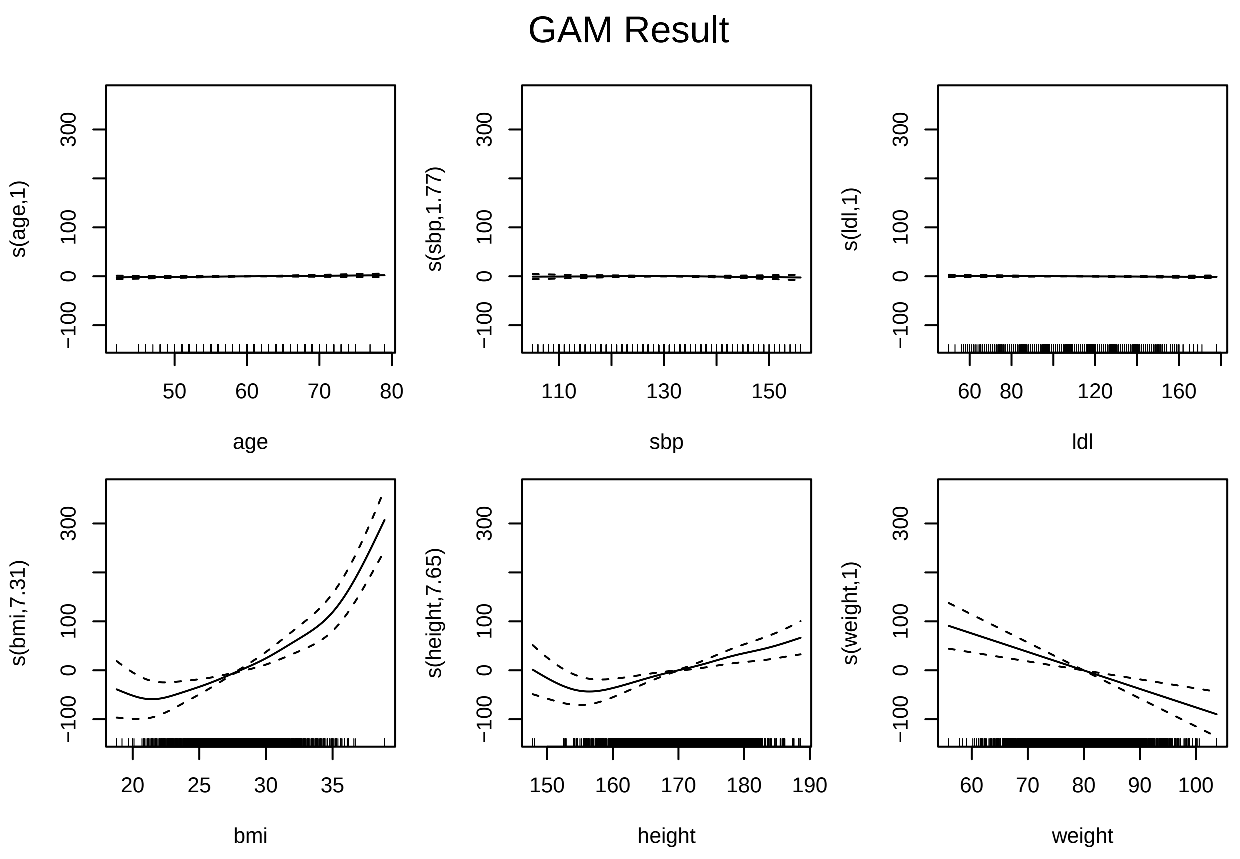
```
set.seed(11)
gam.fit = train(xtrain, ytrain,
                method = "gam",
                trControl = ctrl)
```

```
## Loading required package: mgcv

## Loading required package: nlme

##
## Attaching package: 'nlme'

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

## This is mgcv 1.9-1. For overview type 'help("mgcv-package")'.
```

```r
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:
## 1.00 1.77 1.00 7.31 7.65 1.00  total = 31.73
##
## GCV score: 340.2157
```

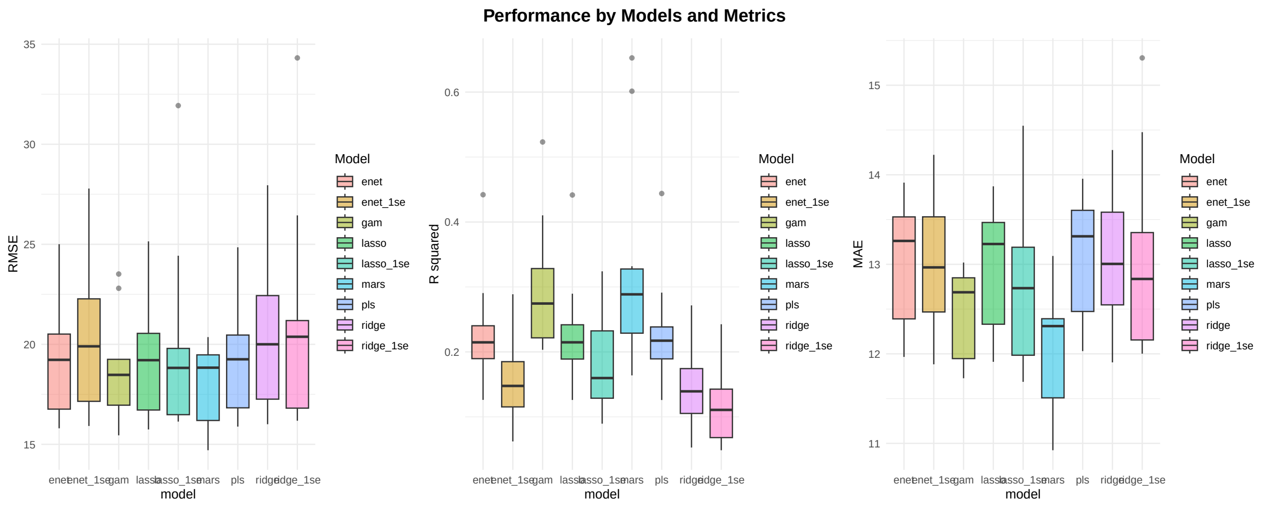
```r
# plot (gam.fit)
par(oma = c(0, 0, 3, 0))
par(mar = c(4, 4, 1, 1), mfrow = c(2, 3))
plot(gam.fit$finalModel)
mtext("GAM Result", side = 3, line = 0.5, outer = TRUE, cex = 1.2)
```



Model Comparation

```r
library(patchwork)
res = resamples(list(lasso = lasso.fit,
                     lasso_1se = lasso.fit.1se,
                     ridge = ridge.fit,
                     ridge_1se = ridge.fit.1se,
                     enet = enet.fit,
                     enet_1se = enet.fit.1se,
                     pls = pls.fit,
                     mars = mars.fit,
                     gam = gam.fit#,
                     #rf = rf.fit,
                     #rf_ctree = ctree.fit
                     ))$value |>
  tibble() |>
  janitor::clean_names() |>
  select(- resample) |>
  pivot_longer(
    everything(),
    names_to = c(".value", "metric"),
    names_pattern = "(.*)_(.*)"
  ) |>
  pivot_longer(c(2:10), names_to = "model", values_to = "result")

plot_rmse = res |>
  filter(metric == "rmse") |>
  ggplot(aes(x = model, y = result, fill = model)) +
  geom_boxplot(alpha = 0.5) +
  labs(y = "RMSE") +
  theme_minimal() +
  guides(fill = guide_legend("Model"))

plot_r_squared = res |>
  filter(metric == "rsquared") |>
  ggplot(aes(x = model, y = result, fill = model)) +
  geom_boxplot(alpha = 0.5) +
  labs(y = "R squared") +
  theme_minimal() +
  guides(fill = guide_legend("Model"))

plot_mae = res |>
  filter(metric == "mae") |>
  ggplot(aes(x = model, y = result, fill = model)) +
  geom_boxplot(alpha = 0.5) +
  labs(y = "MAE") +
  theme_minimal() +
  guides(fill = guide_legend("Model"))

final_plot = plot_rmse + plot_r_squared + plot_mae +
  plot_layout(ncol = 3) +
  plot_annotation(title = "Performance by Models and Metrics",
                  theme = theme(plot.title = element_text(size = 15, face = "bold", hjust = 0.5)))
```

```
final_plot
```

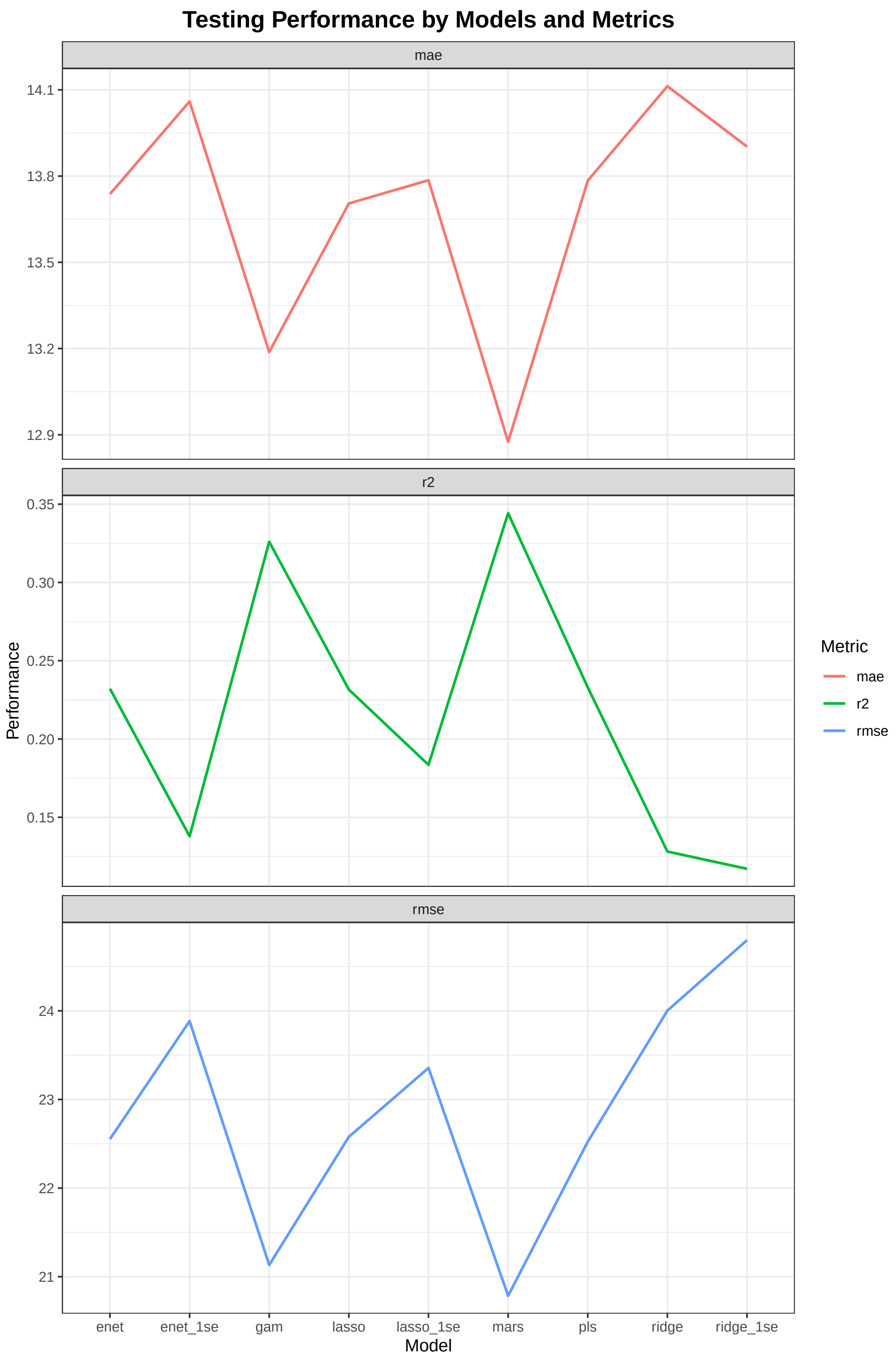


```
# testing data
pred.lasso = predict(lasso.fit, newdata = testing)
perf.lasso = postResample(pred = pred.lasso, obs = testing$recovery_time)
rmse.lasso = sqrt(mean((testing$recovery_time - pred.lasso)^2))
r2.lasso = cor(testing$recovery_time, pred.lasso)^2
mae.lasso = mean(abs(testing$recovery_time - pred.lasso))

pred.lasso.1se = predict(lasso.fit.1se, newdata = testing)
perf.lasso_1se = postResample(pred = pred.lasso.1se, obs = testing$recovery_time)
rmse.lasso_1se = sqrt(mean((testing$recovery_time - pred.lasso.1se)^2))
r2.lasso_1se = cor(testing$recovery_time, pred.lasso.1se)^2
mae.lasso_1se = mean(abs(testing$recovery_time - pred.lasso.1se))

pred.ridge = predict(ridge.fit, newdata = testing)
perf.ridge = postResample(pred = pred.ridge, obs = testing$recovery_time)
rmse.ridge = sqrt(mean((testing$recovery_time - pred.ridge)^2))
r2.ridge = cor(testing$recovery_time, pred.ridge)^2
mae.ridge = mean(abs(testing$recovery_time - pred.ridge))

pred.ridge.1se = predict(ridge.fit.1se, newdata = testing)
perf.ridge_1se = postResample(pred = pred.ridge.1se, obs = testing$recovery_time)
rmse.ridge_1se = sqrt(mean((testing$recovery_time - pred.ridge.1se)^2))
r2.ridge_1se = cor(testing$recovery_time, pred.ridge.1se)^2
mae.ridge_1se = mean(abs(testing$recovery_time - pred.ridge.1se))

pred.enet = predict(enet.fit, newdata = testing)
perf.enet = postResample(pred = pred.enet, obs = testing$recovery_time)
rmse.enet = sqrt(mean((testing$recovery_time - pred.enet)^2))
r2.enet = cor(testing$recovery_time, pred.enet)^2
mae.enet = mean(abs(testing$recovery_time - pred.enet))

pred.enet.1se = predict(enet.fit.1se, newdata = testing)
perf.enet_1se = postResample(pred = pred.enet.1se, obs = testing$recovery_time)
rmse.enet_1se = sqrt(mean((testing$recovery_time - pred.enet.1se)^2))
r2.enet_1se = cor(testing$recovery_time, pred.enet.1se)^2
```

```r
mae.enet_1se = mean(abs(testing$recovery_time - pred.enet.1se))

pred.pls = predict(pls.fit, newdata = testing)
perf.pls = postResample(pred = pred.pls, obs = testing$recovery_time)
rmse.pls = sqrt(mean((testing$recovery_time - pred.pls)^2))
r2.pls = cor(testing$recovery_time, pred.pls)^2
mae.pls = mean(abs(testing$recovery_time - pred.pls))

pred.mars = predict(mars.fit, newdata = xtest)
perf.mars = postResample(pred = pred.mars, obs = testing$recovery_time)
rmse.mars = sqrt(mean((testing$recovery_time - pred.mars)^2))
r2.mars = cor(testing$recovery_time, pred.mars)^2 |> as.numeric()
mae.mars = mean(abs(testing$recovery_time - pred.mars))

pred.gam = predict(gam.fit, newdata = xtest)
perf.gam = postResample(pred = pred.gam, obs = testing$recovery_time)
rmse.gam = sqrt(mean((testing$recovery_time - pred.gam)^2))
r2.gam = cor(testing$recovery_time, pred.gam)^2
mae.gam = mean(abs(testing$recovery_time - pred.gam))

#pred.rf = predict(rf.fit, newdata = xtest)
#perf.rf = postResample(pred = pred.rf, obs = testing$recovery_time)
#rmse.rf = sqrt(mean((testing$recovery_time - pred.rf)^2))
#r2.rf = cor(testing$recovery_time, pred.rf)^2
#mae.rf = mean(abs(testing$recovery_time - pred.rf))

#pred.ctree = predict(ctree.fit, newdata = testing)
#perf.rf.ctree = postResample(pred = pred.ctree, obs = testing$recovery_time)
#rmse.rf_ctree = sqrt(mean((testing$recovery_time - pred.ctree)^2))
#r2.rf_ctree = cor(testing$recovery_time, pred.ctree)^2
#mae.rf_ctree = mean(abs(testing$recovery_time - pred.ctree))

res_test = tibble(
  rmse.lasso, r2.lasso, mae.lasso,
  rmse.lasso_1se, r2.lasso_1se, mae.lasso_1se,
  rmse.ridge, r2.ridge, mae.ridge,
  rmse.ridge_1se, r2.ridge_1se, mae.ridge_1se,
  rmse.enet, r2.enet, mae.enet,
  rmse.enet_1se, r2.enet_1se, mae.enet_1se,
  rmse.pls, r2.pls, mae.pls,
  rmse.mars, r2.mars, mae.mars,
  rmse.gam, r2.gam, mae.gam
#  rmse.rf, r2.rf, mae.rf,
#  rmse.rf_ctree, r2.rf_ctree, mae.rf_ctree
) |>
  pivot_longer(
    everything(),
    names_to = c("metric", "model"),
    names_sep = "\\.",
    values_to = "result"
  )
```

```r
res_test |>
  ggplot(aes(x = model, y = result, group = metric, color = metric)) +
  geom_line(size = 0.8) +
  labs(
    title = "Testing Performance by Models and Metrics",
    x = "Model",
    y = "Performance"
  ) +
  guides(color = guide_legend("Metric")) +
  facet_wrap(~ metric, ncol = 1, scales = "free_y") +
  theme_bw() +
  theme(plot.title = element_text(size = 15, face = "bold", hjust = 0.5))
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



Testing Performance by Models and Metrics
mae
14.1
13.8
13.5
13.2
12.9
r2
0.35
0.30
0.25
0.20
0.15
rmse
24
23
22
21
Performance
Model
enet
enet_1se
gam
lasso
lasso_1se
mars
pls
ridge
ridge_1se
Metric
mae
r2
rmse