P8106 Data Science II Midterm Project Code: Predicting COVID-19 Recovery Time and Identifying Important Risk Factors

Sarah Forrest - sef2183

4/5/2023

Contents

Background **2**

Data **2**

Exploratory Analysis and Data Visualization: **3**

Model Training **5**

Results **14**

Conclusion **17**

Appendix **17**

```
library(tidyverse)
library(caret)
library(mgcv)
library(earth)
```

Background

To gain a better understanding of the factors that predict recovery time from COVID-19 illness, this study was designed to combine three existing cohort studies that have been tracking participants for several years. This 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

This study uses data from the `recovery.RData` file. The dataset contains a variable for the time from COVID-19 infection to recovery (in days), which is the outcome of interest in this study. It also contains 14 predictor variables, including demographic characteristics, personal characteristics, vital measurements, and disease status. The predictors are a mix of continuous and categorical variables.

The `recovery.RData` file consists of data on 10000 participants. A random sample of 2000 participants was used for analysis. Additionally, the random sample of 2000 participants was further split into training and test datasets using the `createDataPartition()` function in R. The training dataset contained 70% of the sample and the test dataset contained the remaining 30%. All data partitions were conducted using a seed set to my UNI number (2183) for reproducibility.

```
set.seed(2183) # for reproducibility - my UNI

# read in dataset
load("data/recovery.Rdata")

# create a random sample of 2000 participants
dat <- dat[sample(1:10000, 2000),]

# remove the id variable from the dataset
dat <- dat %>%
  select(-id)

# specify rows of training data (70% of the dataset)
trRows <- createDataPartition(dat$recovery_time, p = 0.7, list = FALSE)

# training data
dat_train <- dat[trRows, ]
## matrix of predictors
x <- model.matrix(recovery_time~.,dat)[trRows,-1]
## vector of response
y <- dat$recovery_time[trRows]

# test data
dat_test <- dat[-trRows, ]
```

```
## matrix of predictors
x2 <- model.matrix(recovery_time~.,dat)[-trRows,-1]
## vector of response
y2 <- dat$recovery_time[-trRows]
```

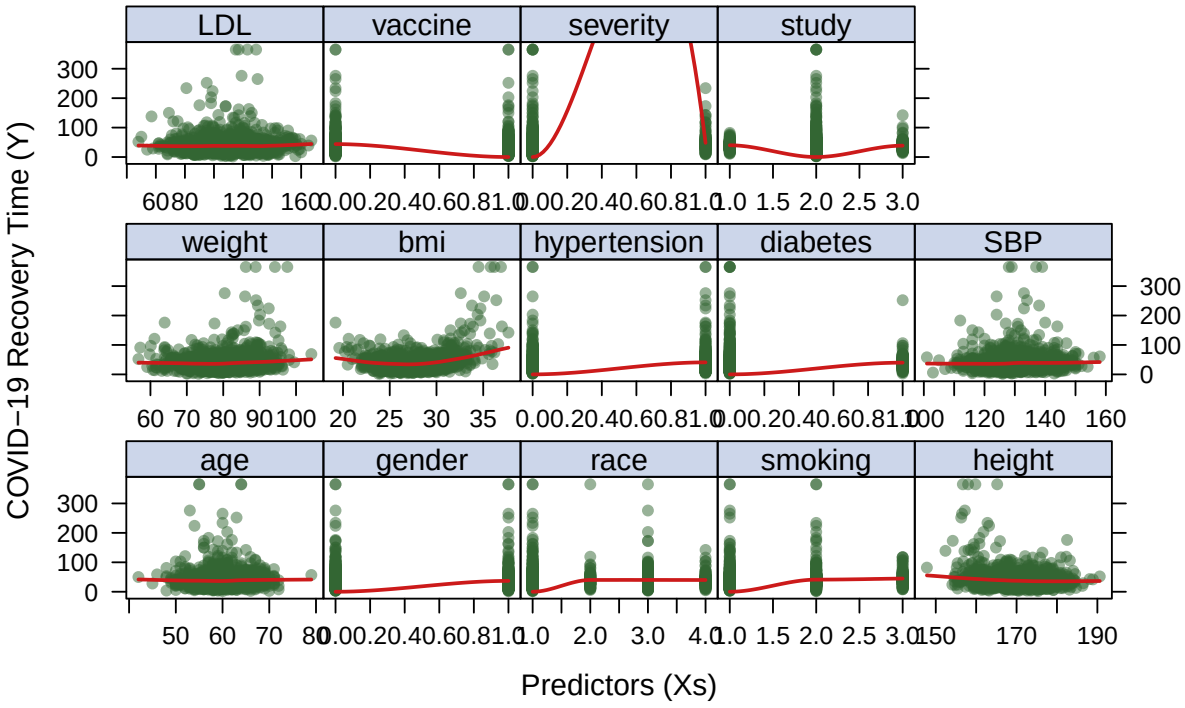
Exploratory Analysis and Data Visualization:

Lattice plots were created using the `featurePlot()` function in the caret package to explore the multivariate dataset and identify patterns or relationships. A plot is created for each of the 14 predictors in the dataset in order to visualize each predictor's association with the outcome, COVID-19 recovery time.

```
# create dataset for exploratory analysis and data visualization
dat_train_viz <- dat_train %>%
  mutate(study = case_when( # turn study (character variable) into a numeric variable
    study == "A" ~ 1,
    study == "B" ~ 2,
    study == "C" ~ 3))

# Find the remaining non-numeric columns
non_numeric_cols <- sapply(dat_train_viz, function(x) !is.numeric(x))

# Convert non-numeric columns to numeric
dat_train_viz[, non_numeric_cols] <- lapply(dat_train_viz[, non_numeric_cols], as.numeric) # turn facto
```

```
# set various graphical parameters (color, line type, background, etc) to control the look of trellis d
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 = dat_train_viz[ ,1:14],
            y = dat_train_viz[ ,15],
            plot = "scatter",
            span = .5,
            labels = c("Predictors (Xs)", "COVID-19 Recovery Time (Y)"),
            main = "Figure 1. Lattice Plot",
            type = c("p", "smooth"))
```

Figure 1. Lattice Plot



Using the lattice plot in Figure 1. the following patterns were observed:

- Patient age does not appear to have a clear association with COVID-19 recovery time. The mostly straight horizontal line shows that both low and high values for age have similar values for recovery time. The age variable appears to be normally distributed, with most data points in the middle between the minimum age (42) and maximum age (79).
- For the gender variable, male patients were assigned a value of 1 and female patients were assigned a value of 0. Therefore, the plot shows that males appear to have a longer recovery time than females in the dataset.
- For the race variable, White patients were assigned a value of 1, Asian patients were assigned 2, Black patients were assigned 3, and Hispanic patients were assigned 4. The plot shows that White patients appear to have a shorter recovery time compared to all other races.
- For the smoking variable, patients who never smoked were assigned a value of 0, former smokers were assigned 1, and current smokers were assigned 2. The plot shows that never smokers appear to have a shorter recovery time compared to former and current smokers. There is a positive association between smoking and recovery time.
- A slight negative association can be observed between patient height and recovery time. Patients with lower height values (shorter patients) appear to have a slightly longer recovery time than patients with higher height values (taller patients).
- A slight positive association can be observed between patient weight and recovery time. Patients with lower weight values appear to have a slightly shorter recovery time than patients with higher weight values.
- A slight U-shaped association can be observed between patient bmi and recovery time. Patients with bmi values near the minimum (19.2) and maximum (37.7) in the dataset have longer recovery times than patients with bmi values in the middle (25-30).
- For the hypertension variable, patients without hypertension were assigned a value of 0 and patients with hypertension were assigned a value of 1. The plot shows that patients with hypertension appear to have a longer recovery time compared to patients without hypertension. There is a positive association

between hypertension and recovery time.
- For the diabetes variable, patients without diabetes were assigned a value of 0 and patients with diabetes were assigned a value of 1. The plot shows that patients with diabetes appear to have a longer recovery time compared to patients without diabetes. There is a positive association between diabetes and recovery time.
- Patient systolic blood pressure (SBP) does not appear to have a clear association with recovery time. The mostly straight horizontal line shows that both low and high values for SBP have similar values for recovery time.
- Patient LDL cholesterol does not appear to have a clear association with recovery time. The mostly straight horizontal line shows that both low and high values for LDL have similar values for recovery time.
- For the vaccine variable, unvaccinated patients at the time of infection were assigned a value of 0 and vaccinated patients were assigned a value of 1. The plot shows that vaccinated patients appear to have a shorter recovery time compared to unvaccinated patients. There is a negative association between vaccination and recovery time.
- For the severity variable, patients without severe COVID-19 infection were assigned a value of 0 and patients with severe COVID-19 infection were assigned a value of 1. The plot shows that patients with severe infections appear to have a longer recovery time compared to patients without severe infections. There is a positive association between severity and recovery time.
- For the study variable, patients in study A were assigned a value of 1, patients in study B were assigned a value of 2, patients in study C were assigned a value of 3. The plot shows that patients in study B had a shorter recovery time compared to patients in studies A and C.

Model Training

Several different models were fit using all predictors to predict time to recovery from COVID-19.

The `train()` function from the caret package was used to fit each model using either the training dataset or a matrix of all the predictors and a vector of the response variable, `recovery_time`, from the training dataset as inputs. Each model was also fit using cross-validation with 10 folds repeated 5-times. The `trainControl()` function was used to specify this cross-validation method, which was called on within the `train()` function using the trControl argument. Additionally, the method argument was used within the `train()` function to specify the type of model to fit. The resulting model object for each model contains the final model (finalModel) and information about the cross-validation performance.

The `predict()` function from the caret package was also used to generate predictions for the test dataset using each final model that was trained using the training dataset. The root mean squared error (RMSE) between the predicted and actual recovery times on the test dataset was calculated in order to evaluate each model's performance and support model comparison.

This approach results in a final model that has been trained using cross-validation, which can help to reduce overfitting and improve the generalization performance of the model on new, unseen data for future prediction purposes.

1. Linear model

A linear model that assumes a linear relationship between the predictor and response variables (linearity), and assumes that the errors are normally distributed (normality) and have constant variance (homoscedasticity),and that the observations are independent of each other (independence). The linear model is the most basic and assumes a linear relationship between the variables, while the other models allow for more flexible relationships. A method = "lm" argument was used within the `train()` function to specify a linear model fit.

```
set.seed(2183)
```

```r
# 10-fold cross-validation repeated 5 times using the best rule
ctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 5)

# Fit a linear regression model using cross-validation on the training dataset
linear_model <- train(recovery_time ~ age + gender + race + smoking + height +
                        weight + bmi + hypertension + diabetes + SBP + LDL +
                        vaccine + severity + study,
                data = dat_train, # training dataset
                method = "lm",
                trControl = ctrl)

# view the model summary and performance on the test set
summary(linear_model$finalModel)
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -100.835  -13.828   -1.023    9.781  286.247
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -3.364e+03  1.976e+02 -17.025  < 2e-16 ***
## age          -2.797e-01  1.804e-01  -1.550 0.121310
## gender       -4.758e+00  1.395e+00  -3.411 0.000666 ***
## race2         2.876e-01  2.966e+00   0.097 0.922767
## race3        -4.189e-01  1.778e+00  -0.236 0.813759
## race4        -1.523e+00  2.509e+00  -0.607 0.543896
## smoking1      4.969e+00  1.563e+00   3.179 0.001509 **
## smoking2      7.510e+00  2.336e+00   3.215 0.001336 **
## height        1.972e+01  1.164e+00  16.939  < 2e-16 ***
## weight       -2.133e+01  1.237e+00 -17.247  < 2e-16 ***
## bmi           6.370e+01  3.513e+00  18.136  < 2e-16 ***
## hypertension  3.905e+00  2.306e+00   1.693 0.090621 .
## diabetes     -2.875e+00  1.966e+00  -1.463 0.143828
## SBP           9.011e-02  1.510e-01   0.597 0.550688
## LDL          -1.173e-02  3.875e-02  -0.303 0.762120
## vaccine      -8.184e+00  1.419e+00  -5.767 9.95e-09 ***
## severity      9.221e+00  2.378e+00   3.877 0.000111 ***
## studyB        5.023e+00  1.809e+00   2.777 0.005558 **
## studyC       -1.602e+00  2.214e+00  -0.724 0.469462
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 25.98 on 1383 degrees of freedom
## Multiple R-squared:  0.3079, Adjusted R-squared:  0.2989
## F-statistic: 34.18 on 18 and 1383 DF,  p-value: < 2.2e-16

# view performance on the test set (RMSE)
test_pred <- predict(linear_model, newdata = dat_test) # test dataset
test_rmse <- sqrt(mean((test_pred - dat_test$recovery_time)^2))
test_rmse
```

```
## [1] 27.30982
```

2. Lasso model

A lasso model is a linear regression model that adds a penalty term to the sum of absolute values of the coefficients to prevent overfitting, which can lead to sparse models by shrinking some coefficients to zero. It has the same assumptions as the linear model. A method = "glmnet" argument was used within the `train()` function to specify a lasso model fit. Additionally, a grid of tuning parameters for the model were set using the tuneGrid argument within the `train()` function. The grid contains 100 values of the lambda parameter, ranging from exp(-1) to exp(5) with equal intervals on the log scale. The alpha parameter is set to 1, indicating that we are only considering Lasso regularization. The `expand.grid()` function is used to generate all possible combinations of alpha and lambda in the grid for tuning.

```
set.seed(2183)

lasso_model <- train(x, y, # training dataset
                     method = "glmnet",
                     tuneGrid = expand.grid(alpha = 1,
                                            lambda = exp(seq(-1, 5, length = 100))),
                     trControl = ctrl)

# view the model summary
summary(lasso_model$finalModel)
##             Length Class      Mode
## a0             93   -none-     numeric
## beta         1674   dgCMatrix  S4
## df             93   -none-     numeric
## dim             2   -none-     numeric
## lambda         93   -none-     numeric
## dev.ratio      93   -none-     numeric
## nulldev         1   -none-     numeric
## npasses         1   -none-     numeric
## jerr            1   -none-     numeric
## offset          1   -none-     logical
## call            5   -none-     call
## nobs            1   -none-     numeric
## lambdaOpt       1   -none-     numeric
## xNames         18   -none-     character
## problemType     1   -none-     character
## tuneValue       2   data.frame list
## obsLevels       1   -none-     logical
## param           0   -none-     list

# view performance on the test set (RMSE)
test_pred <- predict(lasso_model, newdata = x2) # test dataset
test_rmse <- sqrt(mean((test_pred - dat_test$recovery_time)^2))
test_rmse
## [1] 27.73567
```

3. Elastic net model

An elastic net model is linear regression model that combines the penalties of the lasso and ridge regression methods to prevent overfitting, which can result in better prediction accuracy than either method alone. It has the same assumptions as the linear model. A method = "glmnet" argument was used within the

`train()` function to specify a elastic net model fit. Additionally, a grid of tuning parameters for the model were set using the tuneGrid argument within the `train()` function. The grid includes 21 values of the alpha parameter, ranging from 0 to 1 with equal intervals. The lambda parameter is set to a sequence of 50 values, ranging from exp(2) to exp(-2) with equal intervals on the log scale. The expand.grid() function generates all possible combinations of alpha and lambda in the grid for tuning.

```
set.seed(2183)

enet_model <- train(x, y, # training dataset
                    method = "glmnet",
                    tuneGrid = expand.grid(alpha = seq(0, 1, length = 21),
                                           lambda = exp(seq(2, -2, length = 50))),
                    trControl = ctrl)

# view the model summary
summary(enet_model$finalModel)
##             Length Class      Mode
## a0            93   -none-     numeric
## beta        1674   dgCMatrix  S4
## df            93   -none-     numeric
## dim            2   -none-     numeric
## lambda        93   -none-     numeric
## dev.ratio     93   -none-     numeric
## nulldev        1   -none-     numeric
## npasses        1   -none-     numeric
## jerr           1   -none-     numeric
## offset         1   -none-     logical
## call           5   -none-     call
## nobs           1   -none-     numeric
## lambdaOpt      1   -none-     numeric
## xNames        18   -none-     character
## problemType    1   -none-     character
## tuneValue      2   data.frame list
## obsLevels      1   -none-     logical
## param          0   -none-     list

# view performance on the test set (RMSE)
test_pred <- predict(enet_model, newdata = x2) # test dataset
test_rmse <- sqrt(mean((test_pred - dat_test$recovery_time)^2))
test_rmse
## [1] 26.94222
```

4. Partial least squares (PLS) model

A PLS model is a model that seeks to find a low-dimensional representation of the predictor variables that explains the maximum variance in the response variable. It has the same assumptions as the linear model as well as a latent variable assumption, the assumption that the predictor variables are linearly related to the response variable via a set of underlying latent variables. A method = "pls" argument was used within the `train()` function to specify a PLS model fit. Additionally, a tuning parameter for the model were set using the tuneGrid argument within the `train()` function. The tuneGrid object includes a data frame with a single column ncomp that ranges from 1 to 17, representing the number of components used in the model. The preProcess argument is set to "center" and "scale", which means that the training data will be centered and scaled prior to model fitting.

```
set.seed(2183)

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

# view the model summary
summary(pls_model$finalModel)
## Data:    X dimension: 1402 18
##  Y dimension: 1402 1
## Fit method: oscorespls
## Number of components considered: 11
## TRAINING: % variance explained
##           1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps
## X           9.37    15.87    24.74    32.06    38.04    43.55    47.79
## .outcome   14.34    15.29    15.44    15.63    16.10    17.31    21.19
##           8 comps  9 comps  10 comps  11 comps
## X           50.94    54.89     57.94     62.26
## .outcome    26.60    28.97     30.56     30.78

# view performance on the test set (RMSE)
test_pred <- predict(pls_model, newdata = x2) # test dataset
test_rmse <- sqrt(mean((test_pred - dat_test$recovery_time)^2))
test_rmse
## [1] 27.32134
```

5. Generalized additive model (GAM)

A GAM model is a model that extends the linear model by allowing for nonlinear relationships between the predictor and response variables, using flexible functions called splines. It has the same assumptions as the linear model. A method = "gam" argument was used within the `train()` function to specify a GAM fit. Additionally, two separate GAM models were fit: one GAM model using all predictors, and one GAM model with variable selection performed during model training. For the variable selection GAM model, a tuneGrid object was specified within the `train()` function. The tuneGrid object includes a data frame with two arguments: method is set to "GCV.Cp", which represents the generalized cross-validation with the Cp criterion for smoothing parameter selection, and select is set to TRUE, indicating that variable selection will be performed during model training.

```
set.seed(2183)

# fit GAM model using all predictors
gam_model_all <- train(x, y,# training dataset
                   method = "gam",
                   trControl = ctrl,
                   control = gam.control(maxit = 200)) # Adjusted due to failure to converge at default s

# view the model summary
summary(gam_model_all$finalModel)
##
## Family: gaussian
## Link function: identity
##
```

```
## Formula:
## .outcome ~ gender + race2 + race3 + race4 + smoking1 + smoking2 +
##     hypertension + diabetes + vaccine + severity + studyB + studyC +
##     s(age) + s(SBP) + s(LDL) + s(bmi) + s(height) + s(weight)
##
## Parametric coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   42.6484     2.1195  20.122  < 2e-16 ***
## gender        -5.3039     1.2736  -4.165 3.31e-05 ***
## race2         -0.9167     2.6968  -0.340 0.733965
## race3          0.6748     1.6198   0.417 0.677042
## race4         -1.7393     2.2845  -0.761 0.446589
## smoking1       4.6506     1.4235   3.267 0.001114 **
## smoking2       8.0189     2.1284   3.768 0.000172 ***
## hypertension   5.2189     2.0964   2.490 0.012910 *
## diabetes      -2.1693     1.7875  -1.214 0.225124
## vaccine       -8.0292     1.2915  -6.217 6.72e-10 ***
## severity       9.5517     2.1691   4.404 1.15e-05 ***
## studyB         4.2897     1.6510   2.598 0.009470 **
## studyC        -1.5110     2.0152  -0.750 0.453522
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##             edf Ref.df      F  p-value
## s(age)    1.000  1.000  1.640  0.20058
## s(SBP)    1.000  1.000  0.034  0.85404
## s(LDL)    1.000  1.000  1.009  0.31524
## s(bmi)    8.795  8.985 66.151  < 2e-16 ***
## s(height) 7.741  8.605  4.054 5.81e-05 ***
## s(weight) 1.000  1.000  7.639  0.00579 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.425   Deviance explained = 43.8%
## GCV = 567.18  Scale est. = 553.61    n = 1402

# fit GAM model using selection specification
gam_model_select <- train(x, y, # training dataset
                method = "gam",
                tuneGrid = data.frame(method = "GCV.Cp", select = c(TRUE)),
                trControl = ctrl, # 10-fold CV
                control = gam.control(maxit = 200))  # Adjusted due to failure to converge at default

# view the model summary
summary(gam_model_select$finalModel)
##
## Family: gaussian
## Link function: identity
##
## Formula:
## .outcome ~ gender + race2 + race3 + race4 + smoking1 + smoking2 +
##     hypertension + diabetes + vaccine + severity + studyB + studyC +
```

```
##     s(age) + s(SBP) + s(LDL) + s(bmi) + s(height) + s(weight)
##
## Parametric coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   42.7218     1.9674  21.714  < 2e-16 ***
## gender        -5.3009     1.2722  -4.167 3.28e-05 ***
## race2         -0.8627     2.6916  -0.321 0.748624
## race3          0.6957     1.6182   0.430 0.667335
## race4         -1.7361     2.2804  -0.761 0.446606
## smoking1       4.6688     1.4214   3.285 0.001047 **
## smoking2       8.0617     2.1250   3.794 0.000155 ***
## hypertension   5.0312     1.3159   3.823 0.000138 ***
## diabetes      -2.2040     1.7854  -1.234 0.217247
## vaccine       -7.9340     1.2877  -6.161 9.46e-10 ***
## severity       9.5456     2.1636   4.412 1.10e-05 ***
## studyB         4.2338     1.6469   2.571 0.010252 *
## studyC        -1.6042     2.0078  -0.799 0.424426
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                 edf Ref.df      F  p-value
## s(age)    5.026e-01      9  0.121 0.139808
## s(SBP)    2.165e-07      9  0.000 0.969829
## s(LDL)    1.859e-01      9  0.027 0.247466
## s(bmi)    7.715e+00      9 65.842  < 2e-16 ***
## s(height) 7.726e+00      9  4.179 3.17e-06 ***
## s(weight) 1.000e+00      9  1.436 0.000232 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.426   Deviance explained = 43.8%
## GCV = 565.13  Scale est. = 552.99    n = 1402

# view performance on the test set (RMSE) for the model with all predictors
test_pred <- predict(gam_model_all, newdata = x2) # test dataset
test_rmse <- sqrt(mean((test_pred - dat_test$recovery_time)^2))
test_rmse
## [1] 23.78453

# view performance on the test set (RMSE) for the model with select predictors
test_pred <- predict(gam_model_select, newdata = x2) # test dataset
test_rmse <- sqrt(mean((test_pred - dat_test$recovery_time)^2))
test_rmse
## [1] 23.7401
```

6. Multivariate adaptive regression spline (MARS) model

A MARS model is a model that uses piecewise linear or nonlinear functions to model the relationship between the predictor and response variables, which can capture complex nonlinear relationships. It has the same assumptions as the linear model. A method = "earth" argument was used within the `train()` function to specify a MARS model fit. Additionally, the `expand.grid()` function is used to generate a grid of tuning parameters. The mars_grid object includes two arguments: degree is set to 1, 2, and 3, representing the number of possible product hinge functions in a single term, and nprune is set to integers between 2 and 17,

representing the upper bound on the number of terms in the model. The tuneGrid argument in the `train()` function uses the mars_grid object to specify the parameters for model tuning.

```
set.seed(2183)

# create grid of all possible pairs that can take degree and nprune values
mars_grid <- expand.grid(degree = 1:3, # number of possible product hinge functions in 1 term
                         nprune = 2:17) # Upper bound of number of terms in model

mars_model <- train(x, y, # training dataset
                    method = "earth",
                    tuneGrid = mars_grid,
                    trControl = ctrl)

# view the model summary
summary(mars_model$finalModel)
## Call: earth(x=matrix[1402,18], y=c(44,49,38,46,5...), keepxy=TRUE, degree=2,
##             nprune=4)
##
##                       coefficients
## (Intercept)             -43.33869
## h(bmi-23.5)              10.92775
## h(30.9-bmi)              10.82866
## h(bmi-30.9) * studyB     22.96750
##
## Selected 4 of 22 terms, and 2 of 18 predictors (nprune=4)
## Termination condition: Reached nk 37
## Importance: bmi, studyB, age-unused, gender-unused, race2-unused, ...
## Number of terms at each degree of interaction: 1 2 1
## GCV 542.6388    RSS 751582.7    GRSq 0.4367837    RSq 0.4427977

# view performance on the test set (RMSE)
test_pred <- predict(mars_model, newdata = x2) # test dataset
test_rmse <- sqrt(mean((test_pred - dat_test$recovery_time)^2))
test_rmse
## [1] 23.06663
```

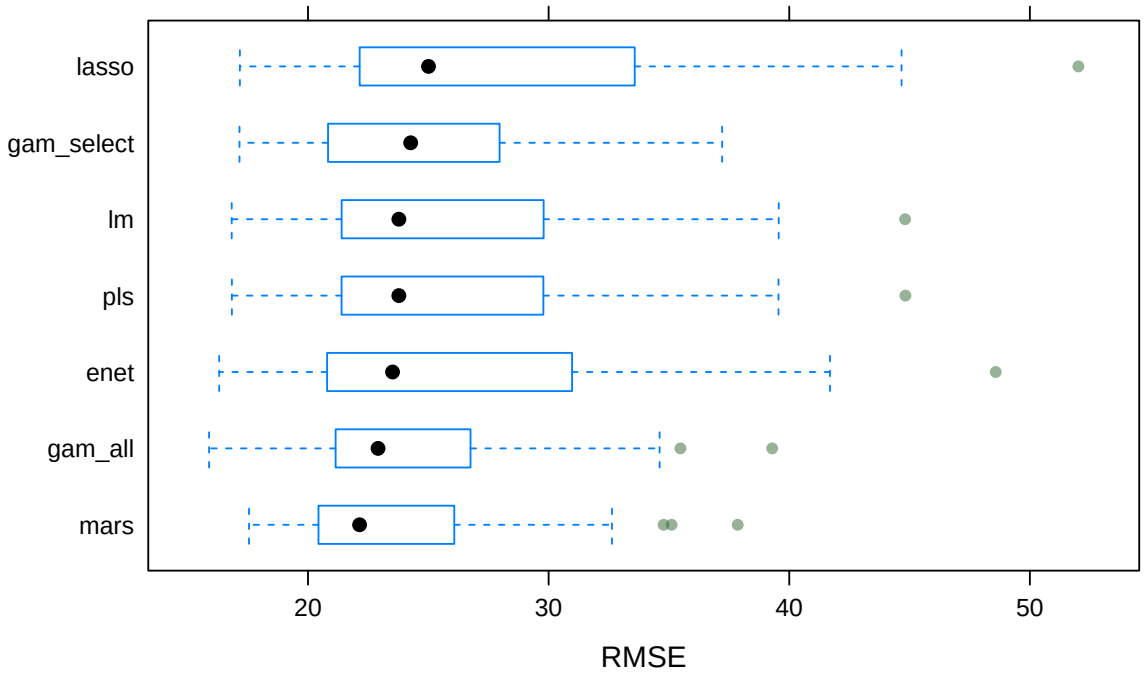
Model comparison

The models were assessed by comparing the RMSE for each model. Low RMSE indicates the best performing model (low prediction error), while high RMSE indicates the worst performing model (high prediction error).

```
set.seed(2183)

resamp <- resamples(list(
  lm = linear_model,
  lasso = lasso_model,
  enet = enet_model,
  pls = pls_model,
  gam_all = gam_model_all,
  gam_select = gam_model_select,
  mars = mars_model
  ))
```

```
summary(resamp)
##
## Call:
## summary.resamples(object = resamp)
##
## Models: lm, lasso, enet, pls, gam_all, gam_select, mars
## Number of resamples: 50
##
## MAE
##                Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## lm         12.73505 15.39825 16.12348 16.47142 17.75496 20.12710    0
## lasso      13.46116 15.56922 16.52594 16.76562 17.59255 21.58307    0
## enet       12.80380 14.94888 16.15967 16.06836 17.07309 20.37231    0
## pls        12.74936 15.38941 16.13187 16.47089 17.75006 20.13645    0
## gam_all    12.23868 14.72597 15.56555 15.59514 16.25176 18.68100    0
## gam_select 12.80144 14.36029 15.49947 15.61425 16.51267 19.37225    0
## mars       13.36260 14.03692 15.32847 15.39812 16.31531 19.08560    0
##
## RMSE
##                Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## lm         16.82876 21.40034 23.77686 25.68307 29.70809 44.81248    0
## lasso      17.16745 22.15436 25.00758 27.85853 33.52794 52.01572    0
## enet       16.30816 20.86376 23.51353 26.01873 30.95194 48.57930    0
## pls        16.83601 21.40035 23.77552 25.68226 29.70413 44.82724    0
## gam_all    15.88988 21.15587 22.91025 24.37900 26.73050 39.29095    0
## gam_select 17.15290 20.86770 24.26783 24.58489 27.94064 37.21158    0
## mars       17.54908 20.46545 22.14328 23.64786 25.93674 37.85683    0
##
## Rsquared
##                  Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lm         0.05285722 0.2198029 0.2767746 0.2885150 0.3387478 0.5731047    0
## lasso      0.01531994 0.1044821 0.1556263 0.1561066 0.1878872 0.3803743    0
## enet       0.04358373 0.2111552 0.2718273 0.2694211 0.3090849 0.5422976    0
## pls        0.05348801 0.2197665 0.2766038 0.2885046 0.3358747 0.5727813    0
## gam_all    0.10729883 0.2735852 0.3530194 0.3695334 0.4524725 0.6629995    0
## gam_select 0.09078299 0.2964500 0.3537030 0.3728204 0.4720036 0.6360189    0
## mars       0.04978937 0.2303193 0.3635106 0.3824594 0.5128183 0.7686433    0

bwplot(resamp,
       metric = "RMSE",
       main = "Figure 2. Model Comparison Plot Using RMSE")
```

Figure 2. Model Comparison Plot Using RMSE



According to Figure 2, the lasso model had the highest median training RMSE (i.e., worst performance), followed by the GAM model using select predictors, the linear model, the pls model, the elastic net model, the GAM model using all predictors, and finally, The MARS model, which had the lowest median training RMSE (i.e., best performance).

The final model for predicting time to recovery from COVID-19 was selected by comparing the median training RMSE for all models created. The MARS model had the lowest value for mean and median RMSE out of all models, however, it only contained 4 terms with 2 of the predictors. A model with so few predictors would not be an ideal model for predicting recovery time and comprehensively identifying important risk factors for long recovery time, as so few risk factors were included. Therefore, the GAM model will all predictors–the second best model in terms of mean and median RMSE–was selected as the final model for predicting time to recovery from COVID-19 in this study.

Results

Model formula

```
# view the model summary for the GAM model using all predictors
summary(gam_model_all$finalModel)
##
## Family: gaussian
## Link function: identity
##
## Formula:
## .outcome ~ gender + race2 + race3 + race4 + smoking1 + smoking2 +
##     hypertension + diabetes + vaccine + severity + studyB + studyC +
##     s(age) + s(SBP) + s(LDL) + s(bmi) + s(height) + s(weight)
```

```
##
## Parametric coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   42.6484     2.1195  20.122  < 2e-16 ***
## gender        -5.3039     1.2736  -4.165 3.31e-05 ***
## race2         -0.9167     2.6968  -0.340 0.733965
## race3          0.6748     1.6198   0.417 0.677042
## race4         -1.7393     2.2845  -0.761 0.446589
## smoking1       4.6506     1.4235   3.267 0.001114 **
## smoking2       8.0189     2.1284   3.768 0.000172 ***
## hypertension   5.2189     2.0964   2.490 0.012910 *
## diabetes      -2.1693     1.7875  -1.214 0.225124
## vaccine       -8.0292     1.2915  -6.217 6.72e-10 ***
## severity       9.5517     2.1691   4.404 1.15e-05 ***
## studyB         4.2897     1.6510   2.598 0.009470 **
## studyC        -1.5110     2.0152  -0.750 0.453522
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##             edf Ref.df      F  p-value
## s(age)    1.000  1.000  1.640  0.20058
## s(SBP)    1.000  1.000  0.034  0.85404
## s(LDL)    1.000  1.000  1.009  0.31524
## s(bmi)    8.795  8.985 66.151  < 2e-16 ***
## s(height) 7.741  8.605  4.054 5.81e-05 ***
## s(weight) 1.000  1.000  7.639  0.00579 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.425   Deviance explained = 43.8%
## GCV = 567.18  Scale est. = 553.61    n = 1402
```

The final GAM model formula uses the `recovery_time` variable as the outcome (Y), and the following terms as predictors. Note that "White" (`race` = 0) was set as the reference category for the `race` variable, "Never Smoker" (`smoking` = 0) was set as the reference category for the `smoking` variable, and "Study A" (`study` = A) was set as the reference category for the `study` variable. An s() around the variable name indicates that a smoothing function was applied to the variable. An asterisk (*) next to the term indicates that the term was statistically significant at the 5% level of significance.

- `gender`*
- Race: Asian (`race2`)
- Race: Black (`race3`)
- Race: Hispanic (`race4`)
- Smoking: former smoker (`smoking1`)*
- Smoking: current smoker (`smoking2`)*
- `hypertension`*
- `diabetes`
- `vaccine`*
- `severity`*
- Study: B (`studyB`)*
- Study: C (`studyC`)
- s(`age`)

- s(SBP)
- s(LDL)
- s(bmi)*
- s(height)*
- s(weight)*

Model interpretation

- On average, being male is associated with a statistically significantly shorter predicted COVID-19 recovery time compared to being female. More specifically, holding all else constant, being male is associated with a 5.3039-day shorter predicted recovery time than being female.
- On average, being a former smoker is associated with a statistically significantly longer predicted COVID-19 recovery time compared to being a never smoker. More specifically, holding all else constant, former smoking is associated with a 4.6506-day longer predicted recovery time than never smoking.
- On average, being a current smoker is associated with a statistically significantly longer predicted COVID-19 recovery time compared to being a never smoker. More specifically, holding all else constant, current smoking is associated with a 8.0189-day longer predicted recovery time than never smoking.
- On average, having hypertension is associated with a statistically significantly longer predicted COVID-19 recovery time compared to not having hypertension. More specifically, holding all else constant, a hypertension diagnosis is associated with a 5.2189-day longer predicted recovery time than no hypertension diagnosis.
- On average, being vaccinated is associated with a statistically significantly shorter predicted COVID-19 recovery time compared to not being vaccinated. More specifically, holding all else constant, vaccination is associated with a 8.0292-day shorter predicted recovery time than no vaccination.
- On average, severe COVID-19 infection is associated with a statistically significantly longer predicted COVID-19 recovery time compared to non severe infection. More specifically, holding all else constant, infection severity is associated with a 9.5517-day longer predicted recovery time than no infection severity.
- On average, being in study B is associated with a statistically significantly longer predicted COVID-19 recovery time compared to study A. More specifically, holding all else constant, study B is associated with a 4.2897-day longer predicted recovery time than study A.

The GAM model also includes several significant smooth terms to capture the nonlinear relationships between the predictors and COVID-19 recovery time. There is a statistically significant relationship between BMI, weight, and height, and predicted COVID-19 recovery time.

Model performance

```
# print the GAM model training RMSE
gam_model_all$results$RMSE
## [1] 24.37900 24.41081
```

```
set.seed(2183)

# calculate and print the GAM model test RMSE
test_pred <- predict(gam_model_all, newdata = x2)
test_rmse <- sqrt(mean((test_pred - dat_test$recovery_time)^2))
test_rmse
## [1] 23.78453
```

All of these factors together explain 43.8% of the deviance in COVID-19 recovery time. Additionally, the model's training error (RMSE using the training dataset) of 24.4 indicates that, on average, the model's predictions on the training data deviate from the actual values by 24.4 units. Meanwhile, the test error

(RMSE using the test dataset) of 23.8 suggests that the model's performance on unseen data is slightly better than its performance on the training data. This could indicate that the model is not overfitting to the training data and is generalizing well to new data.

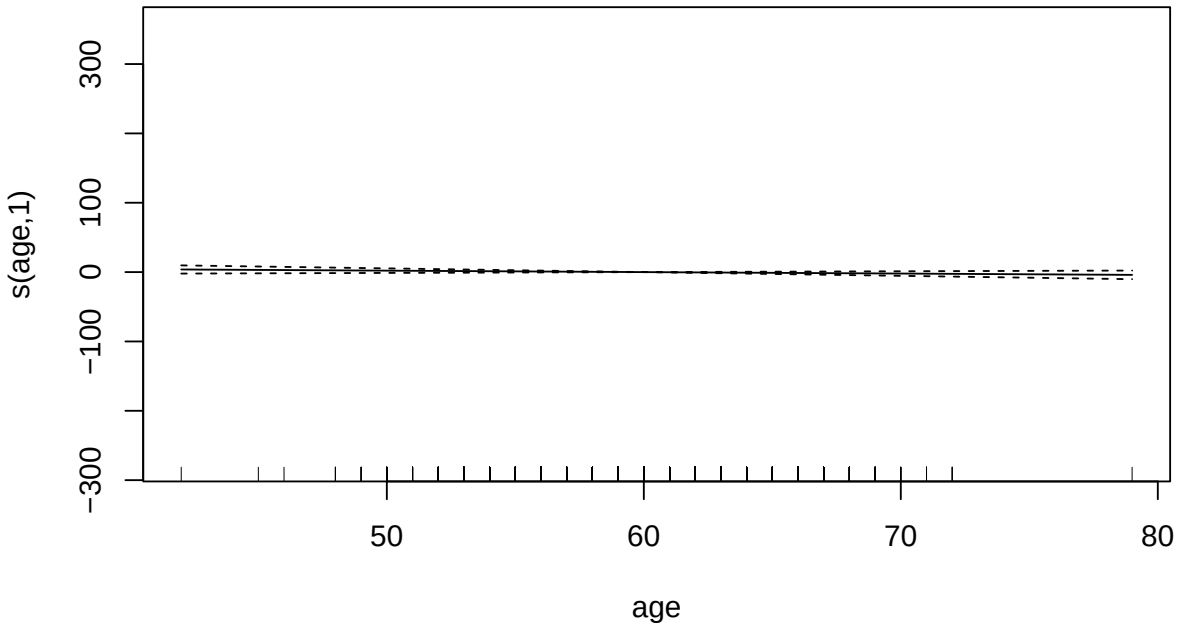
Conclusion

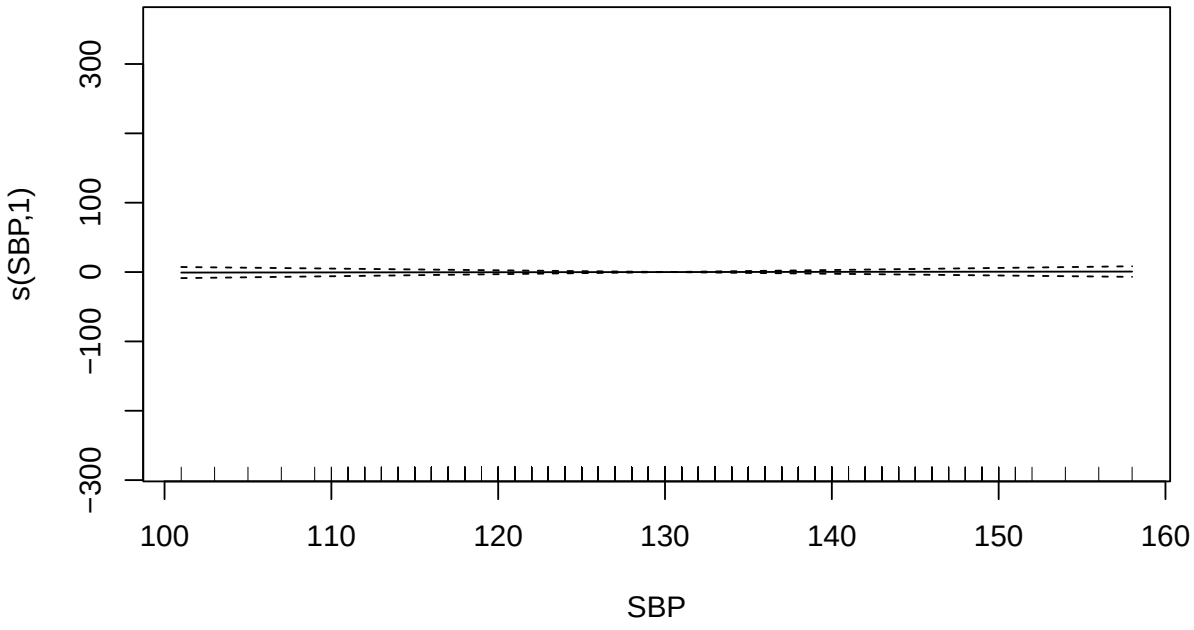
The final GAM model using all predictor variables found that several factors were statistically significant in predicting time to recovery from COVID-19. On average, having a history of former or current smoking, having hypertension, and experiencing severe COVID-19 infection were all significantly associated with a longer predicted recovery time. Additionally, being in study B was associated with a longer recovery time compared to being in study A. On the other hand, being male and being vaccinated was associated with a shorter recovery time. Finally, BMI, height, and weight are also significantly associated with predicted COVID-19 recovery time. The model did not find significant associations with race or diabetes. These insights can be useful in predicting recovery time and informing clinical decisions in the management of COVID-19 patients.

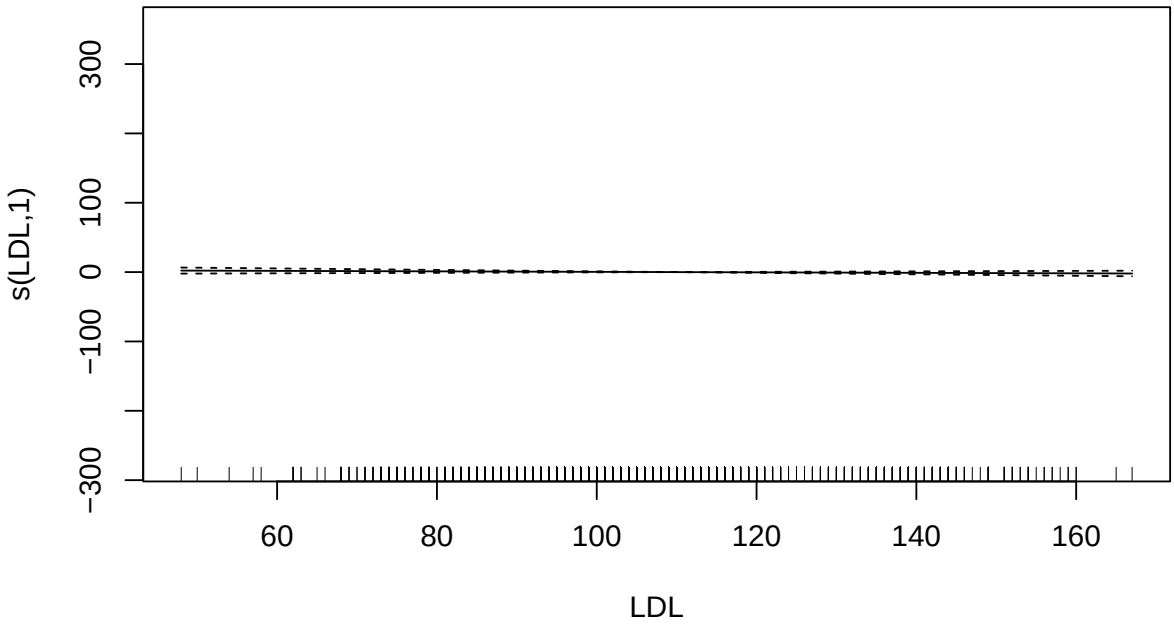
Appendix

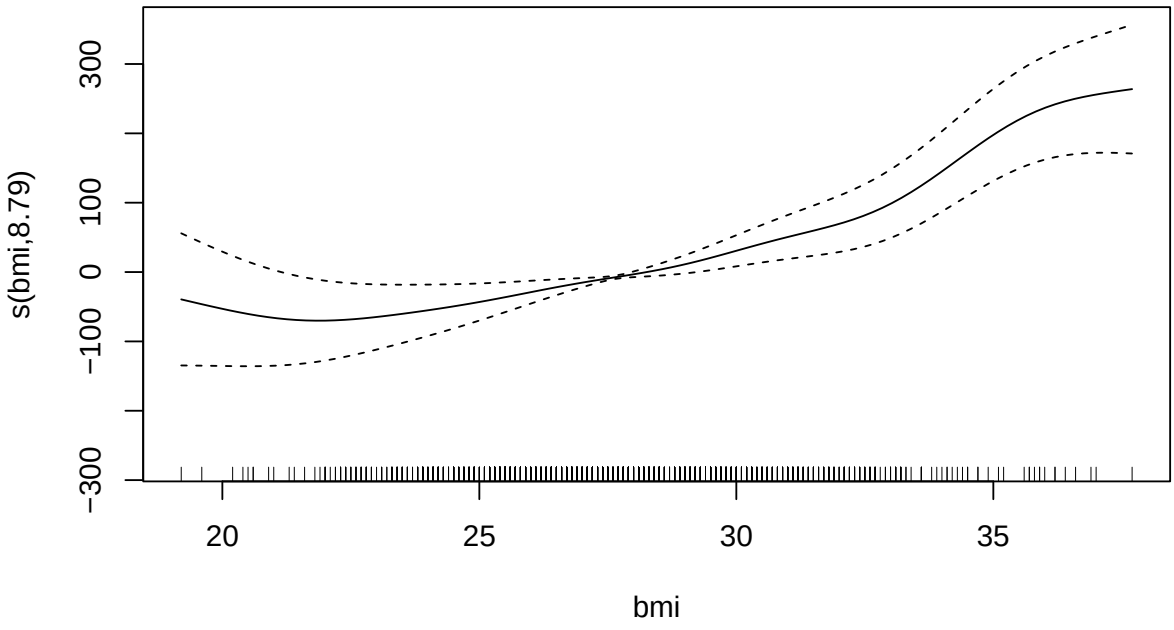
Predictor plots for the final GAM model:

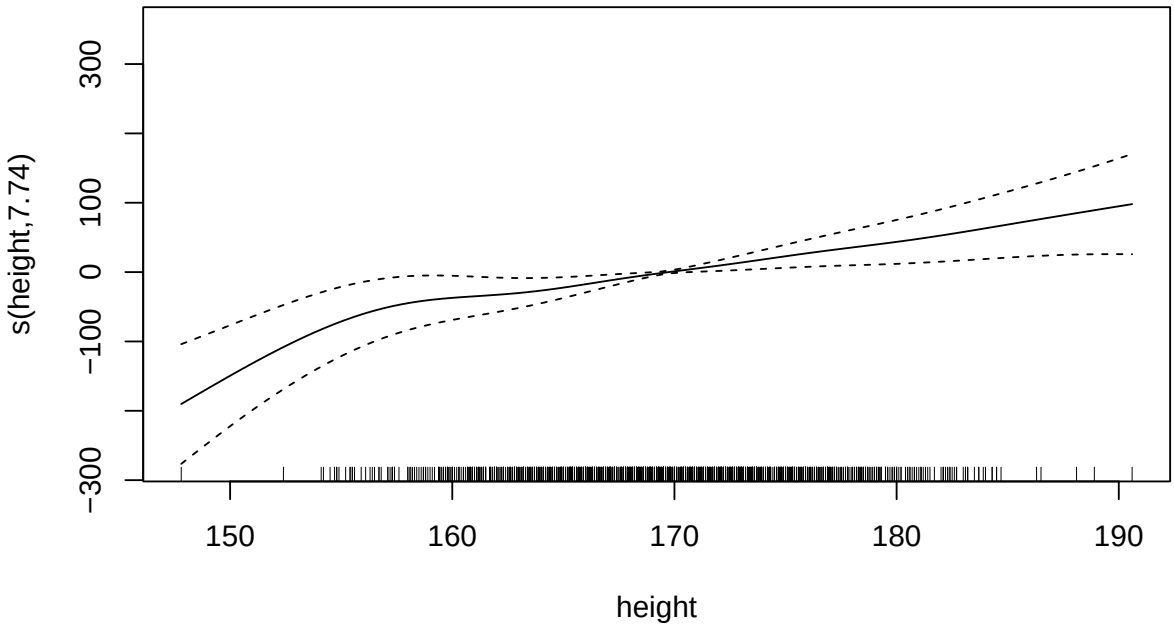
```
set.seed(2183)

# Formula based on final GAM model with all predictors (gam_model_all)
gam_final_model <- gam(recovery_time ~ gender + race + + smoking +
                         hypertension + diabetes + vaccine + severity + study +
                         s(age) + s(SBP) + s(LDL) + s(bmi) + s(height) +
                         s(weight),
              data = dat_train) # training dataset

plot(gam_final_model)
```

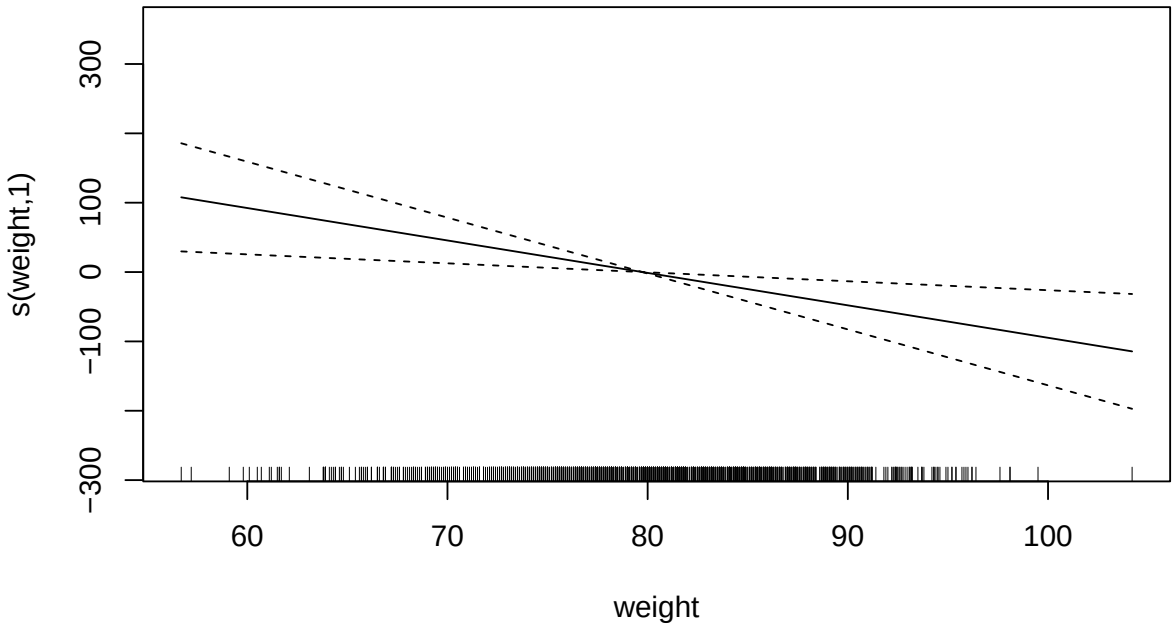

s(age,1)
300
100
0
−100
−300
50
60
70
80
age


s(SBP,1)
300
100
0
−100
−300
100
110
120
130
140
150
160
SBP


s(LDL,1)
300
100
0
−100
−300
60
80
100
120
140
160
LDL


s(bmi,8.79)
300
100
0
−100
−300
20
25
30
35
bmi


s(height,7.74)
300
100
0
−100
−300
150
160
170
180
190
height


s(weight,1)
300
100
0
−100
−300
60
70
80
90
100
weight