

# P8106\_midterm\_project\_v1

Congyu Yang, Yujing Fu, Zhengkun Ou

2025-03-27

```
library(pacman)
p_load(tidyverse, caret, tidymodels, corrplot, ggplot2, plotmo, ggrepel, patchwork, earth, pdp, mgcv, k

## package 'cli' successfully unpacked and MD5 sums checked

## package 'purrr' successfully unpacked and MD5 sums checked

## package 'rlang' successfully unpacked and MD5 sums checked

## package 'tidymodels' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
## C:\Users\Persica\AppData\Local\Temp\RtmpQHRWD6\downloaded_packages
```

## 1. Data Processing

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

```
skimr::skim(dat1)
```

Table 1: Data summary

Name	dat1
Number of rows	5000
Number of columns	14
Column type frequency:	
factor	2
numeric	12
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
race	0	1	FALSE	4	1: 3221, 3: 1036, 4: 465, 2: 278
smoking	0	1	FALSE	3	0: 3010, 1: 1504, 2: 486

#### Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
id	0	1	2500.50	1443.52	1.00	1250.75	2500.50	3750.25	5000.00	
age	0	1	59.97	4.50	44.00	57.00	60.00	63.00	75.00	
gender	0	1	0.49	0.50	0.00	0.00	0.00	1.00	1.00	
height	0	1	170.13	5.94	150.20	166.10	170.10	174.22	192.90	
weight	0	1	80.11	7.06	56.70	75.40	80.10	84.90	106.00	
bmi	0	1	27.74	2.76	18.20	25.80	27.60	29.50	38.80	
diabetes	0	1	0.15	0.36	0.00	0.00	0.00	0.00	1.00	
hypertension	0	1	0.46	0.50	0.00	0.00	0.00	1.00	1.00	
SBP	0	1	129.90	8.00	101.00	124.00	130.00	135.00	155.00	
LDL	0	1	109.91	20.15	43.00	96.00	110.00	124.00	185.00	
time	0	1	108.86	43.42	30.00	76.00	106.00	138.00	270.00	
log_antibody	0	1	10.06	0.60	7.77	9.68	10.09	10.48	11.96	

```
skimr::skim(dat2)
```

Table 4: Data summary

Name	dat2
Number of rows	1000
Number of columns	14
Column type frequency:	
factor	2
numeric	12
Group variables	None

#### Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
race	0	1	FALSE	4	1: 663, 3: 199, 4: 83, 2: 55
smoking	0	1	FALSE	3	0: 601, 1: 296, 2: 103

#### Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
id	0	1	5500.50	288.82	5001.00	5250.75	5500.50	5750.25	6000.00	
age	0	1	60.02	4.45	46.00	57.00	60.00	63.00	75.00	
gender	0	1	0.49	0.50	0.00	0.00	0.00	1.00	1.00	
height	0	1	170.22	6.02	149.40	166.10	170.20	174.20	190.60	

skim_variablen_	missing	complete	_rate	mean	sd	p0	p25	p50	p75	p100	hist
weight	0	1	80.13	7.05	58.80	75.30	80.20	84.40	101.60		
bmi	0	1	27.72	2.82	19.80	25.80	27.60	29.60	35.80		
diabetes	0	1	0.16	0.36	0.00	0.00	0.00	0.00	1.00		
hypertension	0	1	0.46	0.50	0.00	0.00	0.00	1.00	1.00		
SBP	0	1	129.61	8.20	106.00	124.00	130.00	135.00	156.00		
LDL	0	1	110.25	20.32	46.00	96.00	112.00	124.00	174.00		
time	0	1	173.77	46.78	61.00	140.00	171.00	205.00	330.00		
log_antibody	0	1	9.90	0.59	8.05	9.50	9.93	10.31	11.85		

There is no missing value in this dataset.

```
colnames(dat1)
```

```
## [1] "id"          "age"          "gender"       "race"         "smoking"
## [6] "height"      "weight"       "bmi"          "diabetes"     "hypertension"
## [11] "SBP"         "LDL"          "time"         "log_antibody"
```

```
# Clean variable names
```

```
dat1 = dat1 |> janitor::clean_names()
dat2 = dat2 |> janitor::clean_names()
```

We can see there is no NA values in the dataset.

```
# Convert variables to factors
```

```
dat1$gender <- factor(dat1$gender, levels = c(0, 1), labels = c("Female", "Male"))
dat1$race <- factor(dat1$race, levels = c(1, 2, 3, 4), labels = c("White", "Asian", "Black", "Hispanic"))
dat1$smoking <- factor(dat1$smoking, levels = c(0, 1, 2), labels = c("Never", "Former", "Current"))
dat1$diabetes <- factor(dat1$diabetes, levels = c(0, 1), labels = c("No", "Yes"))
dat1$hypertension <- factor(dat1$hypertension, levels = c(0, 1), labels = c("No", "Yes"))

dat2$gender <- factor(dat2$gender, levels = c(0, 1), labels = c("Female", "Male"))
dat2$race <- factor(dat2$race, levels = c(1, 2, 3, 4), labels = c("White", "Asian", "Black", "Hispanic"))
dat2$smoking <- factor(dat2$smoking, levels = c(0, 1, 2), labels = c("Never", "Former", "Current"))
dat2$diabetes <- factor(dat2$diabetes, levels = c(0, 1), labels = c("No", "Yes"))
dat2$hypertension <- factor(dat2$hypertension, levels = c(0, 1), labels = c("No", "Yes"))
```

```
# Summary statistics
```

```
summary_stats <- summary(dat1)
print(summary_stats)
```

```
##          id          age          gender          race          smoking
## Min.    : 1      Min.    :44.00  Female:2573  White   :3221  Never   :3010
## 1st Qu.:1251    1st Qu.:57.00  Male  :2427  Asian   : 278  Former  :1504
## Median :2500    Median :60.00                Black   :1036  Current: 486
## Mean    :2500    Mean    :59.97                Hispanic: 465
## 3rd Qu.:3750    3rd Qu.:63.00
## Max.    :5000    Max.    :75.00
##      height      weight          bmi      diabetes      hypertension
## Min.    :150.2    Min.    : 56.70  Min.    :18.20  No :4228  No :2702
```

```
## 1st Qu.:166.1 1st Qu.: 75.40 1st Qu.:25.80 Yes: 772 Yes:2298
## Median :170.1 Median : 80.10 Median :27.60
## Mean :170.1 Mean : 80.11 Mean :27.74
## 3rd Qu.:174.2 3rd Qu.: 84.90 3rd Qu.:29.50
## Max. :192.9 Max. :106.00 Max. :38.80
## sbp ldl time log_antibody
## Min. :101.0 Min. : 43.0 Min. : 30.0 Min. : 7.765
## 1st Qu.:124.0 1st Qu.: 96.0 1st Qu.: 76.0 1st Qu.: 9.682
## Median :130.0 Median :110.0 Median :106.0 Median :10.089
## Mean :129.9 Mean :109.9 Mean :108.9 Mean :10.064
## 3rd Qu.:135.0 3rd Qu.:124.0 3rd Qu.:138.0 3rd Qu.:10.478
## Max. :155.0 Max. :185.0 Max. :270.0 Max. :11.961
```

```
# Check for missing values
missing_values <- colSums(is.na(dat1))
print(missing_values)
```

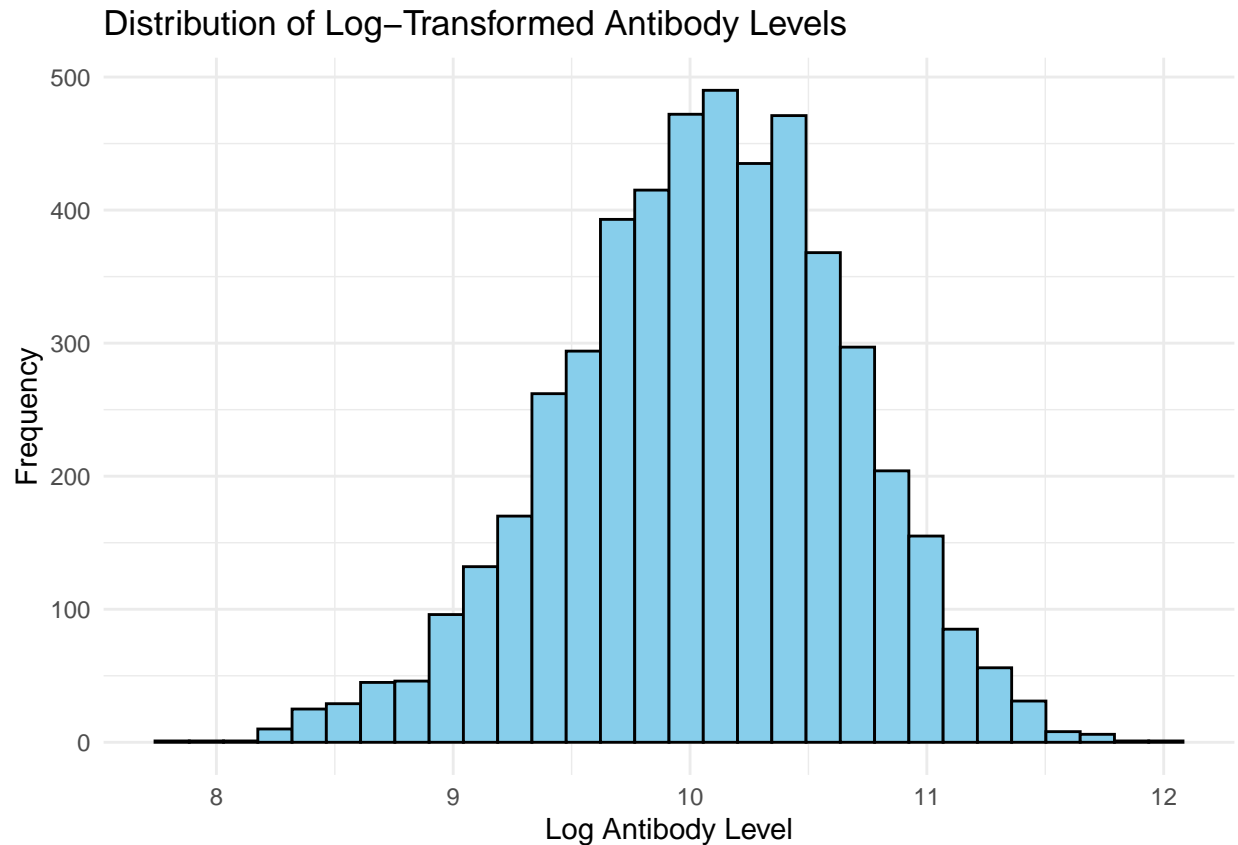
```
## id age gender race smoking height
## 0 0 0 0 0 0
## weight bmi diabetes hypertension sbp ldl
## 0 0 0 0 0 0
## time log_antibody
## 0 0
```

There is no missing value in the dataset.

## 2. Exploratory Data Analysis (EDA)

### 2.1 Distribution of log antibody

```
# Distribution of the response variable
p1 <- ggplot(dat1, aes(x = log_antibody)) +
  geom_histogram(fill = "skyblue", color = "black", bins = 30) +
  labs(title = "Distribution of Log-Transformed Antibody Levels",
       x = "Log Antibody Level", y = "Frequency") +
  theme_minimal()
p1
```



## 2.2 Boxplot of log antibody

```
# Boxplot of log_antibody by gender
p2 <- ggplot(dat1, aes(x = gender, y = log_antibody, fill = gender)) +
  geom_boxplot() +
  labs(title = "Antibody Levels by Gender",
       x = "Gender",
       y = "Log Antibody Level") +
  theme_minimal()

# Boxplot of log_antibody by race
p3 <- ggplot(dat1, aes(x = race, y = log_antibody, fill = race)) +
  geom_boxplot() +
  labs(title = "Antibody Levels by Race/Ethnicity",
       x = "Race/Ethnicity",
       y = "Log Antibody Level") +
  theme_minimal()

# Boxplot of log_antibody by smoking status
p4 <- ggplot(dat1, aes(x = smoking, y = log_antibody, fill = smoking)) +
  geom_boxplot() +
  labs(title = "Antibody Levels by Smoking Status",
       x = "Smoking Status",
       y = "Log Antibody Level") +
```

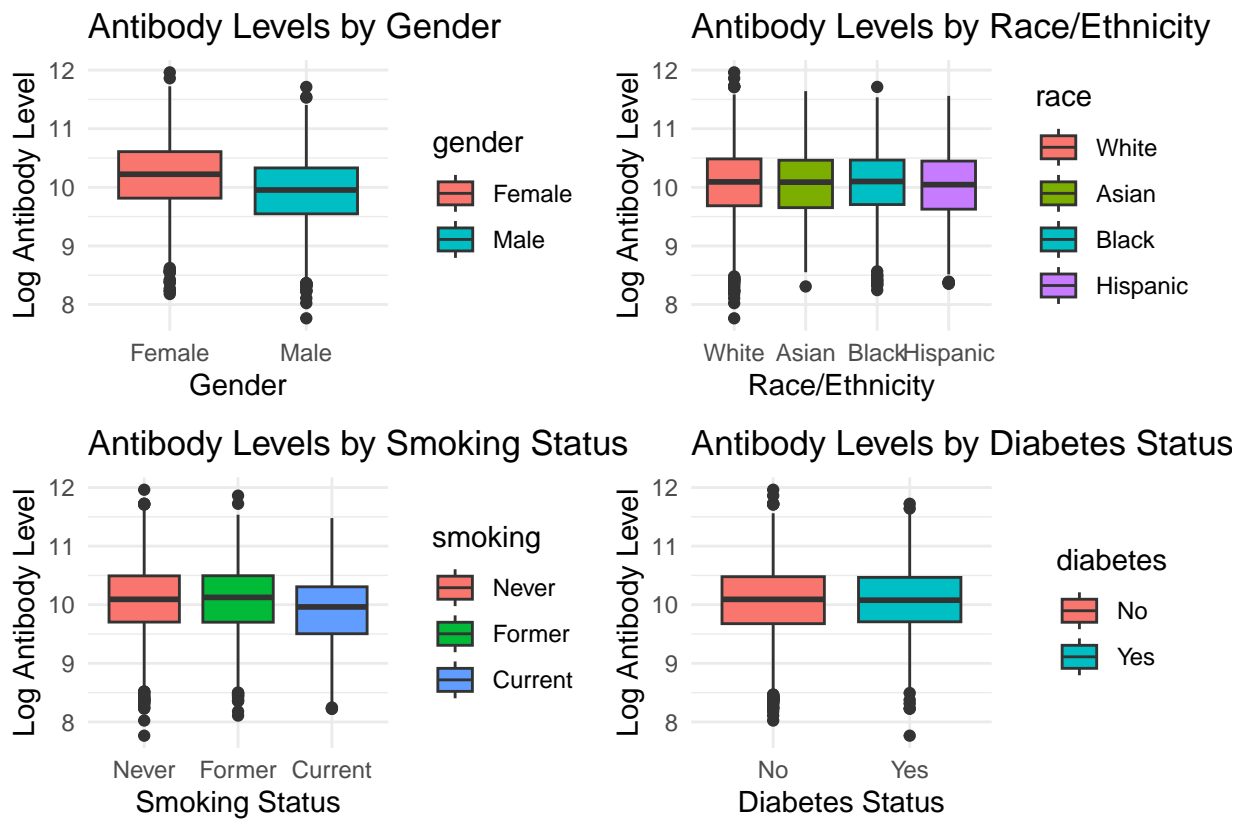
```

theme_minimal()

# Boxplot of log_antibody by diabetes status
p5 <- ggplot(dat1, aes(x = diabetes, y = log_antibody, fill = diabetes)) +
  geom_boxplot() +
  labs(title = "Antibody Levels by Diabetes Status",
       x = "Diabetes Status",
       y = "Log Antibody Level") +
  theme_minimal()

```

p2+p3+p4+p5



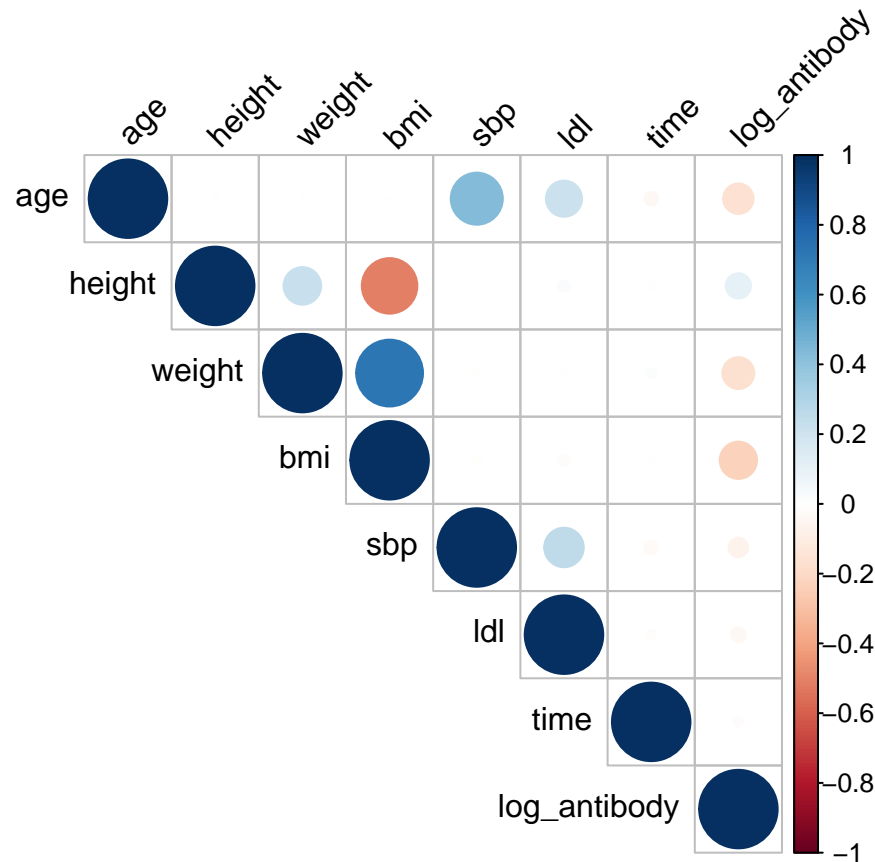
## 2.3 Correlation Plot

```

# Scatterplot matrix for continuous variables
continuous_vars <- dat1 %>%
  select(age, height, weight, bmi, sbp, ldl, time, log_antibody)

# Calculate correlation matrix
correlation_matrix <- cor(continuous_vars)
corrplot(correlation_matrix, method = "circle", type = "upper",
         tl.col = "black", tl.srt = 45)

```

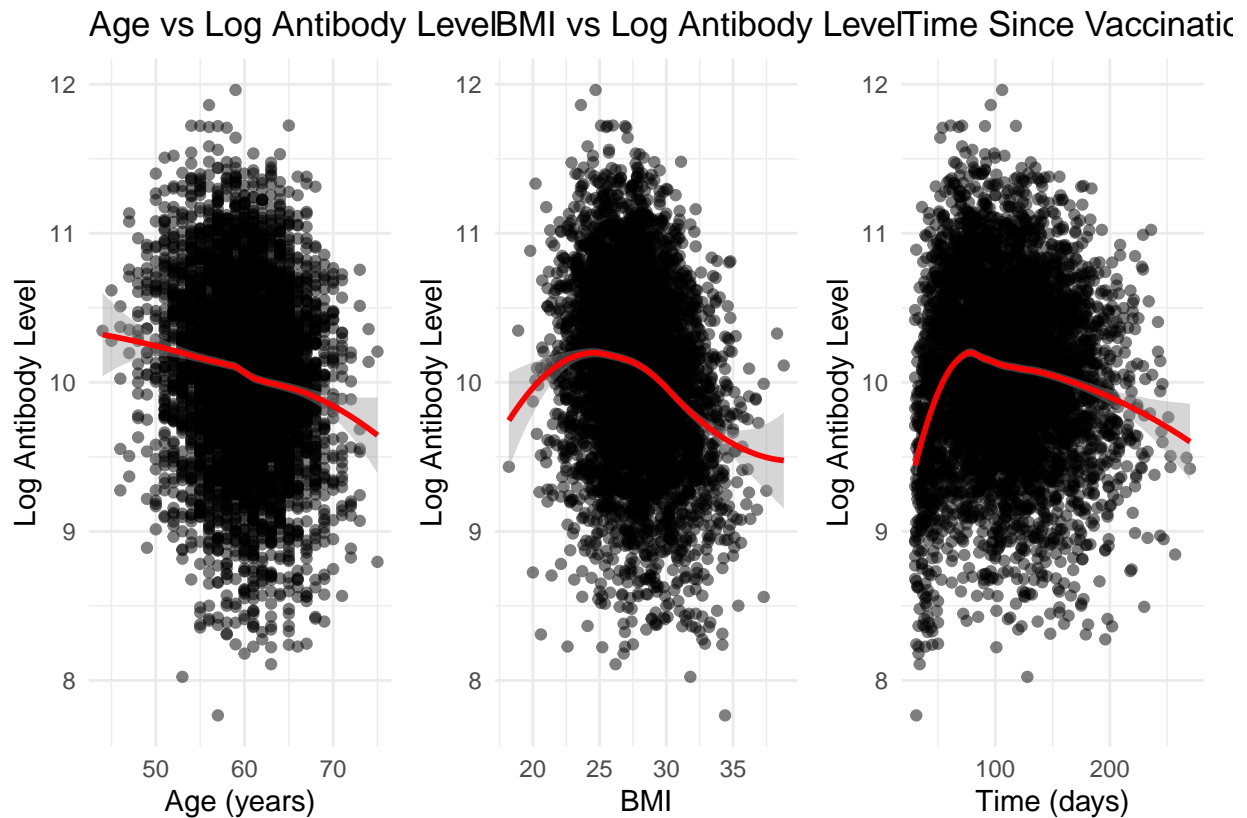


## 2.4 Scatterplots

```
# Create scatterplots for continuous variables vs response
p6 <- ggplot(dat1, aes(x = age, y = log_antibody)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "loess", color = "red") +
  labs(title = "Age vs Log Antibody Level",
       x = "Age (years)", y = "Log Antibody Level") +
  theme_minimal()

p7 <- ggplot(dat1, aes(x = bmi, y = log_antibody)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "loess", color = "red") +
  labs(title = "BMI vs Log Antibody Level",
       x = "BMI", y = "Log Antibody Level") +
  theme_minimal()

p8 <- ggplot(dat1, aes(x = time, y = log_antibody)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "loess", color = "red") +
  labs(title = "Time Since Vaccination vs Log Antibody Level",
       x = "Time (days)", y = "Log Antibody Level") +
  theme_minimal()
```



There are some potential nonlinear trend between `log_antibody` and `bmi`, and between `log_antibody` and `time`.

### 3. Model Training with Cross-validation

#### 3.1 Linear models

```
model_formula <- log_antibody ~ . - id

# Set up cross-validation control
ctrl <- trainControl(method = "cv", number = 10)

# 3.1 Linear Regression
set.seed(123)
lm.fit <- train(
  model_formula,
  data = dat1,
  method = "lm",
  trControl = ctrl
)
summary(lm.fit$finalModel)
```



### 3.1.1 Linear regression

```
##
## 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 ***
## genderMale    -0.2974929   0.0155977  -19.073 < 2e-16 ***
## raceAsian     -0.0060422   0.0344613   -0.175  0.8608
## raceBlack     -0.0075295   0.0196815   -0.383  0.7021
## raceHispanic  -0.0417571   0.0273309   -1.528  0.1266
## smokingFormer  0.0219907   0.0173992    1.264  0.2063
## smokingCurrent -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 ***
## diabetesYes    0.0112795   0.0215643    0.523  0.6010
## hypertensionYes -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
```

```
set.seed(123)
ridge.fit <- train(
  model_formula,
  data = dat1,
  method = "glmnet",
  tuneGrid = expand.grid(alpha = 0,
                        lambda = exp(seq(6, -6, length = 100))),
  trControl = ctrl
)

coef(ridge.fit$finalModel, s = ridge.fit$bestTune$lambda)
```

### 3.1.2 Ridge Regression

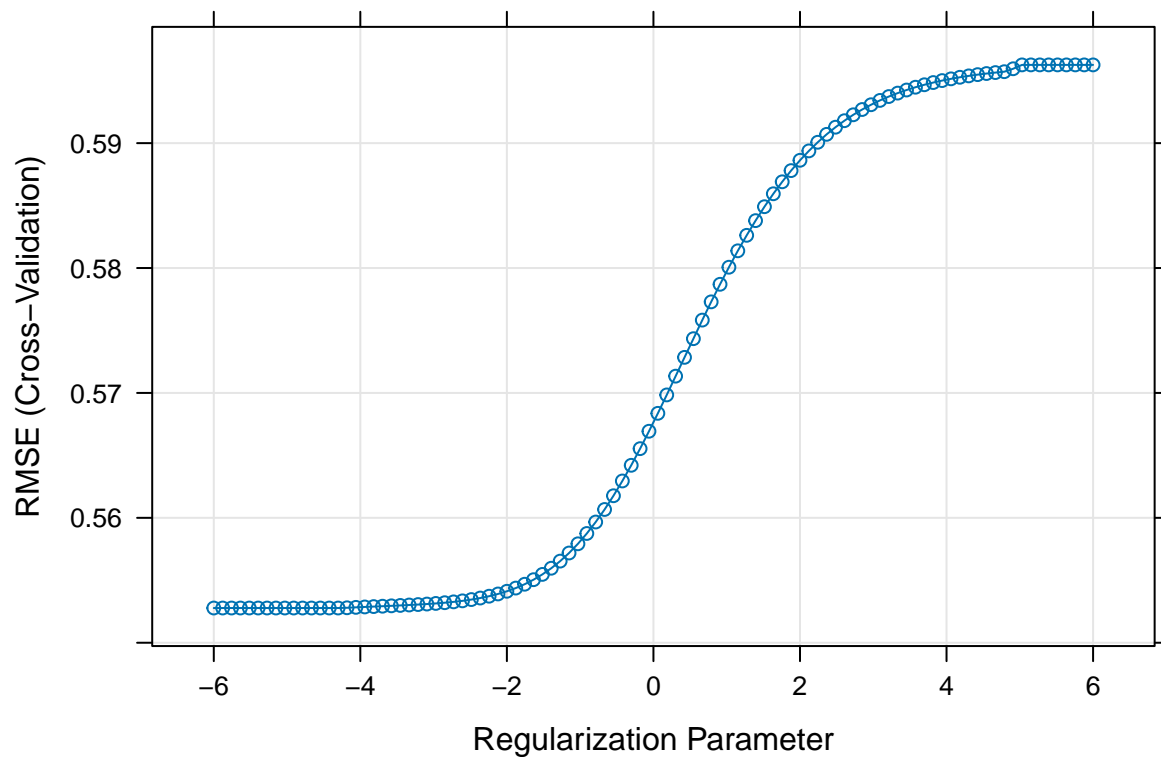
```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                               s1
```

```
## (Intercept)      12.7384560516
## age              -0.0197679168
## genderMale       -0.2880871615
## raceAsian        -0.0038318346
## raceBlack        -0.0066654722
## raceHispanic     -0.0417690245
## smokingFormer    0.0242172365
## smokingCurrent   -0.1847225211
## height           -0.0001949552
## weight           -0.0009185524
## bmi              -0.0473178893
## diabetesYes      0.0113069924
## hypertensionYes -0.0166641972
## sbp              0.0010847735
## ldl              -0.0001614795
## time             -0.0002807285
```

```
print(ridge.fit$bestTune)
```

```
##      alpha      lambda
## 15      0 0.0135275
```

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



```

set.seed(123)
lasso.fit <- train(
  model_formula,
  data = dat1,
  method = "glmnet",
  tuneGrid = expand.grid(alpha = 1,
                        lambda = exp(seq(6, -6, length = 100))),
  trControl = ctrl
)

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

```

### 3.1.3 Lasso Regression

```

## 16 x 1 sparse Matrix of class "dgCMatrix"
##                               s1
## (Intercept)      1.272648e+01
## age             -1.915006e-02
## genderMale      -2.859715e-01
## raceAsian        .
## raceBlack        .
## raceHispanic    -2.472809e-02
## smokingFormer    1.593301e-02
## smokingCurrent  -1.770968e-01
## height           .
## weight          -1.029699e-04
## bmi             -4.801089e-02
## diabetesYes      4.273525e-05
## hypertensionYes .
## sbp              .
## ldl              .
## time            -1.848766e-04

```

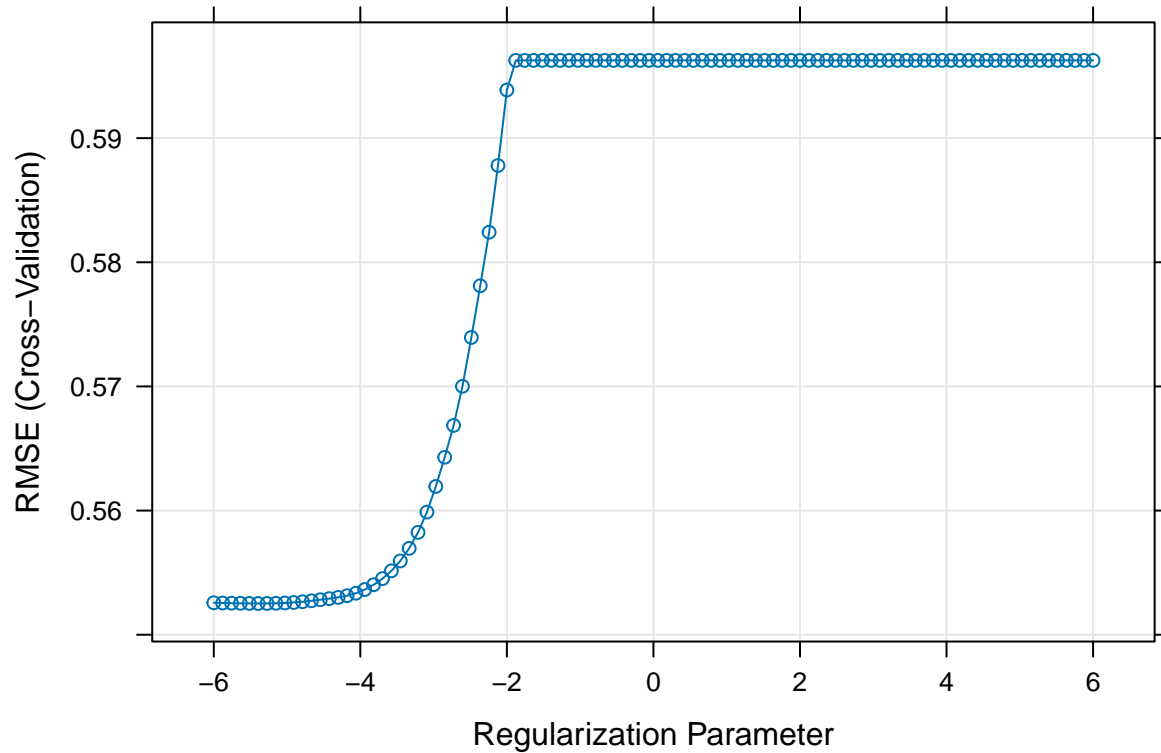
```
print(lasso.fit$bestTune)
```

```

##   alpha      lambda
## 6      1 0.004544037

```

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



```
set.seed(123)
enet.fit <- train(
  model_formula,
  data = dat1,
  method = "glmnet",
  tuneGrid = expand.grid(alpha = seq(0, 1, length = 21),
                        lambda = exp(seq(6, -6, length = 100))),
  trControl = ctrl
)
summary(enet.fit)
```

### 3.1.4 Elastic Net Regression

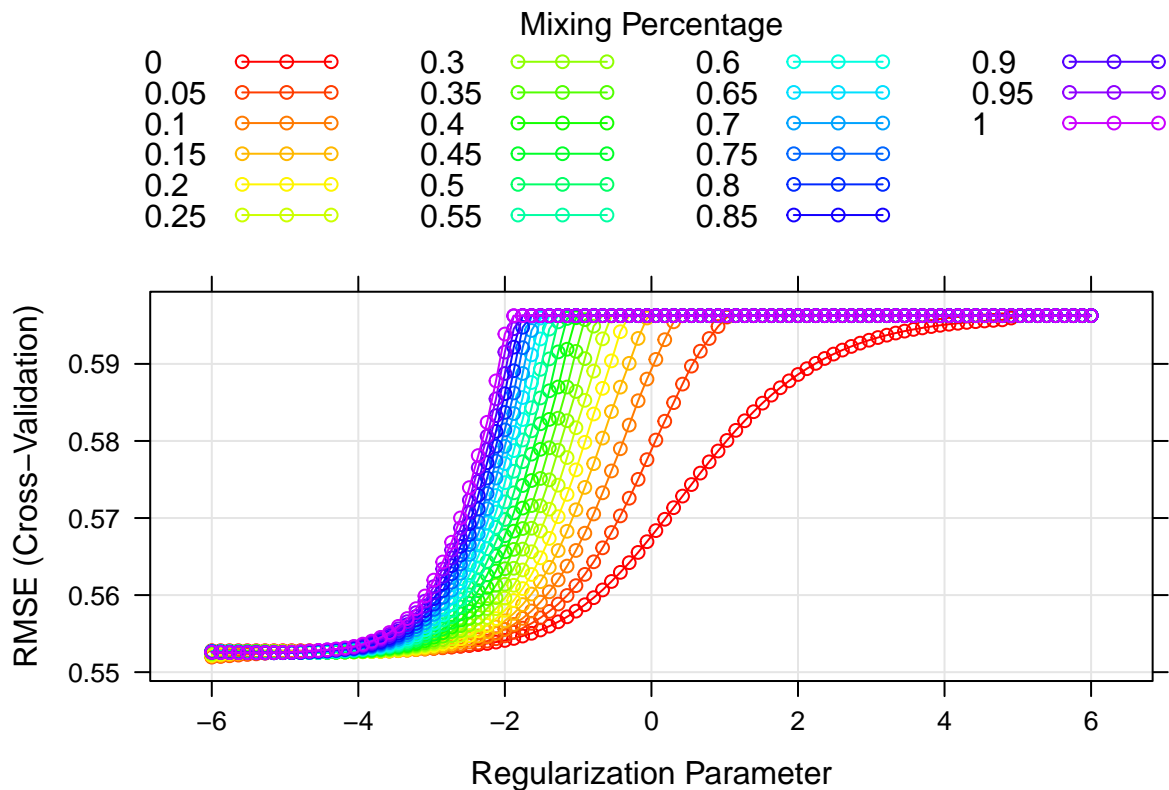
##	Length	Class	Mode
## a0	100	-none-	numeric
## beta	1500	dgCMatrix	S4
## df	100	-none-	numeric
## dim	2	-none-	numeric
## lambda	100	-none-	numeric
## dev.ratio	100	-none-	numeric
## nulldev	1	-none-	numeric
## npasses	1	-none-	numeric
## jerr	1	-none-	numeric

```
## offset      1 -none-    logical
## call        5 -none-    call
## nob          1 -none-    numeric
## lambdaOpt    1 -none-    numeric
## xNames       15 -none-   character
## problemType  1 -none-   character
## tuneValue    2 data.frame list
## obsLevels    1 -none-   logical
## param        0 -none-   list
```

```
print(enet.fit$bestTune)
```

```
##      alpha      lambda
## 101  0.05 0.002478752
```

```
# Plot elastic net results with different colors for each alpha
myCol <- rainbow(25)
myPar <- list(superpose.symbol = list(col = myCol),
              superpose.line = list(col = myCol))
plot(enet.fit, par.settings = myPar, xTrans = log)
```



```
# Show coefficients of the final Elastic Net model
coef_enet <- coef(enet.fit$finalModel, enet.fit$bestTune$lambda)
print(coef_enet)
```

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##              s1
## (Intercept)  16.0225822334
## age         -0.0202906145
## genderMale  -0.2942760076
## raceAsian   -0.0036856007
## raceBlack   -0.0066593257
## raceHispanic -0.0417893824
## smokingFormer 0.0235504017
## smokingCurrent -0.1891969582
## height      -0.0193618230
## weight       0.0193847860
## bmi         -0.1063133265
## diabetesYes  0.0111516911
## hypertensionYes -0.0167677112
## sbp          0.0012484795
## ldl         -0.0001495619
## time        -0.0002879691
```

```
coef_enet <- coef(enet.fit$finalModel, enet.fit$bestTune$lambda)
print(coef_enet)
```

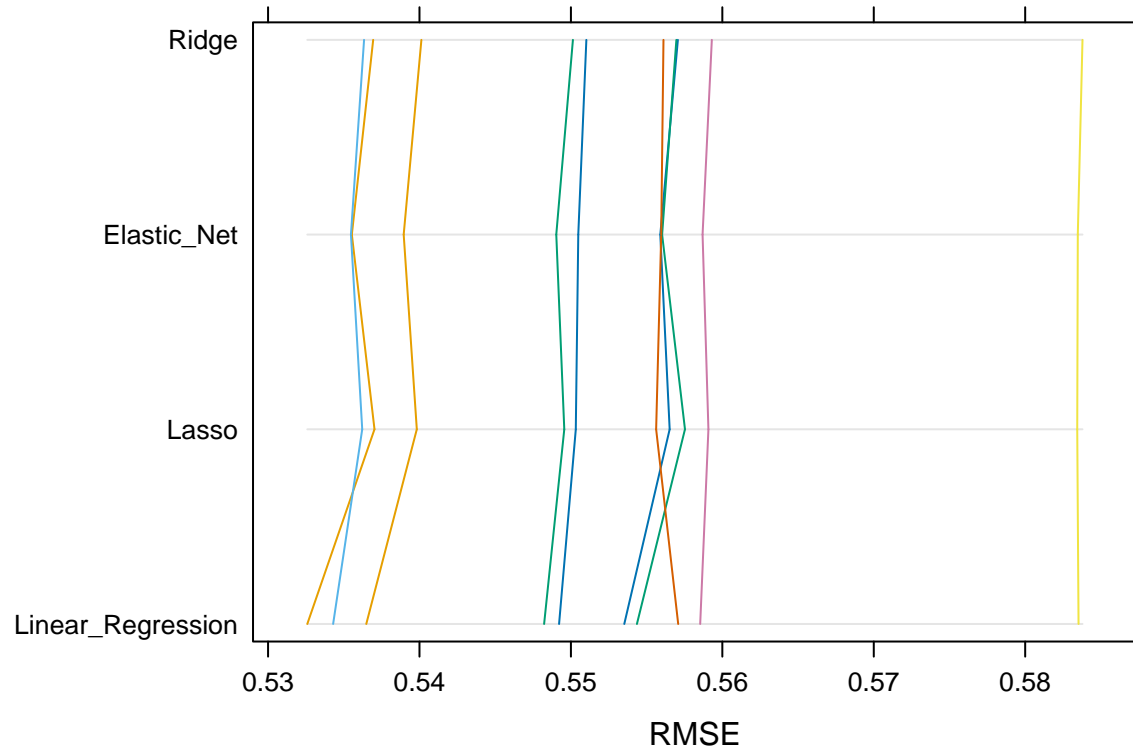
```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##              s1
## (Intercept)  16.0225822334
## age         -0.0202906145
## genderMale  -0.2942760076
## raceAsian   -0.0036856007
## raceBlack   -0.0066593257
## raceHispanic -0.0417893824
## smokingFormer 0.0235504017
## smokingCurrent -0.1891969582
## height      -0.0193618230
## weight       0.0193847860
## bmi         -0.1063133265
## diabetesYes  0.0111516911
## hypertensionYes -0.0167677112
## sbp          0.0012484795
## ldl         -0.0001495619
## time        -0.0002879691
```

```
resamp <- resamples(list(
  Linear_Regression = lm.fit,
  Ridge = ridge.fit,
  Lasso = lasso.fit,
  Elastic_Net = enet.fit
))
summary(resamp)
```

### 3.1.5 Model Comparison (resampling)

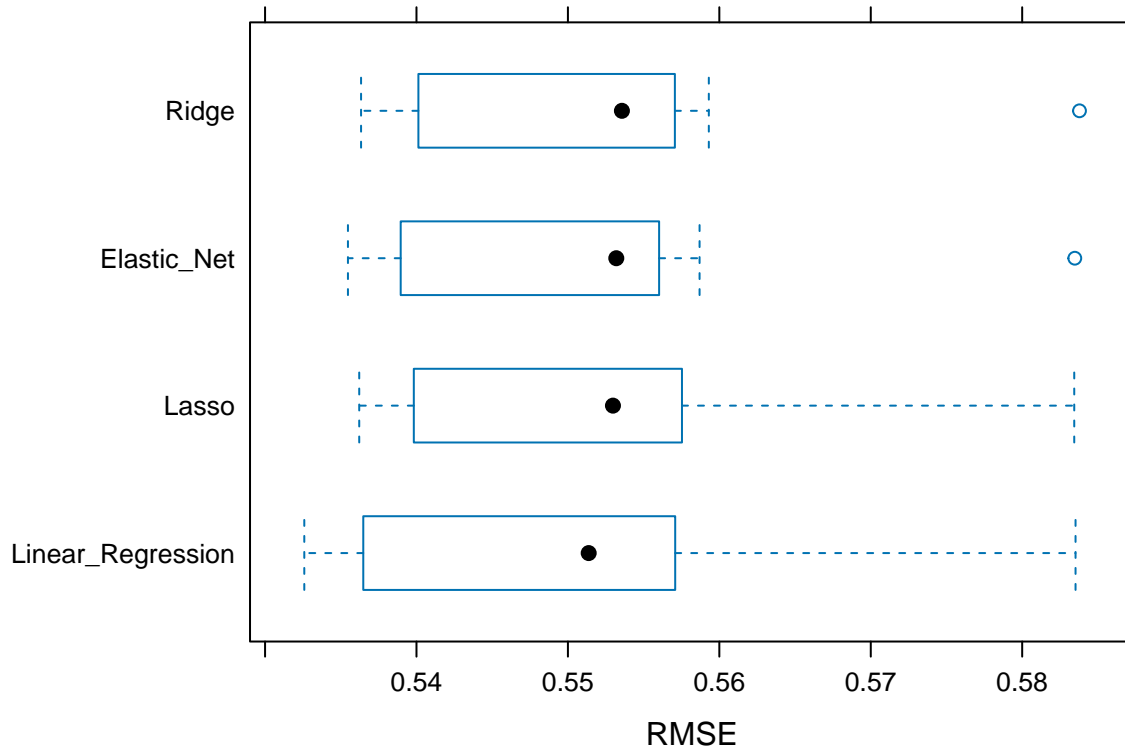
```
##
## Call:
## summary.resamples(object = resamp)
##
## Models: Linear_Regression, Ridge, Lasso, Elastic_Net
## Number of resamples: 10
##
## MAE
##           Min.   1st Qu.   Median     Mean   3rd Qu.     Max.
## Linear_Regression 0.4261808 0.4320149 0.4340128 0.4390390 0.4425761 0.4726409
## Ridge             0.4281731 0.4334230 0.4362590 0.4405051 0.4454071 0.4720434
## Lasso             0.4276920 0.4330647 0.4364914 0.4403860 0.4449243 0.4721327
## Elastic_Net       0.4274328 0.4330542 0.4354391 0.4399269 0.4448176 0.4720702
##           NA's
## Linear_Regression    0
## Ridge                0
## Lasso                0
## Elastic_Net          0
##
## RMSE
##           Min.   1st Qu.   Median     Mean   3rd Qu.     Max.
## Linear_Regression 0.5325907 0.5394231 0.5513718 0.5507854 0.5564032 0.5835238
## Ridge             0.5363406 0.5426274 0.5535668 0.5527796 0.5570390 0.5837848
## Lasso             0.5362188 0.5422580 0.5529754 0.5525153 0.5572793 0.5834368
## Elastic_Net       0.5354741 0.5414764 0.5531955 0.5519547 0.5560081 0.5834749
##           NA's
## Linear_Regression    0
## Ridge                0
## Lasso                0
## Elastic_Net          0
##
## Rsquared
##           Min.   1st Qu.   Median     Mean   3rd Qu.     Max.
## Linear_Regression 0.10033968 0.1319832 0.1534930 0.1479018 0.1622355 0.1906302
## Ridge             0.09354935 0.1329969 0.1421825 0.1418538 0.1502030 0.1878098
## Lasso             0.09475560 0.1342489 0.1433703 0.1428172 0.1503016 0.1912475
## Elastic_Net       0.09585259 0.1339416 0.1458955 0.1443698 0.1542137 0.1902642
##           NA's
## Linear_Regression    0
## Ridge                0
## Lasso                0
## Elastic_Net          0
```

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



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





The RMSE from resampling looks similar to each other.

```
# Function to calculate RMSE
rmse <- function(actual, predicted) {
  sqrt(mean((actual - predicted)^2))
}
```

```
# Make predictions using each model
lm.pred <- predict(lm.fit, newdata = dat2)
ridge.pred <- predict(ridge.fit, newdata = dat2)
lasso.pred <- predict(lasso.fit, newdata = dat2)
enet.pred <- predict(enet.fit, newdata = dat2)
```

```
# Calculate test MSE for each model
lm.test.error <- mean((lm.pred - dat2[, "log_antibody"])^2)
ridge.test.error <- mean((ridge.pred - dat2[, "log_antibody"])^2)
lasso.test.error <- mean((lasso.pred - dat2[, "log_antibody"])^2)
enet.test.error <- mean((enet.pred - dat2[, "log_antibody"])^2)
```

```
test_performance <- data.frame(
  Model = c("Linear Regression", "Ridge Regression", "Lasso Regression",
            "Elastic Net Regression"),
  Test_ERROR = c(lm.test.error, ridge.test.error, lasso.test.error,
                 enet.test.error)
)
print(test_performance)
```

### 3.1.6 Test Set Performance

```
##           Model Test_ERROR
## 1   Linear Regression 0.3229854
## 2   Ridge Regression 0.3258077
## 3   Lasso Regression 0.3288173
## 4 Elastic Net Regression 0.3247256
```

They are very similar to each other. We probably want to choose the Lasso model as it is the simplest model among those four models.

## 3.2 Non-linear Models

**3.2.1 Smoothing Spline** We use scatterplot to explore the relationship between the log antibody level and other variables. Time and bmi tend to show potentially nonlinear trends.

```
x = model.matrix(log_antibody ~ . - id, data = dat1)[, -1]
y = dat1[, "log_antibody"]
x_test = model.matrix(log_antibody ~ . - id, data = dat2)[, -1]
y_test = dat2[, "log_antibody"]
```

```
# choose the best df
fit.ss = smooth.spline(dat1$bmi, dat1$log_antibody)
fit.ss$df
```

```
## [1] 5.769535
```

```
print(fit.ss)
```

```
## Call:
## smooth.spline(x = dat1$bmi, y = dat1$log_antibody)
##
## Smoothing Parameter spar= 0.8460071 lambda= 0.05288648 (13 iterations)
## Equivalent Degrees of Freedom (Df): 5.769535
## Penalized Criterion (RSS): 61.11728
## GCV: 0.3330358
```

```
# plot optimal fit
bmi.grid = seq(from = 17.5, to = 40, by = 1)

pred.ss = predict(fit.ss, x = bmi.grid)
```

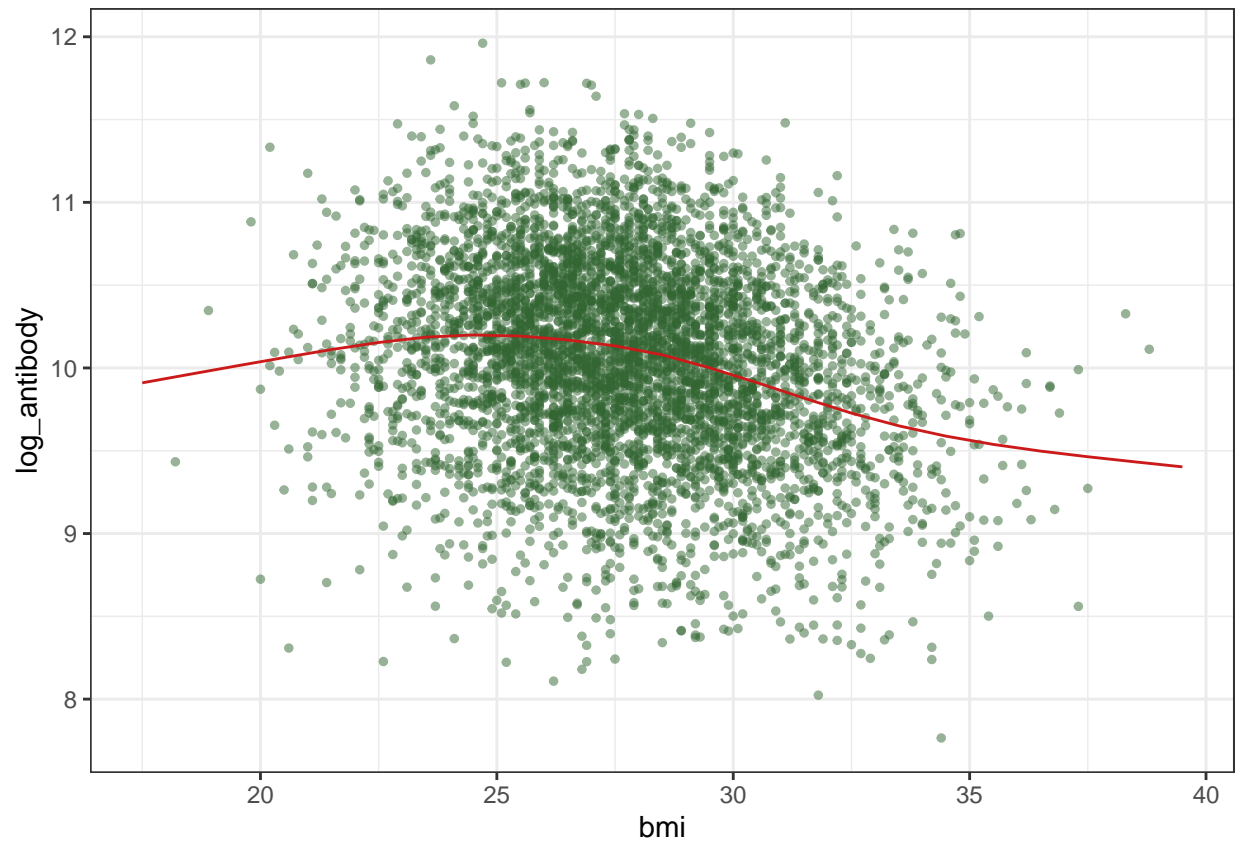
```

pred.ss.df = data.frame(pred = pred.ss$y, bmi = bmi.grid)

p = ggplot(dat1, aes(x = bmi, y = log_antibody)) +
  geom_point(color = rgb(.2, .4, .2, .5), size = 1) + theme_bw()

p +
  geom_line(aes(x = bmi, y = pred), data = pred.ss.df,
    color = rgb(.8, .1, .1, 1)) + theme_bw()

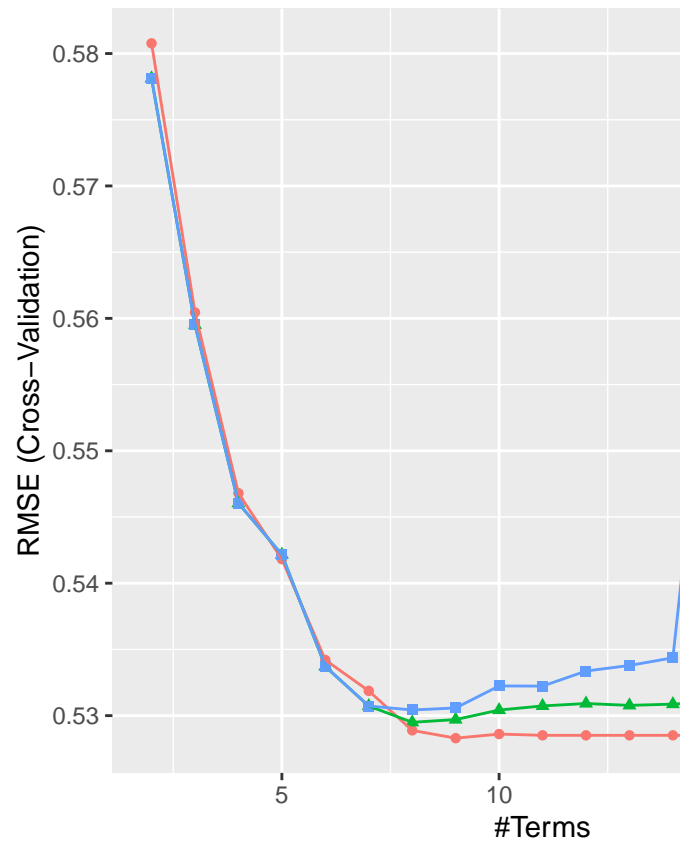
```



```

set.seed(2)
ctrl1 = trainControl(method = "cv", number = 10)
mars_grid = expand.grid(degree = 1:3, nprune = 2:20)
mars.fit = train(x, y, method = "earth", tuneGrid = mars_grid,
  trControl = ctrl1)
ggplot(mars.fit)

```



### 3.2.2 Multivariate Adaptive Regression Splines (MARS)

```
mars.fit$bestTune
```

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

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

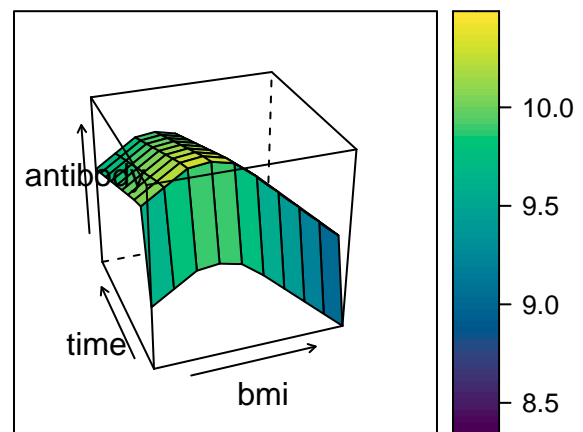
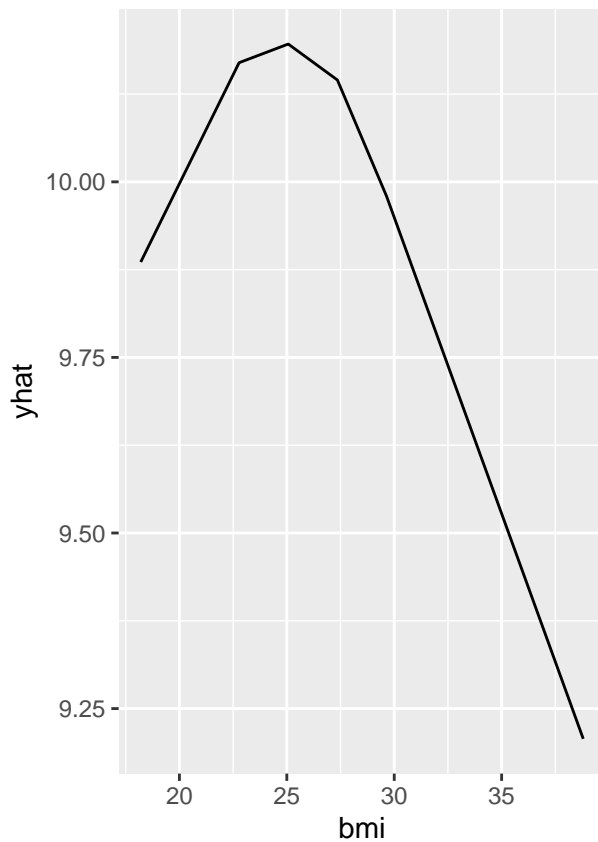
```
##      (Intercept)      h(27.8-bmi)      h(time-57)      h(57-time)      genderMale
## 10.847446930    -0.061997354    -0.002254182    -0.033529326    -0.296290451
##      h(age-59)      h(59-age) smokingCurrent      h(bmi-23.7)
## -0.022957648      0.016138468    -0.205126851    -0.084380175
```

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

```
## Call: earth(x=matrix[5000,15], y=c(10.65,9.889,1...), keepxy=TRUE, degree=1,
##          nprune=9)
##
##          coefficients
## (Intercept)      10.8474469
## genderMale      -0.2962905
## smokingCurrent  -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, genderMale, time, age, smokingCurrent, raceAsian-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

# we choose the relatively important variables to draw partial dependence plot
pdp1 = pdp::partial(mars.fit, pred.var = c("bmi"), grid.resolution = 10) |> autoplot()
pdp2 <- pdp::partial(mars.fit, pred.var = c("bmi", "time"), grid.resolution = 10) |>
pdp::plotPartial(levelplot = FALSE, zlab = "log antibody", drape = TRUE,
screen = list(z = 20, x = -60))
gridExtra::grid.arrange(pdp1, pdp2, ncol = 2)
```



```
set.seed(2)
gam.fit = train(x, y, method = "gam", trControl = ctrl11)

summary(gam.fit)
```

### 3.2.3 Generalized Additive Model (GAM)

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## .outcome ~ genderMale + raceAsian + raceBlack + raceHispanic +
##   smokingFormer + smokingCurrent + diabetesYes + hypertensionYes +
##   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 ***
## genderMale   -0.297837   0.014933 -19.945 < 2e-16 ***
## raceAsian    -0.003296   0.033009  -0.100  0.920
## raceBlack    -0.010509   0.018837  -0.558  0.577
## raceHispanic -0.037424   0.026176  -1.430  0.153
## smokingFormer  0.022219   0.016660   1.334  0.182
## smokingCurrent -0.193175   0.025834  -7.478 8.9e-14 ***
## diabetesYes   0.014230   0.020640   0.689  0.491
## hypertensionYes -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
```

```
gam.fit$bestTune
```

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

```
gam.fit$finalModel
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## .outcome ~ genderMale + raceAsian + raceBlack + raceHispanic +
##   smokingFormer + smokingCurrent + diabetesYes + hypertensionYes +
```

```
##      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
```

```
comparison = resamples(list(MARS = mars.fit, GAM = gam.fit))
summary(comparison)
```

### 3.2.4 Model Comparison (resampling)

```
##
## Call:
## summary.resamples(object = comparison)
##
## 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
```

```
# Make predictions using each mode
pred_ss = predict(fit.ss, x = dat2$bmi)
pred_mars = predict(mars.fit, newdata = x_test)
pred_gam = predict(gam.fit, newdata = x_test)

mse_ss = mean((dat2$log_antibody - pred_ss)^2)
mse_mars = mean((y_test - pred_mars)^2)
mse_gam = mean((y_test - pred_gam)^2)

test_performance_non_linear <- data.frame(
  Model = c("Smoothing Spline", "MARS", "GAM"),
  Test_ERROR = c(mse_ss, mse_mars, mse_gam)
```

```
)
print(test_performance_non_linear)
```

### 3.1.6 Test Set Performance

```
##           Model Test_ERROR
## 1 Smoothing Spline 0.3584620
## 2           MARS  0.2838458
## 3           GAM  0.3249953
```

## 4. Model Comparison between Linear and non-linear Models

```
# MSE comparison

test_mse_table = data.frame(
  Model = c("Linear Regression", "Ridge Regression", "Lasso Regression",
            "Elastic Net Regression", "Smoothing Spline", "MARS", "GAM"),
  MSE = c(lm.test.error, ridge.test.error,
          lasso.test.error, mse_ss, enet.test.error,
          mse_mars, mse_gam)
) |> arrange(MSE)

kable(test_mse_table, sort = TRUE)
```

Model	MSE
MARS	0.2838458
Linear Regression	0.3229854
Smoothing Spline	0.3247256
GAM	0.3249953
Ridge Regression	0.3258077
Lasso Regression	0.3288173
Elastic Net Regression	0.3584620

The best model is the MARS model with the smallest test error.