

Midterm project

Tingyi Li

Contents

Data Partition	2
EDA	2
Feature plot for continuous variables	3
Correlation plot	4
Violin plot for categorical variables	5
Linear models	14
Fit a lasso model	14
Fit an elastic net model	16
Fit a PCR model	18
Nonlinear models	19
MARS	19
GAM	20
Model comparison	24

```
library(ISLR)
library(caret)
library(tidymodels)
library(pls)
library(earth)
library(ggplot2)
library(patchwork)
library(corrplot)
library(gtsummary)
```

```
load("./recovery.RData")
data <- subset(dat, select = -id)
data$study <- ifelse(data$study == "A", 0, 1)

data <- data |>
  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)
```

)

Data Partition

Divide data into training data (80%) and testing data (20%)

```
set.seed(666)
data_split <- initial_split(data, prop = 0.8)

# training data
training_data <- training(data_split)
training_data <- na.omit(training_data)

# testing data
testing_data <- testing(data_split)
testing_data <- na.omit(testing_data)

# matrix of predictors
x <- model.matrix(recovery_time ~ ., training_data) [, -1]
y <- training_data$recovery_time

x2 <- model.matrix(recovery_time ~ ., testing_data) [, -1]
y2 <- testing_data$recovery_time

# 10-fold cross validation
ctrl1 <- trainControl(method = "repeatedcv", number = 10)
```

EDA

```
skimr::skim(data)
```

Table 1: Data summary

Name	data
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

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
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	0: 2000, 1: 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	

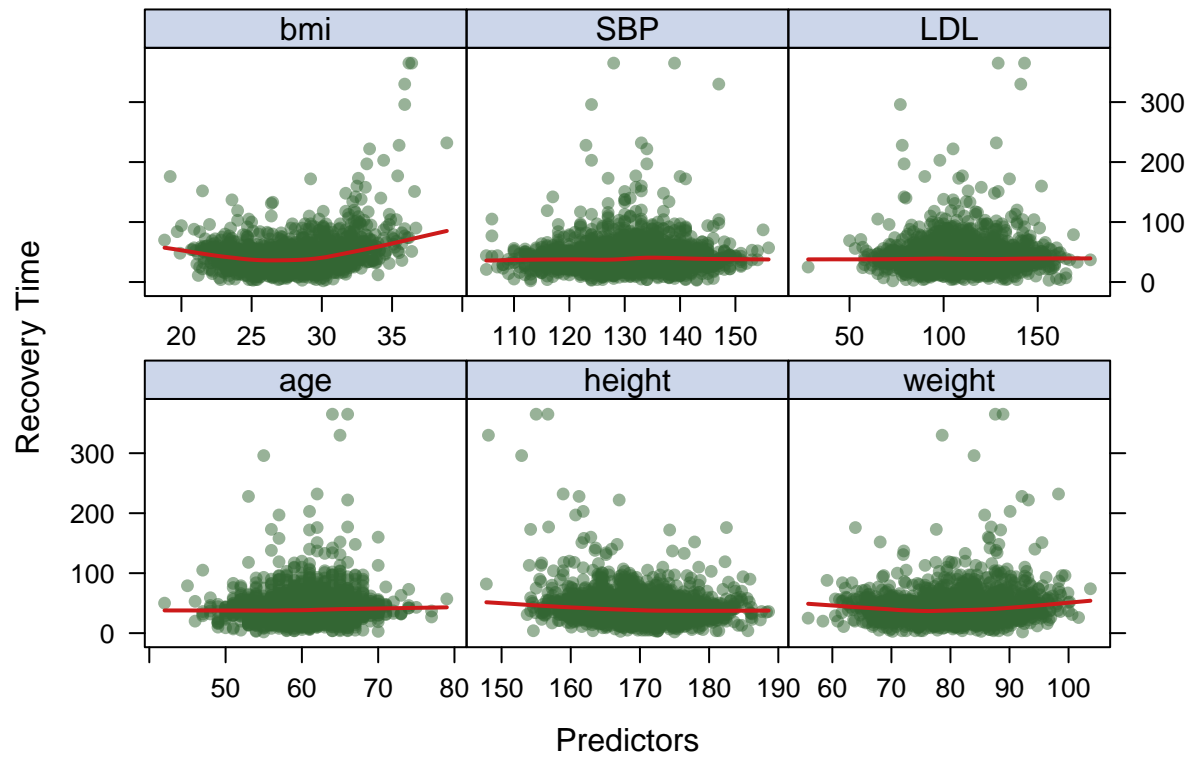
Feature plot for continuous variables

```

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)

trainData <- training_data|>
  dplyr::select('age', 'height', 'weight', 'bmi', 'SBP', 'LDL', 'recovery_time')
featurePlot(x = trainData[, 1:6],
            y = trainData[, 7],
            plot = "scatter",
            span = .5,
            labels = c("Predictors", "Recovery Time"),
            main = "Figure 1: Relationship between Continuous Predictors and Recovery Time",
            type = c("p", "smooth"),
            layout = c(3, 2))

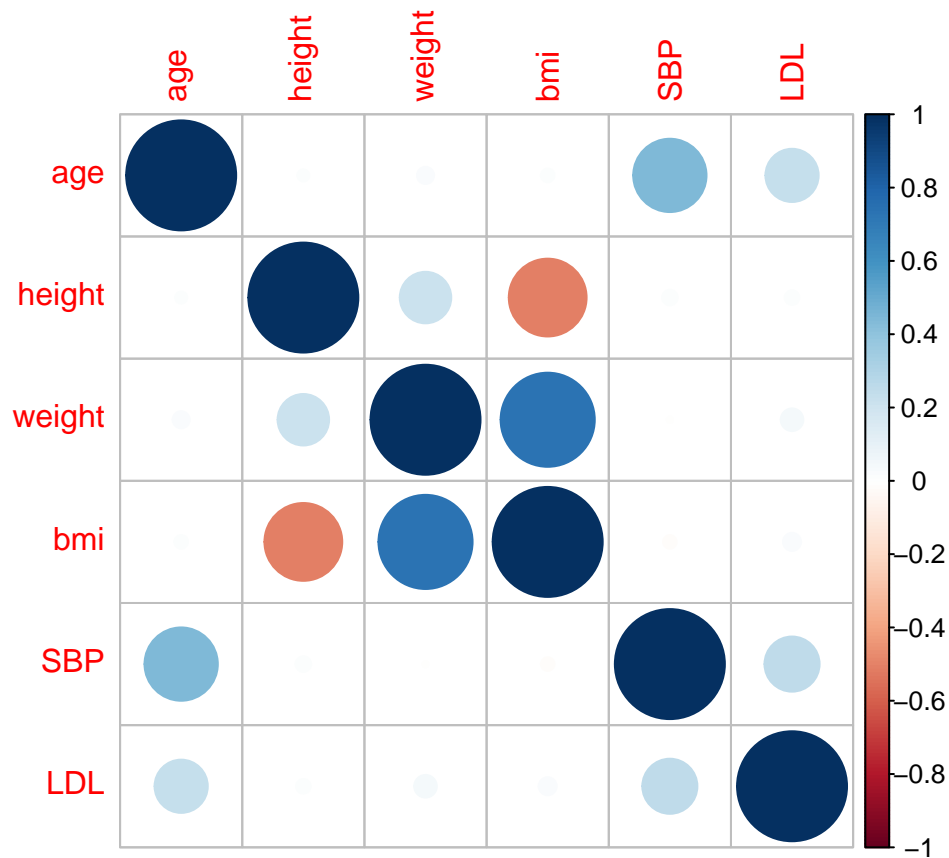
```

Figure 1: Relationship between Continuous Predictors and Recovery Time

Correlation plot

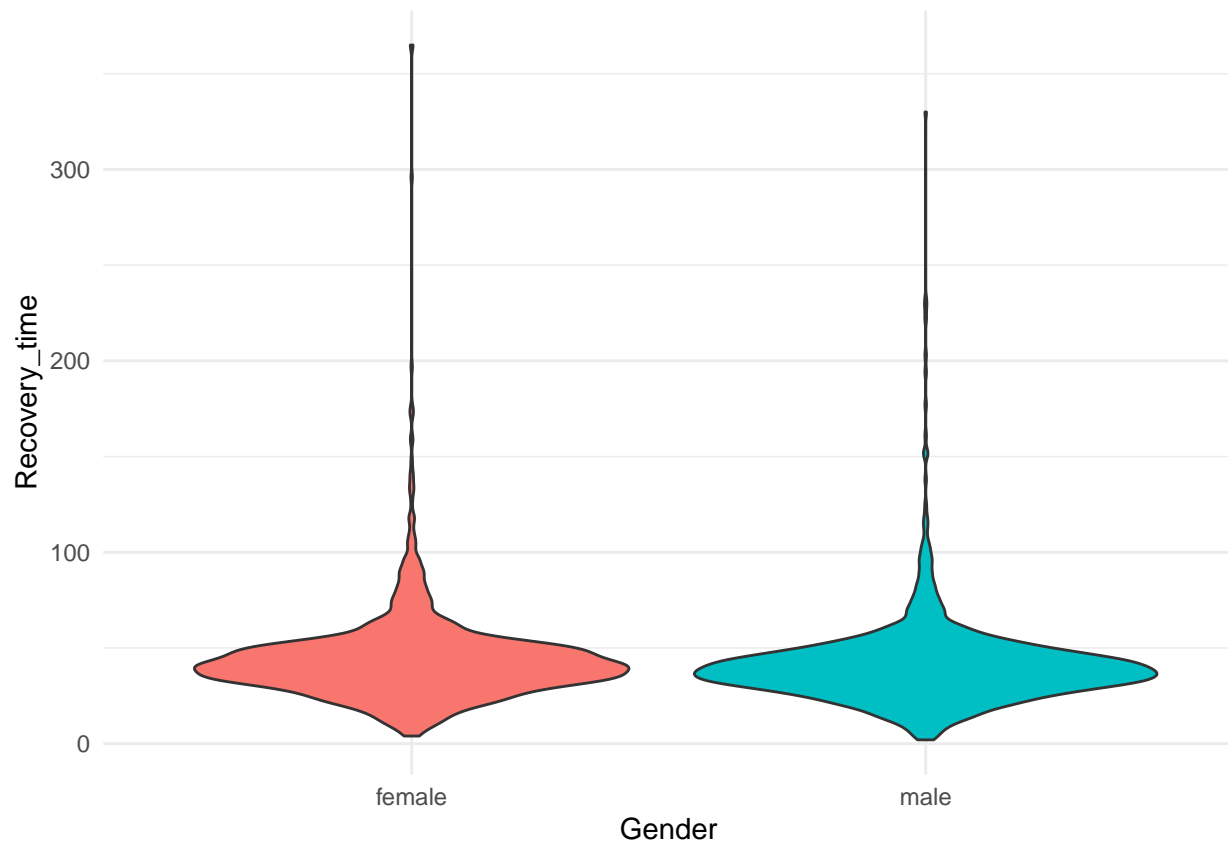
```
corr_data = training_data |>
  dplyr::select('age', 'height', 'weight', 'bmi', 'SBP', 'LDL')

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

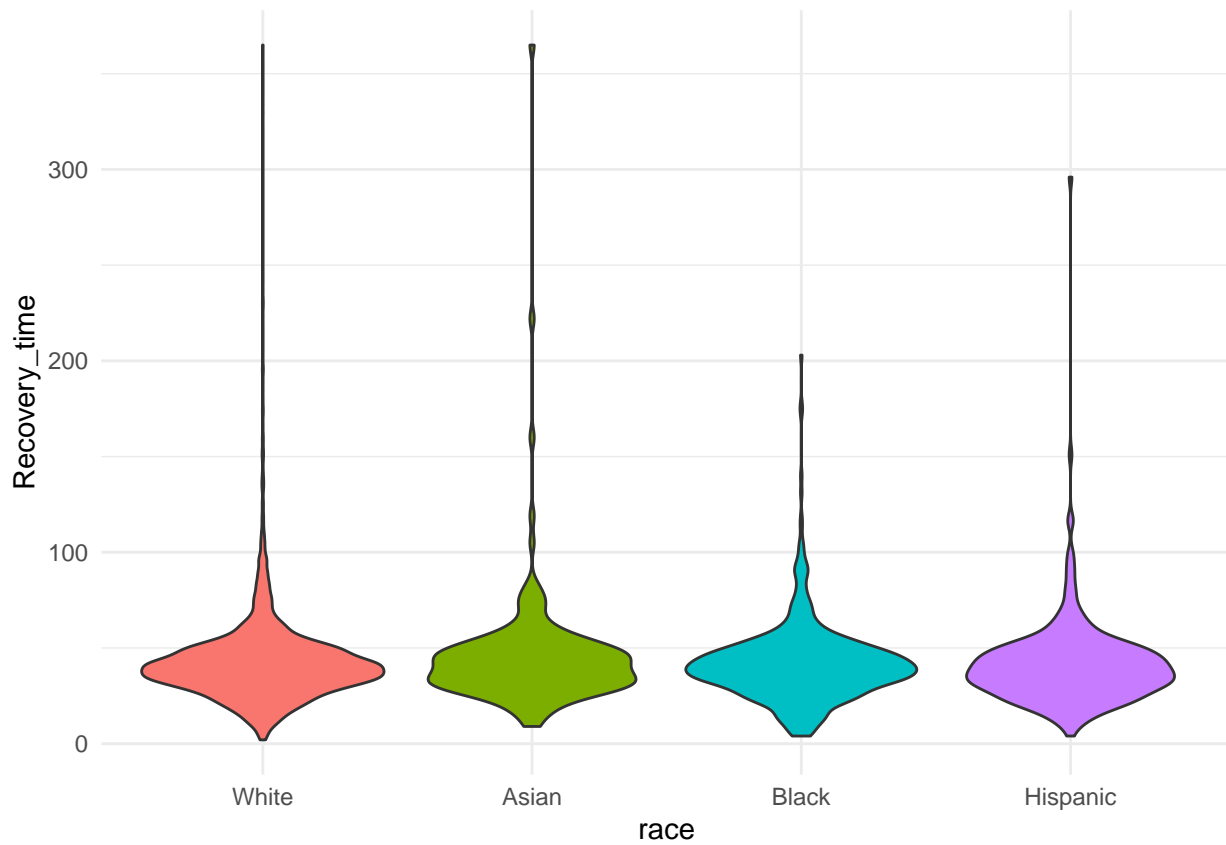


Violin plot for categorical variables

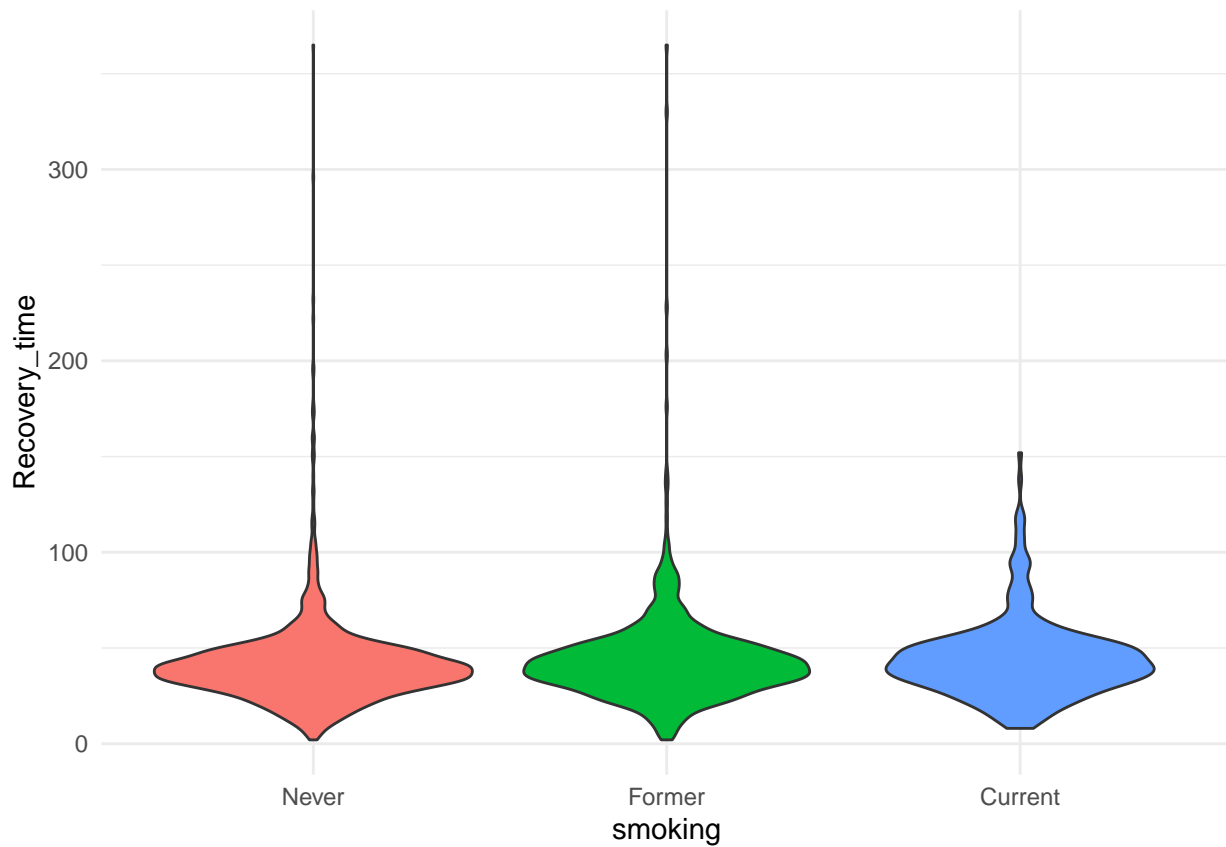
```
gender_plot <- data |>
  ggplot(aes(x = gender, y = recovery_time, fill = gender))+
  geom_violin() +
  scale_x_discrete(labels = c("female", "male")) +
  labs(
    x = "Gender",
    y = "Recovery_time") +
  theme_minimal() + theme(legend.position = "none")
gender_plot
```



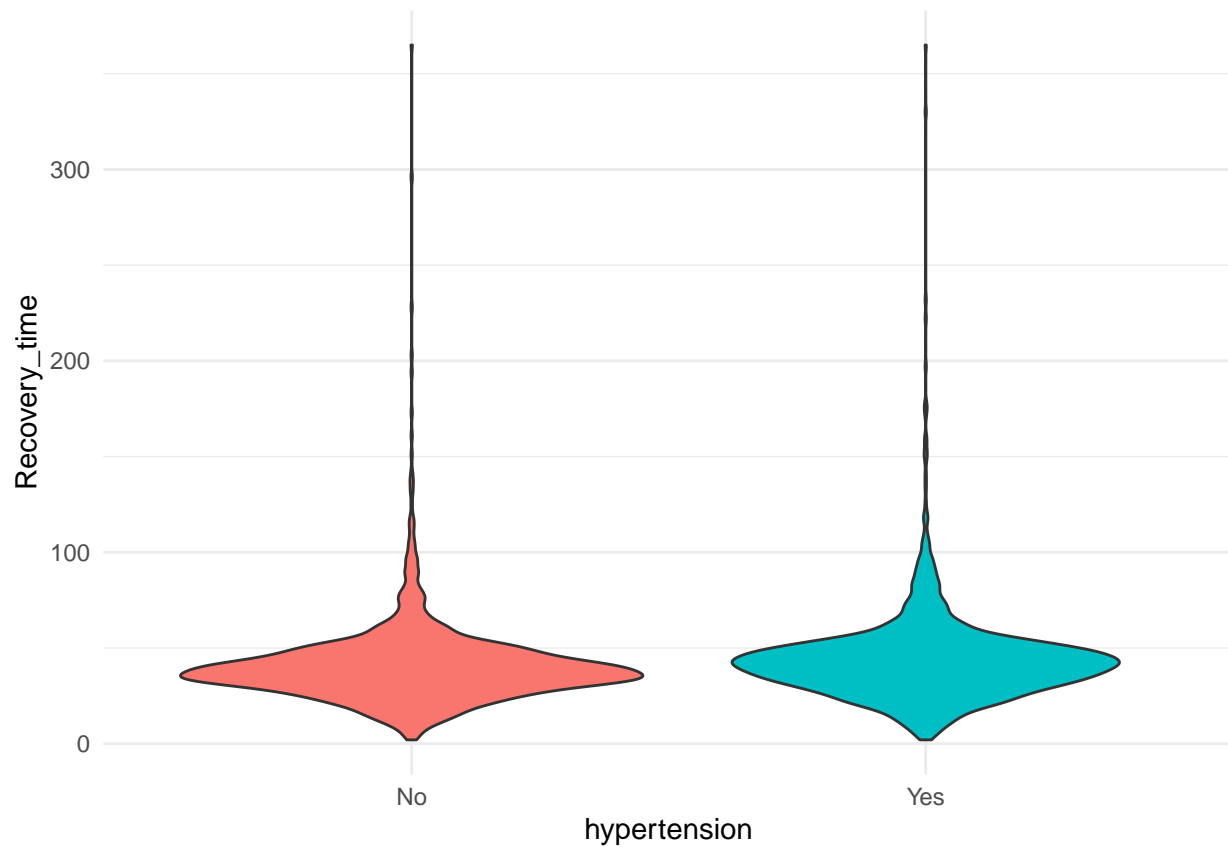
```
race_plot <- data |>
  ggplot(aes(x = race, y = recovery_time, fill = race)) +
  geom_violin() +
  scale_x_discrete(labels = c("White", "Asian", "Black", "Hispanic")) +
  labs(
    x = "race",
    y = "Recovery_time") +
  theme_minimal() + theme(legend.position = "none")
race_plot
```



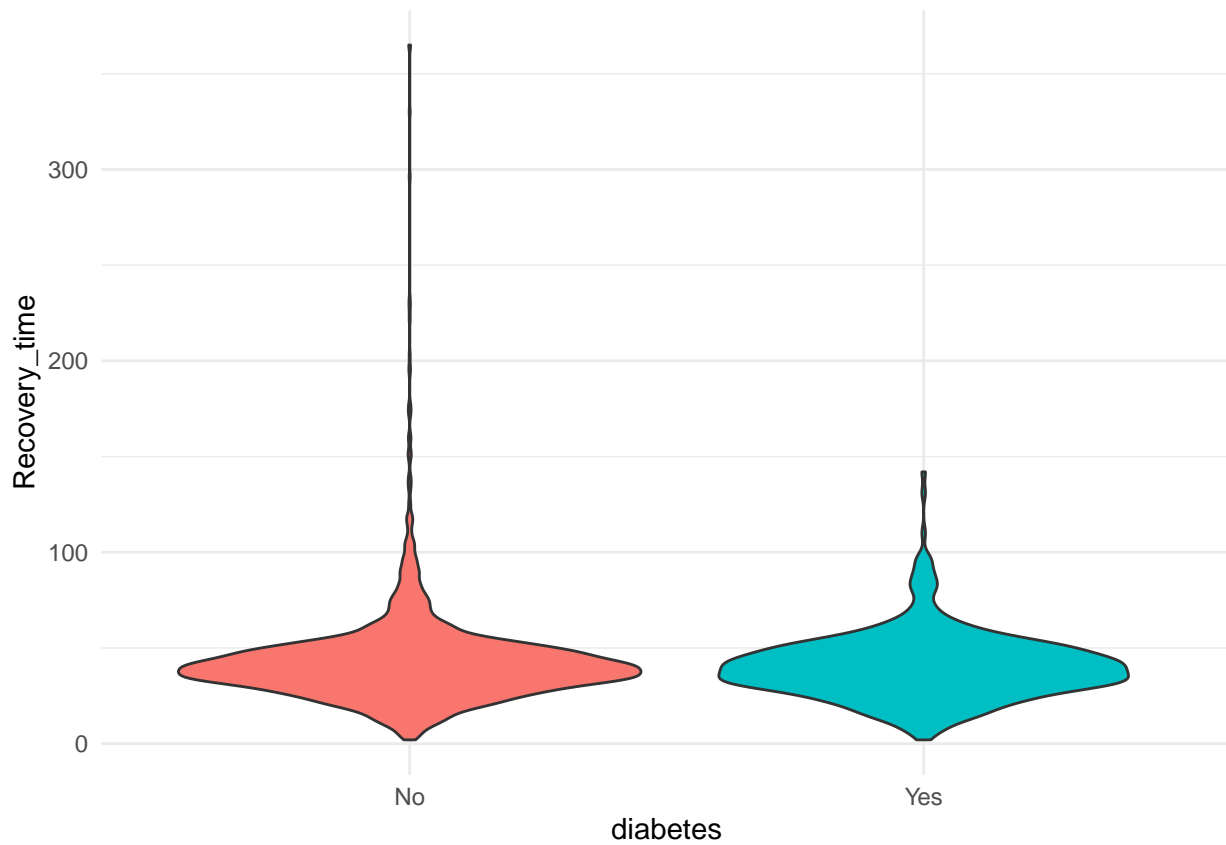
```
smoking_plot <- data |>
  ggplot(aes(x = smoking, y = recovery_time, fill = smoking)) +
  geom_violin() +
  scale_x_discrete(labels = c("Never", "Former", "Current")) +
  labs(
    x = "smoking",
    y = "Recovery_time") +
  theme_minimal() + theme(legend.position = "none")
smoking_plot
```



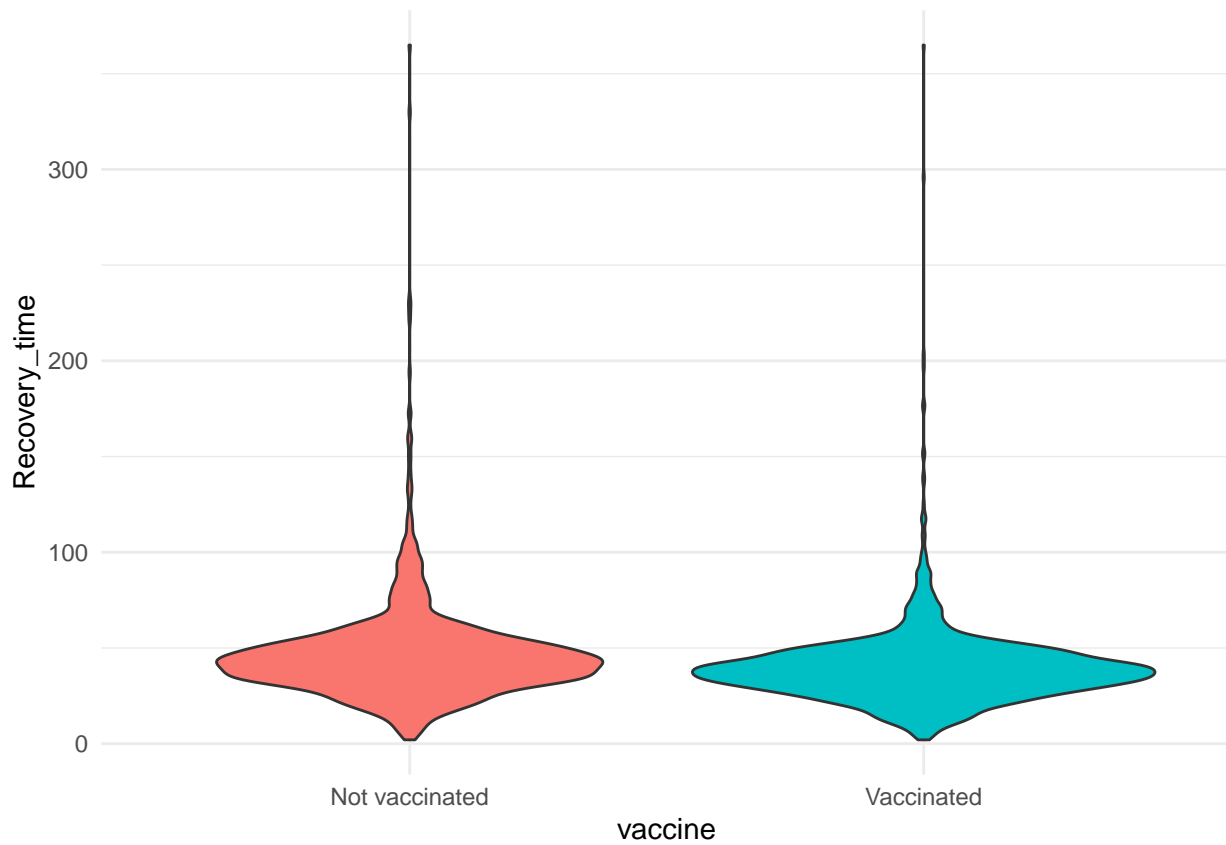
```
hypertension_plot <- data |>  
  ggplot(aes(x = hypertension, y = recovery_time, fill = hypertension)) +  
  geom_violin() +  
  scale_x_discrete(labels = c("No", "Yes")) +  
  labs(  
    x = "hypertension",  
    y = "Recovery_time") +  
  theme_minimal() + theme(legend.position = "none")  
hypertension_plot
```

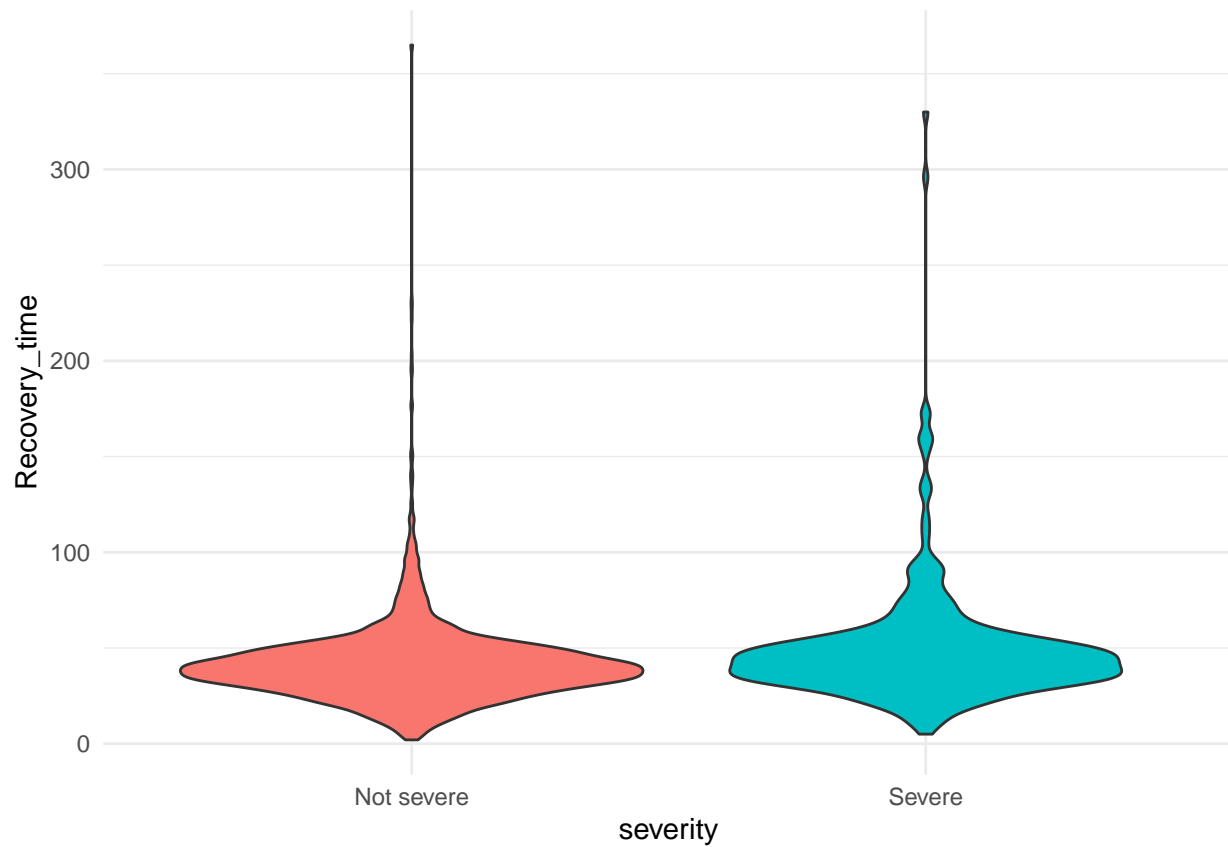
```
diabetes_plot <- data |>
  ggplot(aes(x = diabetes, y = recovery_time, fill = diabetes)) +
  geom_violin() +
  scale_x_discrete(labels = c("No", "Yes")) +
  labs(
    x = "diabetes",
    y = "Recovery_time") +
  theme_minimal() + theme(legend.position = "none")
diabetes_plot
```



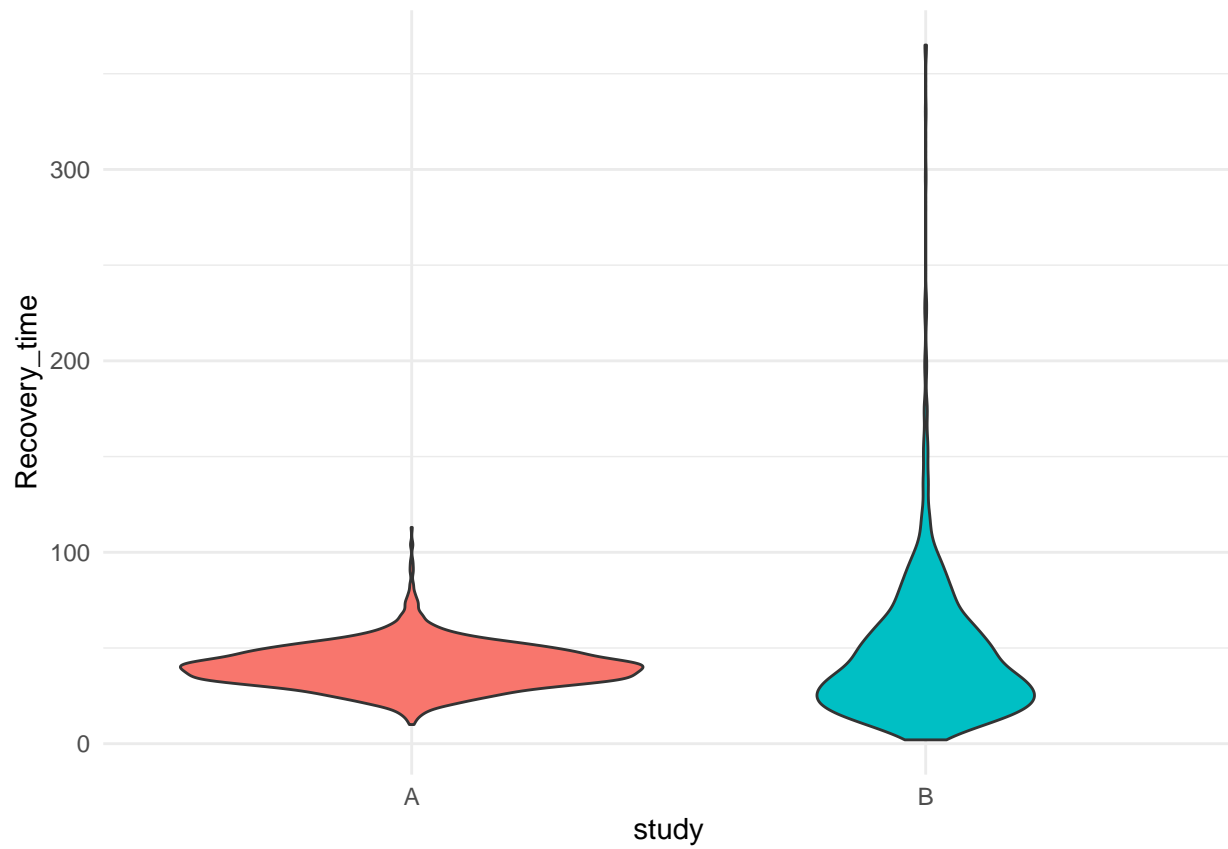
```
vaccine_plot <- data |>
  ggplot(aes(x = vaccine, y = recovery_time, fill = vaccine)) +
  geom_violin() +
  scale_x_discrete(labels = c("Not vaccinated", "Vaccinated")) +
  labs(
    x = "vaccine",
    y = "Recovery_time") +
  theme_minimal() + theme(legend.position = "none")
vaccine_plot
```



```
severity_plot <- data |>
  ggplot(aes(x = severity, y = recovery_time, fill = severity)) +
  geom_violin() +
  scale_x_discrete(labels = c("Not severe", "Severe")) +
  labs(
    x = "severity",
    y = "Recovery_time") +
  theme_minimal() + theme(legend.position = "none")
severity_plot
```

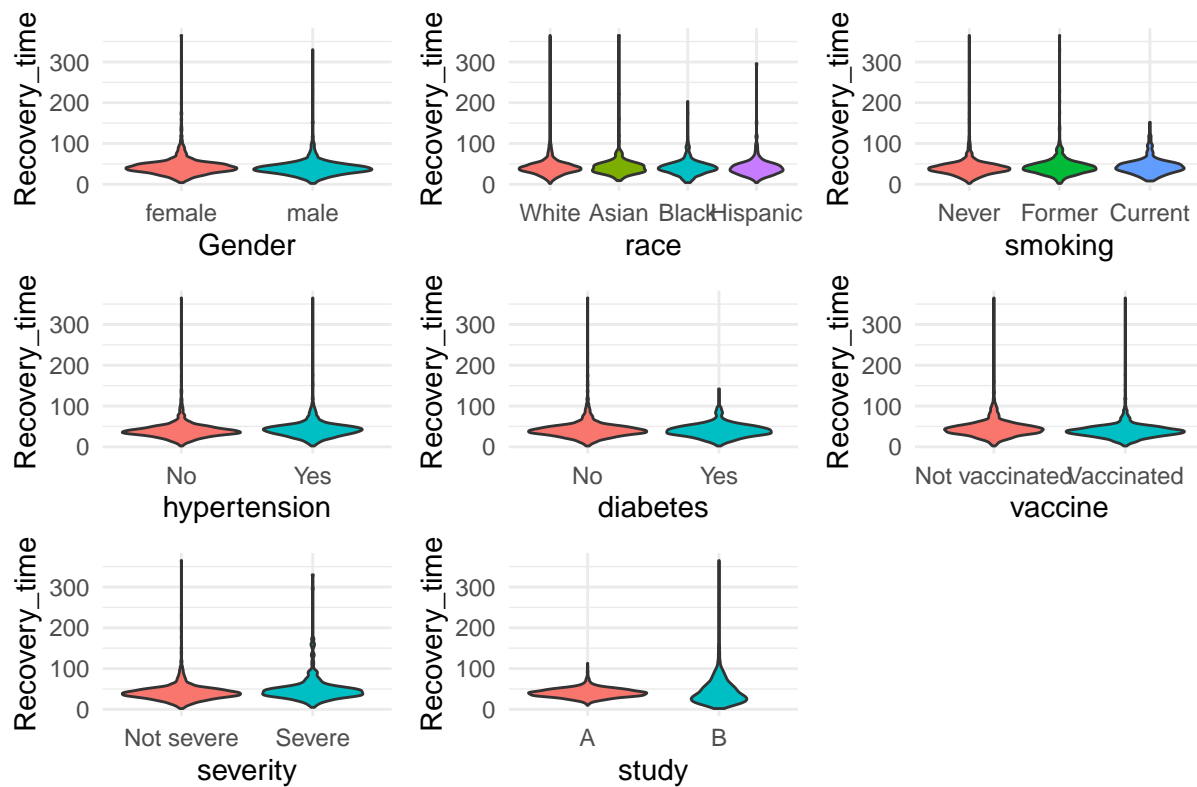


```
study_plot <- data |>
  ggplot(aes(x = study, y = recovery_time, fill = study)) +
  geom_violin() +
  scale_x_discrete(labels = c("A", "B")) +
  labs(
    x = "study",
    y = "Recovery_time") +
  theme_minimal() + theme(legend.position = "none")
study_plot
```



```
combined <- gender_plot + race_plot + smoking_plot + hypertension_plot + diabetes_plot + vaccine_plot +  
combined + plot_annotation(title = "Figure 2: Relationship between Categorical Predictors and Recovery ")
```

Figure 2: Relationship between Categorical Predictors and Recovery Time



Linear models

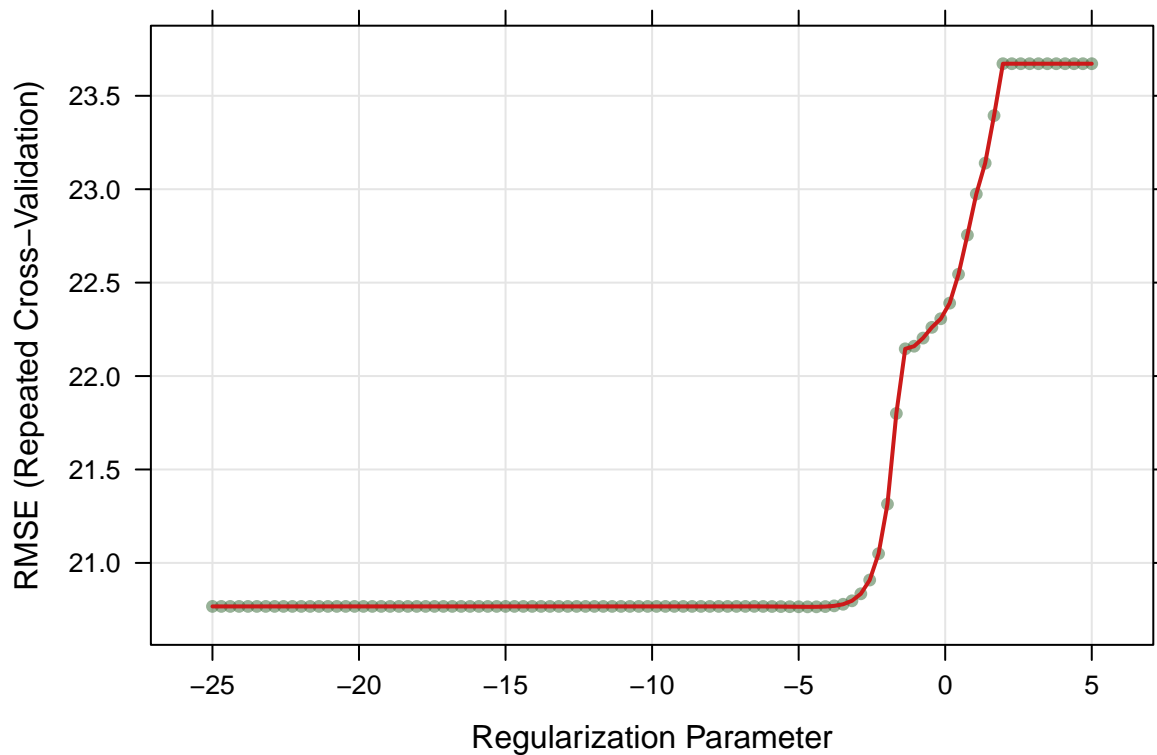
Fit a lasso model

```
set.seed(666)

lasso.fit <- train(recovery_time ~ .,
  data = training_data,
  method = "glmnet",
  tuneGrid = expand.grid(alpha = 1,
    lambda = exp(seq(-25, 5, length = 100))),
  trControl = ctrl1)

## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo,
## : There were missing values in resampled performance measures.

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



```
## Select the best tuning parameter
```

```
lasso.fit$bestTune
```

```
##      alpha      lambda
```

```
## 68      1 0.00912288
```

```
# coefficients in the final model
```

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

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
```

```
##              s1
```

```
## (Intercept) -2.229189e+03
```

```
## age          2.736904e-01
```

```
## gender1     -3.698081e+00
```

```
## race2        1.629521e+00
```

```
## race3       -7.478838e-01
```

```
## race4       -8.493986e-01
```

```
## smoking1     2.764637e+00
```

```
## smoking2     3.624109e+00
```

```
## height       1.302739e+01
```

```
## weight      -1.414814e+01
```

```
## bmi          4.250183e+01
```

```
## hypertension1 3.023576e+00
```

```
## diabetes1    -1.066279e+00
```

```
## SBP         -1.899186e-02
```

```
## LDL         -3.709871e-02
```

```
## vaccine1    -6.014455e+00
```

```
## severity1    7.390563e+00
```

```
## study1       5.298800e+00
```

The selected tuning parameter is 0.00912288.

Test errors

```
set.seed(666)
lasso.pred <- predict(lasso.fit, newdata = testing_data)
mean((lasso.pred - testing_data[, "recovery_time"])^2)
```

```
## [1] 298.3016
```

The test error is 298.3016.

Fit an elastic net model

```
set.seed(666)
enet.fit <- train(recovery_time ~ .,
                  data = training_data,
                  method = "glmnet",
                  tuneGrid = expand.grid(alpha = seq(0, 1, length = 21),
                                         lambda = exp(seq(-25, 5, length = 100))),
                  trControl = ctrl1)
```

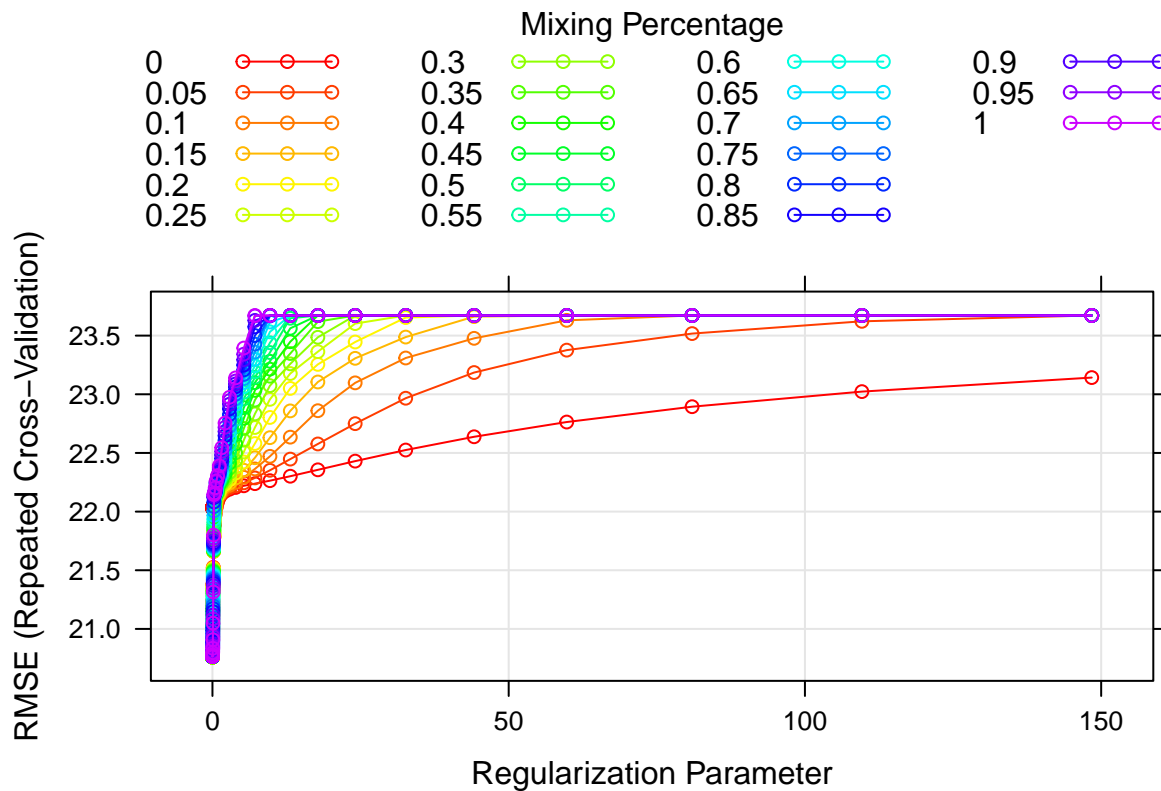
```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo,
## : There were missing values in resampled performance measures.
```

```
## Select the best tuning parameter
enet.fit$bestTune
```

```
##      alpha      lambda
## 365  0.15 0.00367552
```

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

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

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

## 18 x 1 sparse Matrix of class "dgCMatrix"
##              s1
## (Intercept) -2.157409e+03
## age         2.786152e-01
## gender1     -3.708843e+00
## race2       1.666442e+00
## race3      -7.696907e-01
## race4      -9.113345e-01
## smoking1    2.780175e+00
## smoking2    3.645642e+00
## height     1.260703e+01
## weight     -1.370319e+01
## bmi        4.122283e+01
## hypertension1 3.105223e+00
## diabetes1  -1.101528e+00
## SBP        -2.443694e-02
## LDL        -3.746249e-02
## vaccine1   -6.040658e+00
## severity1   7.430536e+00
## study1     5.315665e+00

# test error
enet.pred <- predict(enet.fit, newdata = testing_data)
mean((enet.pred - testing_data[, "recovery_time"])^2)

## [1] 297.1642
```

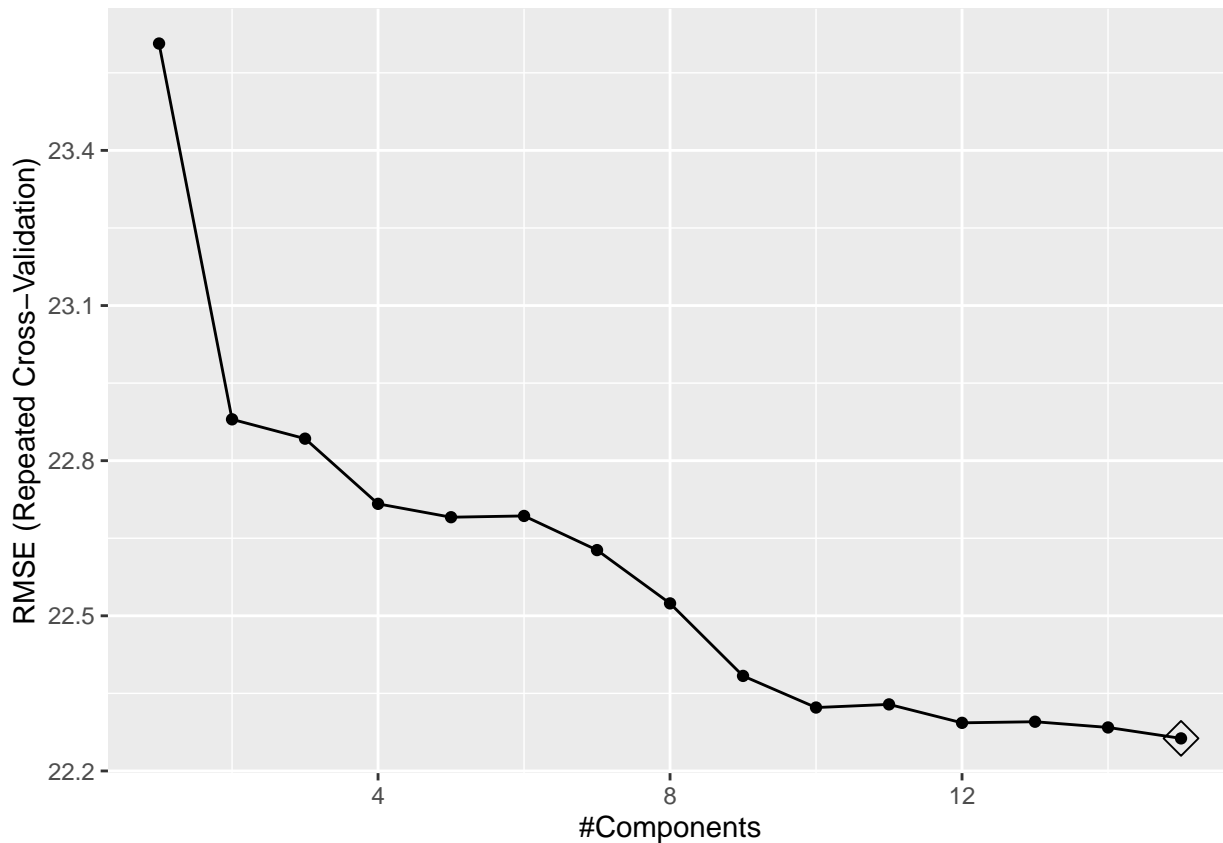
The selected tuning parameter is 0.00367552, and the test error is 297.1642.

Fit a PCR model

```
set.seed(666)
pcr.fit <- train(x, y,
  method = "pcr",
  tuneGrid = data.frame(ncomp = 1:15),
  trControl = ctrl1,
  scale = TRUE)
predy2.pcr2 <- predict(pcr.fit, newdata = x2)
mean((y2 - predy2.pcr2)^2)
```

```
## [1] 323.779
```

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



```
summary(pcr.fit)
```

```
## Data:      X dimension: 2400 17
## Y dimension: 2400 1
## Fit method: svdpc
## Number of components considered: 15
## TRAINING: % variance explained
##      1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps
## X      13.117  23.728  31.171  38.302  45.266  51.762  57.87
## .outcome 0.602   7.376   7.832   9.341   9.353   9.575  10.59
##      8 comps  9 comps 10 comps 11 comps 12 comps 13 comps 14 comps
```

```
## X          63.91    69.84    75.59    81.01    85.98    90.51    94.88
## .outcome   11.16    12.02    12.51    12.51    12.88    12.98    13.16
##          15 comps
## X          98.82
## .outcome   13.32
```

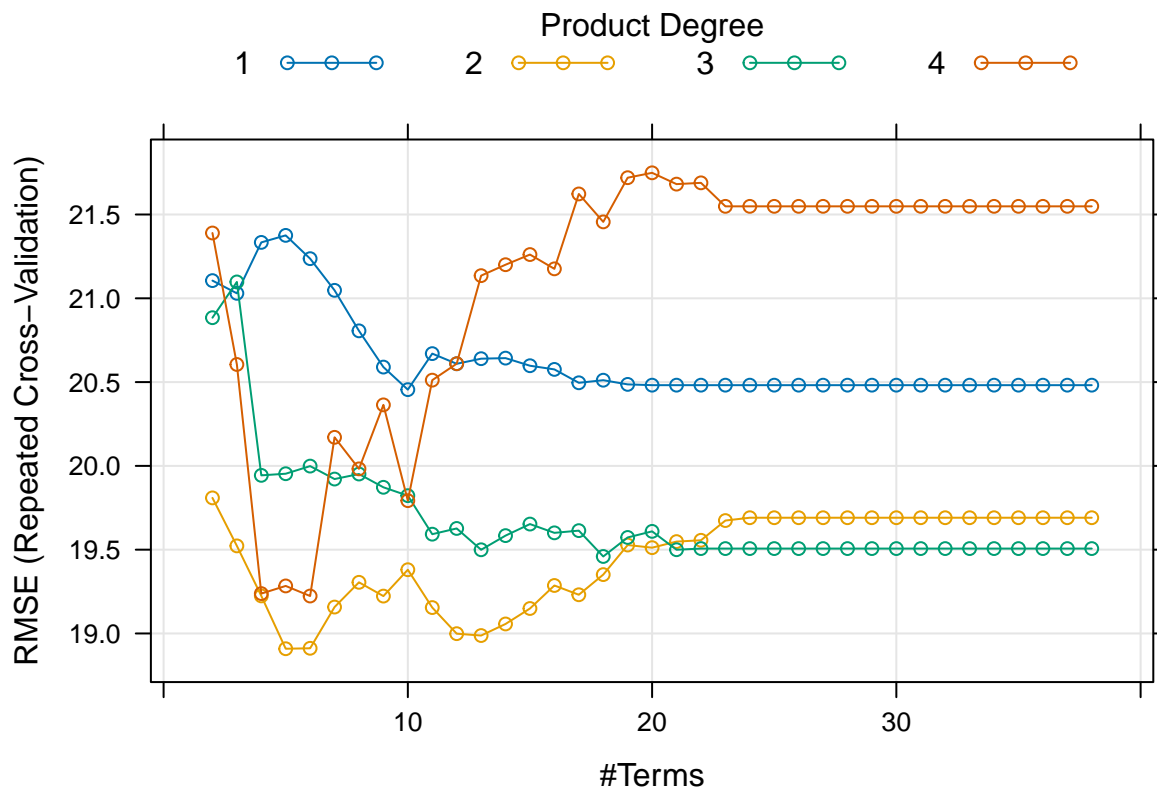
There are 15 components in the model and the test error is 323.779.

Nonlinear models

MARS

```
set.seed(666)
mars.grid <- expand.grid(degree = 1:4,
                        nprune = 2:38)
mars.fit <- train(x, y,
                  method = "earth",
                  tuneGrid = mars.grid,
                  trControl = ctrl1)

plot(mars.fit)
```



```
# best tune
mars.fit$bestTune
```

```
##      nprune degree
## 41         5      2
```

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

```
##              (Intercept)                h(31-bmi)
##              -3.1983530                6.3999877
##      h(bmi-31) * study1                h(bmi-25.2)
##              25.6820131                7.9260754
## h(weight-86.4) * h(bmi-31)
##              -0.6277843
```

Test error for MARS

```
mars.pred <- predict(mars.fit, newdata = x2)
mars.test.error <- mean((mars.pred - y2)^2)
mars.test.error
```

```
## [1] 279.0367
```

The regression function should be: $-3.1983530 + 6.3999877 * h(31-bmi) + 25.6820131 * h(bmi-31) * study1 + 7.9260754 * h(bmi-25.2) - 0.6277843 * h(weight-86.4) * h(bmi-31)$

The test error is 279.0367

GAM

```
set.seed(666)

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

```
## Loading required package: mgcv
## Loading required package: nlme
##
## Attaching package: 'nlme'
## The following object is masked from 'package:dplyr':
##
##      collapse
## This is mgcv 1.9-0. For overview type 'help("mgcv-package")'.
```

```
gam.fit$bestTune
```

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

```
gam.fit$finalModel
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## .outcome ~ gender1 + race2 + race3 + race4 + smoking1 + smoking2 +
##      hypertension1 + diabetes1 + vaccine1 + severity1 + study1 +
##      s(age) + s(SBP) + s(LDL) + s(bmi) + s(height) + s(weight)
##
## Estimated degrees of freedom:
## 1.00 1.75 1.00 8.56 7.24 2.81 total = 34.36
```

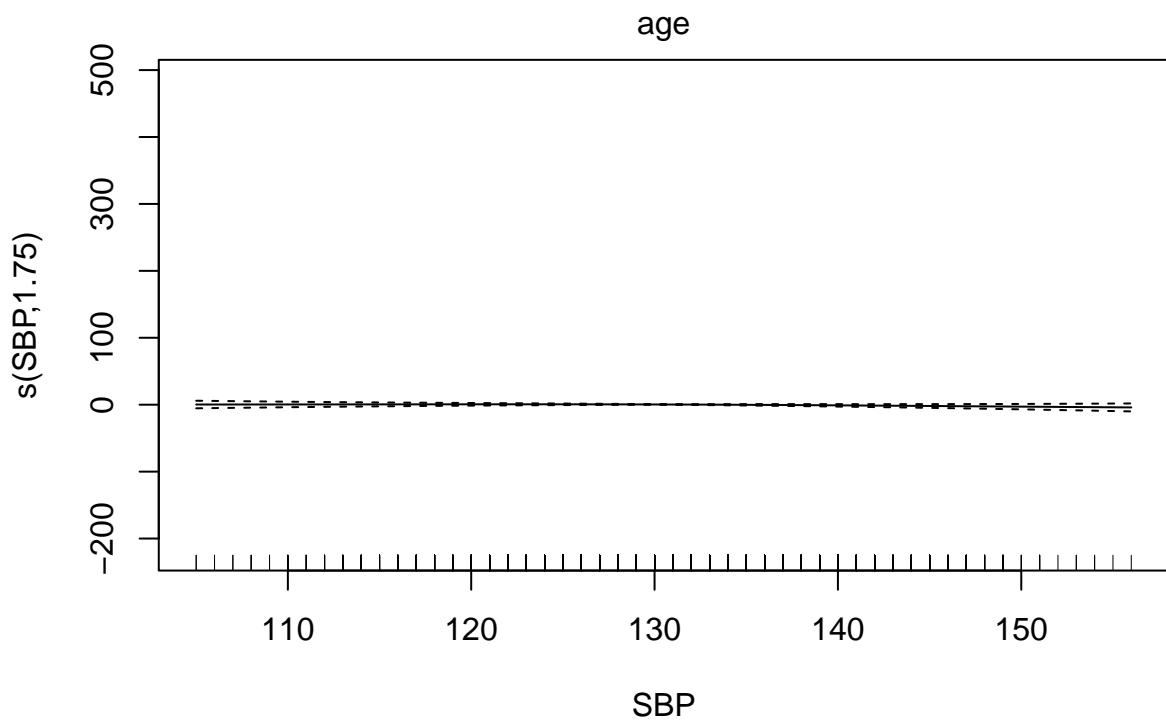
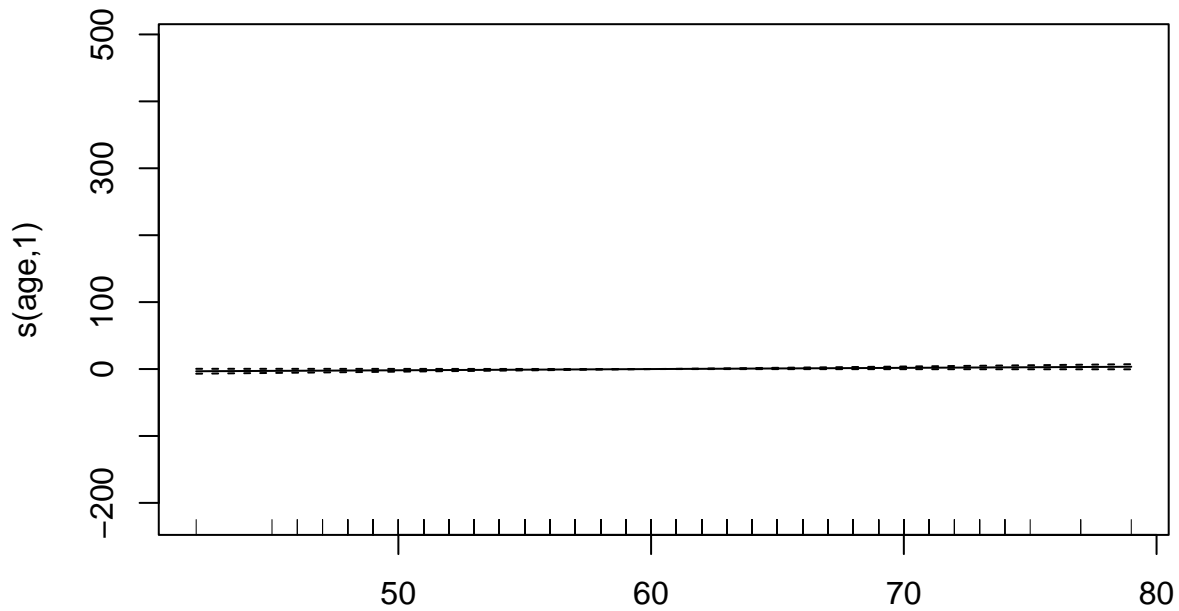
```
##
```

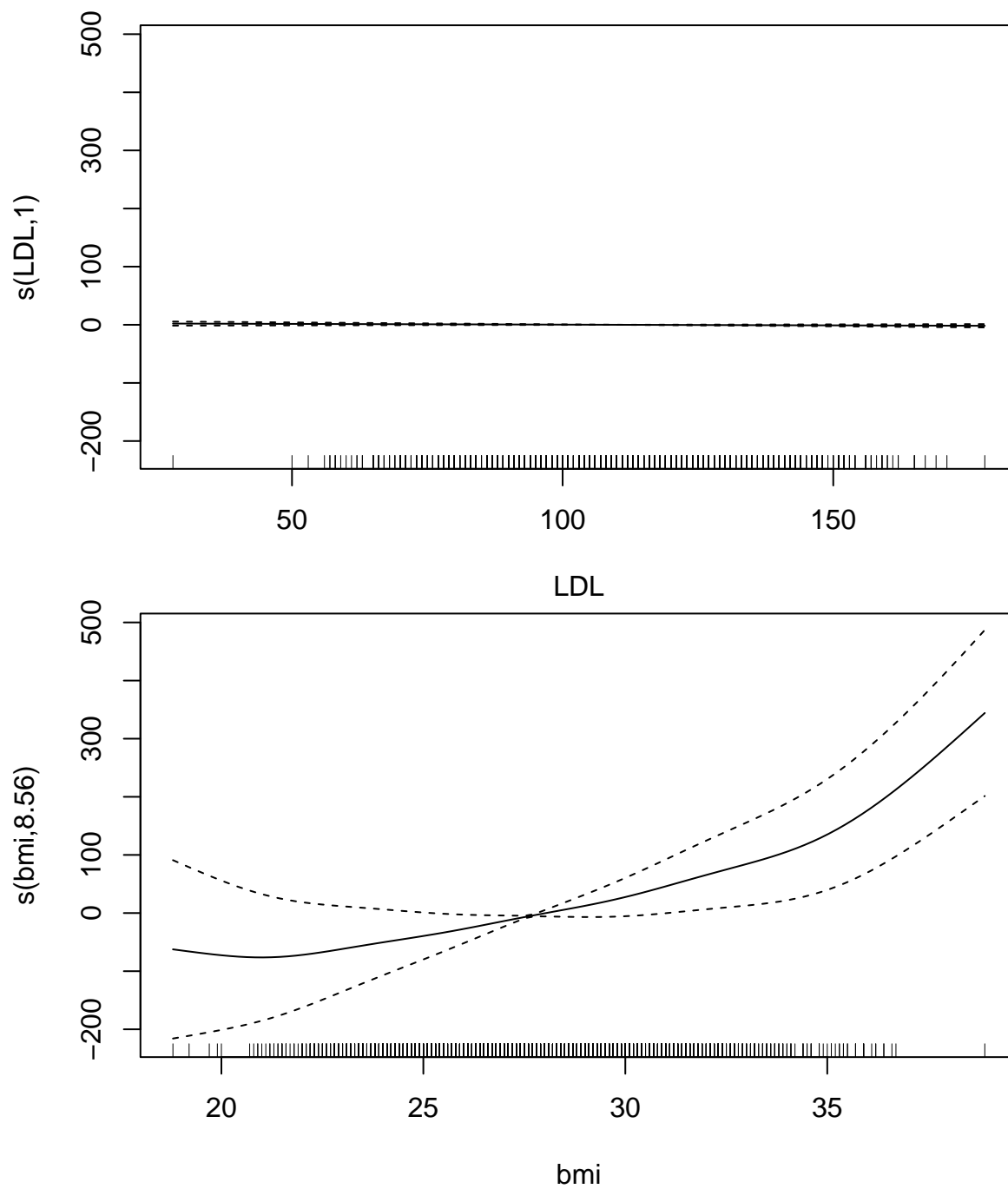
```
## GCV score: 384.8296
```

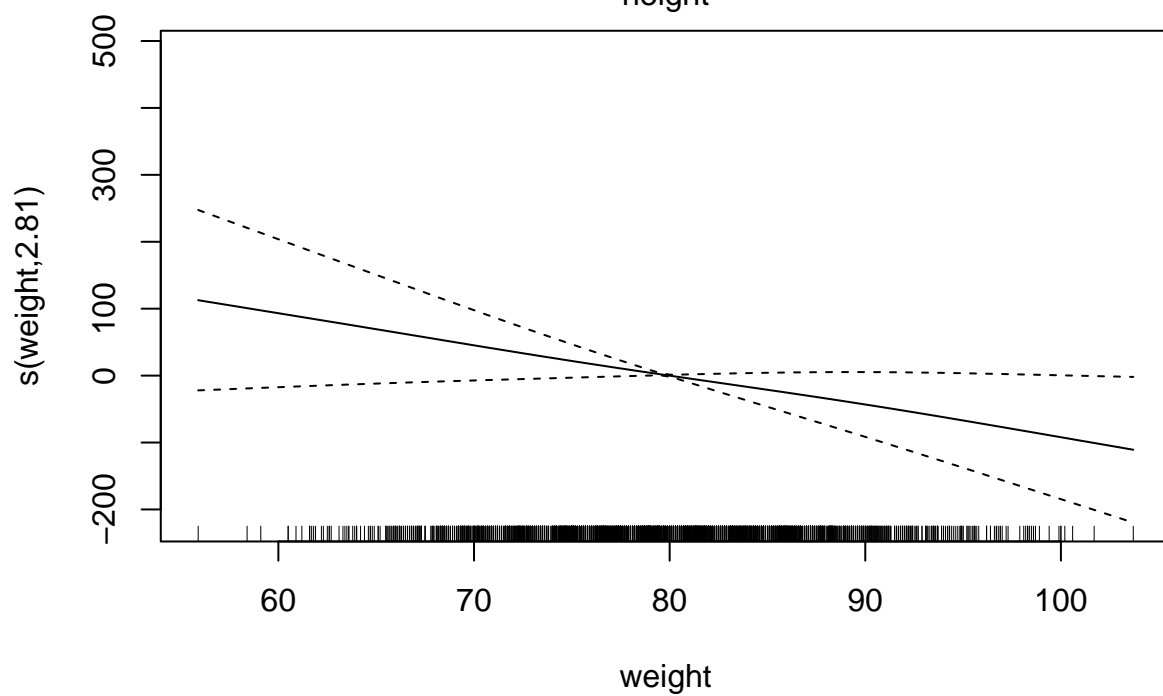
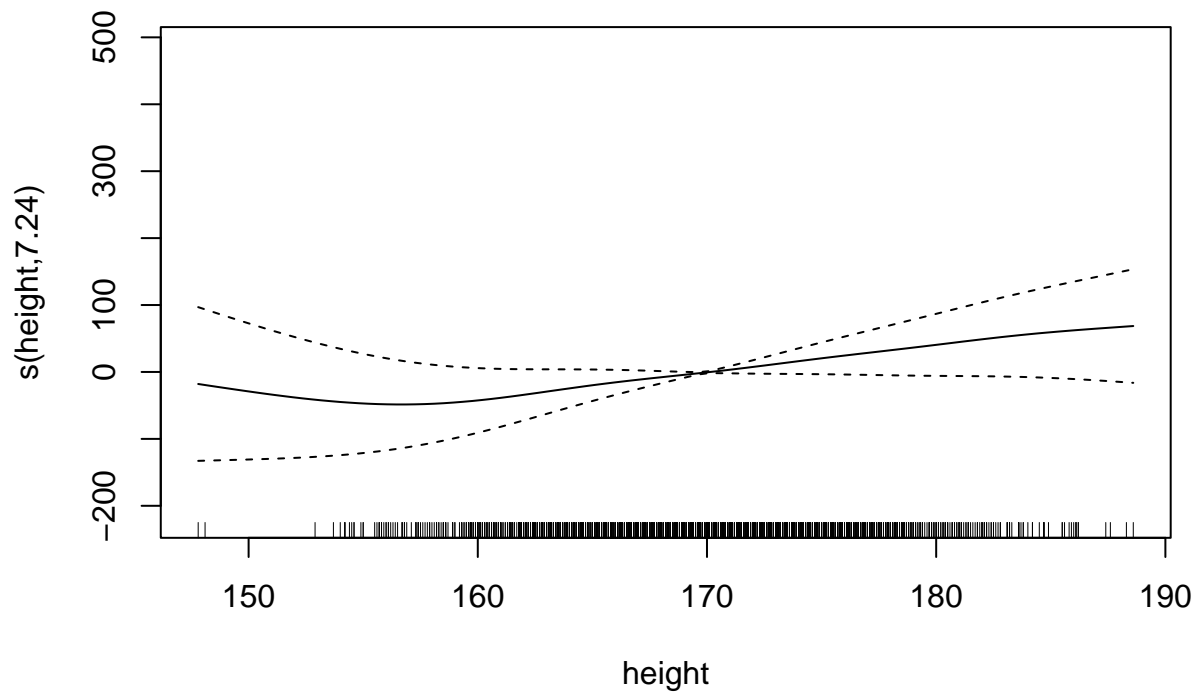
The GAM model includes all the predictors.

Generate plot for GAM

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







Test error for GAM

```
gam.pred <- predict(gam.fit, newdata = x2)
gam.test.error <- mean((gam.pred - y2)^2)
gam.test.error
```

```
## [1] 272.0012
```

The test error for GAM is 272.0012.

Model comparison

```

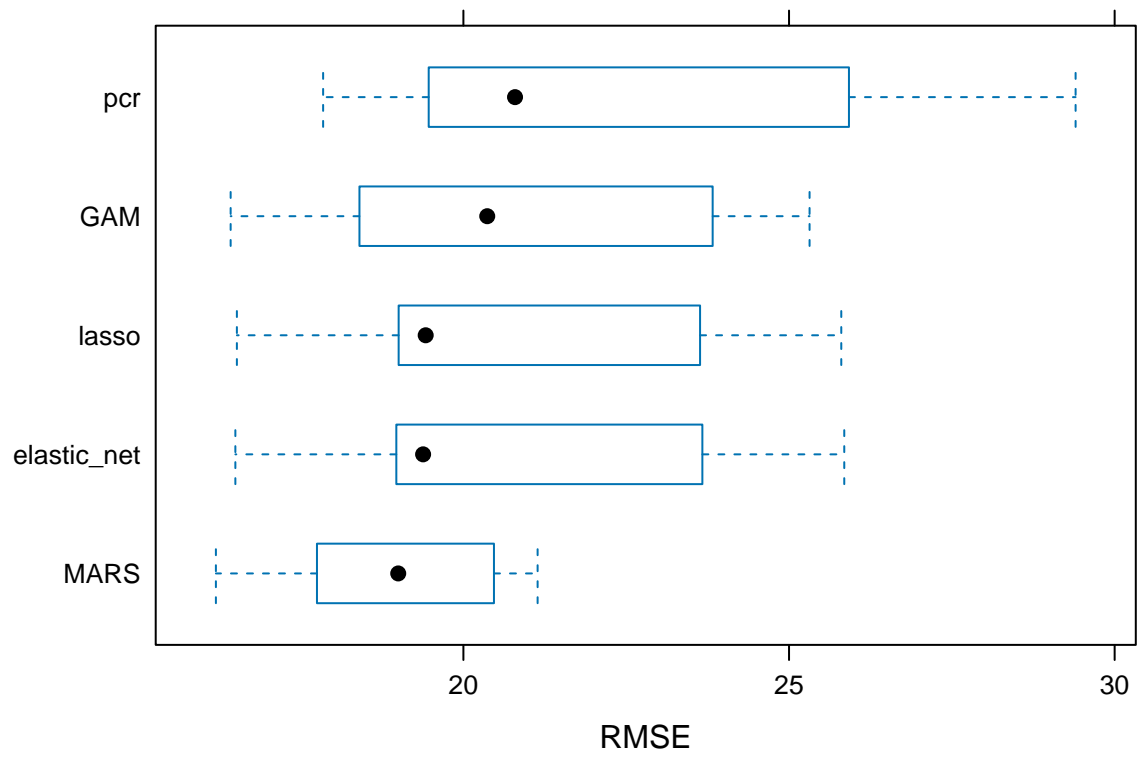
resamp <- resamples(list(lasso = lasso.fit,
                        elastic_net = enet.fit,
                        pcr = pcr.fit,
                        MARS = mars.fit,
                        GAM = gam.fit))

summary(resamp)

##
## Call:
## summary.resamples(object = resamp)
##
## Models: lasso, elastic_net, pcr, MARS, GAM
## Number of resamples: 10
##
## MAE
##           Min.   1st Qu.   Median     Mean   3rd Qu.     Max. NA's
## lasso       12.43437 13.34989 13.55760 13.60117 13.83628 14.80446    0
## elastic_net 12.39330 13.33858 13.50759 13.56655 13.79648 14.79199    0
## pcr         12.79843 13.31630 13.65714 13.79429 14.32701 15.26141    0
## MARS        11.10656 12.33992 12.52758 12.48434 12.95145 13.22857    0
## GAM         11.95625 12.48434 12.90413 13.04720 13.67832 14.43838    0
##
## RMSE
##           Min.   1st Qu.   Median     Mean   3rd Qu.     Max. NA's
## lasso       16.52182 19.01349 19.42116 20.76489 22.92278 25.80217    0
## elastic_net 16.49848 18.99457 19.38126 20.76102 22.93865 25.84810    0
## pcr         17.84596 19.61842 20.79266 22.26300 24.74262 29.40059    0
## MARS        16.19940 17.79482 18.99934 18.90878 20.37825 21.13959    0
## GAM         16.42592 18.51665 20.36608 20.61337 23.26164 25.31540    0
##
## Rsquared
##           Min.   1st Qu.   Median     Mean   3rd Qu.     Max. NA's
## lasso       0.12468127 0.17202217 0.2314610 0.2404963 0.2896872 0.3926191    0
## elastic_net 0.12572907 0.17252570 0.2295363 0.2401821 0.2906348 0.3898854    0
## pcr         0.05140385 0.09539054 0.1072900 0.1273979 0.1285069 0.2390132    0
## MARS        0.12610733 0.18362545 0.2804839 0.3515976 0.5442009 0.6258534    0
## GAM         0.11451833 0.18716669 0.2610268 0.2888378 0.3891996 0.4926768    0

bwplot(resamp, metric = "RMSE")

```

model should be selected as the final model since it has the lowest RMSE value.

Mars