Midterm Project

Minghe Wang (mw3845), Zebang Zhang (zz3309), Xuanyu Guo (xg2451)

2025-03-27

Exploratory Data Analysis

```
load("./data/dat1.RData")
load("./data/dat2.RData")

# no missing data
all(is.na(dat1))
```

```
## [1] FALSE
```

```
all(is.na(dat2))
```

```
## [1] FALSE
```

```
ifelse(all(names(dat1) == names(dat2)), "train and test data have same structure", "train and test data
```

```
## [1] "train and test data have same structure"
```

```
str(dat1)
```

```
## 'data.frame':    5000 obs. of  14 variables:
##  $ id          : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ age         : num  50 71 58 63 56 59 67 62 60 64 ...
##  $ gender      : int  0 1 1 0 1 1 0 1 0 1 ...
##  $ race        : Factor w/ 4 levels "1","2","3","4": 1 1 1 1 1 3 4 1 4 1 ...
##  $ smoking     : Factor w/ 3 levels "0","1","2": 1 1 2 1 1 1 1 1 1 1 ...
##  $ height      : num  176 176 169 167 163 ...
##  $ weight      : num  68.3 69.6 76.9 90 83.9 86.8 91.4 87.7 85.7 76.6 ...
##  $ bmi         : num  22 22.6 27 32.1 31.7 30.8 29.7 28.1 29 31.5 ...
##  $ diabetes    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ hypertension: num  0 1 0 1 0 1 1 0 0 1 ...
##  $ SBP         : num  130 149 127 138 123 132 133 130 129 134 ...
##  $ LDL         : num  82 129 101 93 97 108 89 96 120 135 ...
##  $ time        : num  76 82 168 105 193 143 63 78 61 88 ...
##  $ log_antibody: num  10.65 9.89 10.9 9.91 9.56 ...
```

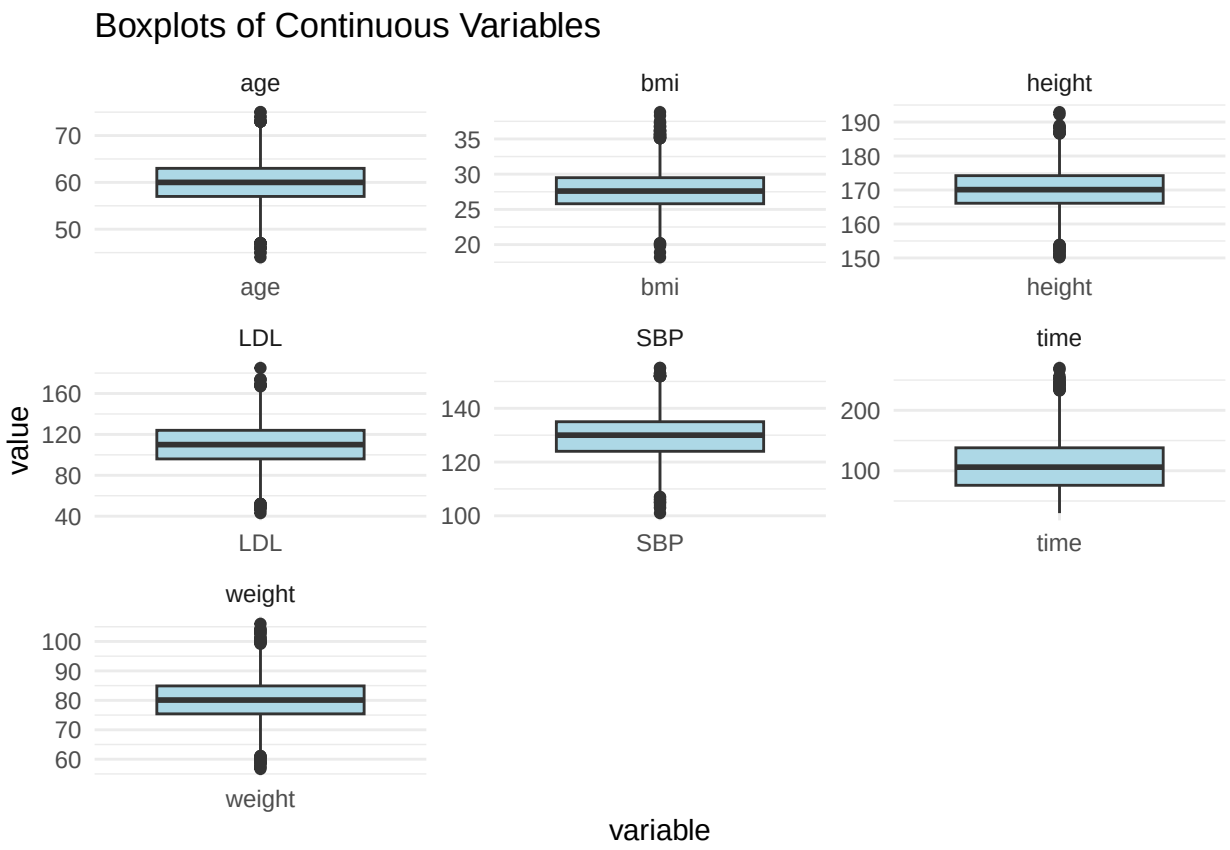
```r
# The 'id' column has no actual meaning, so we remove it.
dat1 <- dat1 %>%
  select(-id)

dat2 <- dat2 %>%
  select(-id)

# Convert categorical variables to labeled factors
convert_factors <- function(df) {
  df %>%
    mutate(
      gender = factor(gender, levels = c(0, 1), labels = c("Female", "Male")),
      race = factor(race, levels = c(1, 2, 3, 4), labels = c("White", "Asian", "Black", "Hispanic")),
      smoking = factor(smoking, levels = c(0, 1, 2), labels = c("Never", "Former", "Current")),
      diabetes = factor(diabetes, levels = c(0, 1), labels = c("No", "Yes")),
      hypertension = factor(hypertension, levels = c(0, 1), labels = c("No", "Yes"))
    )
}

dat1 <- convert_factors(dat1)
dat2 <- convert_factors(dat2)
```

Univariate analysis(continous & categorical)

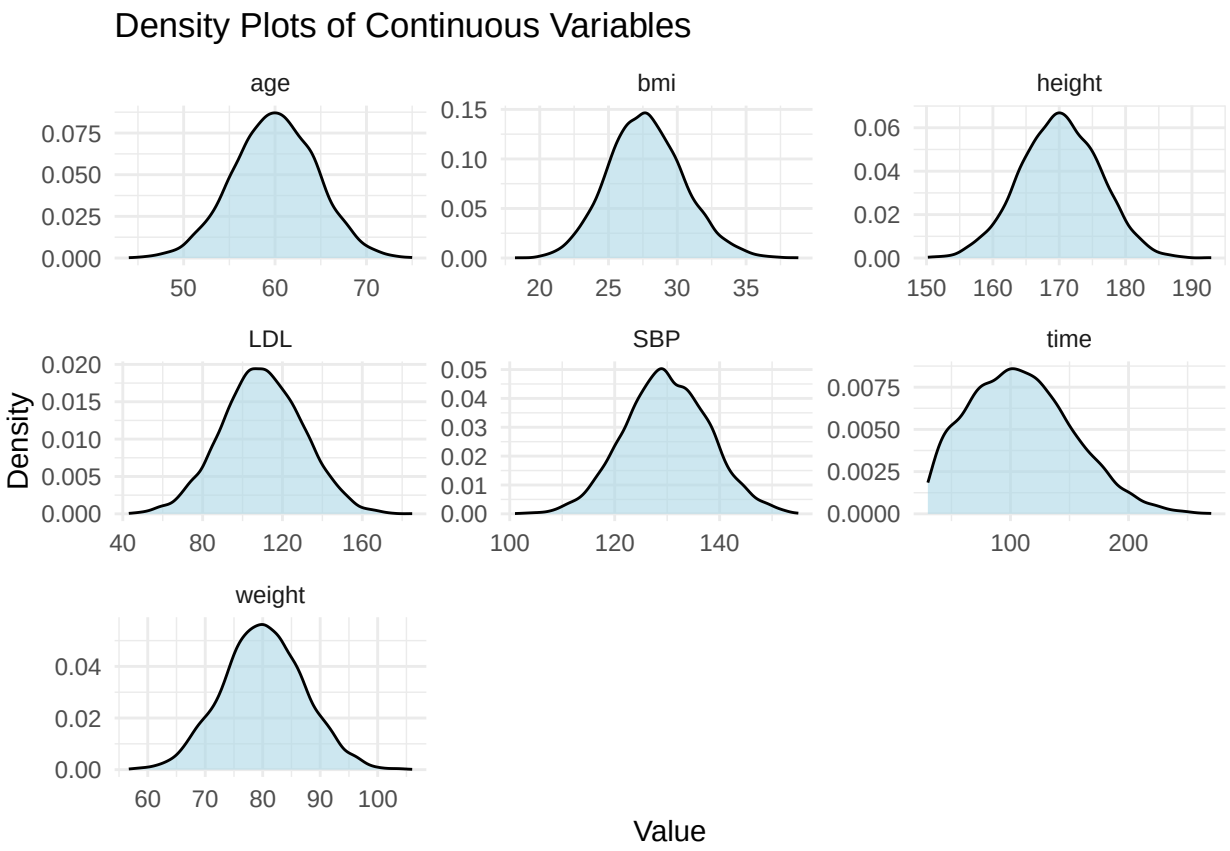
```r
continuous_var <- dat1 %>%
  select(age, height, weight, bmi, SBP, LDL, time)

categorical_var <- dat1 %>%
  select(gender, race, smoking, diabetes, hypertension)

# ---- Continuous Variables ----
# summary
summary(continuous_var)
```

```
##       age             height          weight            bmi
##  Min.   :44.00   Min.   :150.2   Min.   : 56.70   Min.   :18.20
##  1st Qu.:57.00   1st Qu.:166.1   1st Qu.: 75.40   1st Qu.:25.80
##  Median :60.00   Median :170.1   Median : 80.10   Median :27.60
##  Mean   :59.97   Mean   :170.1   Mean   : 80.11   Mean   :27.74
##  3rd Qu.:63.00   3rd Qu.:174.2   3rd Qu.: 84.90   3rd Qu.:29.50
##  Max.   :75.00   Max.   :192.9   Max.   :106.00   Max.   :38.80
##       SBP             LDL             time
##  Min.   :101.0   Min.   : 43.0   Min.   : 30.0
##  1st Qu.:124.0   1st Qu.: 96.0   1st Qu.: 76.0
##  Median :130.0   Median :110.0   Median :106.0
##  Mean   :129.9   Mean   :109.9   Mean   :108.9
##  3rd Qu.:135.0   3rd Qu.:124.0   3rd Qu.:138.0
##  Max.   :155.0   Max.   :185.0   Max.   :270.0
```

```r
# boxplots
continuous_var_long <- continuous_var %>%
  tidyr::pivot_longer(cols = everything(), names_to = "variable", values_to = "value")

ggplot(continuous_var_long, aes(x = variable, y = value)) +
  geom_boxplot(fill = "lightblue") +
  facet_wrap(~variable, scales = "free", ncol = 3) +
  theme_minimal() +
  labs(title = "Boxplots of Continuous Variables")
```



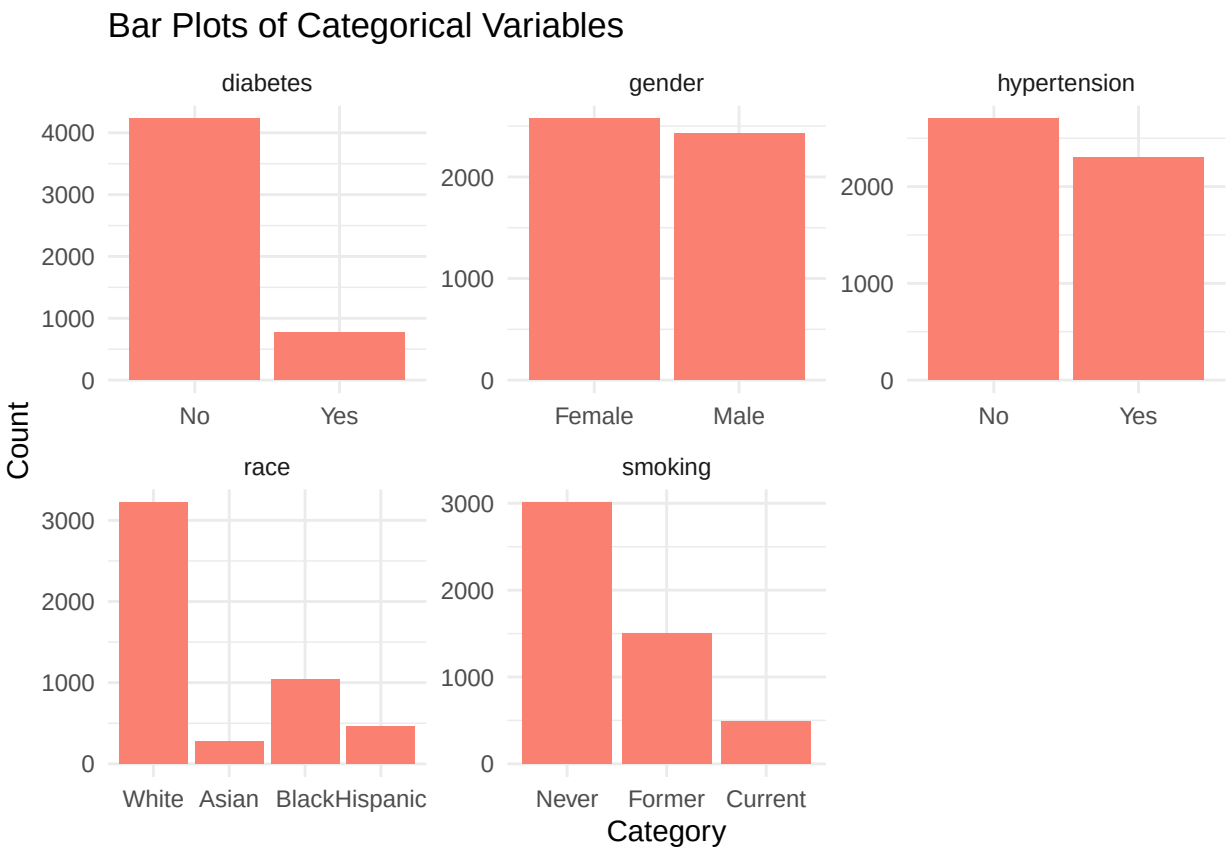

```r
# density plots
ggplot(continuous_var_long, aes(x = value)) +
  geom_density(fill = "lightblue", alpha = 0.6) +
  facet_wrap(~variable, scales = "free", ncol = 3) +
  theme_minimal() +
  labs(title = "Density Plots of Continuous Variables", x = "Value", y = "Density")
```




```
# ---- Categorical Variables ----
# summary
summary(categorical_var)
```

```
##     gender           race         smoking     diabetes   hypertension
##  Female:2573   White   :3221   Never  :3010   No :4228   No :2702
##  Male  :2427   Asian   : 278   Former :1504   Yes: 772   Yes:2298
##                Black   :1036   Current: 486
##                Hispanic: 465
```

```
# bar plots
categorical_var_long <- categorical_var %>%
  tidyr::pivot_longer(cols = everything(), names_to = "variable", values_to = "value")

ggplot(categorical_var_long, aes(x = value)) +
  geom_bar(fill = "salmon") +
  facet_wrap(~variable, scales = "free", ncol = 3) +
  theme_minimal() +
  labs(title = "Bar Plots of Categorical Variables", x = "Category", y = "Count")
```



According to the box plot for continuous variables:

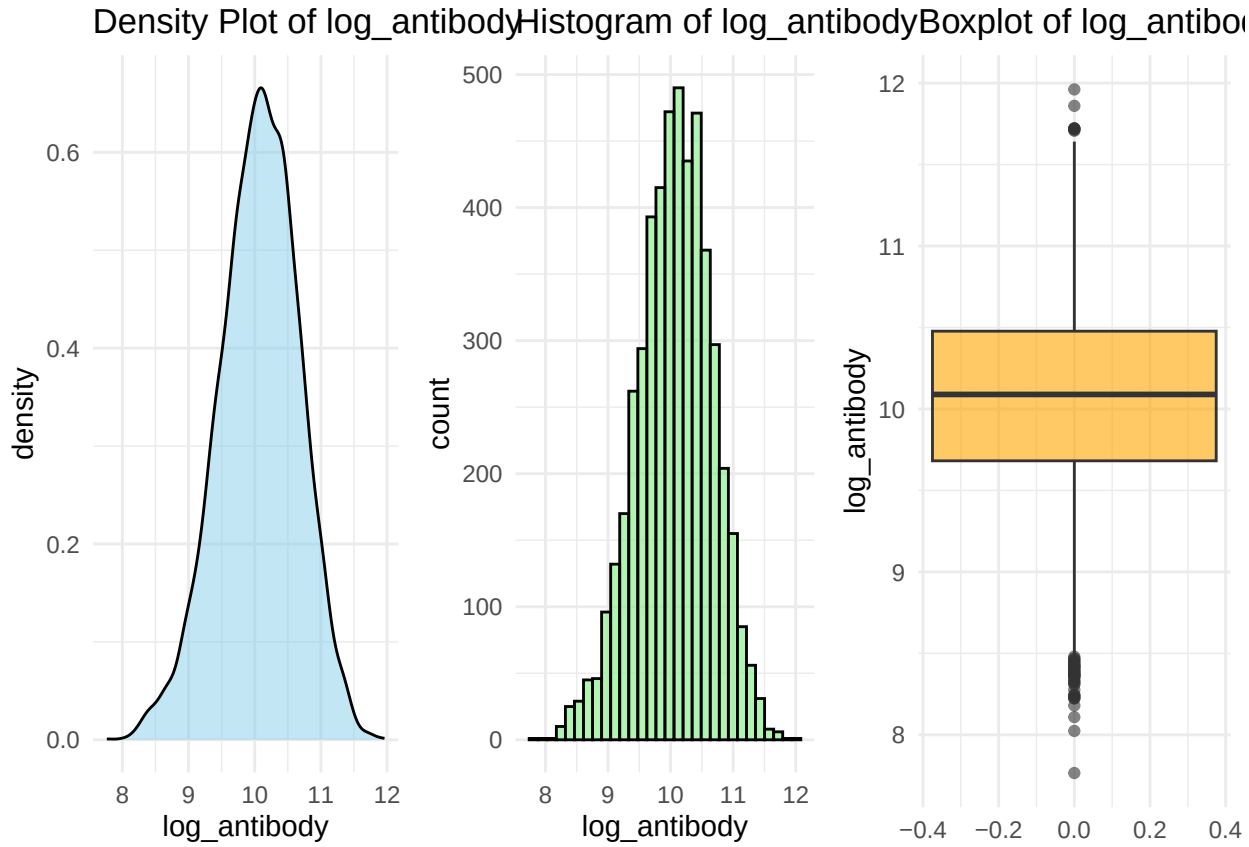
- `Age`, `BMI`, and `SBP` appear reasonably normally distributed, with expected ranges for an adult population; `LDL` cholesterol and `time` since vaccination show a wider range, right-skewness and some outliers, which may impact linear models.

According to the bar plot for categorical variables:

- `Gender` is fairly balanced between Female and Male;
- `Race` is skewed, with a majority of participants identifying as White (Category 1). Other racial/ethnic groups are underrepresented;
- `Smoking` status shows that the majority are never smokers (Category 0), with fewer current and former smokers;
- A large proportion of participants do not have `diabetes`;
- A moderate split exists for `hypertension`, which may contribute meaningfully to clinical outcome variation
- Demographically, the population is balanced by gender but skewed by race and smoking status.

```
# response variable `log_antibody`

# Density plot
p1 <- ggplot(dat1, aes(x = log_antibody)) +
```

```
  geom_density(fill = "skyblue", alpha = 0.5) +
  ggtitle("Density Plot of log_antibody") +
  xlab("log_antibody") +
  theme_minimal()

# Histogram
p2 <- ggplot(dat1, aes(x = log_antibody)) +
  geom_histogram(bins = 30, fill = "lightgreen", color = "black", alpha = 0.7) +
  ggtitle("Histogram of log_antibody") +
  xlab("log_antibody") +
  theme_minimal()

# Boxplot
p3 <- ggplot(dat1, aes(y = log_antibody)) +
  geom_boxplot(fill = "orange", alpha = 0.6) +
  ggtitle("Boxplot of log_antibody") +
  ylab("log_antibody") +
  theme_minimal()

grid.arrange(p1, p2, p3, ncol = 3)
```



`log_antibody` (response) appears fairly symmetrical, which supports its use as a continuous response in linear or GAM models.

Overall, we believe the response variable `log_antibody` is well-behaved, and further correlation analysis(eg. bivariate) is needed.

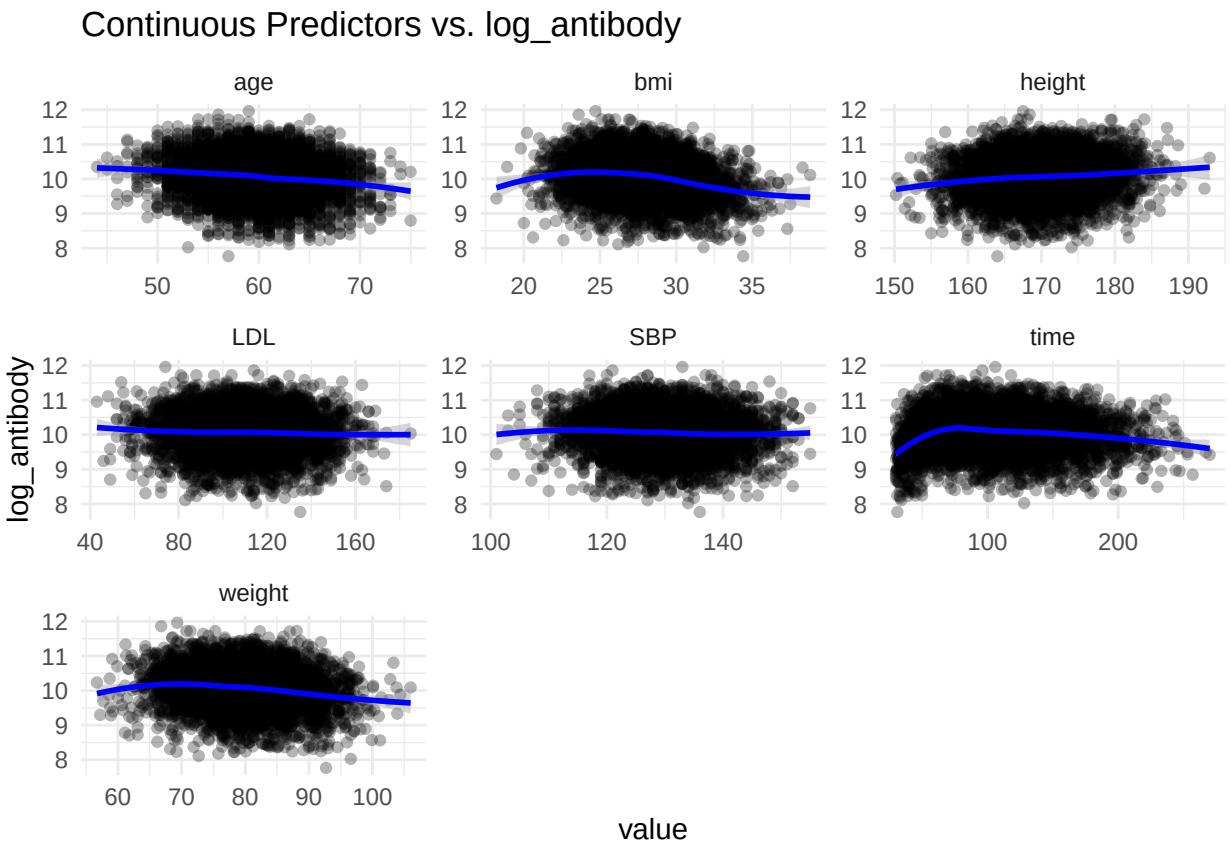
Next, we assess correlations and non-linear trends to guide model form.

Correlation Analysis

We first analyze the relationship between log_antibody (response variable) and continuous variables & correlations among continuous variables themselves.

```
continuous_var_long <- dat1 %>%
  select(age, height, weight, bmi, SBP, LDL, time, log_antibody) %>%
  tidyr::pivot_longer(cols = -log_antibody, names_to = "variable", values_to = "value")

# Scatterplots with smoothing lines
ggplot(continuous_var_long, aes(x = value, y = log_antibody)) +
  geom_point(alpha = 0.3) +
  geom_smooth(method = "loess", color = "blue") +
  facet_wrap(~variable, scales = "free", ncol = 3) +
  theme_minimal() +
  labs(title = "Continuous Predictors vs. log_antibody")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

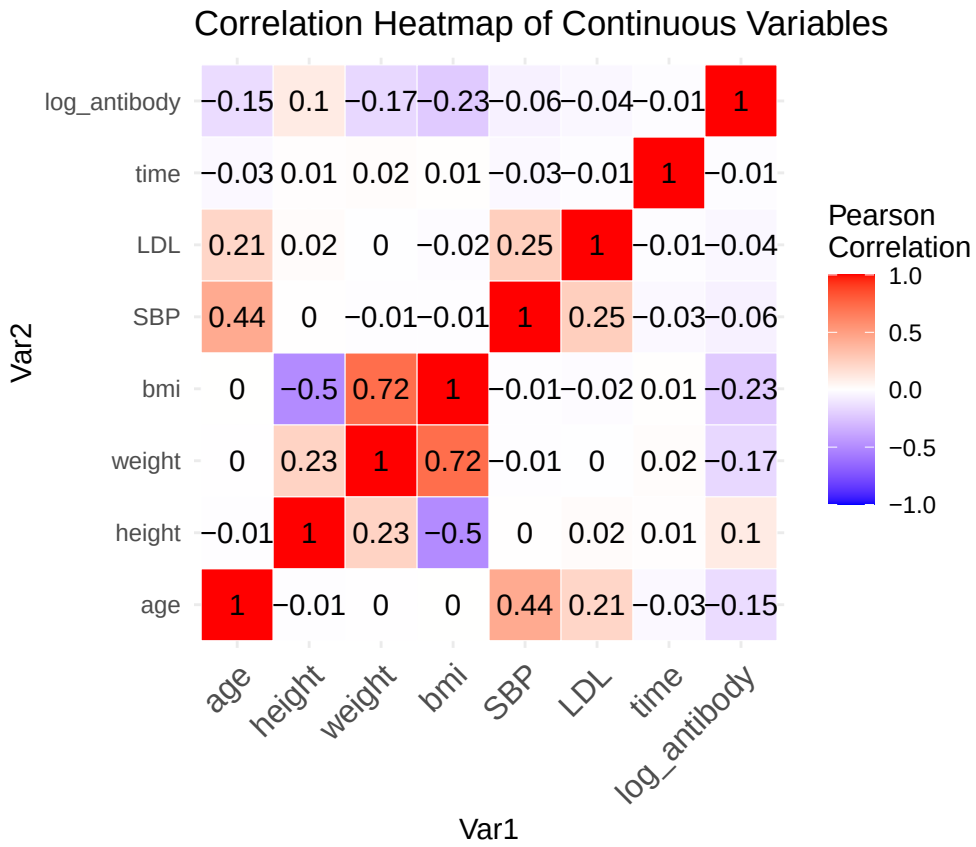


Using LOESS method, we observe linearity between predictors and the response. The plot shows that `bmi`, `time`, and `weight` has clear non linear trend against resopnse `log_antibody`, indicating potential need to use GAM or non linear model.

```r
continous_name <- c("age", "height", "weight", "bmi", "SBP", "LDL", "time", "log_antibody")
dat_cont <- dat1[ , continous_name]

# coefficient matrix
cor_matrix <- cor(dat_cont, use = "complete.obs", method = "pearson")
print(round(cor_matrix, 2))
```

```
##                age height weight   bmi   SBP   LDL  time log_antibody
## age           1.00  -0.01   0.00  0.00  0.44  0.21 -0.03        -0.15
## height       -0.01   1.00   0.23 -0.50  0.00  0.02  0.01         0.10
## weight        0.00   0.23   1.00  0.72 -0.01  0.00  0.02        -0.17
## bmi           0.00  -0.50   0.72  1.00 -0.01 -0.02  0.01        -0.23
## SBP           0.44   0.00  -0.01 -0.01  1.00  0.25 -0.03        -0.06
## LDL           0.21   0.02   0.00 -0.02  0.25  1.00 -0.01        -0.04
## time         -0.03   0.01   0.02  0.01 -0.03 -0.01  1.00        -0.01
## log_antibody -0.15   0.10  -0.17 -0.23 -0.06 -0.04 -0.01         1.00
```

```r
cor_melt <- melt(cor_matrix)

ggplot(cor_melt, aes(Var1, Var2, fill = value)) +
  geom_tile(color = "white") +
  scale_fill_gradient2(low = "blue", high = "red", mid = "white",
                       midpoint = 0, limit = c(-1, 1), space = "Lab",
                       name = "Pearson\nCorrelation") +
  geom_text(aes(label = round(value, 2)), color = "black", size = 4) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, vjust = 1,
                                   size = 12, hjust = 1)) +
  coord_fixed() +
  ggtitle("Correlation Heatmap of Continuous Variables")
```



From the matrix and plot, we can see that the Pearson correlation coefficient between bmi and weight, bmi and height exceed 0.5, indicating that multicollinearity may exist.

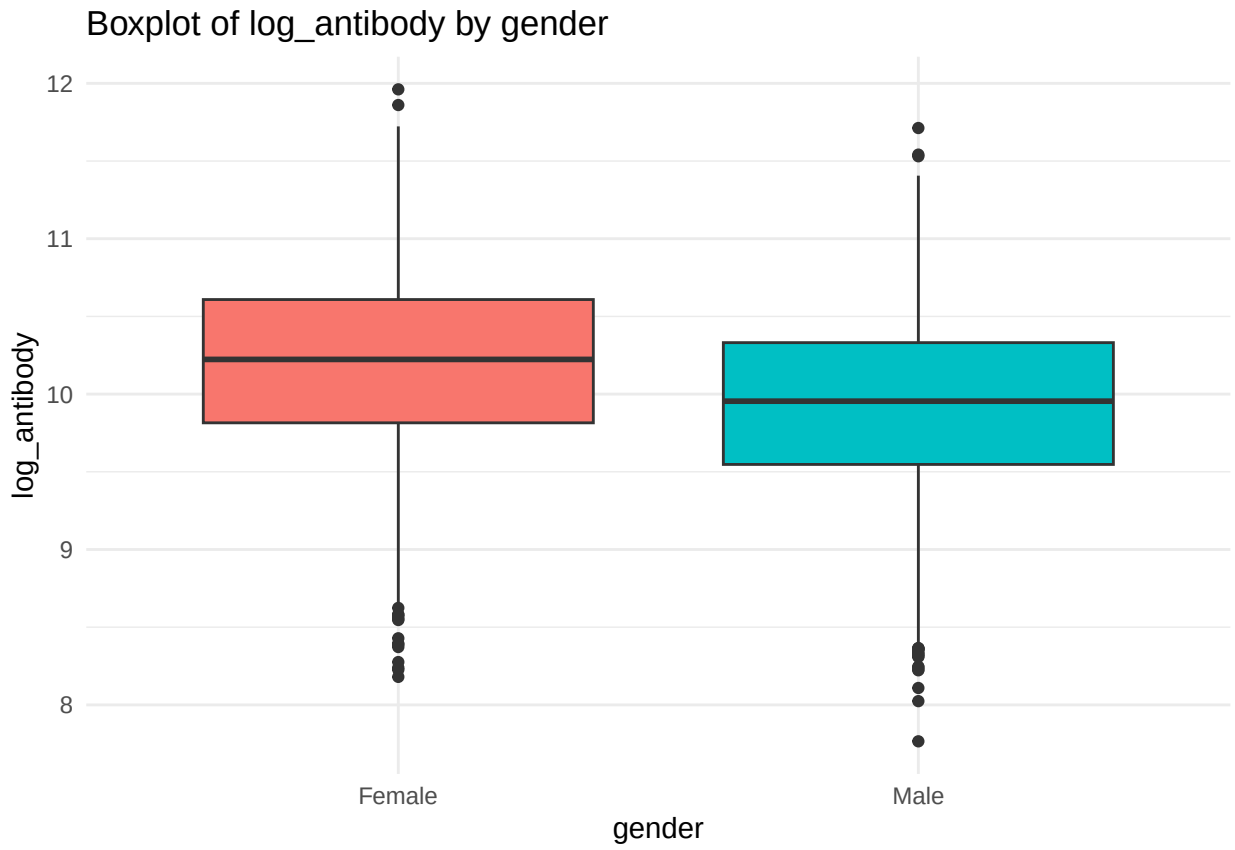
```
# VIF
lm_full <- lm(log_antibody ~ age + gender + smoking + height + weight + bmi +
                diabetes + hypertension + SBP + LDL + time, data = dat1)

vif(lm_full)
```

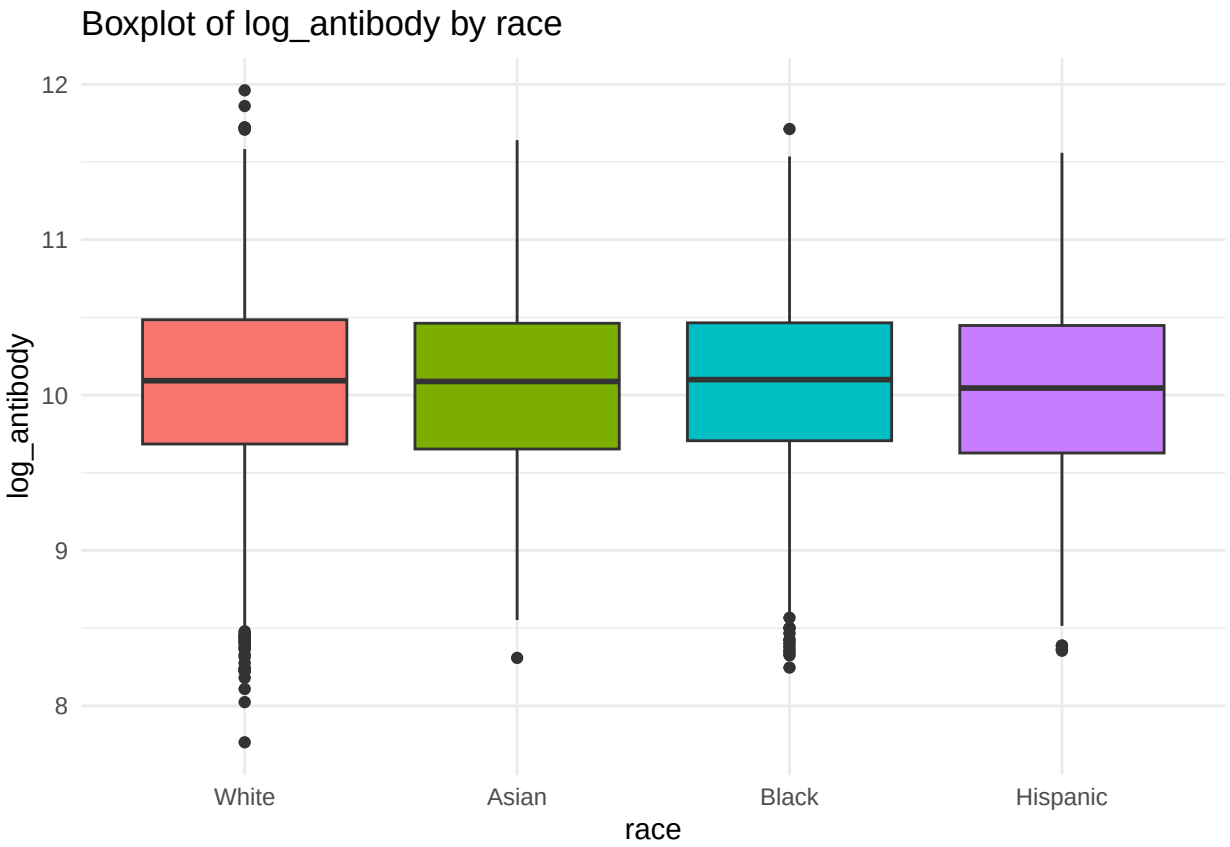
```
##                     GVIF Df GVIF^(1/(2*Df))
## age             1.258104  1        1.121652
## gender          1.002988  1        1.001493
## smoking         1.002682  2        1.000670
## height        107.111548  1       10.349471
## weight        169.112707  1       13.004334
## bmi           213.764468  1       14.620686
## diabetes        1.001898  1        1.000949
## hypertension    2.791341  1        1.670731
## SBP             3.070211  1        1.752202
## LDL             1.085268  1        1.041762
## time            1.002242  1        1.001120
```

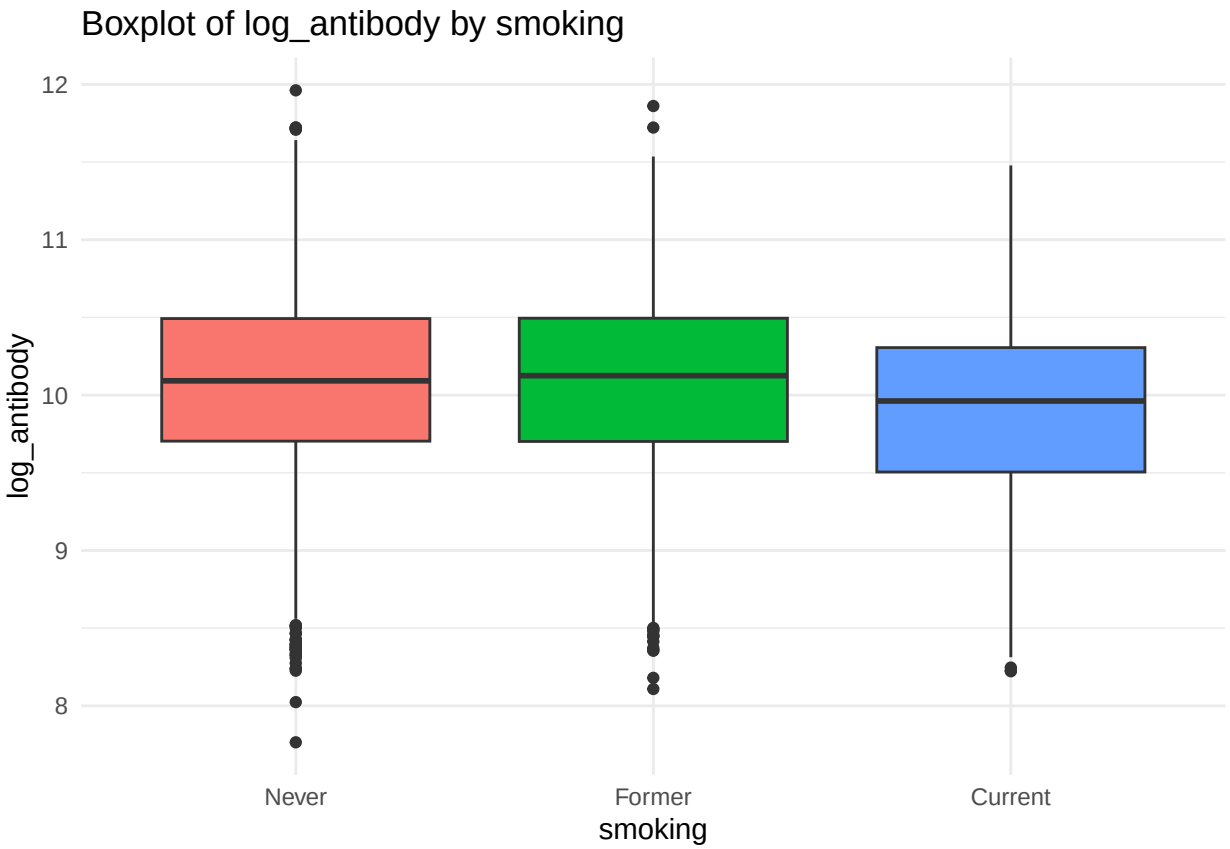
The VIF of bmi, weight and height exceed 10, indicating serious multicollinearity among these variables. Since BMI is a function of weight and height, it is recommended to retain only one of them (e.g., BMI) in the model to avoid redundancy and unstable coefficient estimates.

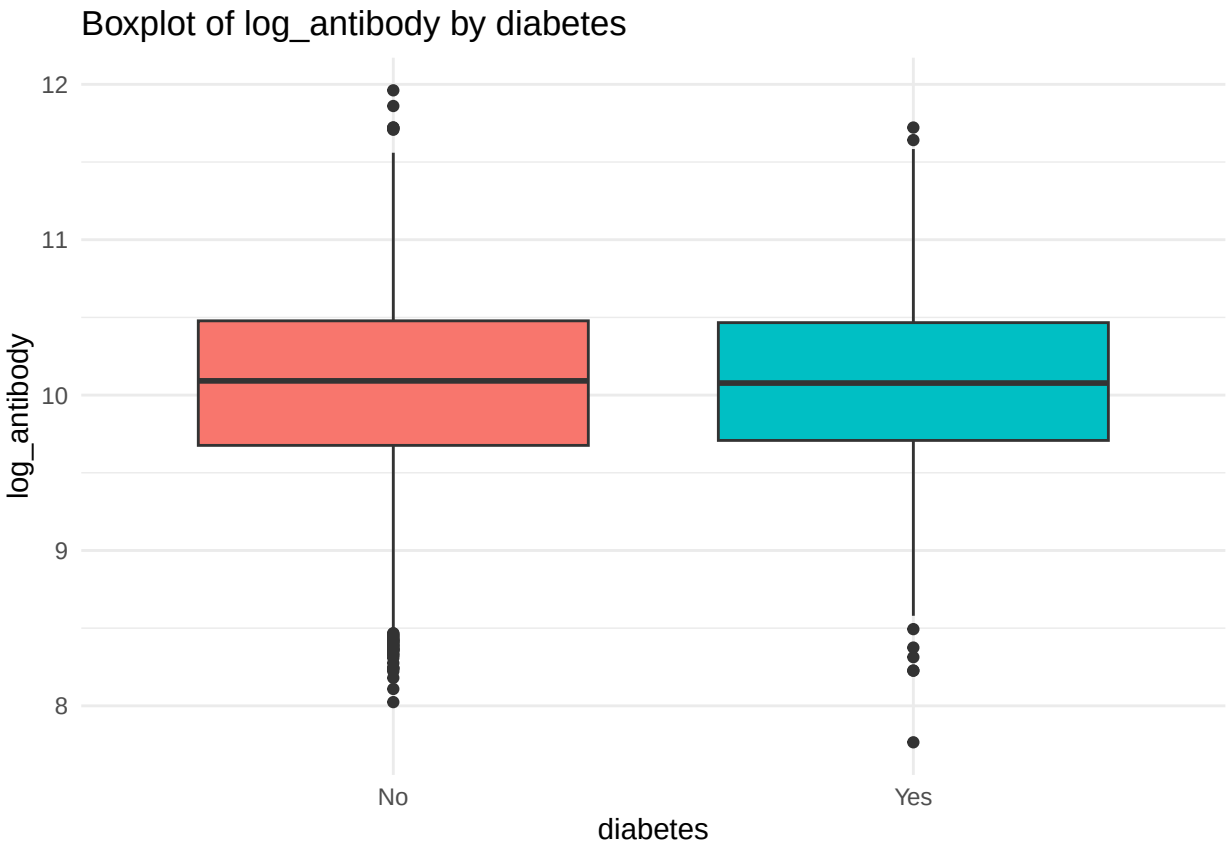
Then for categorical variable, generate boxplots to visualize the distribution of log_antibody across levels of each categorical variable.

```
categorical_name <- c("gender", "race", "smoking", "diabetes", "hypertension")

for (name in categorical_name) {
  p <- ggplot(dat1, aes_string(x = name, y = "log_antibody", fill = name)) +
    geom_boxplot() +
    ggtitle(paste("Boxplot of log_antibody by", name)) +
    theme_minimal() +
    theme(legend.position = "none")
  print(p)
}
```




Boxplot of log_antibody by race
log_antibody
12
11
10
9
8
White
Asian
Black
Hispanic
race


Boxplot of log_antibody by smoking
log_antibody
12
11
10
9
8
Never
Former
Current
smoking

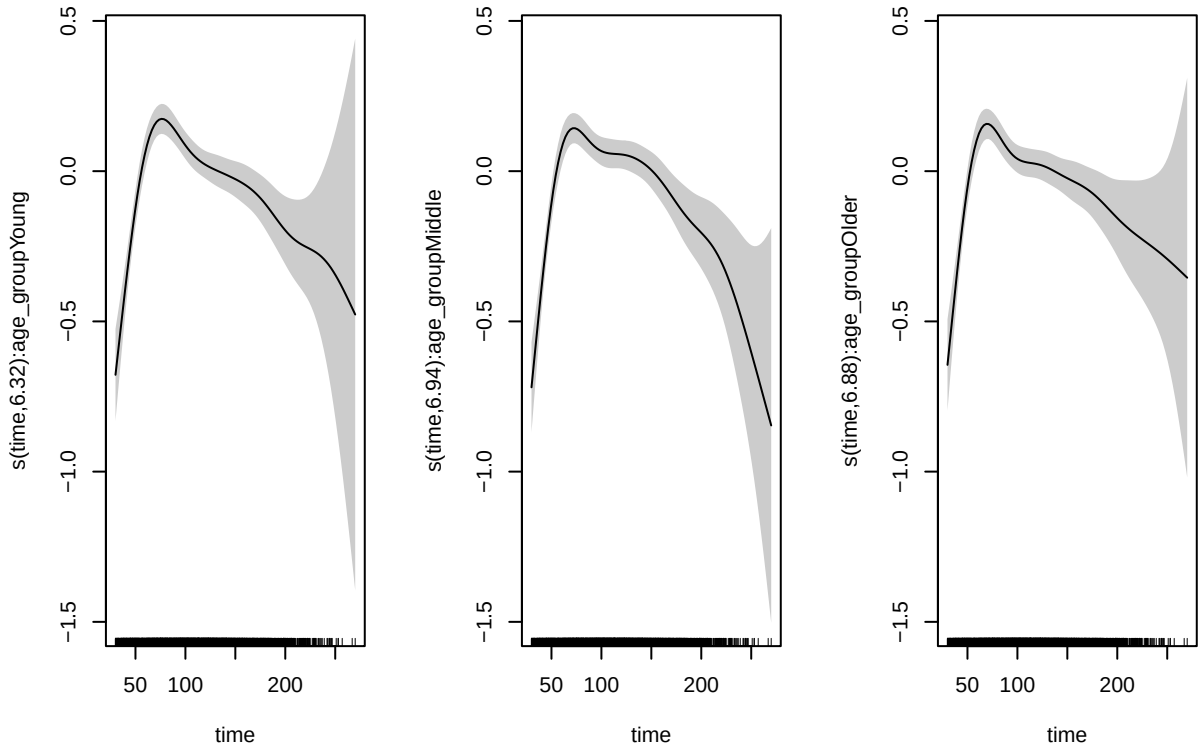

Boxplot of log_antibody by diabetes
log_antibody
12
11
10
9
8
No
Yes
diabetes

The boxplots suggest that the distribution of log_antibody does not differ substantially across the categories of each categorical variable, indicating limited evidence of strong group-level effects.

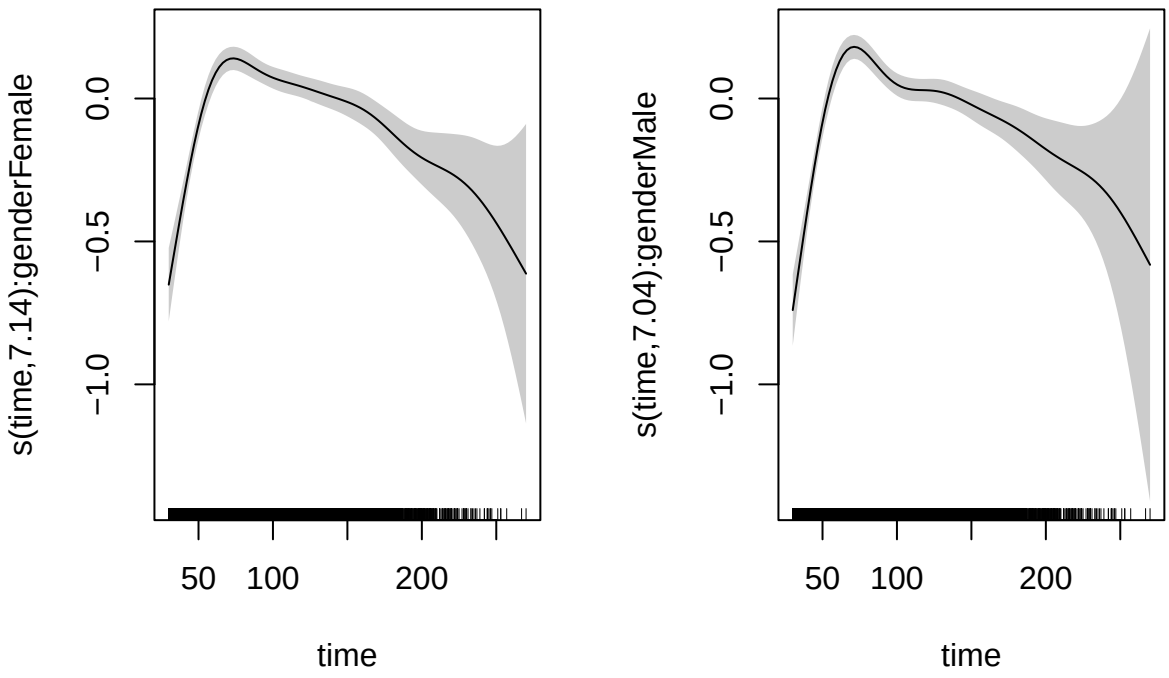
Interaction Analysis

```
dat1_ageGroup <- dat1 %>%
  mutate(age_group = ntile(age, 3)) %>%
  mutate(age_group = factor(age_group, labels = c("Young", "Middle", "Older"))) %>%
  mutate(
    gender = factor(gender, labels = c("Female", "Male")),
    diabetes = factor(diabetes, labels = c("No", "Yes")),
    hypertension = factor(hypertension, labels = c("No", "Yes"))
  )

library(mgcv)

gam_age_interact <- gam(
  log_antibody ~ s(time, by = age_group) + age_group + gender + bmi + SBP + LDL +
    race + smoking + diabetes + hypertension,
  data = dat1_ageGroup,
  method = "GCV.Cp"
)

gam_gender_interact <- gam(
```

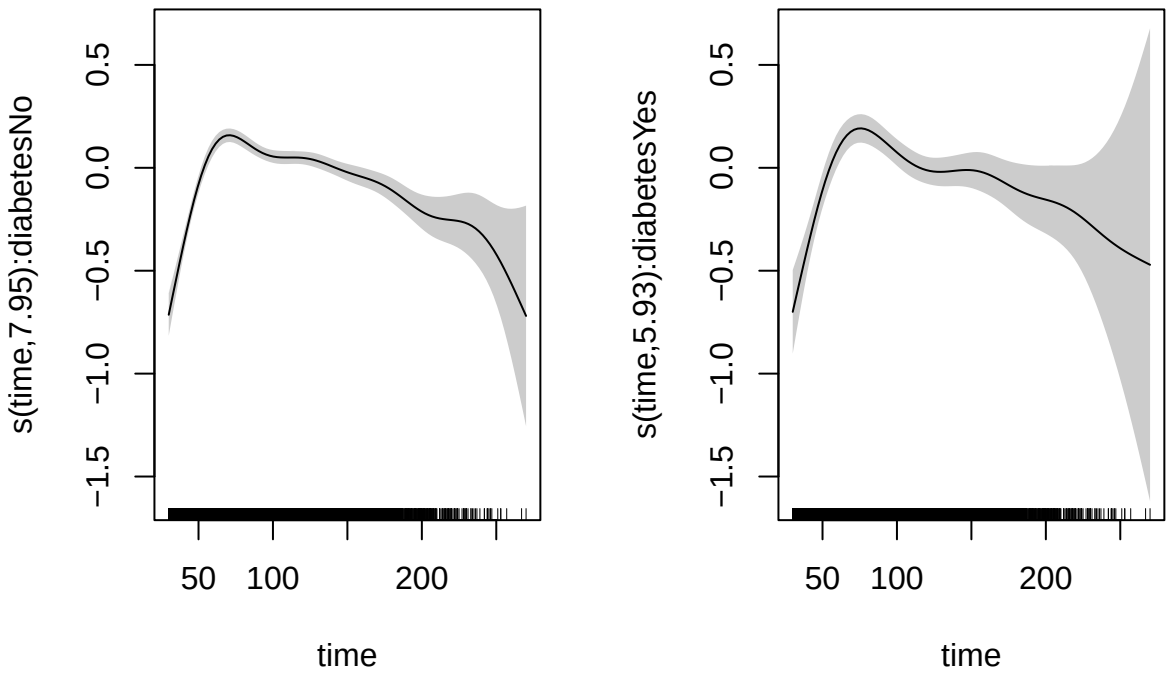
```
  log_antibody ~ s(time, by = gender) + gender + age + bmi + SBP + LDL +
    race + smoking + diabetes + hypertension,
  data = dat1_ageGroup,
  method = "GCV.Cp"
)

gam_diabetes_interact <- gam(
  log_antibody ~ s(time, by = diabetes) + diabetes + age + bmi + SBP + LDL +
    race + gender + smoking + hypertension,
  data = dat1_ageGroup,
  method = "GCV.Cp"
)
gam_hypertension_interact <- gam(
  log_antibody ~ s(time, by = hypertension) + diabetes + age + bmi + SBP + LDL +
    race + gender + smoking + hypertension,
  data = dat1_ageGroup,
  method = "GCV.Cp"
)
# Plotting smooth terms
par(mfrow = c(1, 3))
plot(gam_age_interact, shade = TRUE)
```



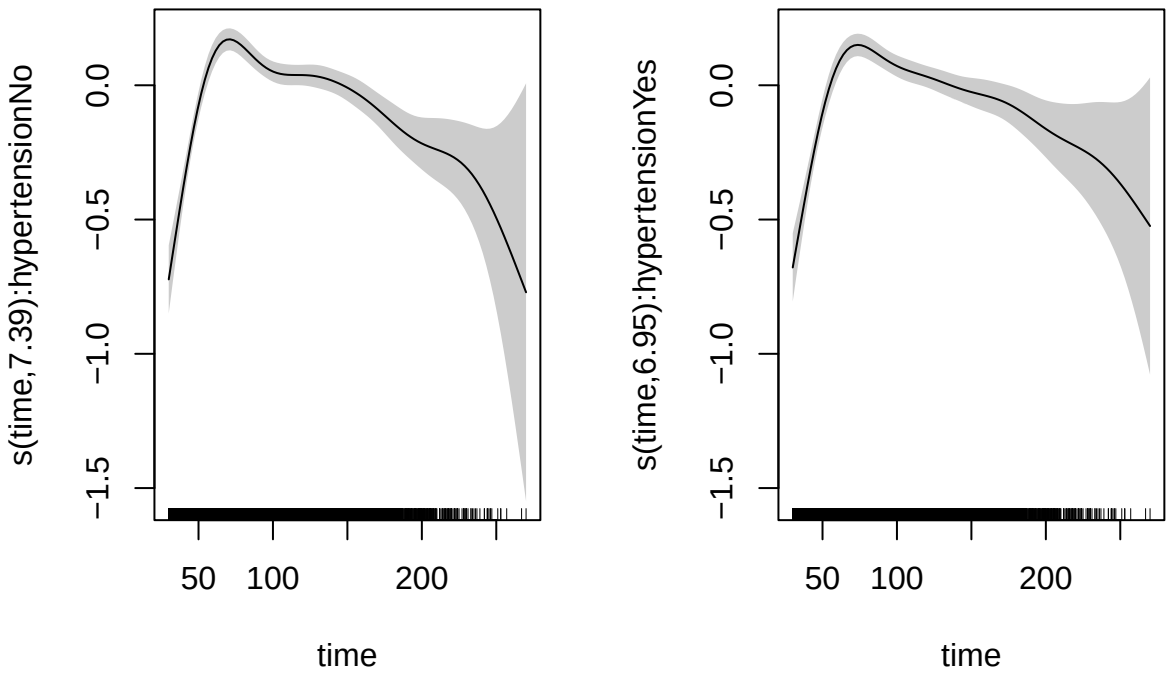

```
par(mfrow = c(1, 2))
plot(gam_gender_interact, shade = TRUE)
```




```
plot(gam_diabetes_interact, shade = TRUE)
```




```
plot(gam_hypertension_interact, shade = TRUE)
```



Model Training

We first fit a Lasso regression model to select important predictors and address multicollinearity. This is particularly useful here, as previous VIF analysis indicated strong multicollinearity among BMI, height, and weight.

```
x <- model.matrix(log_antibody ~ ., data = dat1)[, -1]
y <- dat1$log_antibody

set.seed(2)
cv.lasso <- cv.glmnet(x, y,
                      alpha = 1,
                      lambda = exp(seq(-5, 6, length = 100)))

cv.lasso$lambda.min
```

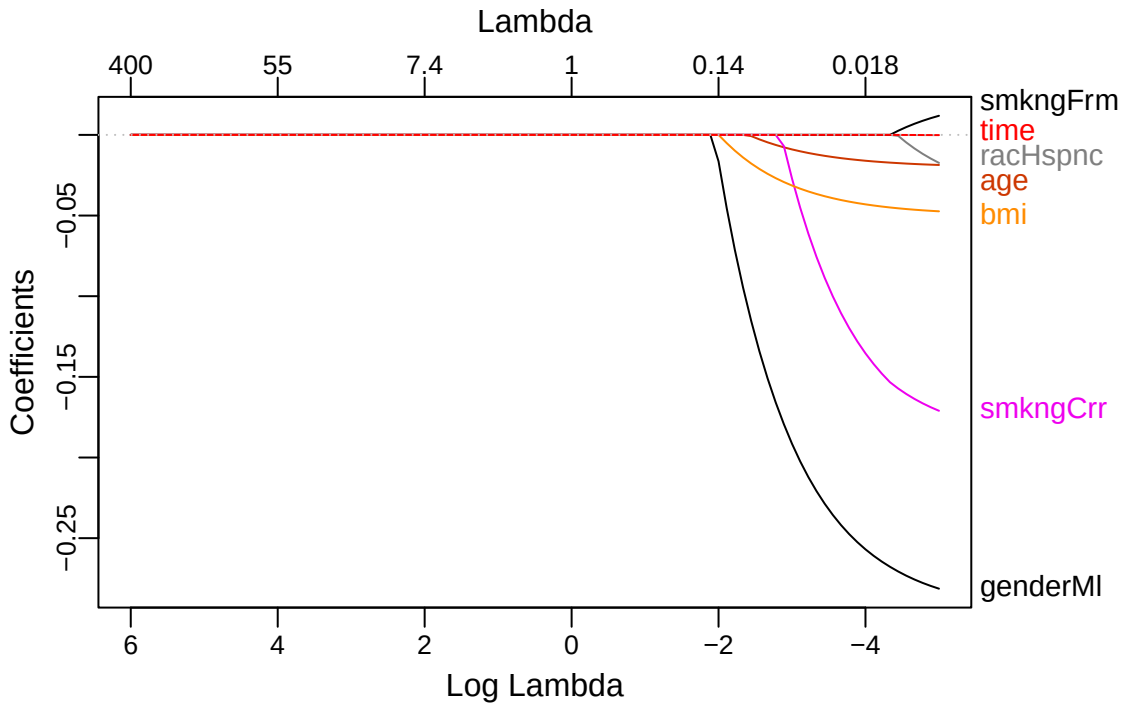
```
## [1] 0.006737947
```

```
lasso_coef <- coef(cv.lasso, s = "lambda.min")
lasso_coef
```

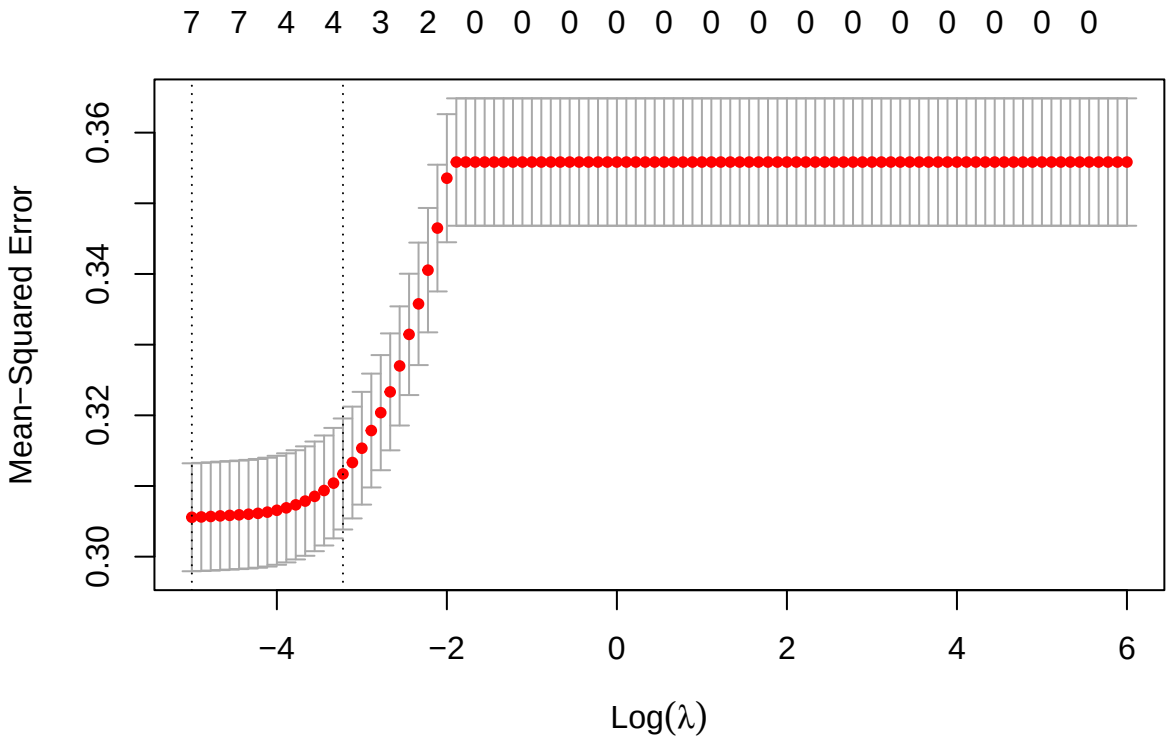
```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                              s1
```

```
## (Intercept)     12.662438353
## age             -0.018645486
## genderMale      -0.281300893
## raceAsian        .
## raceBlack        .
## raceHispanic    -0.017347987
## smokingFormer    0.011848697
## smokingCurrent  -0.171036138
## height           .
## weight           .
## bmi             -0.047381778
## diabetesYes      .
## hypertensionYes  .
## SBP              .
## LDL              .
## time            -0.000131821
```

```
plot_glmnet(cv.lasso$glmnet.fit)
```




```
plot(cv.lasso)
```



From the coefficients under lambda.min, we can see that height and weight are excluded from the Lasso model and bmi is retained. The lasso model helps solve multicollinearity.

```
y_pred_train <- predict(cv.lasso, newx = x, s = "lambda.min")

rmse_train <- sqrt(mean((y_pred_train - y)^2))

print(paste("RMSE on training set (lambda.min):", round(rmse_train, 4)))
```

```
## [1] "RMSE on training set (lambda.min): 0.5518"
```

Then we use the test dataset (dat2) to compute the test RMSE of the lasso model and evaluate model generalizability.

```
x_test <- model.matrix(log_antibody ~ ., data = dat2)[, -1]
y_test <- dat2$log_antibody

y_pred_lasso <- predict(cv.lasso, newx = x_test, s = "lambda.min")

rmse_lasso <- sqrt(mean((y_pred_lasso - y_test)^2))

print(paste("RMSE on test set (lambda.min):", round(rmse_lasso, 4)))
```

```
## [1] "RMSE on test set (lambda.min): 0.5749"
```

Since Lasso is a linear model that assumes additive and linear relationships between predictors and the outcome, we next explore two nonlinear modeling approaches (GAM and MARS) to capture potential non-linearities and interaction effects in the data.

```
# Define trainControl
ctrl1 <- trainControl(method = "cv", number = 10)

train_y <- dat1$log_antibody
train_x <- dat1 %>%
  select(-log_antibody)
```

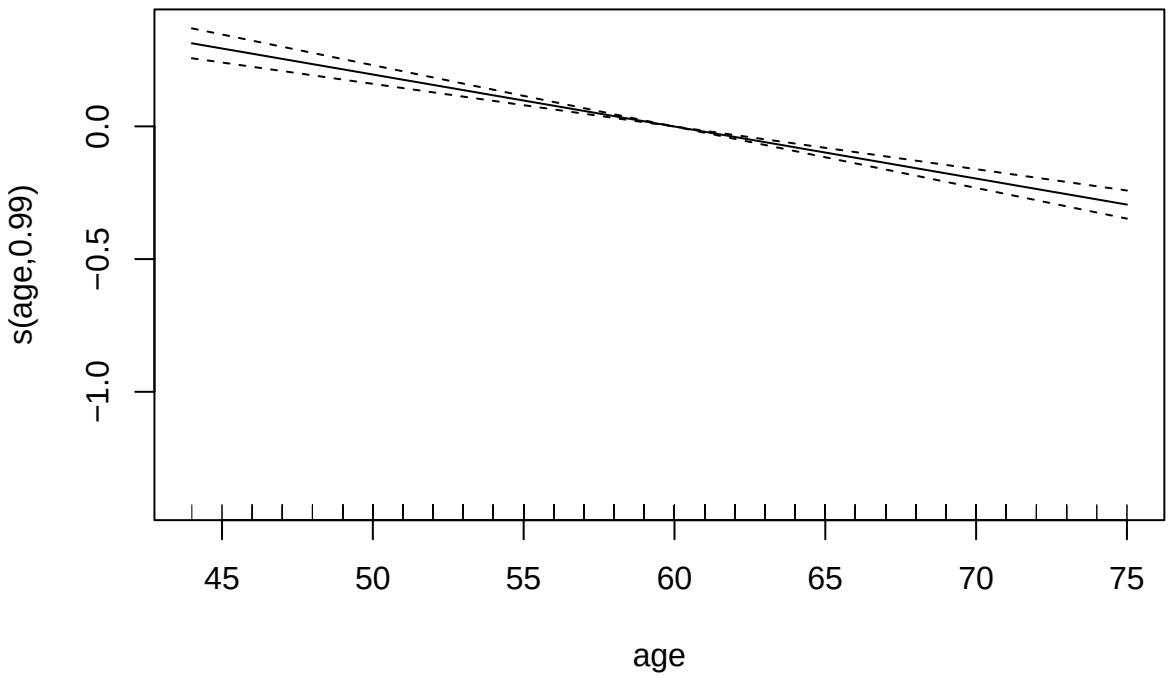
```
# GAM model
set.seed(2)
gam.fit <- train(train_x, train_y,
                 method = "gam",
                 trControl = ctrl1)

gam.fit$bestTune
```

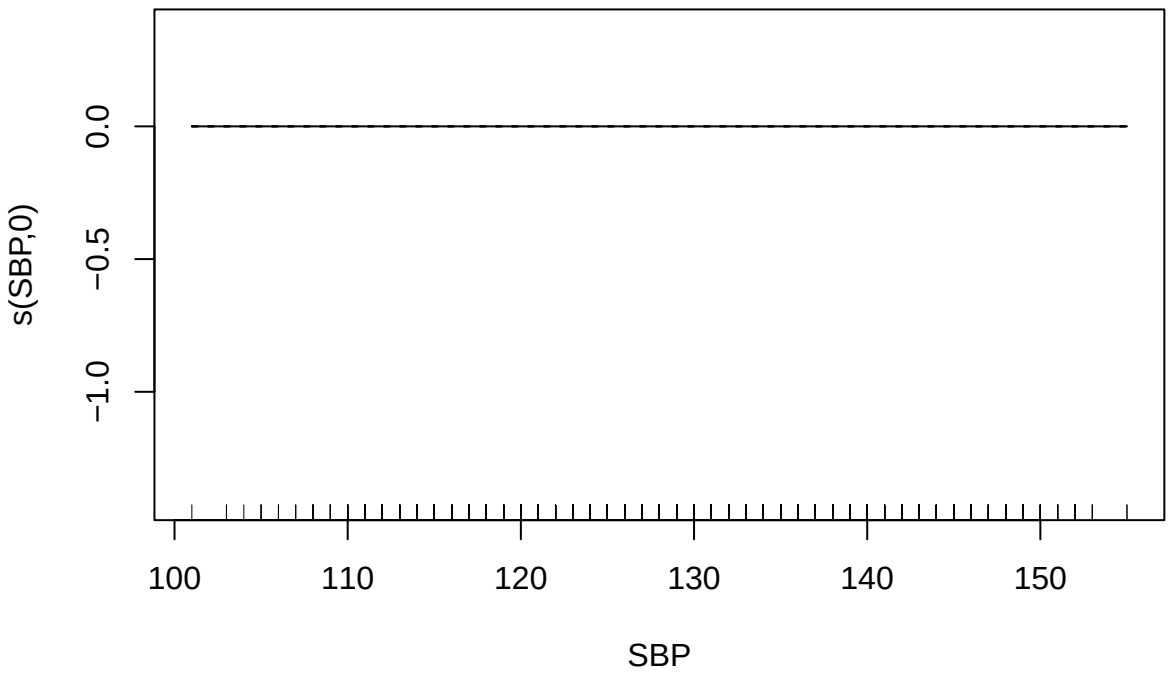
```
##   select method
## 2   TRUE GCV.Cp
```

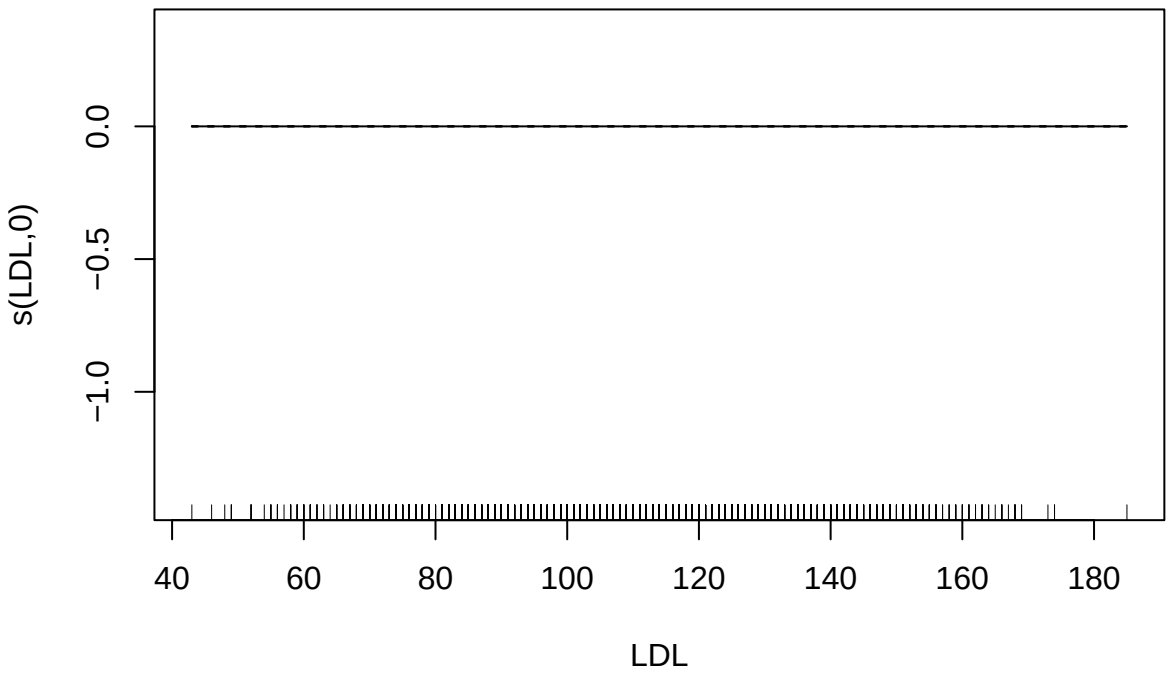
```
gam.fit$finalModel
```

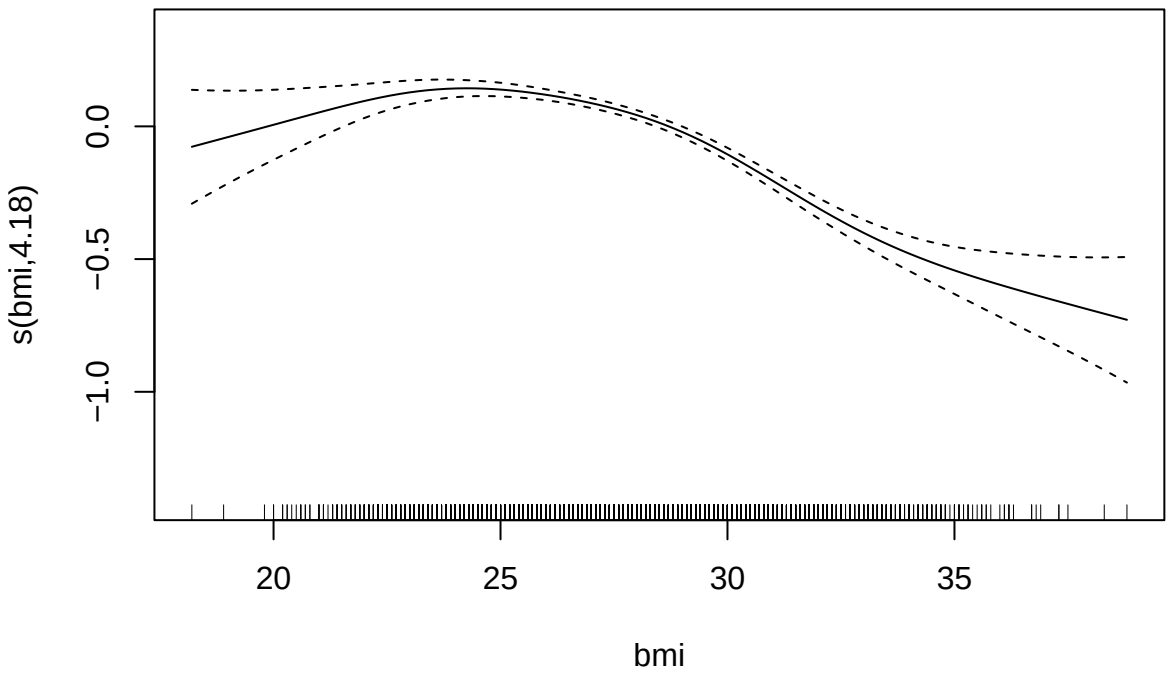
```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## .outcome ~ gender + diabetes + hypertension + smoking + race +
##     s(age) + s(SBP) + s(LDL) + s(bmi) + s(time) + s(height) +
##     s(weight)
##
## Estimated degrees of freedom:
## 0.991 0.000 0.000 4.179 7.892 1.234 0.000
##  total = 23.3
##
## GCV score: 0.2786734
```

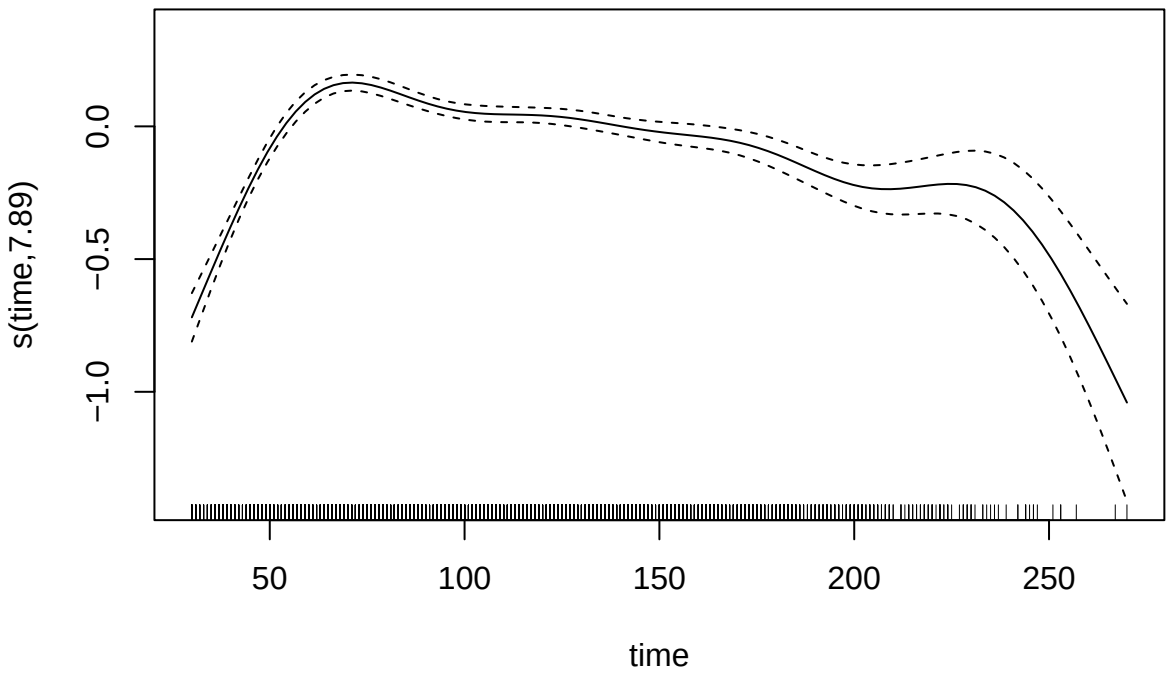
```
plot(gam.fit$finalModel)
```

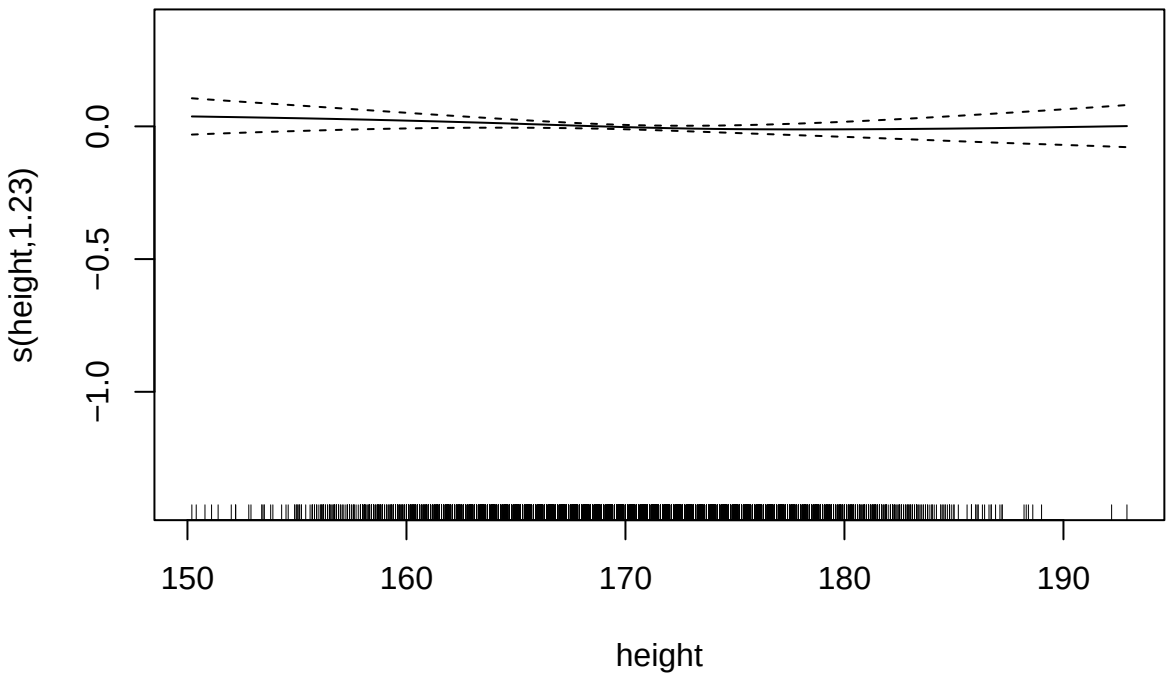

s(age,0.99)
0.0
−0.5
−1.0
45
50
55
60
65
70
75
age

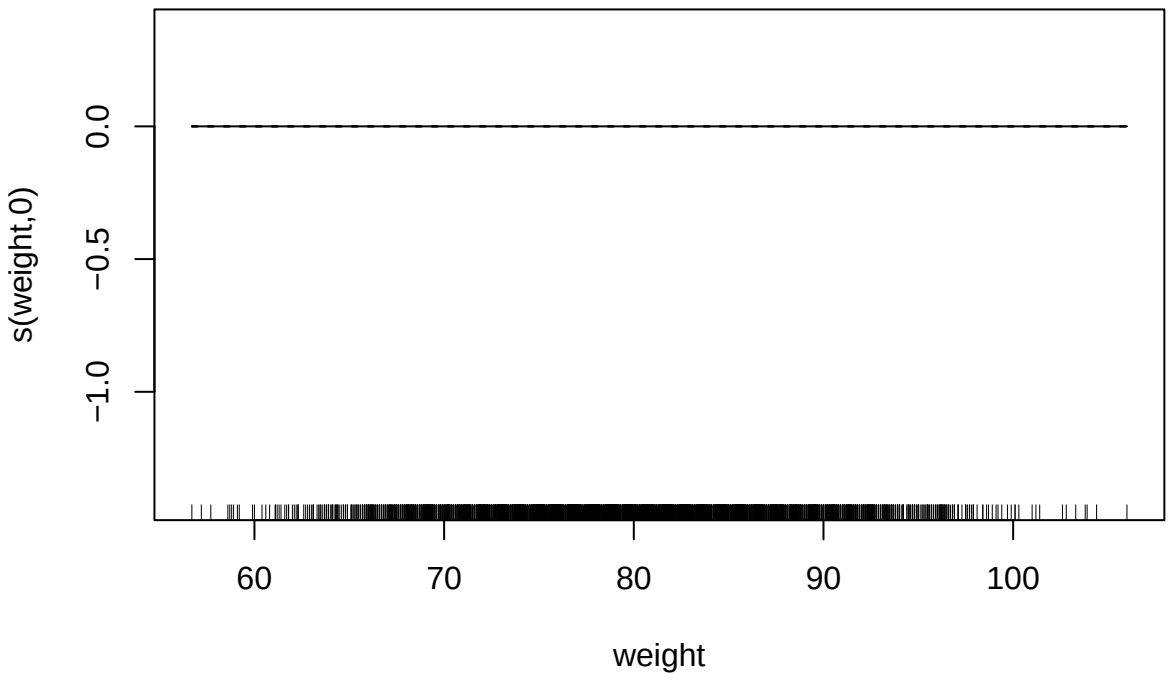

s(SBP,0)
0.0
−0.5
−1.0
100
110
120
130
140
150
SBP


s(LDL,0)
0.0
−0.5
−1.0
40
60
80
100
120
140
160
180
LDL

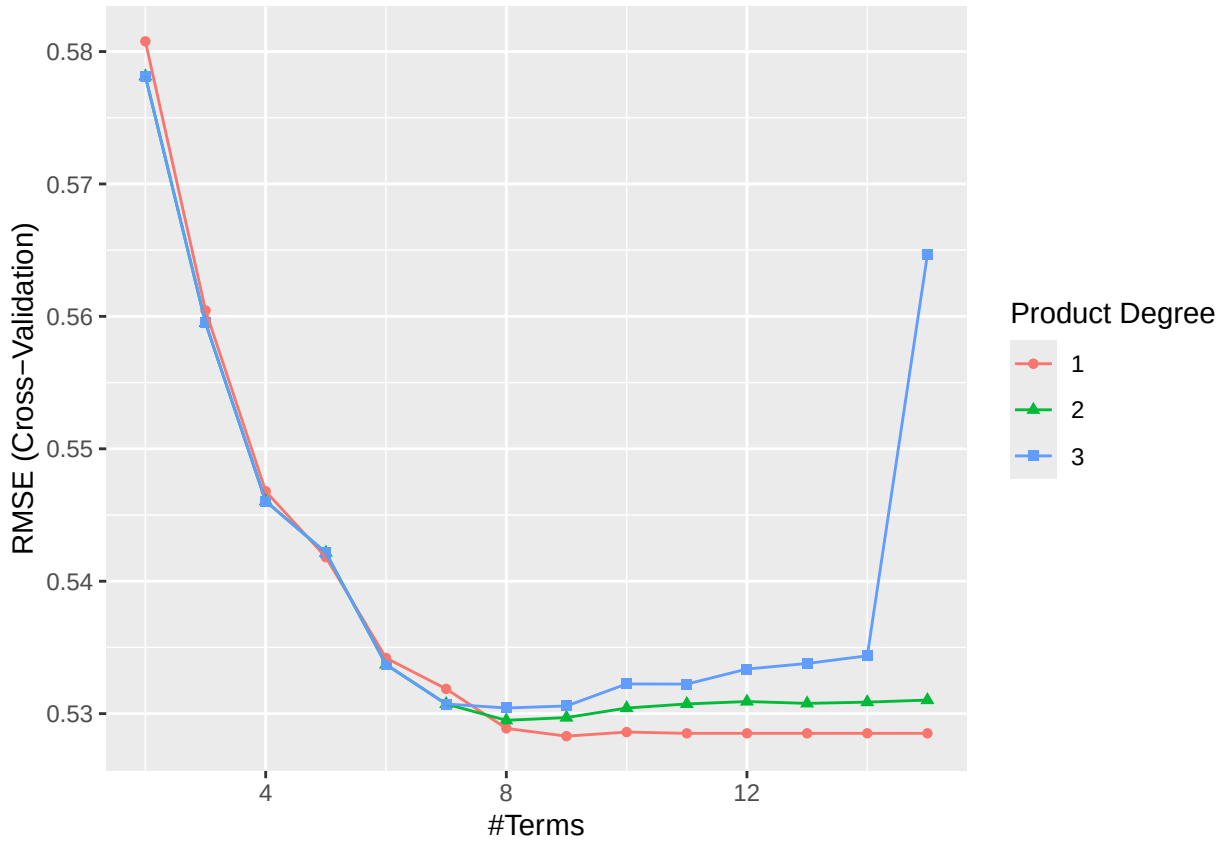

s(bmi,4.18)
0.0
−0.5
−1.0
20
25
30
35
bmi


s(time,7.89)
0.0
−0.5
−1.0
50
100
150
200
250
time


s(height,1.23)
0.0
−0.5
−1.0
150
160
170
180
190
height




```
# MARS model
mars_grid <- expand.grid(degree = 1:3,
                         nprune = 2:15)

set.seed(2)
mars.fit <- train(train_x, train_y,
                  method = "earth",
                  tuneGrid = mars_grid,
                  trControl = ctrl1)

ggplot(mars.fit)
```



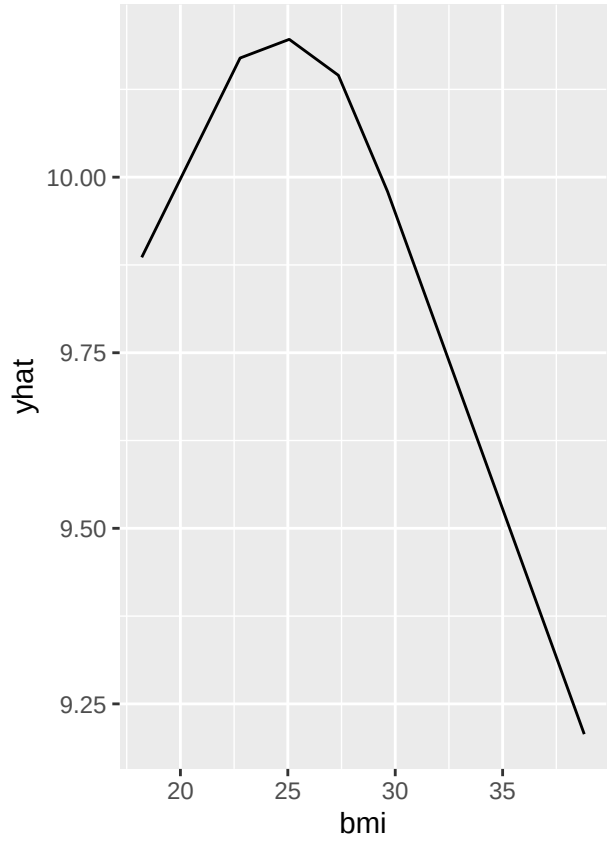

```
mars.fit$bestTune
```

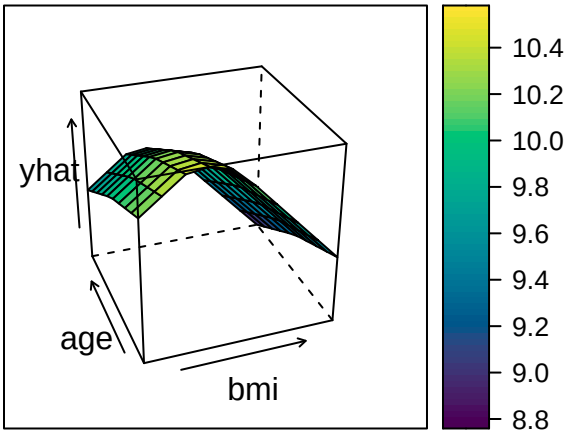
```
##   nprune degree
## 8      9      1
```

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

```
##    (Intercept)    h(27.8-bmi)     h(time-57)     h(57-time)     genderMale
##   10.847446930   -0.061997354   -0.002254182   -0.033529326   -0.296290451
##      h(age-59)      h(59-age) smokingCurrent    h(bmi-23.7)
##   -0.022957648    0.016138468   -0.205126851   -0.084380175
```

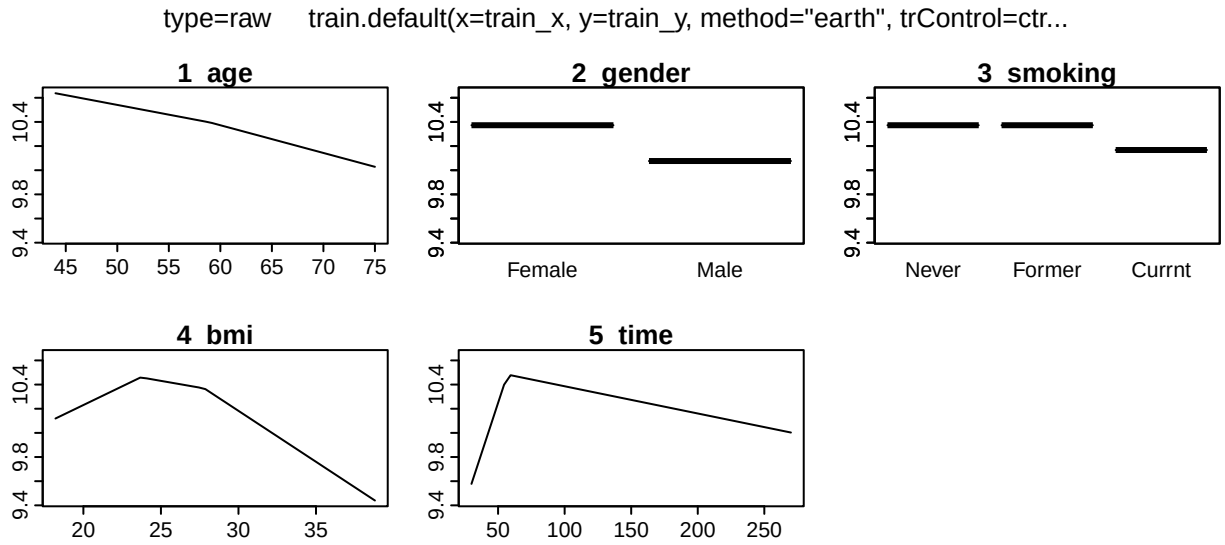
```
p1 <- pdp::partial(mars.fit, pred.var = c("bmi"), grid.resolution = 10) |> autoplot()
p2 <- pdp::partial(mars.fit, pred.var = c("bmi", "age"),
                   grid.resolution = 10) |>
      pdp::plotPartial(levelplot = FALSE, zlab = "yhat", drape = TRUE,
                       screen = list(z = 20, x = -60))

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






```
plotmo(mars.fit,
nresponse = 1,
degree2 = FALSE,
varnames = "age")
```

```
##  plotmo grid:    age gender  race smoking height weight  bmi diabetes
##                   60 Female White   Never  170.1   80.1 27.6       No
##  hypertension SBP LDL time
##            No 130 110  106
```



Model selection

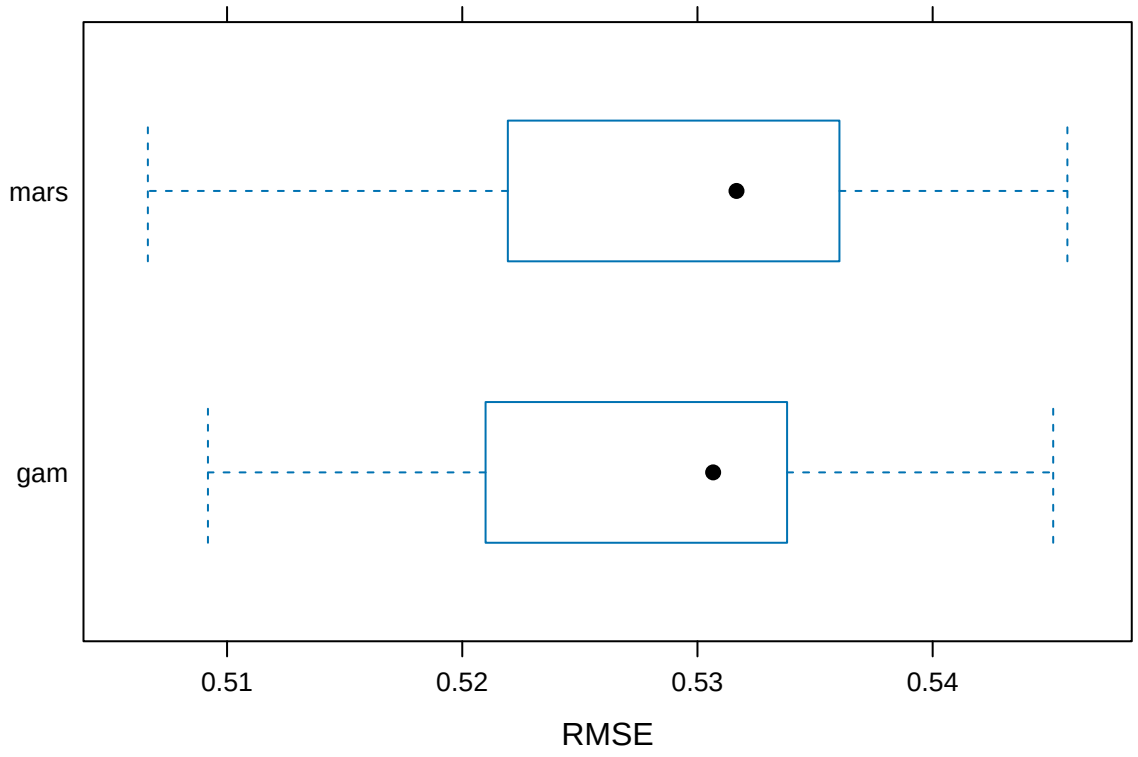
To select the best model, we can compare the cross-validated metrics of the MARS and GAM models using resampling, this helps evaluate and visualize the relative performance of the two models.

```
resamp <- resamples(list(mars = mars.fit, gam = gam.fit))
summary(resamp)
```

```
##
## Call:
## summary.resamples(object = resamp)
##
## Models: mars, gam
## Number of resamples: 10
##
## MAE
##           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## mars 0.4120189 0.4180233 0.4203065 0.4224208 0.4285348 0.4360995    0
## gam  0.4127242 0.4190075 0.4202804 0.4224455 0.4273258 0.4352565    0
##
## RMSE
##           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## mars 0.5066327 0.5230870 0.5316602 0.5282995 0.5354905 0.5457286    0
## gam  0.5091877 0.5223782 0.5306669 0.5279212 0.5336806 0.5451253    0
```

```
##
## Rsquared
##           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## mars 0.1766328 0.1941155 0.2028183 0.2159220 0.2369173 0.2730827    0
## gam  0.1795042 0.1955023 0.2071224 0.2170567 0.2376473 0.2735385    0
```

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



The distribution of RMSE across cross-validation is also similar, with GAM model showing a slightly lower median RMSE compared to MARS model.
Moreover, both models achieve lower average RMSE on the training set (dat1) compared to the Lasso model (Lasso model train RMSE: 0.5518, test RMSE: 0.5749), suggesting better performance. This may be attributed to the presence of nonlinear relationships between the response variable (log_antibody) and some predictors (such as bmi and time), which cannot be effectively captured by the linear structure of Lasso regression.

We should also evaluate the generalizability of the trained MARS and GAM models by computing RMSE on the test set (dat2).

```
# test RMSE of MARS model
mars.pred <- predict(mars.fit, newdata = dat2)
mars_test_rmse = sqrt(mean((mars.pred - dat2[, "log_antibody"])^2))
print(paste("RMSE on test set (MARS model):", round(mars_test_rmse, 4)))
```

```
## [1] "RMSE on test set (MARS model): 0.5328"
```

```r
# test RMSE of GAM model
gam.pred <- predict(gam.fit, newdata = dat2)
gam_test_rmse = sqrt(mean((gam.pred - dat2[, "log_antibody"])^2))
print(paste("RMSE on test set (GAM model):", round(gam_test_rmse, 4)))
```

```
## [1] "RMSE on test set (GAM model): 0.5701"
```

Both MARS and GAM models show very similar performance in cross-validation, with nearly identical mean RMSE values (0.5283 vs 0.5279).
However, on the test set, the MARS model achieves a lower RMSE (0.5328) compared to the GAM model (0.5701), suggesting better generalization.
Given this gap in test performance while maintaining comparable training performance, the MARS model appears to be the better choice in this case.