Data Science II Midterm Project Analysis

Camille Okonkwo

Contents

```r
library(tidymodels)
library(splines)
library(caret)
library(glmnet)
library(table1)
library(kableExtra)
library(summarytools)
library(corrplot)
library(cowplot)
```

Background

To gain a better understanding of the factors that predict recovery time from COVID-19 illness, a study was designed to combine three existing cohort studies that have been tracking participants for several years. The study collects recovery information through questionnaires and medical records, and leverages existing data on personal characteristics prior to the pandemic. The ultimate goal is to develop a prediction model for recovery time and identify important risk factors for long recovery time.

Data

The dataset in `recovery.RData` includes data from 3000 participants.

Here is a description of each variable:

- ID (`id`): Participant ID
- Gender (`gender`): 1 = Male, 0 = Female
- Race/ethnicity (`race`): 1 = White, 2 = Asian, 3 = Black, 4 = Hispanic
- Smoking (`smoking`): Smoking status; 0 = Never smoked, 1 = Former smoker, 2 = Current smoker
- Height (`height`): Height (in centimeters)
- Weight (`weight`): Weight (in kilograms)
- BMI (`bmi`): Body Mass Index; BMI = weight (in kilograms) / height (in meters) squared
- Hypertension (`hypertension`): 0 = No, 1 = Yes
- Diabetes (`diabetes`): 0 = No, 1 = Yes
- Systolic blood pressure (`SBP`): Systolic blood pressure (in mm/Hg)
- LDL cholesterol (`LDL`): LDL (low-density lipoprotein) cholesterol (in mg/dL)
- Vaccination status at the time of infection (`vaccine`): 0 = Not vaccinated, 1 = Vaccinated
- Severity of COVID-19 infection (`severity`): 0 = Not severe, 1= Severe
- Study (`study`): The study (A/B) that the participant belongs to
- Time to recovery (`recovery_time`): Time from COVID-19 infection to recovery in days

Data Preparation

Partition the dataset into two parts: a matrix of predictors and a vector of response.

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

dat = dat |>
  select(-id)

# matrix of predictors & vector of response for data set exploration
x.dat = model.matrix(recovery_time ~., dat)[, -1]
y.dat = dat$recovery_time
```

Exploratory analysis and data visualization

Descriptive Statistics Table

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

dfSummary(dat)
```

```
## ### Data Frame Summary
## **dat**
## **Dimensions:** 3000 x 15
## **Duplicates:** 0
##
## ----------------------------------------------------------------------------------------------------
## No   Variable         Stats / Values              Freqs (% of Valid)    Graph                    Vali
## ---- ---------------- --------------------------- --------------------- ------------------------ ---
## 1    age\             Mean (sd) : 60.2 (4.5)\     34 distinct values    \ \ \ \ \ \ : .\         3000
##      [numeric]        min < med < max:\                                 \ \ \ \ \ \ : :\         (100
##                       42 < 60 < 79\                                     \ \ \ \ \ \ : :\
##                       IQR (CV) : 6 (0.1)                                \ \ \ \ . : : .\
##                                                                         \ \ \ \ : : : :
##
## 2    gender\          Min  : 0\                   0 : 1544 (51.5%)\     IIIIIIIIII \             3000
##      [integer]        Mean : 0.5\                 1 : 1456 (48.5%)      IIIIIIIII                (100
##                       Max  : 1
##
## 3    race\            1\. 1\                      1967 (65.6%)\         IIIIIIIIIIIII \          3000
##      [factor]         2\. 2\                      158 ( 5.3%)\          I \                      (100
##                       3\. 3\                      604 (20.1%)\          IIII \
##                       4\. 4                       271 ( 9.0%)           I
##
## 4    smoking\         1\. 0\                      1822 (60.7%)\         IIIIIIIIIIII \           3000
##      [factor]         2\. 1\                      859 (28.6%)\          IIIII \                  (100
##                       3\. 2                       319 (10.6%)           II
##
## 5    height\          Mean (sd) : 169.9 (6)\      313 distinct values   \ \ \ \ \ \ \ \ : :\     3000
##      [numeric]        min < med < max:\                                 \ \ \ \ \ \ \ \ : :\     (100
##                       147.8 < 169.9 < 188.6\                            \ \ \ \ \ \ . : : .\
##                       IQR (CV) : 7.9 (0)                                \ \ \ \ \ \ : : : :\
##                                                                         \ \ \ \ . : : : : .
##
## 6    weight\          Mean (sd) : 80 (7.1)\       364 distinct values   \ \ \ \ \ \ \ \ : .\     3000
##      [numeric]        min < med < max:\                                 \ \ \ \ \ \ \ \ : :\     (100
##                       55.9 < 79.8 < 103.7\                              \ \ \ \ \ \ : : : :\
##                       IQR (CV) : 9.6 (0.1)                              \ \ \ \ . : : : : .\
##                                                                         \ \ . : : : : : : .
##
## 7    bmi\             Mean (sd) : 27.8 (2.8)\     163 distinct values   \ \ \ \ \ \ . :\         3000
##      [numeric]        min < med < max:\                                 \ \ \ \ \ \ : : :\       (100
##                       18.8 < 27.6 < 38.9\                               \ \ \ \ \ \ : : :\
##                        IQR (CV) : 3.7 (0.1)                              \ \ \ \ : : : : :\
##                                                                         \ \ . : : : : : .
##
## 8    hypertension\    Min   : 0\                  0 : 1508 (50.3%)\    IIIIIIIIII \             3000
##      [numeric]        Mean : 0.5\                 1 : 1492 (49.7%)     IIIIIIIII                (100
##                        Max   : 1
##
## 9    diabetes\        Min   : 0\                  0 : 2537 (84.6%)\    IIIIIIIIIIIIIIII \       3000
##      [integer]        Mean : 0.2\                 1 :  463 (15.4%)     III                      (100
##                        Max   : 1
##
## 10   SBP\             Mean (sd) : 130.5 (8)\      52 distinct values    \ \ \ \ \ \ \ \ : .\     3000
##      [numeric]        min < med < max:\                                 \ \ \ \ \ \ \ \ : : .\   (100
##                        105 < 130 < 156\                                  \ \ \ \ \ \ : : : :\
##                        IQR (CV) : 11 (0.1)                               \ \ \ \ . : : : : .\
##                                                                         \ \ . : : : : : : .
##
## 11   LDL\             Mean (sd) : 110.5 (19.8)\   114 distinct values   \ \ \ \ \ \ \ \ \ \ :\   3000
##      [numeric]        min < med < max:\                                 \ \ \ \ \ \ \ \ : : .\   (100
##                        28 < 110 < 178\                                   \ \ \ \ \ \ \ \ : : :\
##                        IQR (CV) : 27 (0.2)                               \ \ \ \ \ \ . : : : .\
##                                                                         \ \ \ \ . : : : : : .
##
## 12   vaccine\         Min   : 0\                  0 : 1212 (40.4%)\    IIIIIIII \               3000
##      [integer]        Mean : 0.6\                 1 : 1788 (59.6%)     IIIIIIIIIII              (100
##                        Max   : 1
##
## 13   severity\        Min   : 0\                  0 : 2679 (89.3%)\    IIIIIIIIIIIIIIIII \      3000
##      [integer]        Mean : 0.1\                 1 :  321 (10.7%)     II                       (100
##                        Max   : 1
##
## 14   study\           1\. A\                       2000 (66.7%)\        IIIIIIIIIIIII \          3000
##      [character]      2\. B                        1000 (33.3%)         IIIIII                   (100
##
## 15   recovery_time\   Mean (sd) : 42.2 (23.2)\    140 distinct values   : :\                     3000
##      [numeric]        min < med < max:\                                 : :\                     (100
##                        2 < 39 < 365\                                     : :\
##                        IQR (CV) : 18 (0.5)                               : :\
##                                                                         : : .
## ----------------------------------------------------------------------------------------------------
```

```
units(dat_ds$Height) <- "cm"
units(dat_ds$Weight) <- "kg"
units(dat_ds$`Body Mass Index`) <- "kg/m^2"
units(dat_ds$`Systolic Blood Pressure`) <- "mm/Hg"
units(dat_ds$`Low-density lipoprotein cholesterol`) <- "mg/dL"
units(dat_ds$`Time from COVID-19 infection to recovery`) <- "days"

descriptive_table <- table1(~ Age + Gender + `Race/Ethnicity` + `Smoking status` + Height + Weight + `B
                            data = dat_ds,
                            overall = "Total",
                            caption = "Descriptive Characteristics of Participants, Stratified by Sourc

ds = t1kable(descriptive_table)
```

```
ds
```

There are no missing values in the dataset. The distribution of the demographic variables `age`, `gender`, `race` are about the same between treatment groups. Mean `height`, `weight`, `BMI`, `SBP` and `LDL` variables are also similarly distributed between groups. There are more people who are vaccinated than not vaccinated in study group A and B, and also there are more participants who are reported to have not severe COVID-19 infections. `recovery_time` mean and SD is higher for Study B. There is also a larger interval range.

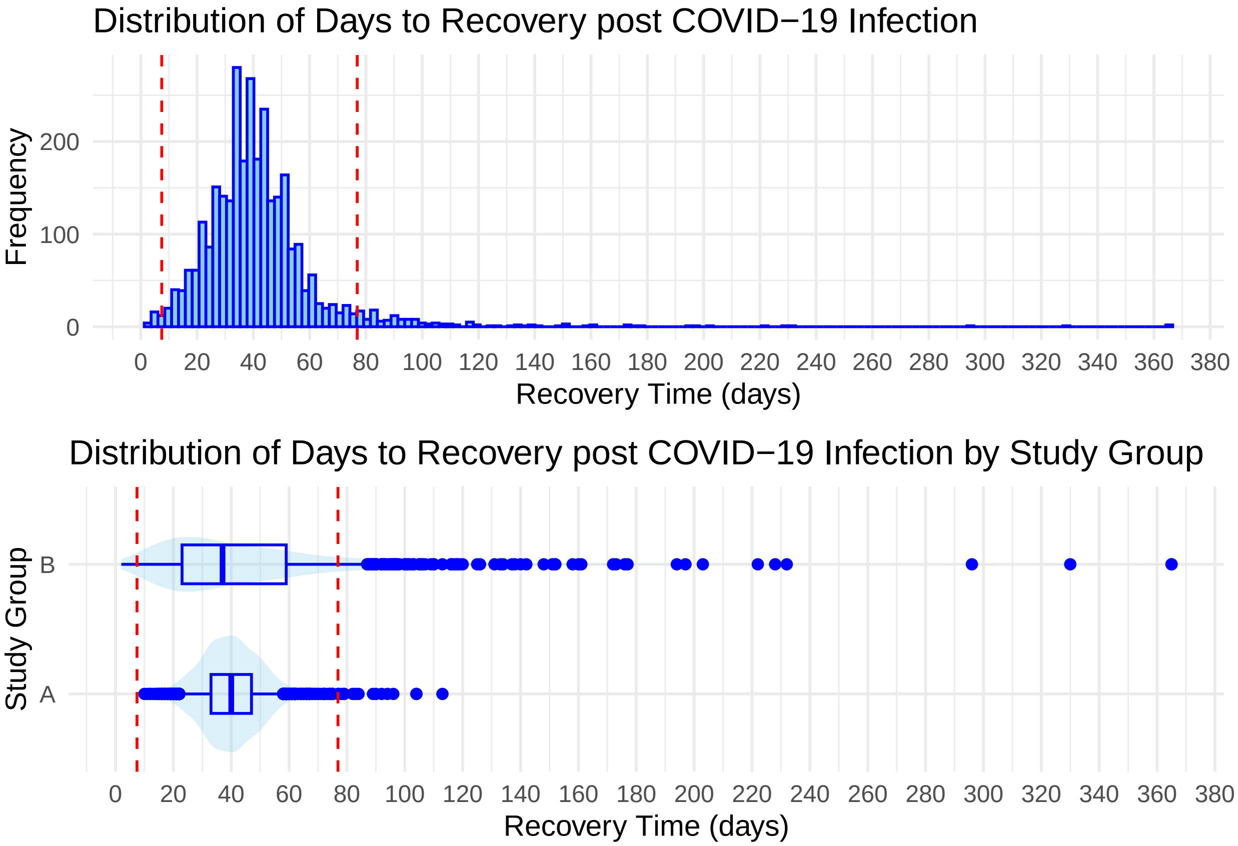
Response Variable Exploration

```
# Calculate mean and standard deviation
mean_value = mean(dat$recovery_time)
sd_value = sd(dat$recovery_time)

# Define upper and lower bounds
outlier_coeff = 1.5
outlier_high = mean_value + outlier_coeff * sd_value
outlier_low = mean_value - outlier_coeff * sd_value

# recovery_time boxplot
boxplot_recovery =
  dat |>
  ggplot(aes(x = recovery_time, y = study)) +
  geom_violin(fill = "skyblue", alpha = 0.3, color= NA) +
  geom_boxplot(fill = NA, color = "blue",
               width = 0.3, coef = outlier_coeff/2) +
  geom_vline(xintercept = c(outlier_low, outlier_high),
             color = "red",linetype = "dashed", size = .5) +
  labs(title = "Distribution of Days to Recovery post COVID-19 Infection by Study Group",
       x = "Recovery Time (days)", y = "Study Group") +
  theme_minimal() +
  scale_x_continuous(
    breaks = seq(0, 400, by = 20),
    labels = seq(0, 400, by = 20)
  )

# recovery_time histogram
histogram_recovery =
  dat |>
  ggplot(aes(x = recovery_time)) +
  geom_histogram(bins = 150, fill = "skyblue", color = "blue") +
  geom_vline(xintercept = c(outlier_low, outlier_high),
             color = "red", linetype = "dashed", size = .5) +
  labs(title = "Distribution of Days to Recovery post COVID-19 Infection",
       x = "Recovery Time (days)", y = "Frequency") +
  theme_minimal() +
  scale_x_continuous(
    breaks = seq(0, 400, by = 20),
    labels = seq(0, 400, by = 20)
  )

combined_recovery =
  plot_grid(histogram_recovery, boxplot_recovery, ncol = 1)
```

Table 1: Descriptive Characteristics of Participants, Stratified by Source Study

	Study A	Study B	Total
	(N=2000)	(N=1000)	(N=3000)
Age			
Mean (SD)	17.2 (4.52)	17.2 (4.38)	17.2 (4.47)
Median [Min, Max]	17.0 [1.00, 34.0]	17.0 [2.00, 33.0]	17.0 [1.00, 34.0]
Gender			
Female	1036 (51.8%)	508 (50.8%)	1544 (51.5%)
Male	964 (48.2%)	492 (49.2%)	1456 (48.5%)
Race/Ethnicity			
White	1312 (65.6%)	655 (65.5%)	1967 (65.6%)
Asian	108 (5.4%)	50 (5.0%)	158 (5.3%)
Black	408 (20.4%)	196 (19.6%)	604 (20.1%)
Hispanic	172 (8.6%)	99 (9.9%)	271 (9.0%)
Smoking status			
Never smoked	1225 (61.3%)	597 (59.7%)	1822 (60.7%)
Former smoker	557 (27.9%)	302 (30.2%)	859 (28.6%)
Current smoker	218 (10.9%)	101 (10.1%)	319 (10.6%)
Height (cm)			
Mean (SD)	160 (58.8)	161 (59.1)	160 (58.9)
Median [Min, Max]	160 [1.00, 313]	161 [2.00, 312]	160 [1.00, 313]
Weight (kg)			
Mean (SD)	181 (70.0)	182 (70.5)	182 (70.2)
Median [Min, Max]	178 [1.00, 364]	182 [3.00, 358]	180 [1.00, 364]
Body Mass Index (kg/m^2)			
Mean (SD)	77.6 (27.5)	77.6 (28.3)	77.6 (27.8)
Median [Min, Max]	77.0 [1.00, 162]	76.0 [2.00, 163]	76.5 [1.00, 163]
Hypertension			
No	998 (49.9%)	510 (51.0%)	1508 (50.3%)
Yes	1002 (50.1%)	490 (49.0%)	1492 (49.7%)
Diabetes			
No	1678 (83.9%)	859 (85.9%)	2537 (84.6%)
Yes	322 (16.1%)	141 (14.1%)	463 (15.4%)
Systolic Blood Pressure (mm/Hg)			
Mean (SD)	26.6 (8.02)	26.3 (7.88)	26.5 (7.97)
Median [Min, Max]	27.0 [1.00, 52.0]	26.0 [1.00, 51.0]	26.0 [1.00, 52.0]
Low-density lipoprotein cholesterol (mg/dL)			
Mean (SD)	58.3 (19.7)	58.7 (19.7)	58.4 (19.7)
Median [Min, Max]	58.0 [1.00, 114]	58.0 [3.00, 112]	58.0 [1.00, 114]
Vaccination status at the time of infection			
Not vaccinated	797 (39.9%)	415 (41.5%)	1212 (40.4%)
Vaccinated	1203 (60.2%)	585 (58.5%)	1788 (59.6%)
Severity of COVID-19 infection			
Not severe	1785 (89.3%)	894 (89.4%)	2679 (89.3%)
Severe	215 (10.8%)	106 (10.6%)	321 (10.7%)
Time from COVID-19 infection to recovery (days)			
Mean (SD)	39.4 (11.1)	42.8 (28.1)	40.5 (18.7)
Median [Min, Max]	39.0 [9.00, 107]	36.0 [1.00, 140]	38.0 [1.00, 140]

```
# Show the combined plot
print(combined_recovery)
```


Distribution of Days to Recovery post COVID−19 Infection

Distribution of Days to Recovery post COVID−19 Infection by Study Group

Using a cut-off based on the standard deviation ± 1.5 times the mean, these outliers will be excluded from future analysis. This cutoff was used to keep the distributions of study groups A & B similar.

```
# removing recovery_time outliers
dat2 =
  dat |>
  filter(recovery_time > outlier_low & recovery_time < outlier_high)

x.dat2 = model.matrix(recovery_time ~., dat2)[, -1]
y.dat2 = dat2$recovery_time
```

Univariate Analysis of Predictors

```
predictor_variables <- c("age", "gender", "race", "smoking", "height", "weight", "bmi", "hypertension",

univar_linear <- lapply(predictor_variables, function(var) {
  formula <- as.formula(paste("recovery_time ~ ", var))
  model <- lm(formula, data = dat2)
    tidy(model) |>
      filter(term != "(Intercept)") |>
      mutate(p.value.sig = ifelse(p.value < 0.05, "*", ""))
})
```

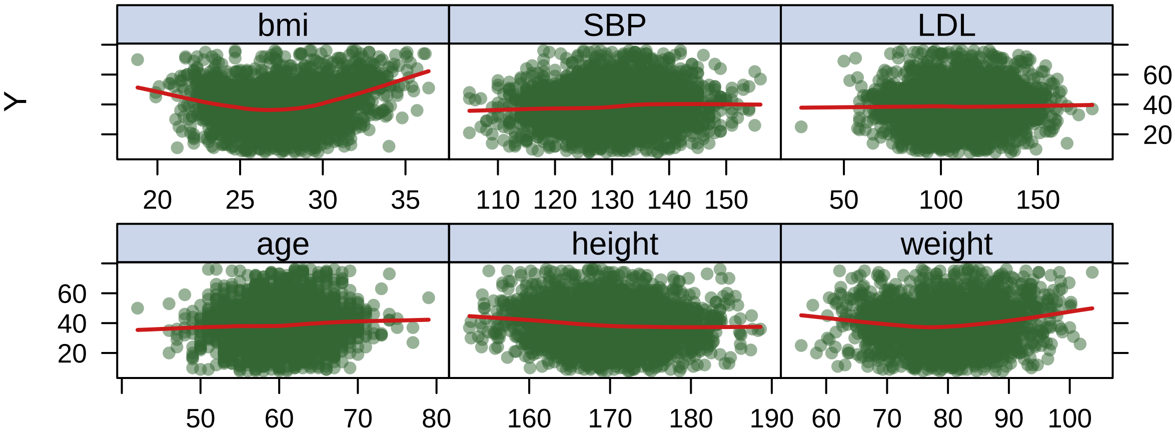
```r
univar_summary <- bind_rows(univar_linear)

knitr::kable(univar_summary)
```

term	estimate	std.error	statistic	p.value	p.value.sig
age	0.2136917	0.0544670	3.9233274	0.0000894	*
gender	-1.9514002	0.4874708	-4.0031118	0.0000641	*
race2	1.2550540	1.1025566	1.1383125	0.2550866	
race3	0.1351247	0.6237843	0.2166208	0.8285195	
race4	-1.0451010	0.8630697	-1.2109115	0.2260307	
smoking1	1.9492952	0.5544858	3.5155005	0.0004458	*
smoking2	1.9728440	0.8126980	2.4275240	0.0152644	*
height	-0.2283756	0.0417856	-5.4654101	0.0000001	*
weight	0.1401333	0.0346808	4.0406647	0.0000547	*
bmi	0.7637000	0.0915412	8.3426892	0.0000000	*
hypertension	2.5708820	0.4863755	5.2857972	0.0000001	*
diabetes	-0.5725136	0.6745191	-0.8487730	0.3960796	
SBP	0.1406887	0.0305709	4.6020517	0.0000044	*
LDL	0.0026741	0.0124015	0.2156233	0.8292969	
vaccine	-4.0462641	0.4945390	-8.1818908	0.0000000	*
severity	3.4951056	0.8013362	4.3615970	0.0000134	*
studyB	-3.7239735	0.5296797	-7.0306142	0.0000000	*

`race`, `diabetes`, and `LDL` appear to be insignificant predictors of `recovery_time` according to the univariate linear analysis.

Feature Plot

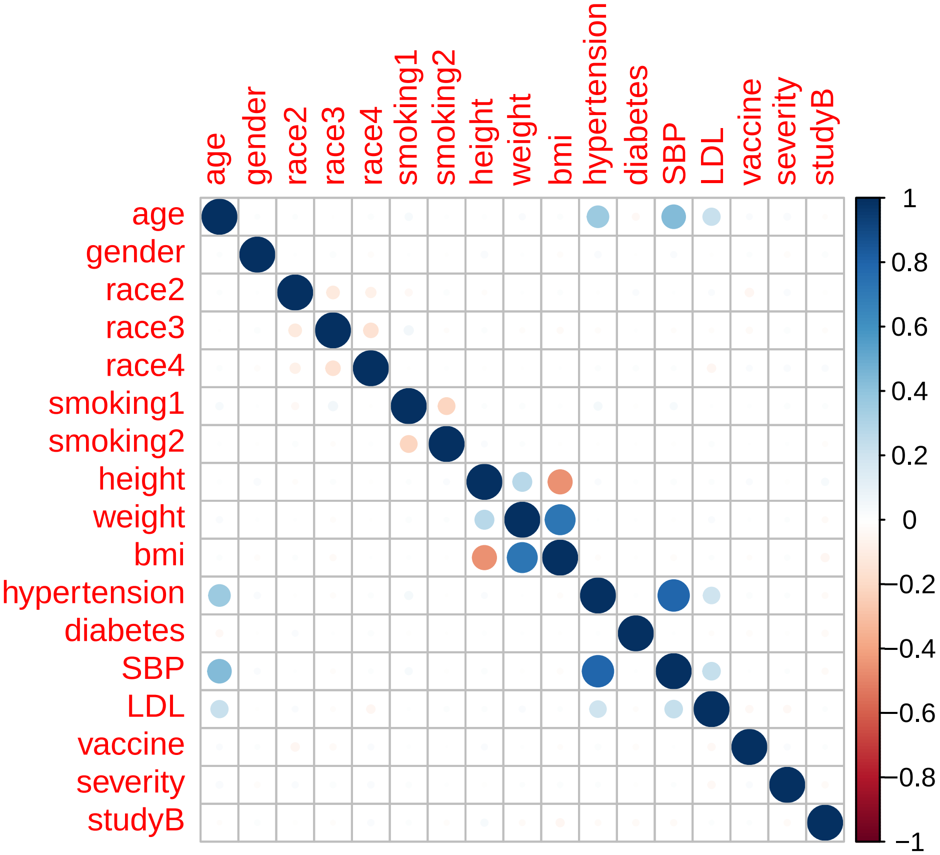
```r
theme1 <- trellis.par.get()
theme1$plot.symbol$col <- rgb(.2, .4, .2, .5)
theme1$plot.symbol$pch <- 16
theme1$plot.line$col <- rgb(.8, .1, .1, 1)
theme1$plot.line$lwd <- 2
theme1$strip.background$col <- rgb(.0, .2, .6, .2)
trellis.par.set(theme1)

featurePlot(
  x.dat2[, -c(2, 3, 4, 5, 6, 7, 11, 12, 15, 16, 17) ],
  y.dat2,
  plot = "scatter",
  labels = c("", "Y"),
  type = c("p", "smooth"),
  layout = c(3, 3))
```



From the feature plot of the continuous variables, there appears to be no strong linear correlations with our response variable `recovery_time`. `bmi` and `weight` however show a potential non-linear relationship. A GAM or MARS model may be best.

Correlation Matrix

```
corrplot(cor(x.dat2), method = "circle", type = "full")
```



The correlation matrix between predictors indicates multicollinearity between `bmi` and `weight`, `sbp` and `hypertension`, and potentially `bmi` and `height`.

Model Training in caret

Test and Train Data Preparation

```
set.seed(1234)

# create a random split of 80% training and 20% test data
data_split <- initial_split(data = dat2, prop = 0.8)

# partitioned datasets
training_data = training(data_split)
testing_data = testing(data_split)

# training data
x <- model.matrix(recovery_time ~ ., training_data)[, -1] # matrix of predictors
head(x)
```

```
##   age gender race2 race3 race4 smoking1 smoking2 height weight  bmi
## 1  62      0     0     0     0        0        0  178.2   82.0 25.8
## 2  56      0     0     0     0        0        0  169.6   67.1 23.3
## 3  65      1     0     1     0        1        0  171.1   81.0 27.7
## 4  60      0     0     0     0        0        0  170.3   67.1 23.1
## 5  58      0     0     0     0        1        0  178.5   86.3 27.1
## 6  62      1     0     0     0        0        0  180.3   87.8 27.0
##   hypertension diabetes SBP LDL vaccine severity studyB
## 1            0        1 126 125       0        0      0
## 2            0        0 127 120       1        0      0
## 3            1        0 134 127       1        0      1
## 4            1        0 131 106       0        0      0
## 5            0        1 128 101       1        0      0
## 6            1        0 132  93       1        0      1
```

```
y <- training_data$recovery_time # vector of response

# testing data
x2 <- model.matrix(recovery_time ~ .,testing_data)[, -1] # matrix of predictors
y2 <- testing_data$recovery_time # vector of response

# setting a 10-fold cross-validation
ctrl <- trainControl(method = "cv",
                     number = 10,
                     selectionFunction = "best")
```

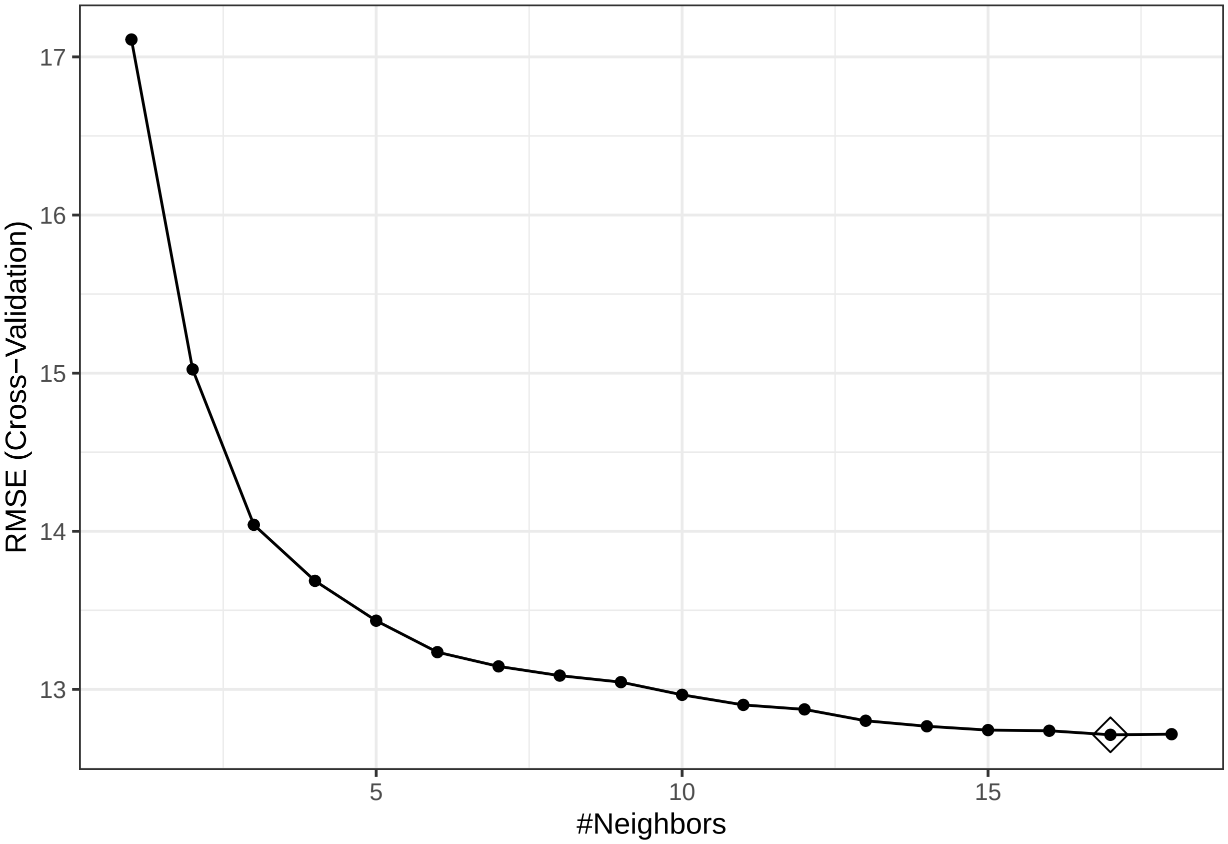
Linear Model

```
set.seed(1234)

# fit a linear model
lm.fit <- train(x, y,
                method = "lm",
                trControl = ctrl)

summary(lm.fit)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -32.606  -7.947  -0.366   7.419  41.007
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -610.42736   82.74540  -7.377 2.26e-13 ***
## age             0.13115    0.06437   2.037 0.041718 *
## gender         -1.79175    0.51499  -3.479 0.000513 ***
## race2           0.09511    1.17945   0.081 0.935735
## race3          -0.07002    0.66065  -0.106 0.915604
## race4          -1.05422    0.91719  -1.149 0.250509
## smoking1        1.62216    0.58159   2.789 0.005329 **
## smoking2        1.84076    0.87570   2.102 0.035660 *
## height          3.67238    0.48468   7.577 5.14e-14 ***
## weight         -3.97488    0.51468  -7.723 1.70e-14 ***
## bmi            12.06885    1.48630   8.120 7.60e-16 ***
## hypertension    2.08798    0.85600   2.439 0.014796 *
## diabetes       -0.74181    0.70678  -1.050 0.294029
## SBP             0.03753    0.05588   0.672 0.501871
## LDL            -0.01596    0.01357  -1.176 0.239635
## vaccine        -4.16566    0.52810  -7.888 4.75e-15 ***
## severity        3.31438    0.85245   3.888 0.000104 ***
## studyB         -3.17935    0.56684  -5.609 2.29e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.21 on 2242 degrees of freedom
## Multiple R-squared:  0.114,  Adjusted R-squared:  0.1073
## F-statistic: 16.97 on 17 and 2242 DF,  p-value: < 2.2e-16
```

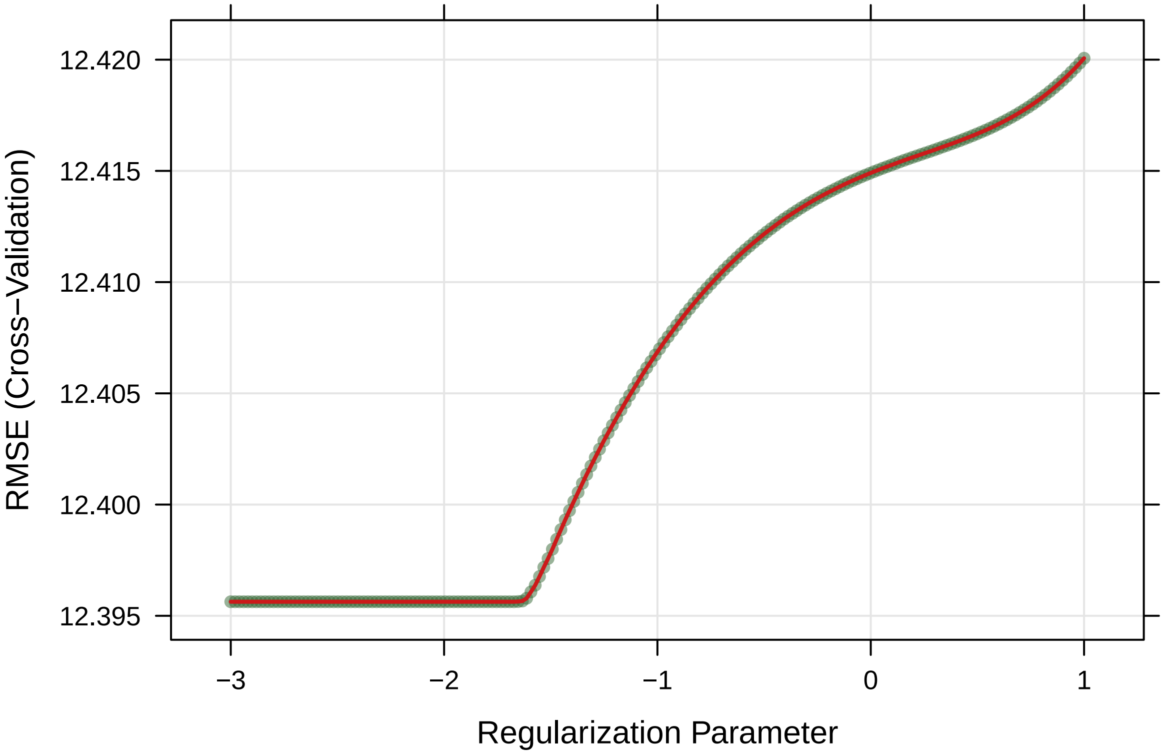
KNN

```
# knn using `caret`
set.seed(1234)

knn.fit <- train(x, y,
                 method = "knn",
                 trControl = ctrl,
                 tuneGrid = expand.grid(k = seq(from = 1, to = 18, by = 1)))

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



Ridge

```
# ridge using `caret`
set.seed(1234)

ridge.fit <- train(x, y,
                   method = "glmnet",
                   tuneGrid = expand.grid(alpha = 0,
                                          lambda = exp(seq(1, -3, length=200))),
                   trControl = ctrl)

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



```
ridge.fit$bestTune
```

```
##    alpha    lambda
## 67     0 0.1876143
```

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

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                          s1
## (Intercept)  -30.49023868
## age            0.13242533
## gender        -1.65395214
## race2          0.18309154
## race3         -0.02022505
## race4         -1.19609142
## smoking1       1.60322256
## smoking2       1.77198128
## height         0.25909783
## weight        -0.34396112
## bmi            1.57632541
## hypertension   1.97627218
## diabetes      -0.67505824
## SBP            0.04202812
## LDL           -0.01516204
## vaccine       -4.04781244
## severity       3.16755179
## studyB        -3.25401415
```

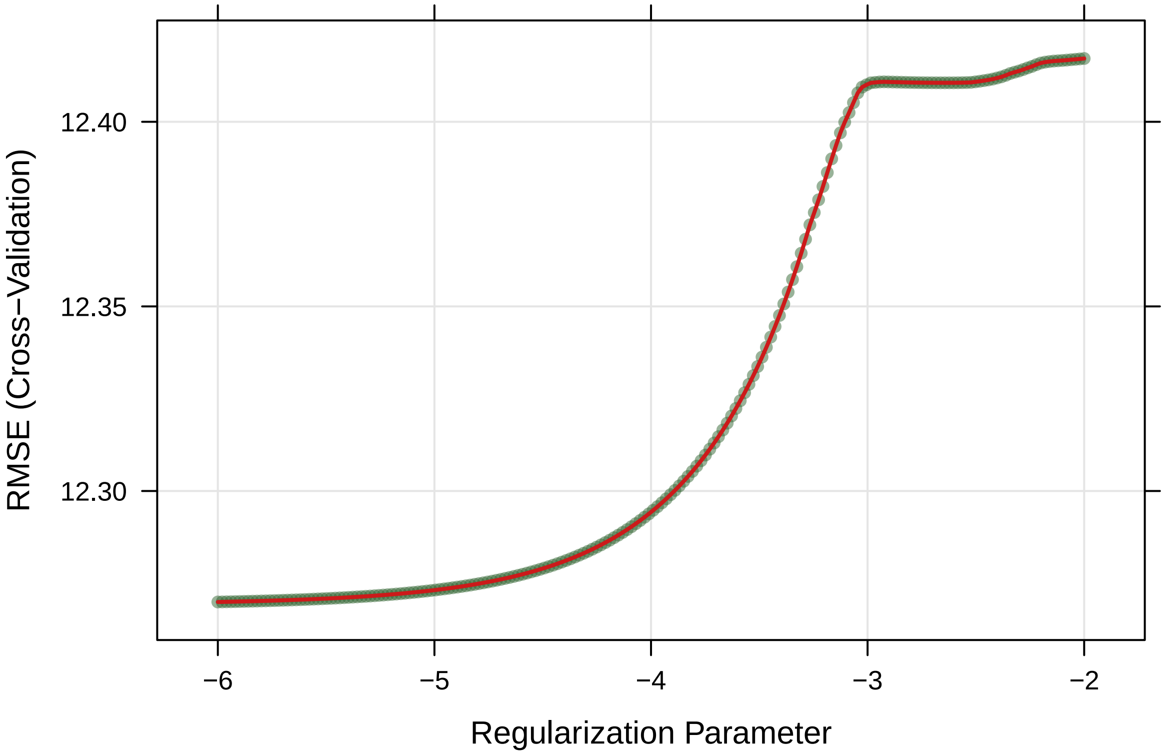
```
ridge.pred <- predict(ridge.fit, newdata = model.matrix(recovery_time ~ ., testing_data)[,-1])
```

```
# test error
mean((ridge.pred - testing_data[, "recovery_time"])^2)
```

```
## [1] 157.8367
```

Lasso

```
set.seed(1234)

# lasso using caret
lasso.fit <- train(x, y,
                   method = "glmnet",
                   tuneGrid = expand.grid(alpha = 1,
                                          lambda = exp(seq(-2, -6, length=200))),
                   trControl = ctrl)

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



```
lasso.fit$bestTune
```

```
##   alpha      lambda
## 1     1 0.002478752
```

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

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                        s1
## (Intercept)  -558.57633451
## age             0.13098133
## gender         -1.77644890
## race2            0.09390994
## race3           -0.05885829
## race4           -1.05801795
## smoking1         1.61611238
## smoking2         1.82699300
## height           3.36730473
## weight          -3.65051514
## bmi             11.13258080
## hypertension     2.08074008
## diabetes        -0.73015772
## SBP              0.03754484
## LDL             -0.01573433
## vaccine         -4.15530518
## severity         3.29781450
## studyB          -3.18500501
```

```
lasso.pred <- predict(lasso.fit, newdata = model.matrix(recovery_time ~ ., testing_data)[,-1])

# test error
mean((lasso.pred - testing_data[, "recovery_time"])^2)
```

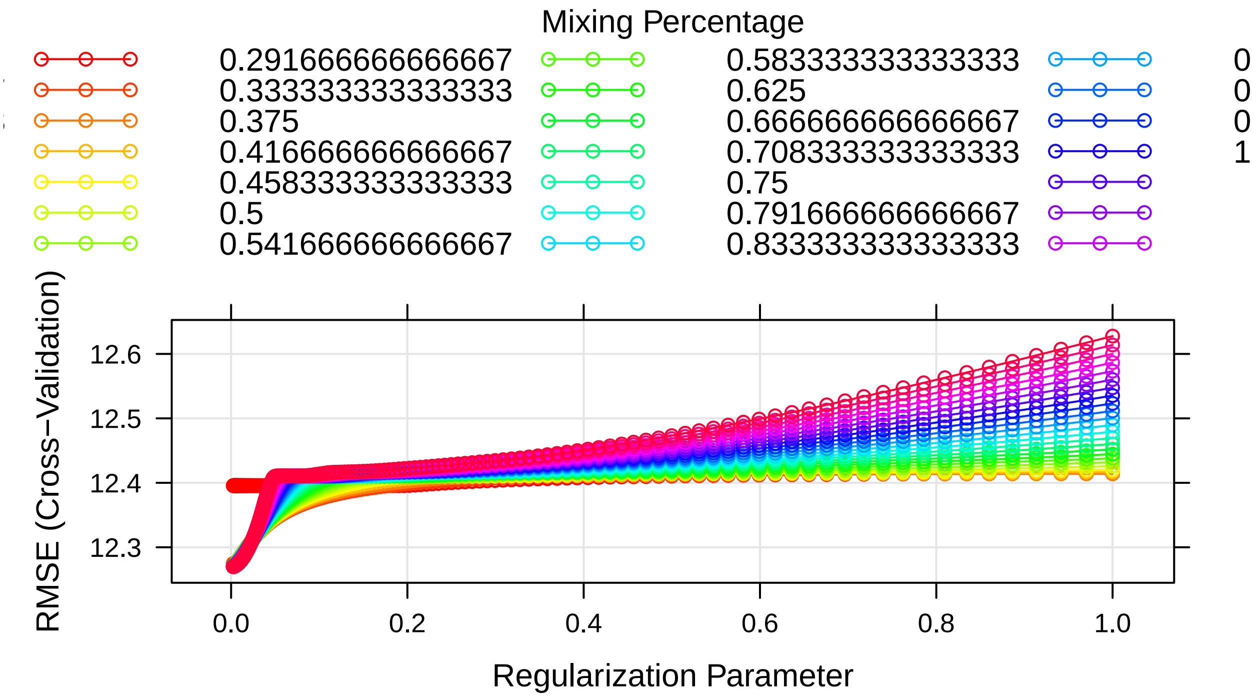
```
## [1] 150.7619
```

Elastic Net

```
set.seed(1234)

# elastic net using caret
enet.fit <- train(x, y,
                  method = "glmnet",
                  tuneGrid = expand.grid(alpha = seq(0, 1, length =
                                                        25),
                                         lambda = exp(seq(0, -6, length=200))),
                  trControl = ctrl)

enet.fit$bestTune
```

```
##      alpha      lambda
## 4801     1 0.002478752
```

```
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)
```

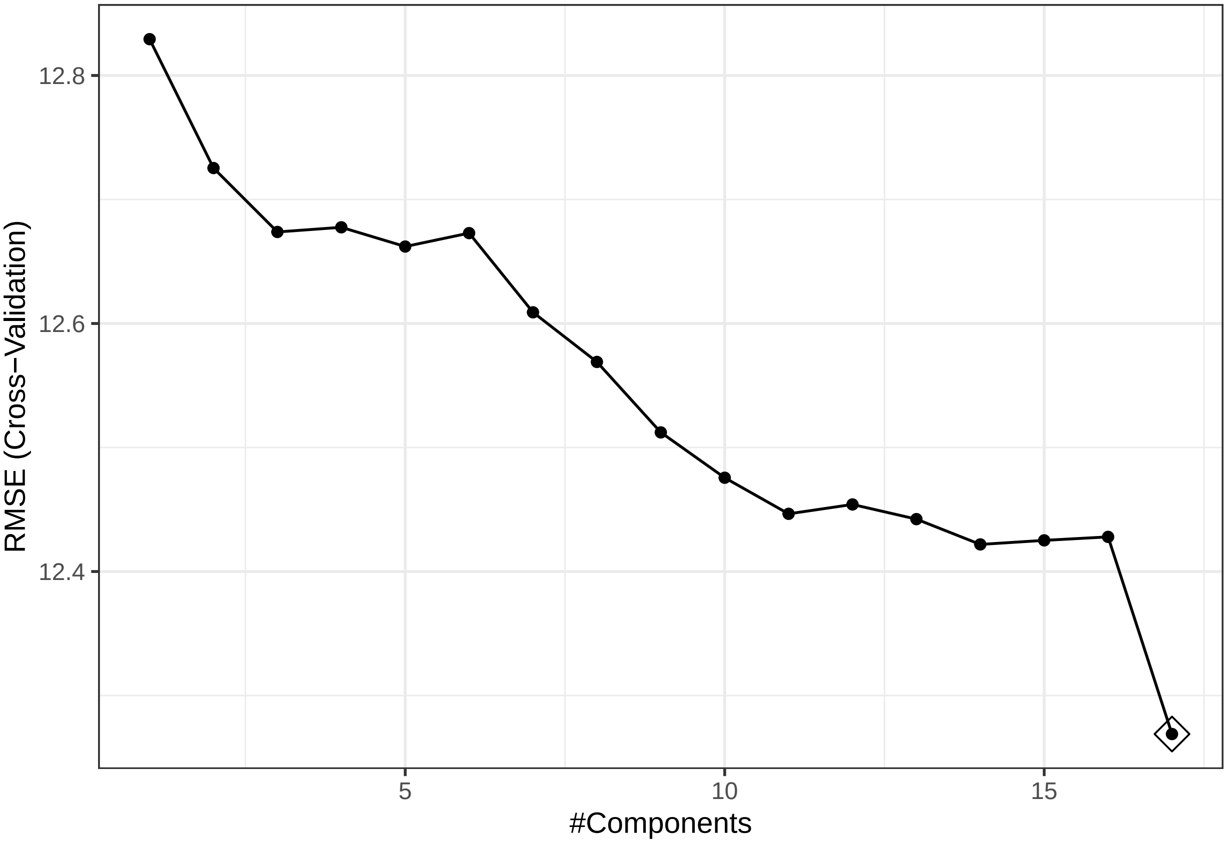
```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                         s1
## (Intercept)  -558.57633451
## age             0.13098133
## gender         -1.77644890
## race2           0.09390994
## race3          -0.05885829
## race4          -1.05801795
## smoking1        1.61611238
## smoking2        1.82699300
## height          3.36730473
## weight         -3.65051514
## bmi            11.13258080
## hypertension    2.08074008
## diabetes       -0.73015772
## SBP             0.03754484
## LDL            -0.01573433
## vaccine        -4.15530518
## severity        3.29781450
## studyB         -3.18500501
```

```
enet.pred <- predict(enet.fit, newdata = model.matrix(recovery_time ~ ., testing_data)[,-1])

# test error
mean((enet.pred - testing_data[, "recovery_time"])^2)
```

```
## [1] 150.7619
```

PCR

```
set.seed(1234)

# pcr using caret
pcr.fit <- train(x, y,
                 method = "pcr",
                 tuneGrid = data.frame(ncomp = 1:17),
                 trControl = ctrl,
                 preProcess = c("center", "scale"))

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

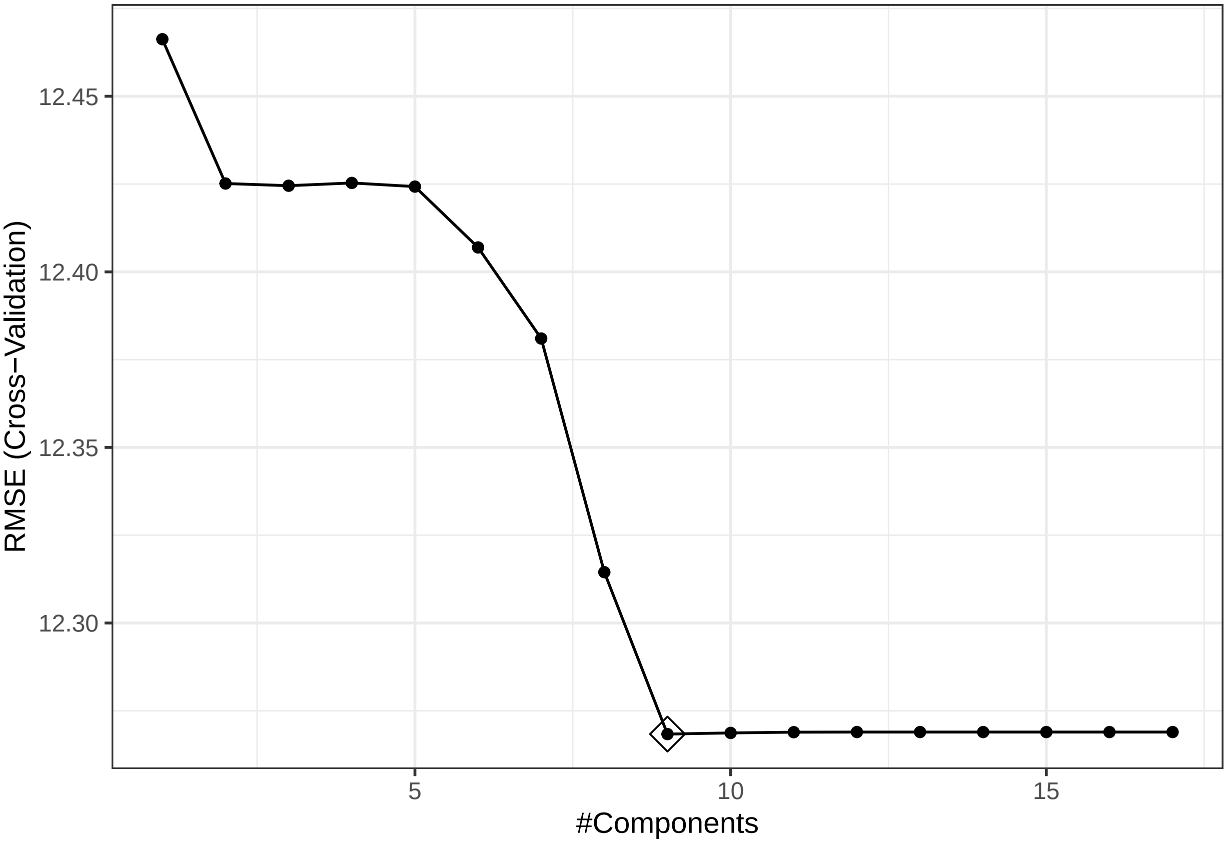


```
pcr.pred <- predict(pcr.fit, newdata = model.matrix(recovery_time ~ ., testing_data)[,-1])

# test error
mean((pcr.pred - testing_data[, "recovery_time"])^2)
```

```
## [1] 150.392
```

PLS

```
set.seed(1234)

# pls using caret
pls.fit <- train(x, y,
                 method = "pls",
                 tuneGrid = data.frame(ncomp = 1:17),
                 trControl = ctrl,
                 preProcess = c("center", "scale"))

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



```
pls.pred <- predict(pls.fit, newdata = model.matrix(recovery_time ~ ., testing_data)[,-1])

# test error
mean((pls.pred - testing_data[, "recovery_time"])^2)
```

```
## [1] 150.3027
```

GAM

```
set.seed(1234)

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

gam.fit$bestTune
```

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

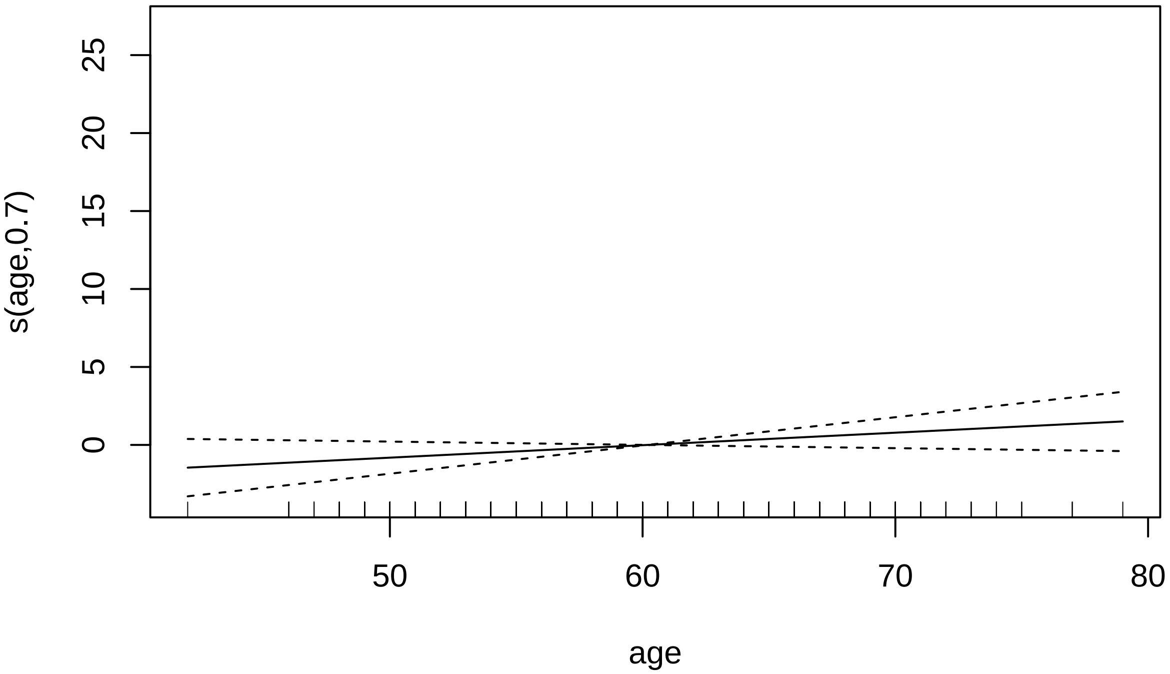
```
gam.fit$finalModel
```

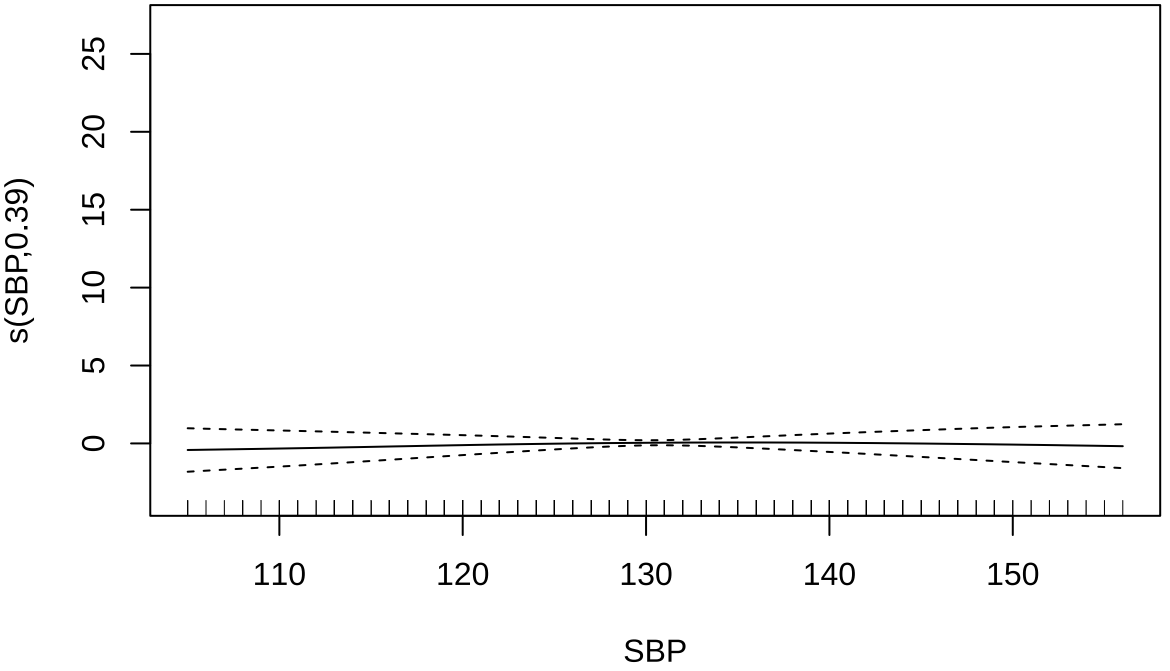
```
## 
## Family: gaussian 
## Link function: identity 
## 
## Formula:
## .outcome ~ gender + race2 + race3 + race4 + smoking1 + smoking2 + 
##     hypertension + diabetes + vaccine + severity + studyB + s(age) + 
##     s(SBP) + s(LDL) + s(bmi) + s(height) + s(weight)
## 
## Estimated degrees of freedom:
## 0.704 0.390 0.000 4.225 0.000 0.154  total = 17.47 
## 
## GCV score: 142.3996
```

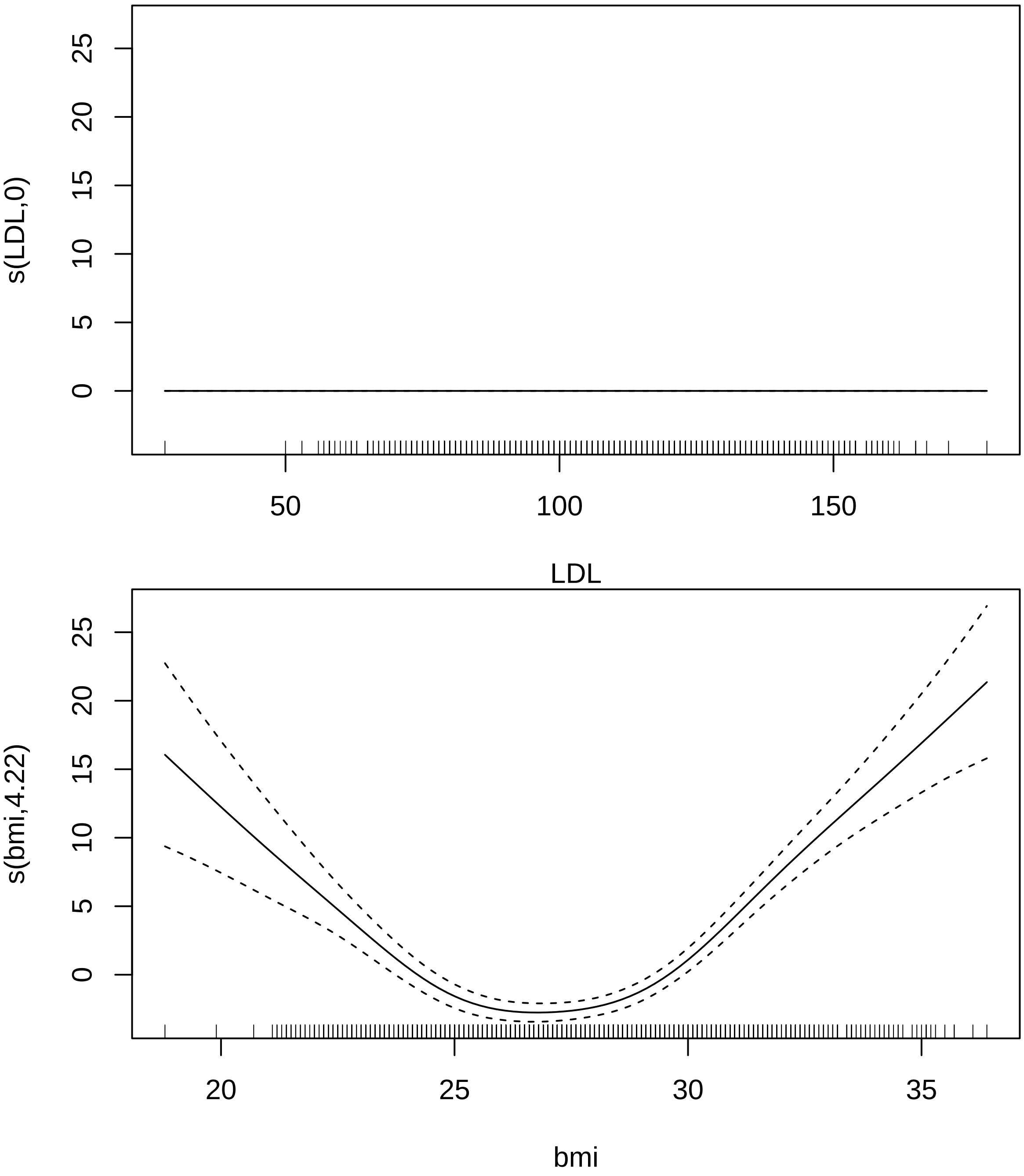
```
coef(gam.fit$finalModel)
```

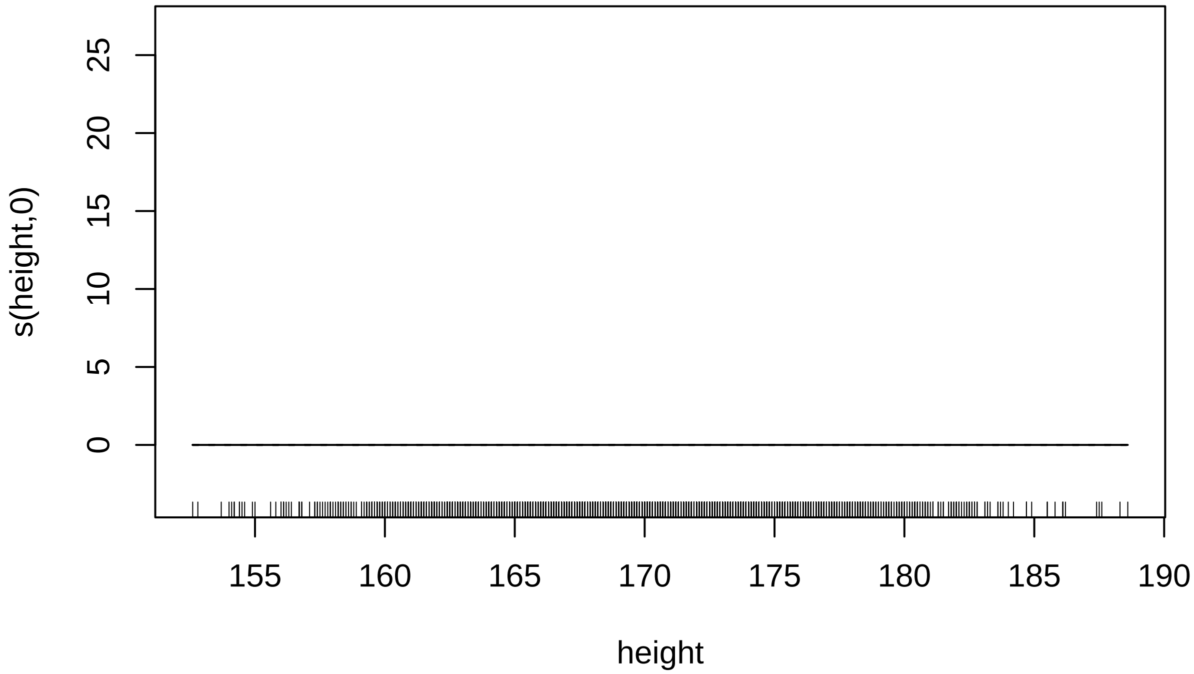
```
##   (Intercept)        gender         race2         race3         race4 
##  4.123351e+01 -1.931404e+00 -2.975488e-01  5.273024e-02 -9.135777e-01 
##      smoking1      smoking2  hypertension      diabetes       vaccine 
##  1.790470e+00  2.069706e+00  2.571950e+00 -9.285276e-01 -4.115053e+00 
##      severity        studyB      s(age).1      s(age).2      s(age).3 
##  3.251854e+00 -3.087649e+00  5.535420e-10 -8.881891e-12 -7.419265e-11 
##      s(age).4      s(age).5      s(age).6      s(age).7      s(age).8 
## -6.394802e-11  5.795303e-11 -4.018612e-11 -4.433160e-11 -7.088224e-11 
##      s(age).9      s(SBP).1      s(SBP).2      s(SBP).3      s(SBP).4 
##  3.629859e-01  3.296078e-02 -1.632182e-03  1.105556e-02 -3.364956e-02 
##      s(SBP).5      s(SBP).6      s(SBP).7      s(SBP).8      s(SBP).9 
## -1.539162e-02 -3.597018e-02  1.642948e-02  2.066440e-01  1.389608e-10 
##      s(LDL).1      s(LDL).2      s(LDL).3      s(LDL).4      s(LDL).5 
##  1.330210e-09 -1.132407e-10  3.335205e-10 -2.176277e-10  1.903933e-10 
##      s(LDL).6      s(LDL).7      s(LDL).8      s(LDL).9      s(bmi).1 
##  1.868300e-10  1.353590e-10  1.425515e-09 -3.907979e-11  3.838852e+00 
##      s(bmi).2      s(bmi).3      s(bmi).4      s(bmi).5      s(bmi).6 
##  1.948281e-01 -2.137512e+00  2.297442e+00 -1.393186e+00 -2.358484e+00 
##      s(bmi).7      s(bmi).8      s(bmi).9   s(height).1   s(height).2 
##  1.934460e+00 -1.300727e+01  1.509188e-11 -9.874895e-10  4.592469e-11 
##   s(height).3   s(height).4   s(height).5   s(height).6   s(height).7 
## -1.234193e-10  4.641937e-11  8.320244e-11  5.841208e-11 -5.236157e-11 
##   s(height).8   s(height).9   s(weight).1   s(weight).2   s(weight).3 
## -5.658923e-11 -9.223178e-11 -4.440034e-02  1.607775e-03  8.086106e-03 
##   s(weight).4   s(weight).5   s(weight).6   s(weight).7   s(weight).8 
##  2.034728e-03 -3.770430e-03  3.154619e-03  2.928095e-03 -5.424874e-03 
##   s(weight).9 
## -1.009868e-09
```

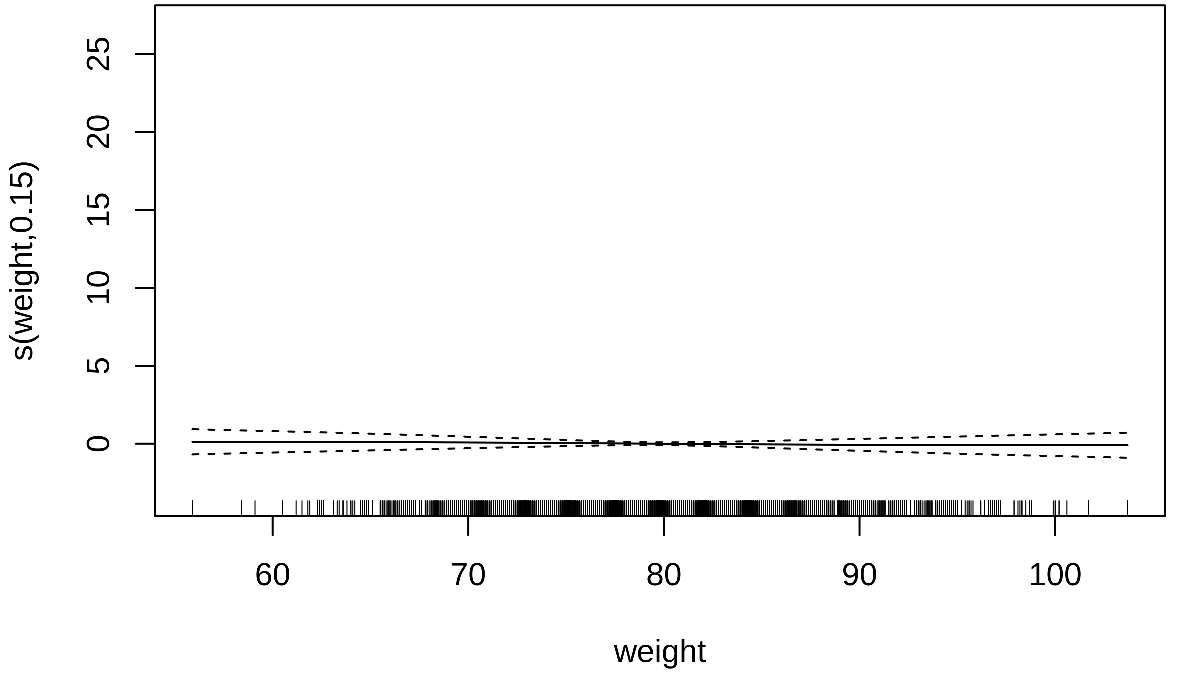
```
plot(gam.fit$finalModel)
```


s(age,0.7)
0
5
10
15
20
25
50
60
70
80
age


s(SBP,0.39)
0
5
10
15
20
25
110
120
130
140
150
SBP


s(LDL,0)
0
5
10
15
20
25
50
100
150
LDL
s(bmi,4.22)
0
5
10
15
20
25
20
25
30
35
bmi





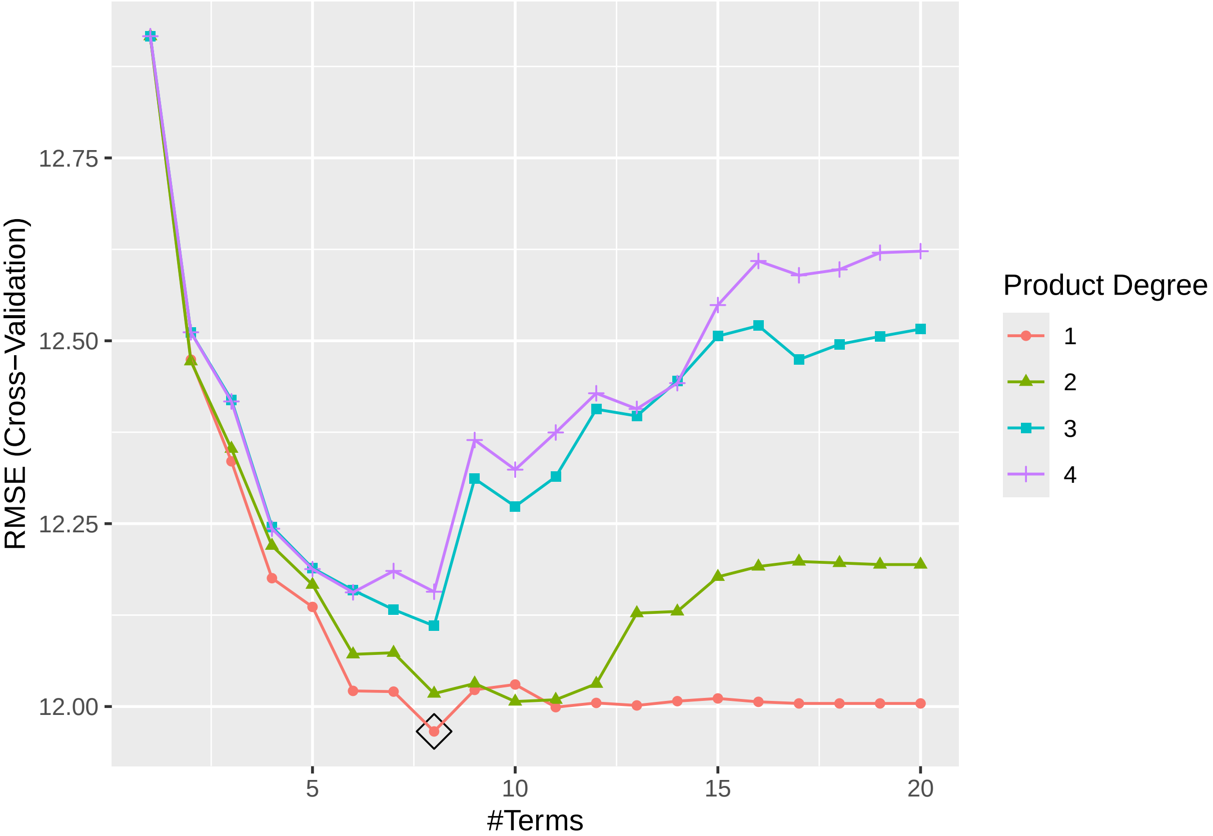
```
gam.pred <- predict(gam.fit, newdata = model.matrix(recovery_time ~ ., testing_data)[,-1])

# test error
mean((gam.pred - testing_data[, "recovery_time"])^2)
```

```
## [1] 141.6812
```

MARS

```
# set grid
mars_grid <- expand.grid(degree = 1:4, nprune = 1:20)
```

```
set.seed(1234)

# fit a MARS model
mars.fit <- train(x, y,
                  method = "earth",
                  tuneGrid = mars_grid,
                  trControl = ctrl)
# plot
ggplot(mars.fit, highlight = TRUE)
```



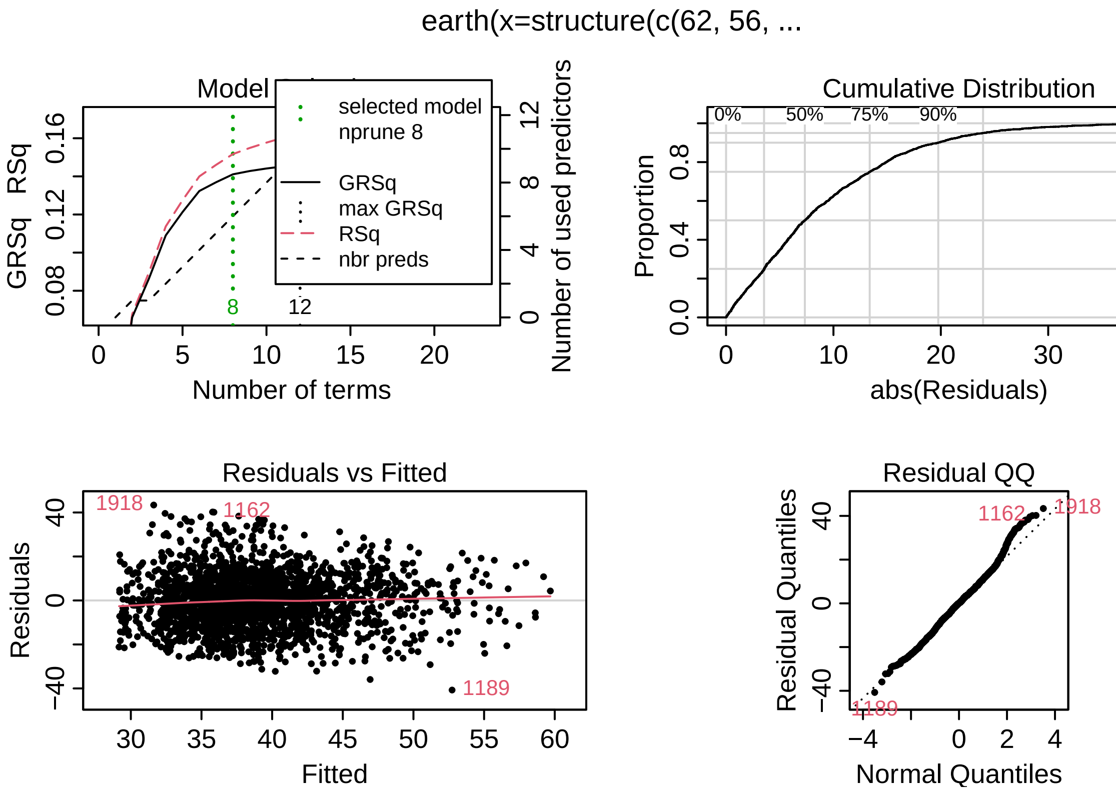
```
# best tuning parameters
mars.fit$bestTune
```

```
##   nprune degree
## 8      8      1
```

```
# regression function
mars.fit$finalModel
```

```
## Selected 8 of 23 terms, and 6 of 17 predictors (nprune=8)
## Termination condition: Reached nk 35
## Importance: bmi, vaccine, hypertension, studyB, severity, gender, ...
## Number of terms at each degree of interaction: 1 7 (additive model)
## GCV 143.4816    RSS 319978.2    GRSq 0.1410789    RSq 0.1516921
```

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



```
# report the regression function
summary(mars.fit)
```

```
## Call: earth(x=matrix[2260,17], y=c(45,19,72,51,3...), keepxy=TRUE, degree=1,
##             nprune=8)
##
##                 coefficients
## (Intercept)        27.320355
## gender             -1.941630
## hypertension        2.979712
## vaccine            -4.115995
## severity            3.283503
## studyB             -3.081043
## h(bmi-24.5)         2.890216
## h(28.3-bmi)         3.356805
##
## Selected 8 of 23 terms, and 6 of 17 predictors (nprune=8)
## Termination condition: Reached nk 35
## Importance: bmi, vaccine, hypertension, studyB, severity, gender, ...
## Number of terms at each degree of interaction: 1 7 (additive model)
## GCV 143.4816    RSS 319978.2    GRSq 0.1410789    RSq 0.1516921
```

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

```
##   (Intercept)  h(28.3-bmi)      vaccine hypertension       studyB  h(bmi-24.5)
##     27.320355     3.356805    -4.115995     2.979712    -3.081043     2.890216
##      severity       gender
##      3.283503    -1.941630
```

```
mars.pred <- predict(pls.fit, newdata = model.matrix(recovery_time ~ ., testing_data)[,-1])

# test error
mean((mars.pred - testing_data[, "recovery_time"])^2)
```

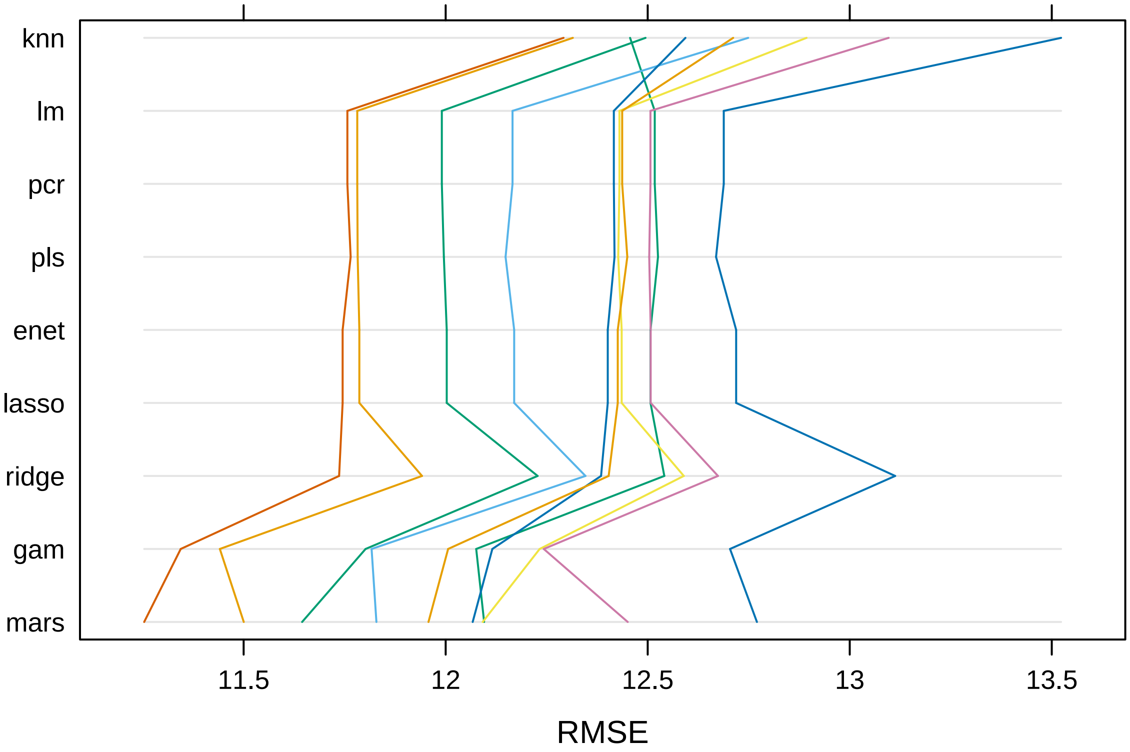
```
## [1] 150.3027
```

Model Comparison

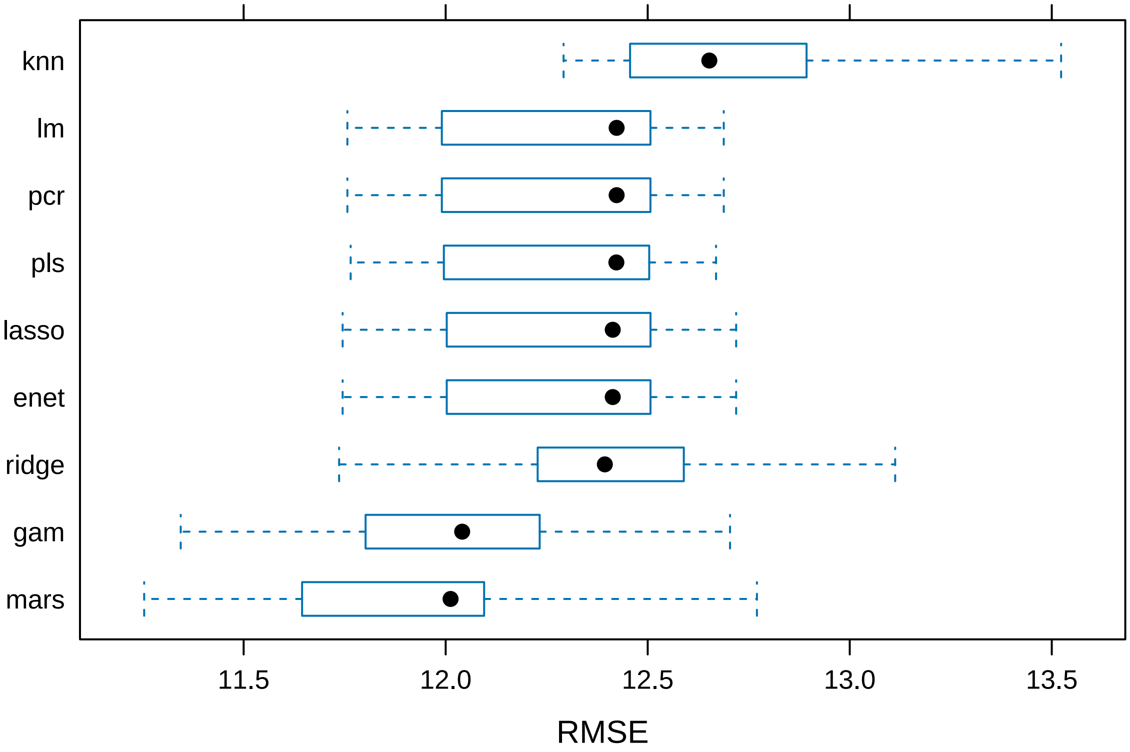
```
# compare models
resamp <- resamples(list(knn = knn.fit, ridge = ridge.fit, lasso = lasso.fit, enet = enet.fit, pcr = pc

summary(resamp)
```

```
##
## Call:
## summary.resamples(object = resamp)
##
## Models: knn, ridge, lasso, enet, pcr, pls, gam, mars, lm
## Number of resamples: 10
##
## MAE
##           Min.  1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## knn   9.594714 9.832788 10.002697 10.032591 10.100388 10.633725    0
## ridge 9.341803 9.523540  9.670731  9.672395  9.885507  9.990540    0
## lasso 9.185240 9.461357  9.584954  9.583899  9.777227  9.859142    0
## enet  9.185240 9.461357  9.584954  9.583899  9.777227  9.859142    0
## pcr   9.185364 9.448089  9.594017  9.583875  9.785251  9.853309    0
## pls   9.192663 9.445604  9.589508  9.581007  9.787731  9.834013    0
## gam   8.828814 9.154373  9.285417  9.324935  9.584407  9.795384    0
## mars  8.848736 9.081705  9.277586  9.295759  9.435707  9.840393    0
## lm    9.185364 9.448089  9.594017  9.583875  9.785251  9.853309    0
##
## RMSE
##           Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## knn   12.29179 12.46598 12.65252 12.71240 12.85704 13.52312    0
## ridge 11.73626 12.25722 12.39396 12.39563 12.57733 13.11248    0
## lasso 11.74499 12.04435 12.41351 12.26994 12.48909 12.71894    0
## enet  11.74499 12.04435 12.41351 12.26994 12.48909 12.71894    0
## pcr   11.75673 12.03423 12.42310 12.26894 12.48932 12.68823    0
## pls   11.76487 12.03372 12.42244 12.26837 12.49022 12.66913    0
## gam   11.34430 11.80555 12.04085 11.97803 12.20338 12.70372    0
## mars  11.25383 11.69066 12.01217 11.96595 12.09428 12.77033    0
## lm    11.75673 12.03423 12.42310 12.26894 12.48932 12.68823    0
##
## Rsquared
##             Min.    1st Qu.     Median       Mean    3rd Qu.       Max. NA's
## knn   0.02272082 0.02713223 0.04120697 0.04456071 0.04572346 0.09749271    0
## ridge 0.04667203 0.06516412 0.07799313 0.08063997 0.09383248 0.12440799    0
## lasso 0.06224598 0.07553564 0.10064334 0.09912251 0.11574606 0.14659394    0
## enet  0.06224598 0.07553564 0.10064334 0.09912251 0.11574606 0.14659394    0
## pcr   0.06146512 0.07450289 0.10089174 0.09951578 0.11892238 0.14691822    0
## pls   0.05986147 0.07401944 0.09995100 0.09959624 0.12097281 0.14673931    0
## gam   0.10031290 0.12075487 0.14322245 0.14135377 0.15371549 0.19436006    0
## mars  0.11392110 0.12028391 0.13635560 0.14316046 0.16578685 0.18614318    0
## lm    0.06146512 0.07450289 0.10089174 0.09951578 0.11892238 0.14691822    0
```

```
parallelplot(resamp, metric = "RMSE")
```



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



MARS has lowest mean and median RMSE -> model I pick. GAM is also a good choice since it incorporates non-linear terms by adding the smoothing function, as well as linear terms. GAM also performs model selection for us. We will compare the performance of GAM and MARS on the testing data to determine which model is best.

Testing Data Analysis

```r
# MARS Prediction
mars.pred = predict(mars.fit, newdata = model.matrix(recovery_time ~ .,testing_data)[,-1])

# MARS test error
mean((mars.pred - testing_data[, "recovery_time"])^2)
```

```
## [1] 144.2482
```

```r
# GAM Prediction
gam.pred = predict(gam.fit, newdata = model.matrix(recovery_time ~ .,testing_data)[,-1])

# GAM test error
mean((gam.pred - testing_data[, "recovery_time"])^2)
```

```
## [1] 141.6812
```

Based on the testing data, we should actually go with the GAM model over the MARS since the test error is lower.