# DS2_midterm

My An Huynh, Soomin You

2025-03-30

```
library(tidyverse)
library(MASS)
library(caret)
library(corrplot)
library(mgcv)
library(patchwork)
library(ggplot2)
library(gtsummary)
```

## Load Data

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

## Train Data Summary Statistics

```
train = dat1 |>
  dplyr::select(-id)

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

tbl_summary(train)
```

First, the train data was loaded and separated into predictor variable matrix and response variable vector. A summary statistics table was made to summarize the mean and proportions of the variables in the train data.

| **Characteristic** | **N = 5,000**[1] |
|---|---|
| age | 60.0 (57.0, 63.0) |
| gender | 2,427 (49%) |
| race | |
| 1 | 3,221 (64%) |
| 2 | 278 (5.6%) |
| 3 | 1,036 (21%) |
| 4 | 465 (9.3%) |
| smoking | |
| 0 | 3,010 (60%) |
| 1 | 1,504 (30%) |
| 2 | 486 (9.7%) |
| height | 170.1 (166.1, 174.3) |
| weight | 80 (75, 85) |
| bmi | 27.60 (25.80, 29.50) |
| diabetes | 772 (15%) |
| hypertension | 2,298 (46%) |
| SBP | 130 (124, 135) |
| LDL | 110 (96, 124) |
| time | 106 (76, 138) |
| log_antibody | 10.09 (9.68, 10.48) |

[1] Median (Q1, Q3); n (%)

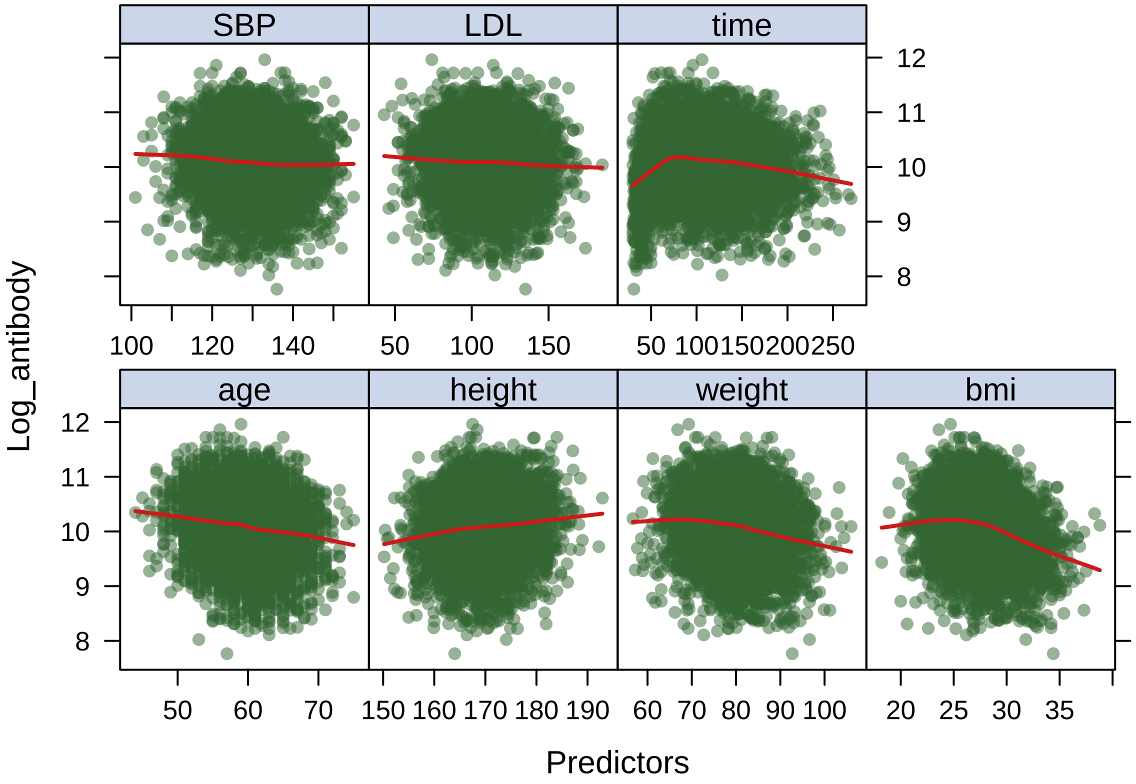
The following is a brief explanation of the predictor variables and the response variable summarized in the table above.

- race (1 = White, 2 = Asian, 3 = Black, 4 = Hispanic)
- gender (0 = Female, 1 = Male)
- smoking levels (0 = non-smoker, 1 = former smoker, 2 = current smoker)
- hypertension (0 = no hypertension, 1 = hypertension)
- diabetes (0 = non-diabetic, 1 = diabetic)
- SBP (systolic blood pressure in mmHg)
- LDL (LDL cholesterol in mg/dL)
- age (years)
- height (cm)
- weight (kg)
- BMI (kg/m^2)
- time (time passed since vaccination in days).
- log_antibody (log transformed antibody level)

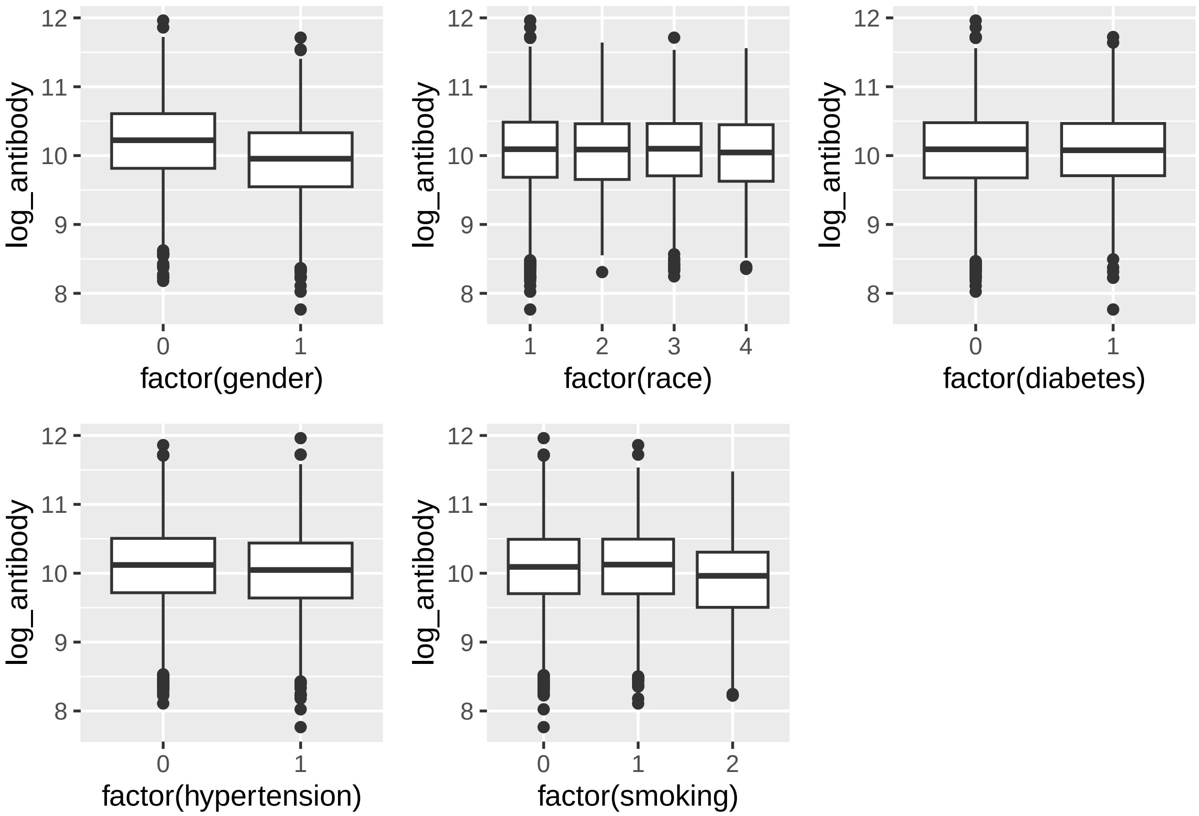
## Train Data Exploratory Data Analysis

```
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
```

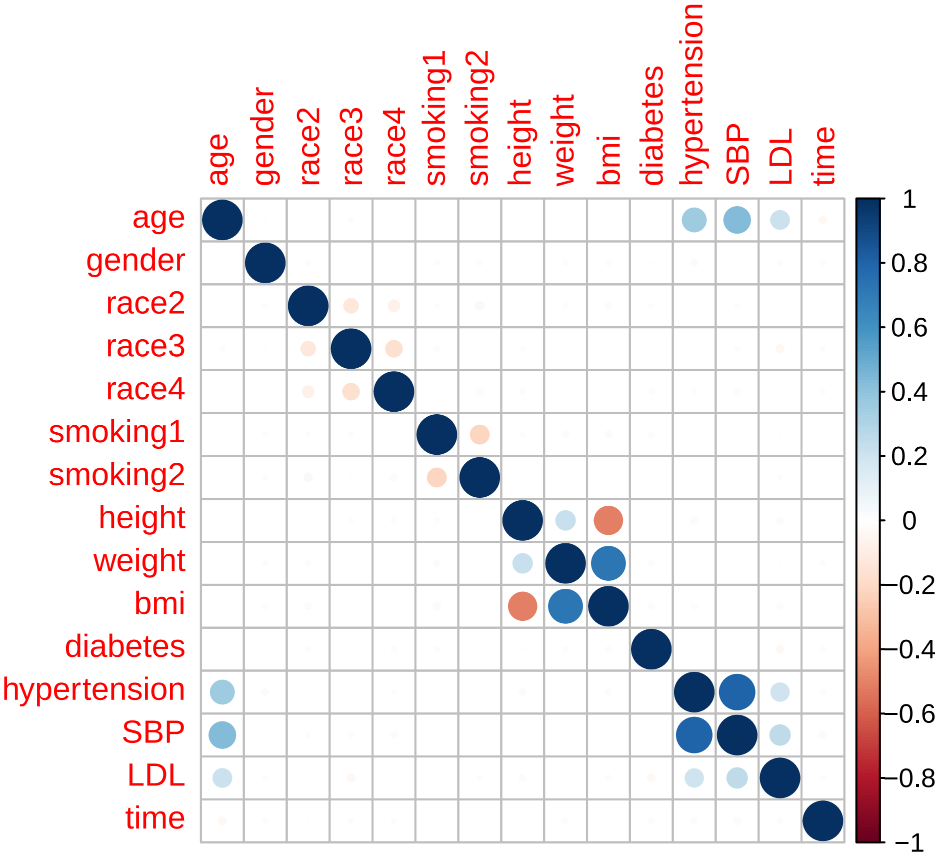
```r
theme1$strip.background$col = rgb(.0, .2, .6, .2)
trellis.par.set(theme1)

featurePlot(x = train[, -c(2,3,4,8,9,13)],
            y = train$log_antibody,
            plot = "scatter",
            span = 0.5,
            type = c("p", "smooth"),
            labels = c("Predictors", "Log_antibody"))
```



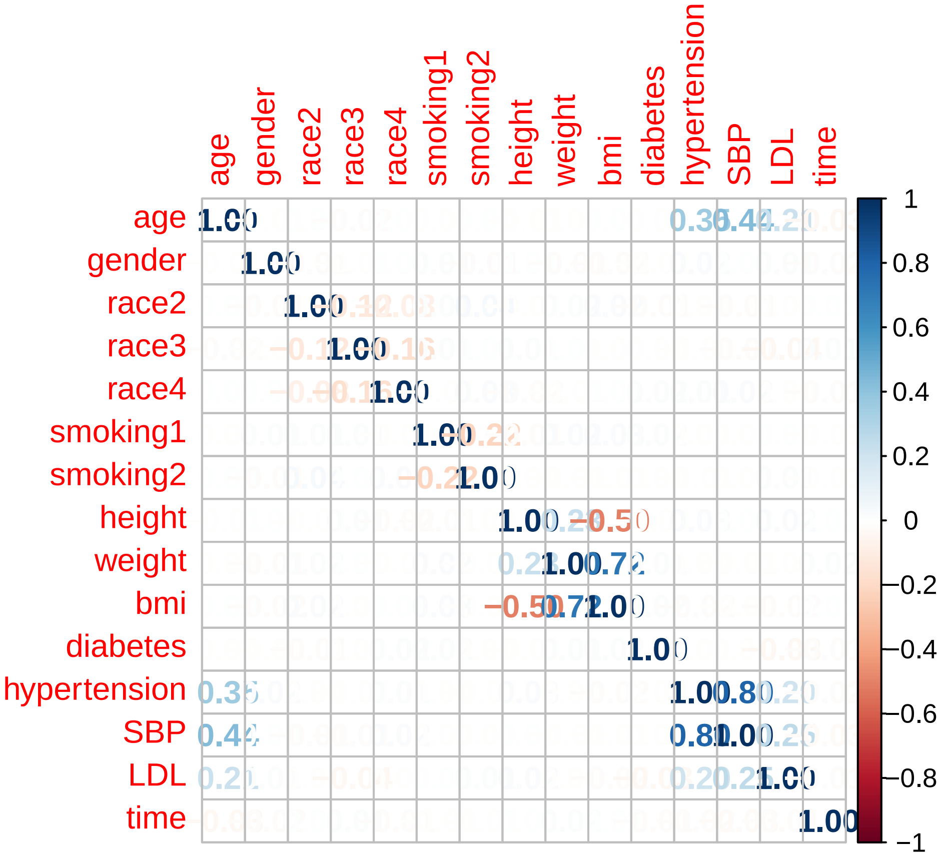
```r
# boxplot for factor/binary variables
p1 = ggplot(aes(x = factor(gender), y = log_antibody), data = train) +
  geom_boxplot()
p2 = ggplot(aes(x = factor(race), y = log_antibody), data = train) +
  geom_boxplot()
p3 = ggplot(aes(x = factor(diabetes), y = log_antibody), data = train) +
  geom_boxplot()
p4 = ggplot(aes(x = factor(hypertension), y = log_antibody), data = train) +
  geom_boxplot()
p5 = ggplot(aes(x = factor(smoking), y = log_antibody), data = train) +
  geom_boxplot()
p1 + p2 + p3 + p4 + p5
```



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



```
corrplot(cor(x), method = "number", type = "full")
```



Then, the visualization of the train data was checked. Scatterplots were made for the numeric predictor variables and boxplots were made for the factor or binary variables.

According to the scatterplots for `Systolic blood pressure (SBP)`, `LDL cholesterol (LDL)`, `Time since vaccination (time)`, `Age (age)`, `Height (height)`, `Weight (weight)`, `BMI (bmi)`, these variables indicate relatively linear relationship with the response variable `Log-transformed antibody level (log_antibody)`. Also, the boxplots indicate that the the factor and binary predictors show Gaussian characteristic.

Correlation plot was also checked for the train data and no strong multicolinearity was observed.

## Test Data

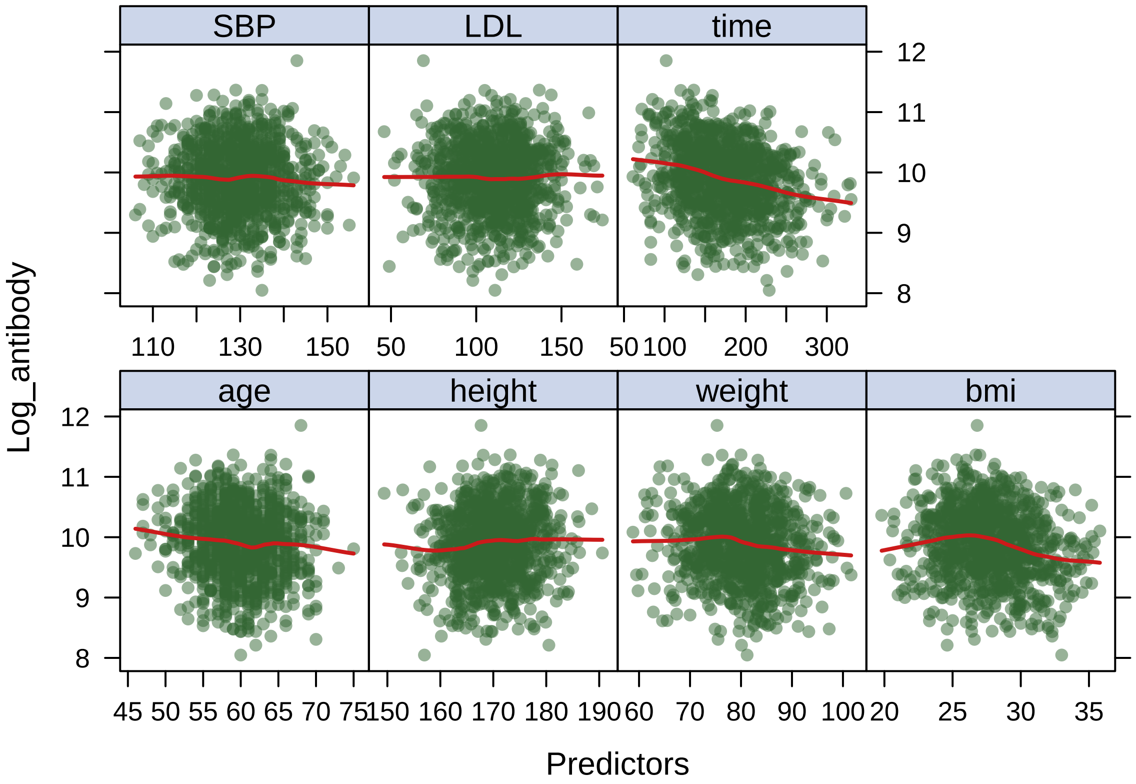
Then, the exploratory analysis was performed for the independent test data that has the same structure as the train data.

```
test = dat2 |>
  dplyr::select(-id)

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

tbl_summary(test)
```

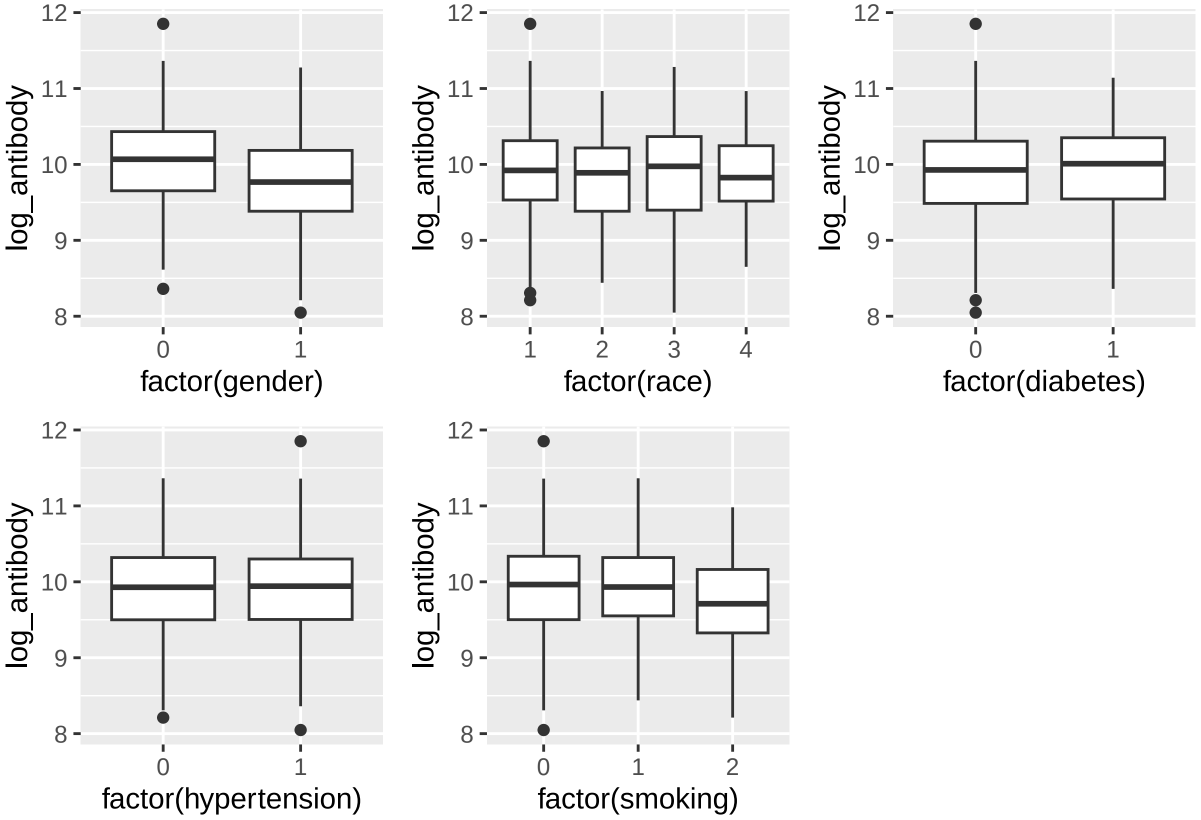
| **Characteristic** | **N = 1,000**[1] |
|---|---|
| age | 60.0 (57.0, 63.0) |
| gender | 491 (49%) |
| race | |
| 1 | 663 (66%) |
| 2 | 55 (5.5%) |
| 3 | 199 (20%) |
| 4 | 83 (8.3%) |
| smoking | |
| 0 | 601 (60%) |
| 1 | 296 (30%) |
| 2 | 103 (10%) |
| height | 170.2 (166.1, 174.2) |
| weight | 80 (75, 84) |
| bmi | 27.60 (25.80, 29.60) |
| diabetes | 157 (16%) |
| hypertension | 456 (46%) |
| SBP | 130 (124, 135) |
| LDL | 112 (96, 124) |
| time | 171 (140, 205) |
| log_antibody | 9.93 (9.50, 10.32) |

[1] Median (Q1, Q3); n (%)

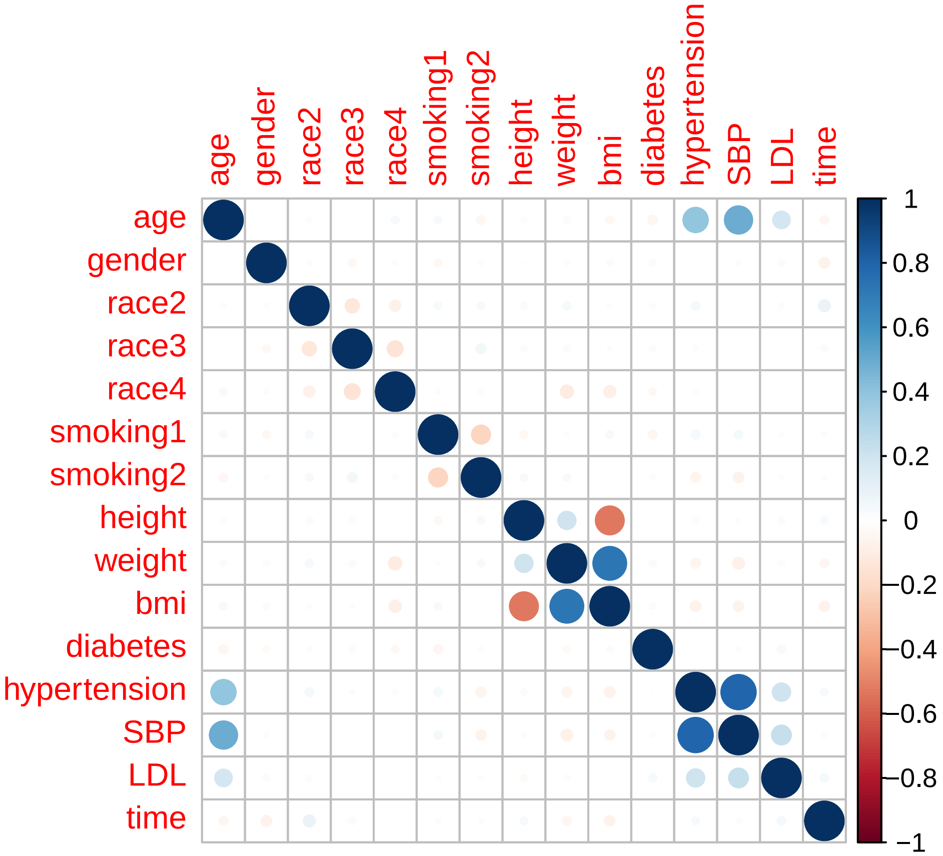
```
featurePlot(x = test[, -c(2,3,4,8,9,13)],
            y = y_test,
            plot = "scatter",
            span = 0.5,
            type = c("p", "smooth"),
            labels = c("Predictors", "Log_antibody"))
```



```
p1_test = ggplot(aes(x = factor(gender), y = log_antibody), data = test) +
  geom_boxplot()
p2_test = ggplot(aes(x = factor(race), y = log_antibody), data = test) +
  geom_boxplot()
p3_test = ggplot(aes(x = factor(diabetes), y = log_antibody), data = test) +
  geom_boxplot()
p4_test = ggplot(aes(x = factor(hypertension), y = log_antibody), data = test) +
  geom_boxplot()
p5_test = ggplot(aes(x = factor(smoking), y = log_antibody), data = test) +
  geom_boxplot()
p1_test + p2_test + p3_test + p4_test + p5_test
```



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



According to the summary statistics and visualization plots, it was be confirmed that the structure of the test data is similar to that of the train data.

## Model training

To find an optimal prediction model of antibody levels that show the impact of the demographic and clinical factors, we decided to build various models and compare. We used linear, elastic net, PLS, MARS and GAM models as shown below.

```
#linear
ctrl1 = trainControl(method = "cv", number = 10)

set.seed(1)
linear_mod = train(x, y,
                   method = "lm",
                   trControl = ctrl1)
summary(linear_mod)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.14396 -0.35840  0.02944  0.37802  1.65090
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)  26.6751961  2.3149812  11.523  < 2e-16 ***
## age          -0.0205979  0.0019385 -10.626  < 2e-16 ***
## gender       -0.2974929  0.0155977 -19.073  < 2e-16 ***
## race2        -0.0060422  0.0344613  -0.175   0.8608    
## race3        -0.0075295  0.0196815  -0.383   0.7021    
## race4        -0.0417571  0.0273309  -1.528   0.1266    
## smoking1      0.0219907  0.0173992   1.264   0.2063    
## smoking2     -0.1934834  0.0269576  -7.177 8.15e-13 ***
## height       -0.0821381  0.0135622  -6.056 1.49e-09 ***
## weight        0.0859034  0.0143481   5.987 2.29e-09 ***
## bmi          -0.2977935  0.0412612  -7.217 6.10e-13 ***
## diabetes      0.0112795  0.0215643   0.523   0.6010    
## hypertension -0.0179106  0.0260931  -0.686   0.4925    
## SBP           0.0015181  0.0017049   0.890   0.3733    
## LDL          -0.0001645  0.0004028  -0.409   0.6829    
## time         -0.0003011  0.0001795  -1.677   0.0936 .  
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.5503 on 4984 degrees of freedom
## Multiple R-squared:  0.1513, Adjusted R-squared:  0.1488 
## F-statistic: 59.25 on 15 and 4984 DF,  p-value: < 2.2e-16
```

```r
# test error
lm_pred = predict(linear_mod, newdata = x_test)
lm_rmse = sqrt(mean((y_test - lm_pred)^2))

cat("Test error for the linear model: ", lm_rmse, "\n")
```

```
## Test error for the linear model:  0.568318
```

```r
# elastic net
set.seed(1)
enet_mod = train(x = x,
                 y = y,
                 method = "glmnet",
                 tuneGrid = expand.grid(alpha = seq(0, 1, length = 25),
                                        lambda = exp(seq(-2, 2, length = 100))),
                 trControl = ctrl1)
```

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

```r
best_alpha = enet_mod$bestTune$alpha
best_lambda = enet_mod$bestTune$lambda

# test error
enet_pred = predict(enet_mod, newdata = x_test)
enet_rmse = sqrt(mean((y_test - enet_pred)^2))

cat("The optimal alpha for the elastic net model: ", best_alpha, "\n")
```

```
## The optimal alpha for the elastic net model:  0
```

```r
cat("The optimal alpha for the elastic net model: ", best_lambda, "\n")
```

```
## The optimal alpha for the elastic net model:  0.1353353
```

```r
cat("Test error for the linear model: ", enet_rmse, "\n")
```

```
## Test error for the linear model:  0.5728115
```

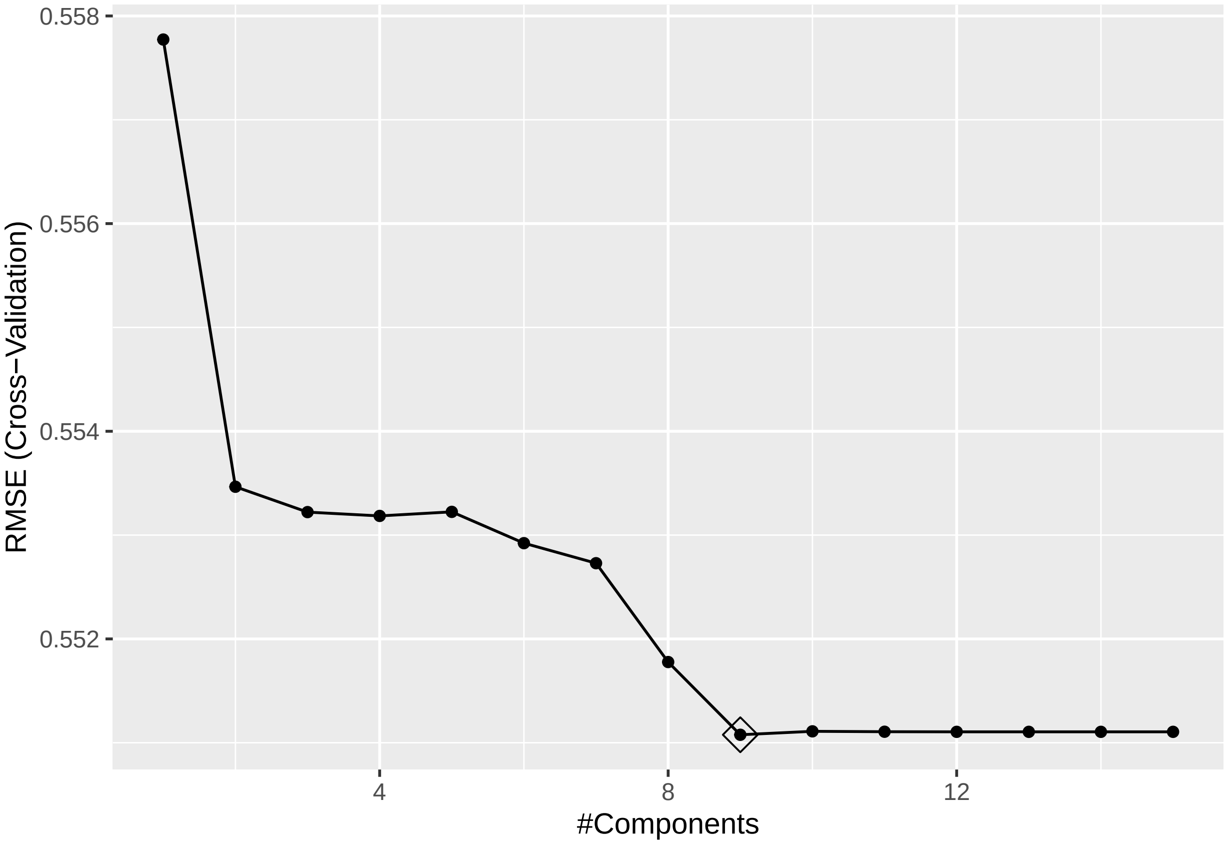
```r
# pls model
set.seed(1)
pls_mod = train(x = x,
                y = y,
                method = "pls",
                tuneGrid = data.frame(ncomp = 1:15),
                trControl = ctrl1,
                preProcess = c("center", "scale"))
summary(pls_mod)
```

```
## Data:    X dimension: 5000 15
##  Y dimension: 5000 1
## Fit method: oscorespls
## Number of components considered: 9
## TRAINING: % variance explained
##           1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps
## X           10.77    19.88    31.77    38.37    43.94    47.54    53.19
## .outcome    12.83    14.29    14.38    14.41    14.42    14.45    14.51
##           8 comps  9 comps
## X           54.88    60.39
## .outcome    15.03    15.13
```

```r
# test error
pls_pred = predict(pls_mod, newdata = x_test)
pls_rmse = sqrt(mean((y_test - pls_pred)^2))

cat("Test error for the linear model: ", pls_rmse, "\n")
```

```
## Test error for the linear model:  0.5690832
```

```r
ggplot(pls_mod, highlight = TRUE)
```



```r
# mars
set.seed(1)
mars_mod = train(x = x,
                 y = y,
                 method = "earth",
                 trControl = ctrl1)
```

```
## Loading required package: earth

## Loading required package: Formula

## Loading required package: plotmo

## Loading required package: plotrix
```

```r
summary(mars_mod)
```

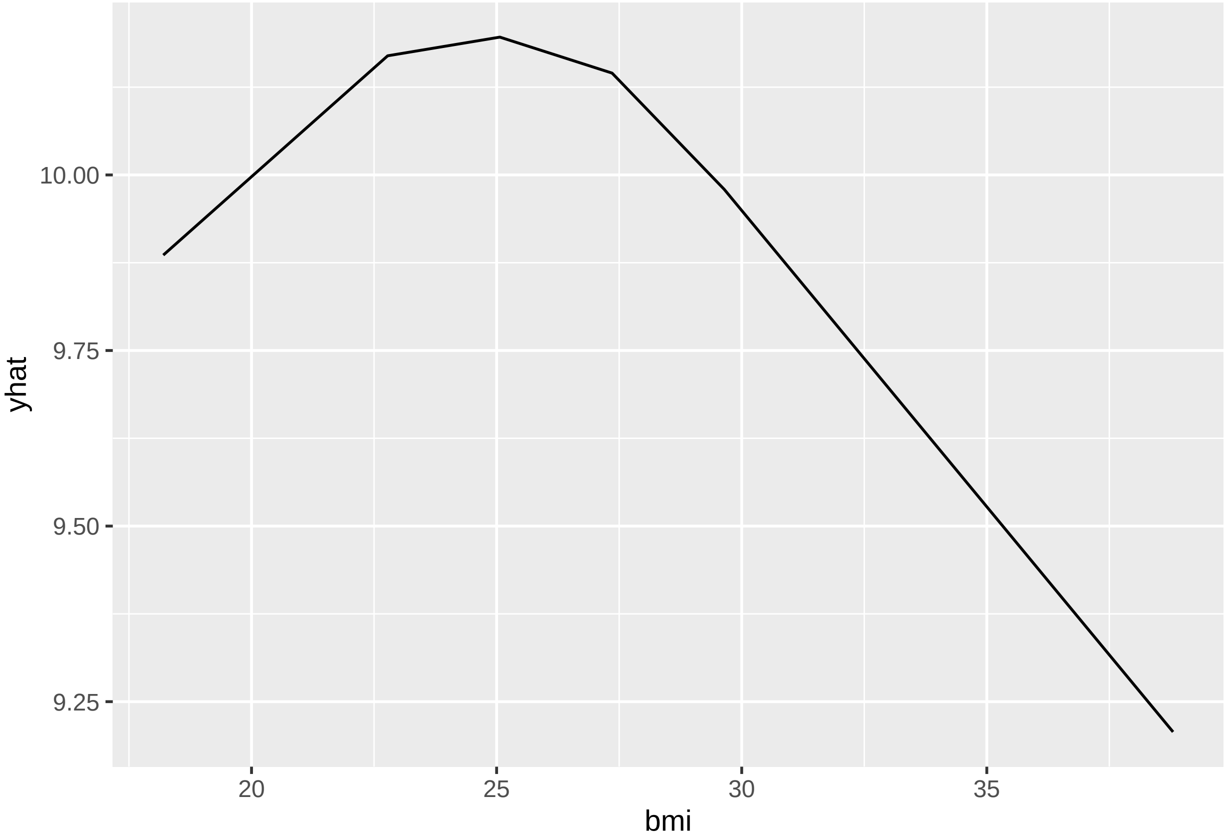
```
## Call: earth(x=matrix[5000,15], y=c(10.65,9.889,1...), keepxy=TRUE, degree=1,
##             nprune=9)
##
##             coefficients
## (Intercept)   10.8474469
## gender        -0.2962905
## smoking2      -0.2051269
## h(59-age)      0.0161385
## h(age-59)     -0.0229576
## h(bmi-23.7)   -0.0843802
## h(27.8-bmi)   -0.0619974
## h(57-time)    -0.0335293
## h(time-57)    -0.0022542
```

```
##
## Selected 9 of 10 terms, and 5 of 15 predictors (nprune=9)
## Termination condition: RSq changed by less than 0.001 at 10 terms
## Importance: bmi, gender, time, age, smoking2, race2-unused, race3-unused, ...
## Number of terms at each degree of interaction: 1 8 (additive model)
## GCV 0.2787787    RSS 1384.431    GRSq 0.2166152    RSq 0.2216218
```

```
# test error
mars_pred = predict(mars_mod, newdata = x_test)
mars_rmse = sqrt(mean((y_test - mars_pred)^2))

cat("Test error for the linear model: ", mars_rmse, "\n")
```

```
## Test error for the linear model:  0.5327718
```

```
pdp::partial(mars_mod, pred.var = c("bmi"), grid.resolution = 10) |>
  autoplot()
```



To better understand the relationship, partial dependence plot (PDP) for BMI predictor variable and the response variable was plotted.

```
# gam
set.seed(1)
gam_mod = train(x = x,
                y = y,
                method = "gam",
                trControl = ctrl1)

gam_mod$bestTune
```

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

```r
gam_mod$finalModel
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## .outcome ~ gender + race2 + race3 + race4 + smoking1 + smoking2 +
##     diabetes + hypertension + 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
```

```r
summary(gam_mod)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## .outcome ~ gender + race2 + race3 + race4 + smoking1 + smoking2 +
##     diabetes + hypertension + s(age) + s(SBP) + s(LDL) + s(bmi) +
##     s(time) + s(height) + s(weight)
##
## Parametric coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  10.228177   0.015328 667.269  < 2e-16 ***
## gender       -0.297837   0.014933 -19.945  < 2e-16 ***
## race2        -0.003296   0.033009  -0.100    0.920
## race3        -0.010509   0.018837  -0.558    0.577
## race4        -0.037424   0.026176  -1.430    0.153
## smoking1      0.022219   0.016660   1.334    0.182
## smoking2     -0.193175   0.025834  -7.478 8.9e-14 ***
## diabetes      0.014230   0.020640   0.689    0.491
## hypertension -0.007678   0.015995  -0.480    0.631
## ---
## 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)    9.908e-01      9 13.733  <2e-16 ***
## s(SBP)    6.175e-07      9  0.000   0.765
## s(LDL)    6.648e-07      9  0.000   0.639
## s(bmi)    4.179e+00      9 41.897  <2e-16 ***
## s(time)   7.892e+00      9 44.960  <2e-16 ***
## s(height) 1.234e+00      9  0.278   0.121
## s(weight) 2.262e-06      9  0.000   0.666
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =   0.22   Deviance explained = 22.4%
## GCV = 0.27867  Scale est. = 0.27738   n = 5000
```

```
# test error
gam_pred = predict(gam_mod, newdata = x_test)
gam_rmse = sqrt(mean((y_test - gam_pred)^2))

cat("Test error for the linear model: ", gam_rmse, "\n")
```

```
## Test error for the linear model:  0.5700836
```

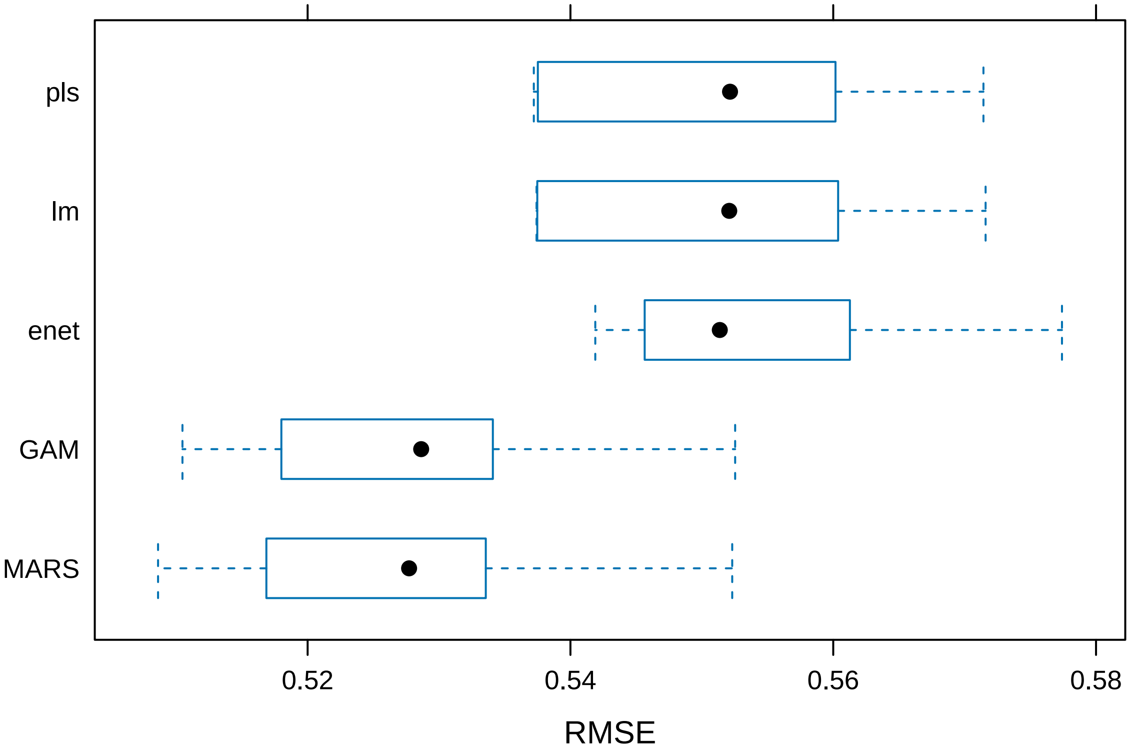
## Model validation

To evaluate the robustness and generalizability of the prediction model using the new dataset, test data, the five fitted models were compared using resampling.

```
set.seed(1)
resamp = resamples(list(lm = linear_mod,
                        enet = enet_mod,
                        pls = pls_mod,
                        MARS = mars_mod,
                        GAM = gam_mod))
summary(resamp)
```

```
##
## Call:
## summary.resamples(object = resamp)
##
## Models: lm, enet, pls, MARS, GAM
## Number of resamples: 10
##
## MAE
##           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lm   0.4206892 0.4321712 0.4404316 0.4391934 0.4453954 0.4598289    0
## enet 0.4283094 0.4374741 0.4398703 0.4416173 0.4449426 0.4658299    0
## pls  0.4205572 0.4324090 0.4404365 0.4392051 0.4454607 0.4596797    0
## MARS 0.4033817 0.4179151 0.4234065 0.4221856 0.4275553 0.4418075    0
## GAM  0.4042397 0.4194058 0.4242789 0.4228546 0.4275021 0.4421737    0
##
## RMSE
##           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lm   0.5374189 0.5385233 0.5520867 0.5511050 0.5589697 0.5715949    0
## enet 0.5418907 0.5460887 0.5513664 0.5541943 0.5592095 0.5774083    0
## pls  0.5372125 0.5386023 0.5521459 0.5510768 0.5588196 0.5714301    0
## MARS 0.5086158 0.5180140 0.5277205 0.5277239 0.5330971 0.5523110    0
## GAM  0.5104739 0.5194562 0.5286409 0.5286279 0.5334558 0.5525356    0
##
## Rsquared
##            Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lm   0.08863443 0.1400413 0.1491380 0.1475987 0.1614569 0.1965523    0
## enet 0.09065183 0.1294793 0.1409665 0.1404523 0.1642418 0.1736794    0
## pls  0.08861598 0.1403744 0.1493405 0.1476804 0.1609204 0.1975762    0
## MARS 0.16452631 0.2016386 0.2149715 0.2179857 0.2395080 0.2656206    0
## GAM  0.16168005 0.1988396 0.2126428 0.2152374 0.2350941 0.2640126    0
```

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



According to the model comparison via 10-fold cross validation, the mean RMSE of the MARS model is the lowest for this specific dataset and predefined seed. Hence, MARS is chosen as the final model to predict the relationship between log-transformed antibody levels and the predictors of interest. Also, MARS offers better interpretability.