datascience_2_midterm

2025-03-27

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v readr     2.1.5
## v ggplot2   3.5.1     v stringr   1.5.1
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.4     v tidyr     1.3.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
library(survival)
library(ggsurvfit)
library(gtsummary)
library(glue)
library(labelled)
library(rpart.plot)
```

```
## Loading required package: rpart
```

```r
library(rpart)
library(party)
```

```
## Loading required package: grid
## Loading required package: mvtnorm
## Loading required package: modeltools
```

```
## Loading required package: stats4
## Loading required package: strucchange
## Loading required package: zoo
##
## Attaching package: 'zoo'
##
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
##
## Loading required package: sandwich
##
## Attaching package: 'strucchange'
##
## The following object is masked from 'package:stringr':
##
##     boundary
##
##
## Attaching package: 'party'
##
## The following object is masked from 'package:gtsummary':
##
##     where
##
## The following object is masked from 'package:dplyr':
##
##     where
```

```
library(partykit)
```

```
## Loading required package: libcoin
##
## Attaching package: 'partykit'
##
## The following objects are masked from 'package:party':
##
##     cforest, ctree, ctree_control, edge_simple, mob, mob_control,
##     node_barplot, node_bivplot, node_boxplot, node_inner, node_surv,
##     node_terminal, varimp
```

```
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:survival':
##
##     cluster
##
## The following object is masked from 'package:purrr':
##
##     lift
```

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

```
#mutating factor variables and creating labels for baseline characteristic table
table_data = dat1 |>
  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 smoked", "Former smoker", "Curre
         diabetes = factor(diabetes, levels = c(0,1), labels = c("No","Yes")),
         hypertension = factor(hypertension, levels = c(0,1), labels = c("No", "Yes")))

#creating table labels
var_label(table_data) = list(
  age = "Age (in years)",
  gender = "Gender",
  race = "Race/ethnicity",
  smoking = "Smoking status",
  height = "Height (in centimeters)",
  weight = "Weight (in kilograms)",
  bmi = "BMI (body mass index)",
  diabetes = "Diabetes",
  hypertension = "Hypertension",
  SBP = "Systolic Blood Pressure (mmHg)",
  LDL = "LDL Cholesterol (mg/dL)",
  time = "Time since vaccination (in days)"
)
```

```
#outputting table
table = table_data |>
  select(age, gender, race, smoking, height, weight, bmi, diabetes, hypertension, SBP, LDL, time) |>
  tbl_summary(
    missing_text = "(Missing)",
    statistic = list(all_continuous() ~ "{median} ± {sd}")
  )

print(table)
```

```
## <div id="crdplkhmfk" style="padding-left:0px;padding-right:0px;padding-top:10px;padding-bottom:10px;
##   <style>#crdplkhmfk table {
##   font-family: system-ui, 'Segoe UI', Roboto, Helvetica, Arial, sans-serif, 'Apple Color Emoji', 'Se
##   -webkit-font-smoothing: antialiased;
##   -moz-osx-font-smoothing: grayscale;
## }
##
## #crdplkhmfk thead, #crdplkhmfk tbody, #crdplkhmfk tfoot, #crdplkhmfk tr, #crdplkhmfk td, #crdplkhmfk
##   border-style: none;
## }
##
## #crdplkhmfk p {
##   margin: 0;
##   padding: 0;
## }
##
```

```
## #crdplkhmfk .gt_table {
##   display: table;
##   border-collapse: collapse;
##   line-height: normal;
##   margin-left: auto;
##   margin-right: auto;
##   color: #333333;
##   font-size: 16px;
##   font-weight: normal;
##   font-style: normal;
##   background-color: #FFFFFF;
##   width: auto;
##   border-top-style: solid;
##   border-top-width: 2px;
##   border-top-color: #A8A8A8;
##   border-right-style: none;
##   border-right-width: 2px;
##   border-right-color: #D3D3D3;
##   border-bottom-style: solid;
##   border-bottom-width: 2px;
##   border-bottom-color: #A8A8A8;
##   border-left-style: none;
##   border-left-width: 2px;
##   border-left-color: #D3D3D3;
## }
##
## #crdplkhmfk .gt_caption {
##   padding-top: 4px;
##   padding-bottom: 4px;
## }
##
## #crdplkhmfk .gt_title {
##   color: #333333;
##   font-size: 125%;
##   font-weight: initial;
##   padding-top: 4px;
##   padding-bottom: 4px;
##   padding-left: 5px;
##   padding-right: 5px;
##   border-bottom-color: #FFFFFF;
##   border-bottom-width: 0;
## }
##
## #crdplkhmfk .gt_subtitle {
##   color: #333333;
##   font-size: 85%;
##   font-weight: initial;
##   padding-top: 3px;
##   padding-bottom: 5px;
##   padding-left: 5px;
##   padding-right: 5px;
##   border-top-color: #FFFFFF;
##   border-top-width: 0;
## }
```

```
##
## #crdplkhmfk .gt_heading {
##   background-color: #FFFFFF;
##   text-align: center;
##   border-bottom-color: #FFFFFF;
##   border-left-style: none;
##   border-left-width: 1px;
##   border-left-color: #D3D3D3;
##   border-right-style: none;
##   border-right-width: 1px;
##   border-right-color: #D3D3D3;
## }
##
## #crdplkhmfk .gt_bottom_border {
##   border-bottom-style: solid;
##   border-bottom-width: 2px;
##   border-bottom-color: #D3D3D3;
## }
##
## #crdplkhmfk .gt_col_headings {
##   border-top-style: solid;
##   border-top-width: 2px;
##   border-top-color: #D3D3D3;
##   border-bottom-style: solid;
##   border-bottom-width: 2px;
##   border-bottom-color: #D3D3D3;
##   border-left-style: none;
##   border-left-width: 1px;
##   border-left-color: #D3D3D3;
##   border-right-style: none;
##   border-right-width: 1px;
##   border-right-color: #D3D3D3;
## }
##
## #crdplkhmfk .gt_col_heading {
##   color: #333333;
##   background-color: #FFFFFF;
##   font-size: 100%;
##   font-weight: normal;
##   text-transform: inherit;
##   border-left-style: none;
##   border-left-width: 1px;
##   border-left-color: #D3D3D3;
##   border-right-style: none;
##   border-right-width: 1px;
##   border-right-color: #D3D3D3;
##   vertical-align: bottom;
##   padding-top: 5px;
##   padding-bottom: 6px;
##   padding-left: 5px;
##   padding-right: 5px;
##   overflow-x: hidden;
## }
##
```

```
## #crdplkhmfk .gt_column_spanner_outer {
##   color: #333333;
##   background-color: #FFFFFF;
##   font-size: 100%;
##   font-weight: normal;
##   text-transform: inherit;
##   padding-top: 0;
##   padding-bottom: 0;
##   padding-left: 4px;
##   padding-right: 4px;
## }
##
## #crdplkhmfk .gt_column_spanner_outer:first-child {
##   padding-left: 0;
## }
##
## #crdplkhmfk .gt_column_spanner_outer:last-child {
##   padding-right: 0;
## }
##
## #crdplkhmfk .gt_column_spanner {
##   border-bottom-style: solid;
##   border-bottom-width: 2px;
##   border-bottom-color: #D3D3D3;
##   vertical-align: bottom;
##   padding-top: 5px;
##   padding-bottom: 5px;
##   overflow-x: hidden;
##   display: inline-block;
##   width: 100%;
## }
##
## #crdplkhmfk .gt_spanner_row {
##   border-bottom-style: hidden;
## }
##
## #crdplkhmfk .gt_group_heading {
##   padding-top: 8px;
##   padding-bottom: 8px;
##   padding-left: 5px;
##   padding-right: 5px;
##   color: #333333;
##   background-color: #FFFFFF;
##   font-size: 100%;
##   font-weight: initial;
##   text-transform: inherit;
##   border-top-style: solid;
##   border-top-width: 2px;
##   border-top-color: #D3D3D3;
##   border-bottom-style: solid;
##   border-bottom-width: 2px;
##   border-bottom-color: #D3D3D3;
##   border-left-style: none;
##   border-left-width: 1px;
```

```
##   border-left-color: #D3D3D3;
##   border-right-style: none;
##   border-right-width: 1px;
##   border-right-color: #D3D3D3;
##   vertical-align: middle;
##   text-align: left;
## }
##
## #crdplkhmfk .gt_empty_group_heading {
##   padding: 0.5px;
##   color: #333333;
##   background-color: #FFFFFF;
##   font-size: 100%;
##   font-weight: initial;
##   border-top-style: solid;
##   border-top-width: 2px;
##   border-top-color: #D3D3D3;
##   border-bottom-style: solid;
##   border-bottom-width: 2px;
##   border-bottom-color: #D3D3D3;
##   vertical-align: middle;
## }
##
## #crdplkhmfk .gt_from_md > :first-child {
##   margin-top: 0;
## }
##
## #crdplkhmfk .gt_from_md > :last-child {
##   margin-bottom: 0;
## }
##
## #crdplkhmfk .gt_row {
##   padding-top: 8px;
##   padding-bottom: 8px;
##   padding-left: 5px;
##   padding-right: 5px;
##   margin: 10px;
##   border-top-style: solid;
##   border-top-width: 1px;
##   border-top-color: #D3D3D3;
##   border-left-style: none;
##   border-left-width: 1px;
##   border-left-color: #D3D3D3;
##   border-right-style: none;
##   border-right-width: 1px;
##   border-right-color: #D3D3D3;
##   vertical-align: middle;
##   overflow-x: hidden;
## }
##
## #crdplkhmfk .gt_stub {
##   color: #333333;
##   background-color: #FFFFFF;
##   font-size: 100%;
```

```
##   font-weight: initial;
##   text-transform: inherit;
##   border-right-style: solid;
##   border-right-width: 2px;
##   border-right-color: #D3D3D3;
##   padding-left: 5px;
##   padding-right: 5px;
## }
##
## #crdplkhmfk .gt_stub_row_group {
##   color: #333333;
##   background-color: #FFFFFF;
##   font-size: 100%;
##   font-weight: initial;
##   text-transform: inherit;
##   border-right-style: solid;
##   border-right-width: 2px;
##   border-right-color: #D3D3D3;
##   padding-left: 5px;
##   padding-right: 5px;
##   vertical-align: top;
## }
##
## #crdplkhmfk .gt_row_group_first td {
##   border-top-width: 2px;
## }
##
## #crdplkhmfk .gt_row_group_first th {
##   border-top-width: 2px;
## }
##
## #crdplkhmfk .gt_summary_row {
##   color: #333333;
##   background-color: #FFFFFF;
##   text-transform: inherit;
##   padding-top: 8px;
##   padding-bottom: 8px;
##   padding-left: 5px;
##   padding-right: 5px;
## }
##
## #crdplkhmfk .gt_first_summary_row {
##   border-top-style: solid;
##   border-top-color: #D3D3D3;
## }
##
## #crdplkhmfk .gt_first_summary_row.thick {
##   border-top-width: 2px;
## }
##
## #crdplkhmfk .gt_last_summary_row {
##   padding-top: 8px;
##   padding-bottom: 8px;
##   padding-left: 5px;
```

```
##   padding-right: 5px;
##   border-bottom-style: solid;
##   border-bottom-width: 2px;
##   border-bottom-color: #D3D3D3;
## }
##
## #crdplkhmfk .gt_grand_summary_row {
##   color: #333333;
##   background-color: #FFFFFF;
##   text-transform: inherit;
##   padding-top: 8px;
##   padding-bottom: 8px;
##   padding-left: 5px;
##   padding-right: 5px;
## }
##
## #crdplkhmfk .gt_first_grand_summary_row {
##   padding-top: 8px;
##   padding-bottom: 8px;
##   padding-left: 5px;
##   padding-right: 5px;
##   border-top-style: double;
##   border-top-width: 6px;
##   border-top-color: #D3D3D3;
## }
##
## #crdplkhmfk .gt_last_grand_summary_row_top {
##   padding-top: 8px;
##   padding-bottom: 8px;
##   padding-left: 5px;
##   padding-right: 5px;
##   border-bottom-style: double;
##   border-bottom-width: 6px;
##   border-bottom-color: #D3D3D3;
## }
##
## #crdplkhmfk .gt_striped {
##   background-color: rgba(128, 128, 128, 0.05);
## }
##
## #crdplkhmfk .gt_table_body {
##   border-top-style: solid;
##   border-top-width: 2px;
##   border-top-color: #D3D3D3;
##   border-bottom-style: solid;
##   border-bottom-width: 2px;
##   border-bottom-color: #D3D3D3;
## }
##
## #crdplkhmfk .gt_footnotes {
##   color: #333333;
##   background-color: #FFFFFF;
##   border-bottom-style: none;
##   border-bottom-width: 2px;
```

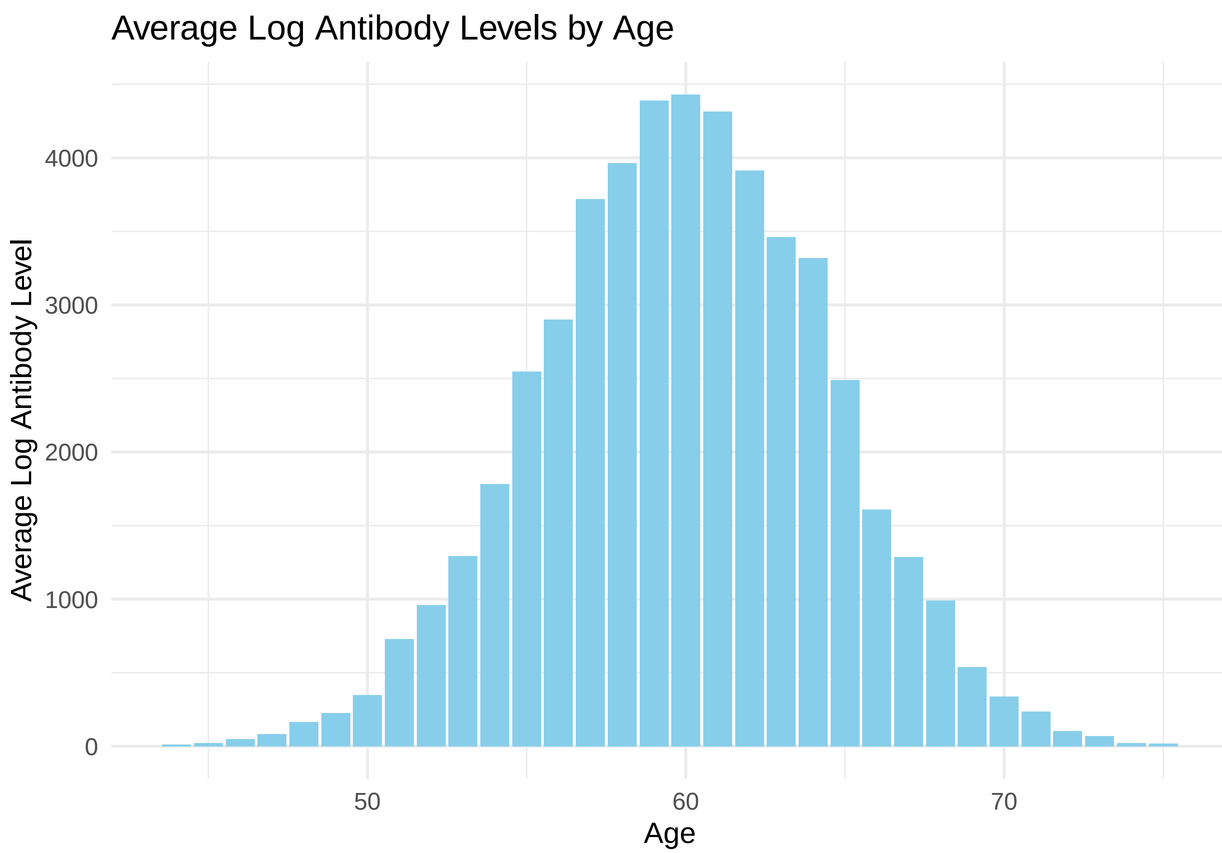
```
##   border-bottom-color: #D3D3D3;
##   border-left-style: none;
##   border-left-width: 2px;
##   border-left-color: #D3D3D3;
##   border-right-style: none;
##   border-right-width: 2px;
##   border-right-color: #D3D3D3;
## }
##
## #crdplkhmfk .gt_footnote {
##   margin: 0px;
##   font-size: 90%;
##   padding-top: 4px;
##   padding-bottom: 4px;
##   padding-left: 5px;
##   padding-right: 5px;
## }
##
## #crdplkhmfk .gt_sourcenotes {
##   color: #333333;
##   background-color: #FFFFFF;
##   border-bottom-style: none;
##   border-bottom-width: 2px;
##   border-bottom-color: #D3D3D3;
##   border-left-style: none;
##   border-left-width: 2px;
##   border-left-color: #D3D3D3;
##   border-right-style: none;
##   border-right-width: 2px;
##   border-right-color: #D3D3D3;
## }
##
## #crdplkhmfk .gt_sourcenote {
##   font-size: 90%;
##   padding-top: 4px;
##   padding-bottom: 4px;
##   padding-left: 5px;
##   padding-right: 5px;
## }
##
## #crdplkhmfk .gt_left {
##   text-align: left;
## }
##
## #crdplkhmfk .gt_center {
##   text-align: center;
## }
##
## #crdplkhmfk .gt_right {
##   text-align: right;
##   font-variant-numeric: tabular-nums;
## }
##
## #crdplkhmfk .gt_font_normal {
```

```
##   font-weight: normal;
## }
##
## #crdplkhmfk .gt_font_bold {
##   font-weight: bold;
## }
##
## #crdplkhmfk .gt_font_italic {
##   font-style: italic;
## }
##
## #crdplkhmfk .gt_super {
##   font-size: 65%;
## }
##
## #crdplkhmfk .gt_footnote_marks {
##   font-size: 75%;
##   vertical-align: 0.4em;
##   position: initial;
## }
##
## #crdplkhmfk .gt_asterisk {
##   font-size: 100%;
##   vertical-align: 0;
## }
##
## #crdplkhmfk .gt_indent_1 {
##   text-indent: 5px;
## }
##
## #crdplkhmfk .gt_indent_2 {
##   text-indent: 10px;
## }
##
## #crdplkhmfk .gt_indent_3 {
##   text-indent: 15px;
## }
##
## #crdplkhmfk .gt_indent_4 {
##   text-indent: 20px;
## }
##
## #crdplkhmfk .gt_indent_5 {
##   text-indent: 25px;
## }
##
## #crdplkhmfk .katex-display {
##   display: inline-flex !important;
##   margin-bottom: 0.75em !important;
## }
##
## #crdplkhmfk div.Reactable > div.rt-table > div.rt-thead > div.rt-tr.rt-tr-group-header > div.rt-th-g
##   height: 0px !important;
## }
```

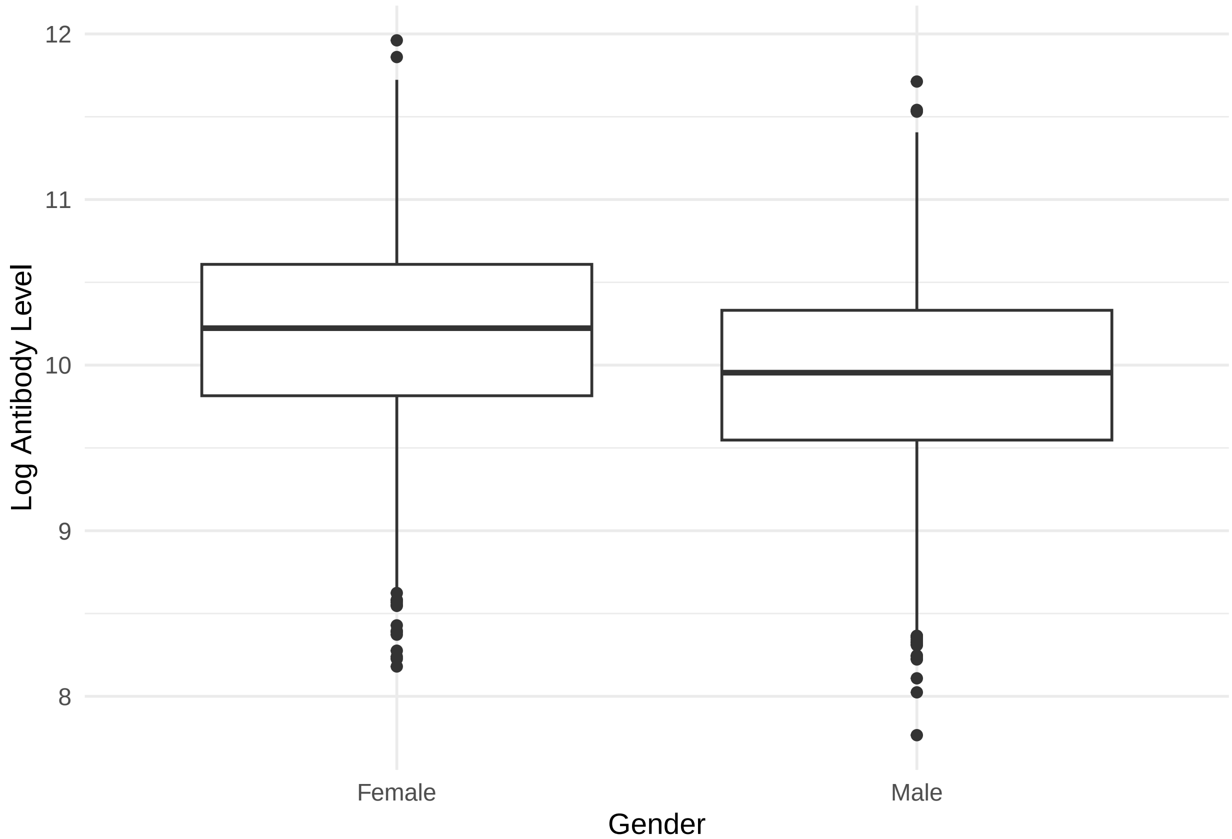
```
## </style>
##   <table class="gt_table" data-quarto-disable-processing="false" data-quarto-bootstrap="false">
##   <thead>
##     <tr class="gt_col_headings">
##       <th class="gt_col_heading gt_columns_bottom_border gt_left" rowspan="1" colspan="1" scope="col
##       <th class="gt_col_heading gt_columns_bottom_border gt_center" rowspan="1" colspan="1" scope="co
##     </tr>
##   </thead>
##   <tbody class="gt_table_body">
##     <tr><td headers="label" class="gt_row gt_left">Age (in years)</td>
## <td headers="stat_0" class="gt_row gt_center">60.0 ± 4.5</td></tr>
##     <tr><td headers="label" class="gt_row gt_left">Gender</td>
## <td headers="stat_0" class="gt_row gt_center"><br /></td></tr>
##     <tr><td headers="label" class="gt_row gt_left">    Female</td>
## <td headers="stat_0" class="gt_row gt_center">2,573 (51%)</td></tr>
##     <tr><td headers="label" class="gt_row gt_left">    Male</td>
## <td headers="stat_0" class="gt_row gt_center">2,427 (49%)</td></tr>
##     <tr><td headers="label" class="gt_row gt_left">Race/ethnicity</td>
## <td headers="stat_0" class="gt_row gt_center"><br /></td></tr>
##     <tr><td headers="label" class="gt_row gt_left">    White</td>
## <td headers="stat_0" class="gt_row gt_center">3,221 (64%)</td></tr>
##     <tr><td headers="label" class="gt_row gt_left">    Asian</td>
## <td headers="stat_0" class="gt_row gt_center">278 (5.6%)</td></tr>
##     <tr><td headers="label" class="gt_row gt_left">    Black</td>
## <td headers="stat_0" class="gt_row gt_center">1,036 (21%)</td></tr>
##     <tr><td headers="label" class="gt_row gt_left">    Hispanic</td>
## <td headers="stat_0" class="gt_row gt_center">465 (9.3%)</td></tr>
##     <tr><td headers="label" class="gt_row gt_left">Smoking status</td>
## <td headers="stat_0" class="gt_row gt_center"><br /></td></tr>
##     <tr><td headers="label" class="gt_row gt_left">    Never smoked</td>
## <td headers="stat_0" class="gt_row gt_center">3,010 (60%)</td></tr>
##     <tr><td headers="label" class="gt_row gt_left">    Former smoker</td>
## <td headers="stat_0" class="gt_row gt_center">1,504 (30%)</td></tr>
##     <tr><td headers="label" class="gt_row gt_left">    Current smoker</td>
## <td headers="stat_0" class="gt_row gt_center">486 (9.7%)</td></tr>
##     <tr><td headers="label" class="gt_row gt_left">Height (in centimeters)</td>
## <td headers="stat_0" class="gt_row gt_center">170.1 ± 5.9</td></tr>
##     <tr><td headers="label" class="gt_row gt_left">Weight (in kilograms)</td>
## <td headers="stat_0" class="gt_row gt_center">80 ± 7</td></tr>
##     <tr><td headers="label" class="gt_row gt_left">BMI (body mass index)</td>
## <td headers="stat_0" class="gt_row gt_center">27.60 ± 2.76</td></tr>
##     <tr><td headers="label" class="gt_row gt_left">Diabetes</td>
## <td headers="stat_0" class="gt_row gt_center">772 (15%)</td></tr>
##     <tr><td headers="label" class="gt_row gt_left">Hypertension</td>
## <td headers="stat_0" class="gt_row gt_center">2,298 (46%)</td></tr>
##     <tr><td headers="label" class="gt_row gt_left">Systolic Blood Pressure (mmHg)</td>
## <td headers="stat_0" class="gt_row gt_center">130 ± 8</td></tr>
##     <tr><td headers="label" class="gt_row gt_left">LDL Cholesterol (mg/dL)</td>
## <td headers="stat_0" class="gt_row gt_center">110 ± 20</td></tr>
##     <tr><td headers="label" class="gt_row gt_left">Time since vaccination (in days)</td>
## <td headers="stat_0" class="gt_row gt_center">106 ± 43</td></tr>
##   </tbody>
##
##   <tfoot class="gt_footnotes">
```

```
##     <tr>
##       <td class="gt_footnote" colspan="2"><span class="gt_footnote_marks" style="white-space:nowrap;
##     </tr>
##   </tfoot>
## </table>
## </div>
```

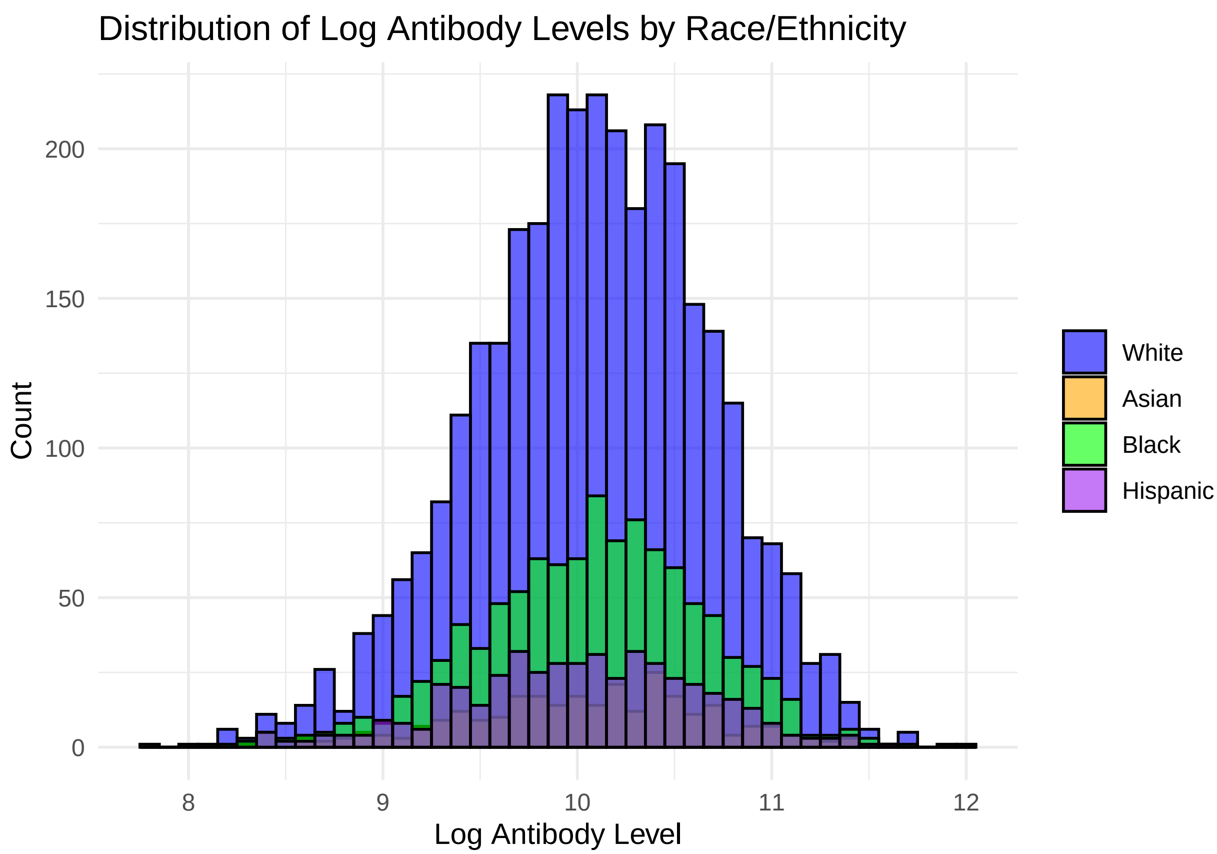
```r
##Distribution of Log Antibody Levels by Race/Ethnicity
age_antibody <- ggplot(table_data, aes(x = age, y = log_antibody)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  labs(x = "Age", y = "Average Log Antibody Level",
       title = "Average Log Antibody Levels by Age") +
  theme_minimal()

age_antibody
```



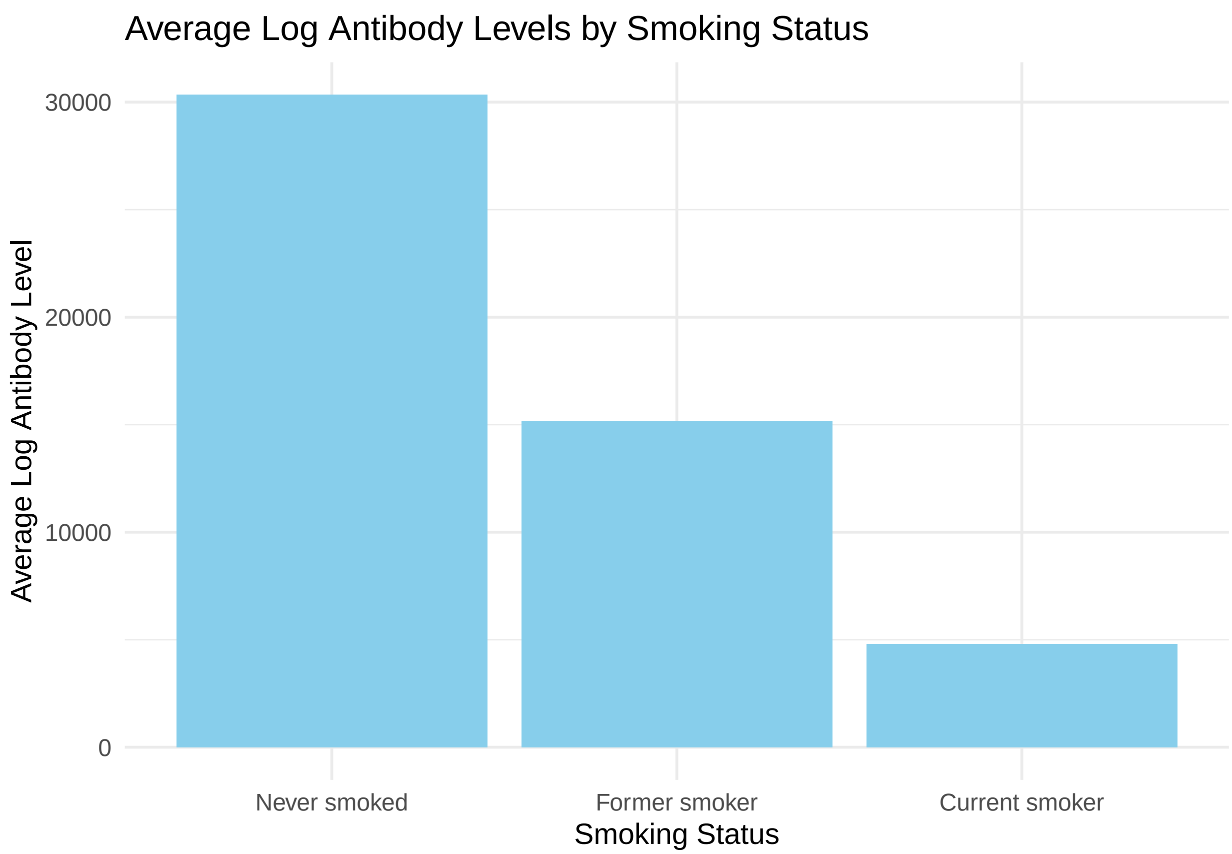

```r
#Average Log Antibody Levels by Gender"
gender_antibody <- ggplot(table_data, aes(x = factor(gender), y = log_antibody)) +
  geom_boxplot() +
  labs(x = "Gender", y = "Log Antibody Level") +
  theme_minimal()

gender_antibody
```




```r
#Distribution of Log Antibody Levels by Race/Ethnicity
race_antibody <- ggplot(table_data, aes(x = log_antibody, fill = factor(race))) +
  geom_histogram(binwidth = 0.1, position = "identity", alpha = 0.6, color = "black") +
  labs(x = "Log Antibody Level", y = "Count", title = "Distribution of Log Antibody Levels by Race/Ethni
  scale_fill_manual(values = c("blue", "orange", "green", "purple"),
                    labels = c("1" = "White", "2" = "Asian", "3" = "Black", "4" = "Hispanic")) +
  theme_minimal() +
  theme(legend.title = element_blank())

race_antibody
```




```r
#Average Log Antibody Levels by Smoking Status
smoking_antibody <- ggplot(table_data, aes(x = factor(smoking), y = log_antibody)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  labs(x = "Smoking Status", y = "Average Log Antibody Level",
       title = "Average Log Antibody Levels by Smoking Status") +
  scale_x_discrete(labels = c("0" = "Never Smoked",
                              "1" = "Former Smoker",
                              "2" = "Current Smoker")) +
  theme_minimal()

smoking_antibody
```

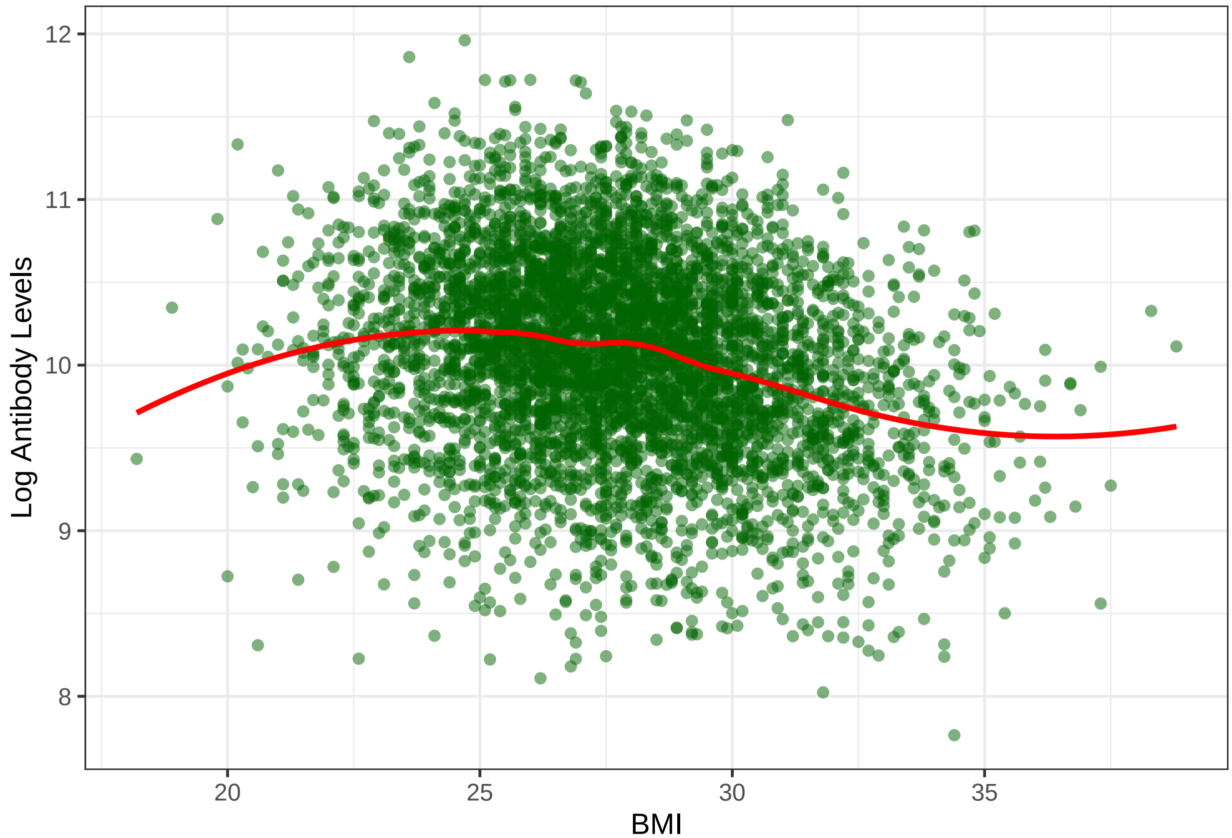



```
#Distribution of Log Antibody Levels by BMI
bmi_antibody <- ggplot(table_data, aes_string(x = "bmi", y = "log_antibody")) +
  geom_point(color = "darkgreen", alpha = 0.5) +
  geom_smooth(method = "loess", span = 0.5, color = "red", se = FALSE) +
  theme_bw() +
  labs(x = "BMI", y = "Log Antibody Levels")
```

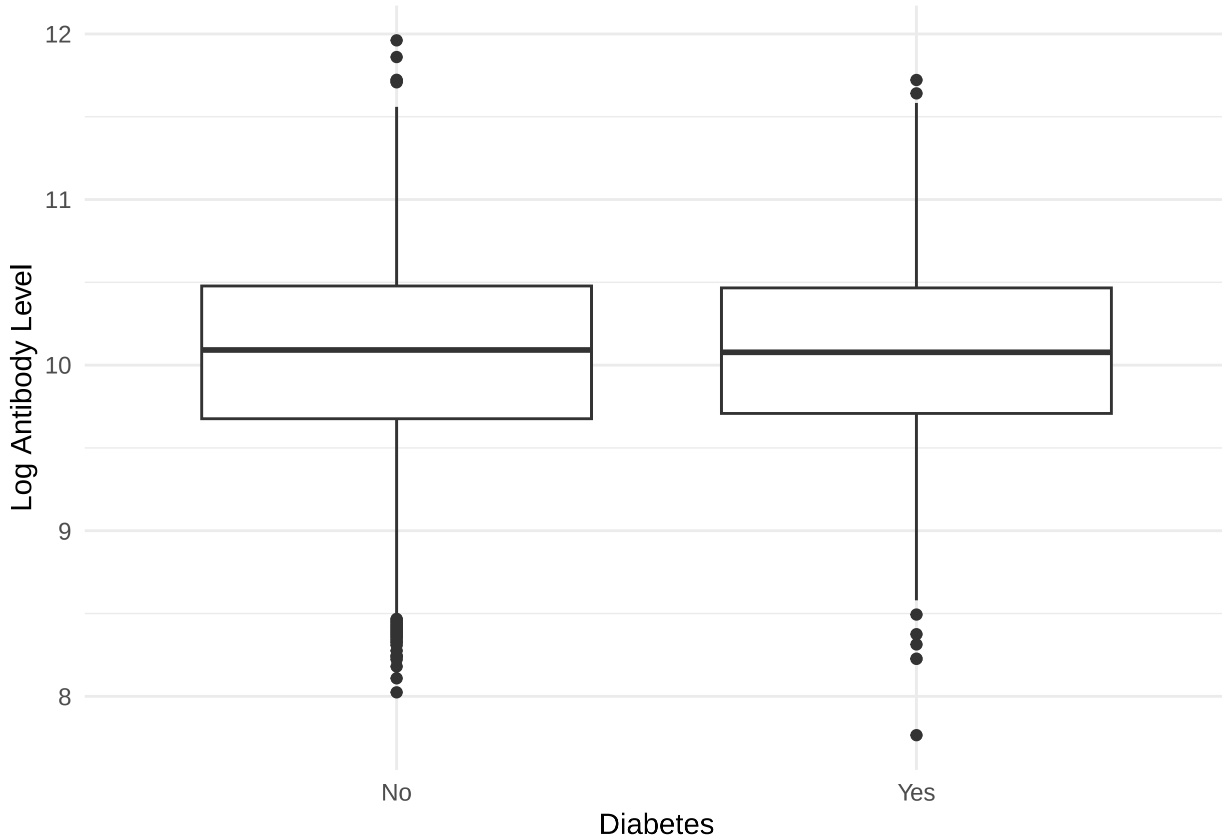
```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
bmi_antibody
```

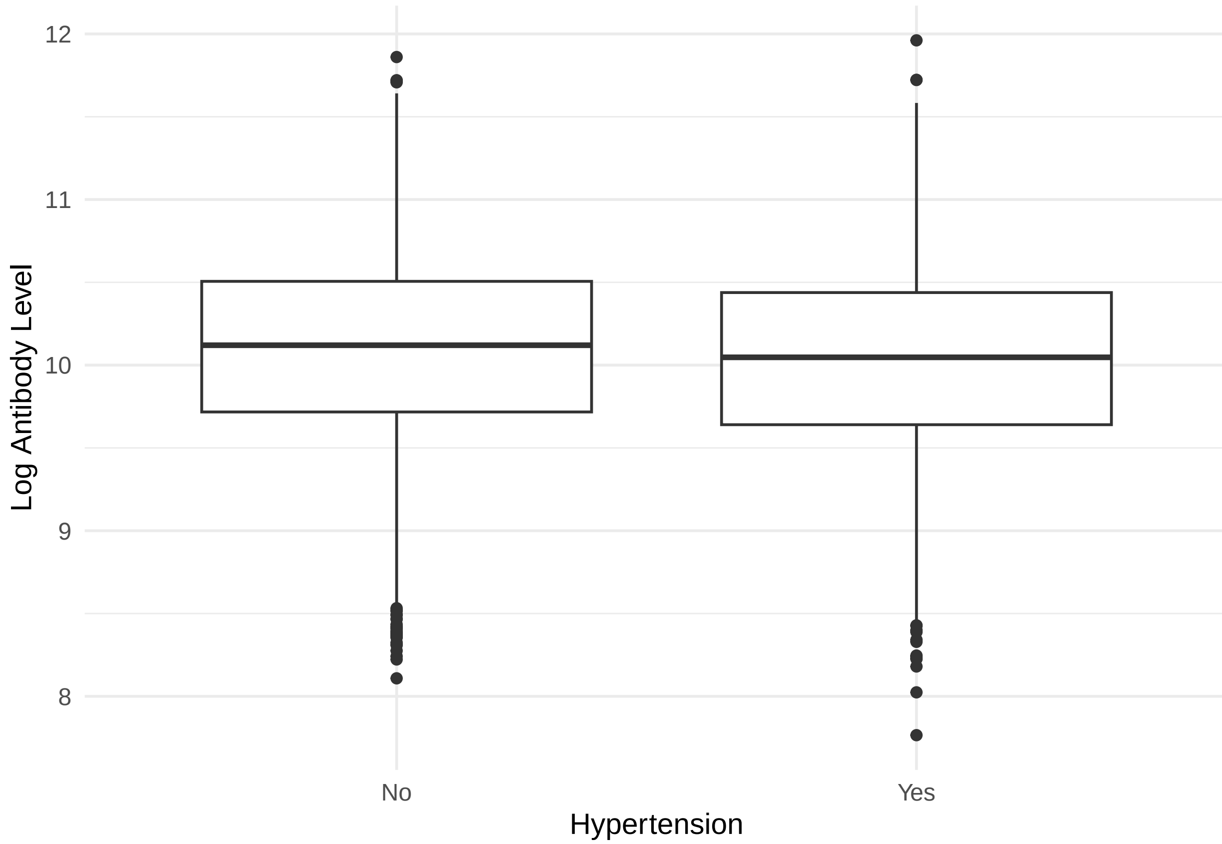
```
## `geom_smooth()` using formula = 'y ~ x'
```




```r
#Average Log Antibody levels by diabetes status
diabetes_antibody <- ggplot(table_data, aes(x = factor(diabetes), y = log_antibody)) +
  geom_boxplot() +
  labs(x = "Diabetes", y = "Log Antibody Level") +
  theme_minimal()

diabetes_antibody
```

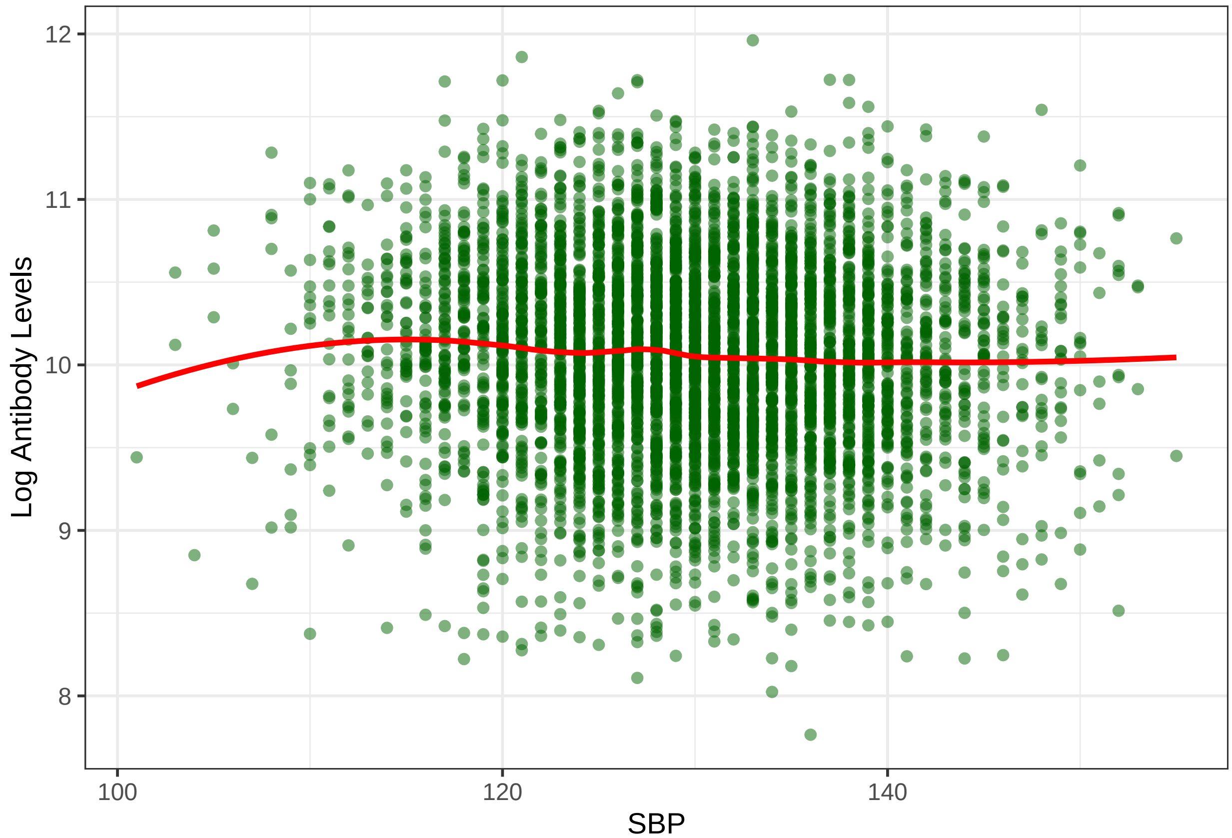



```r
#Average Log Antibody levels by hypertension status
hypertension_antibody <- ggplot(table_data, aes(x = factor(hypertension), y = log_antibody)) +
  geom_boxplot() +
  labs(x = "Hypertension", y = "Log Antibody Level") +
  theme_minimal()

hypertension_antibody
```



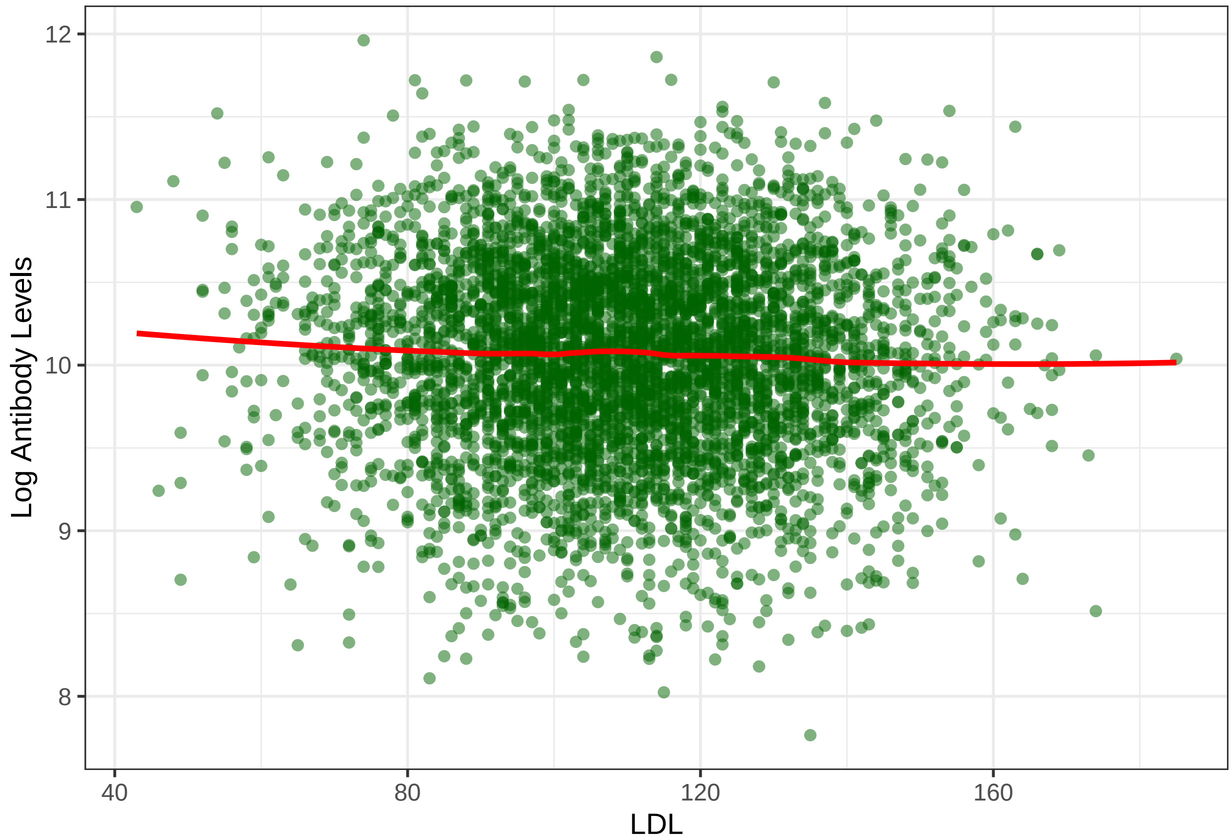

```
#Distribution of Log Antibody Levels by SBP
sbp_antibody <- ggplot(table_data, aes_string(x = "SBP", y = "log_antibody")) +
  geom_point(color = "darkgreen", alpha = 0.5) +
  geom_smooth(method = "loess", span = 0.5, color = "red", se = FALSE) +
  theme_bw() +
  labs(x = "SBP", y = "Log Antibody Levels")

sbp_antibody
```

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



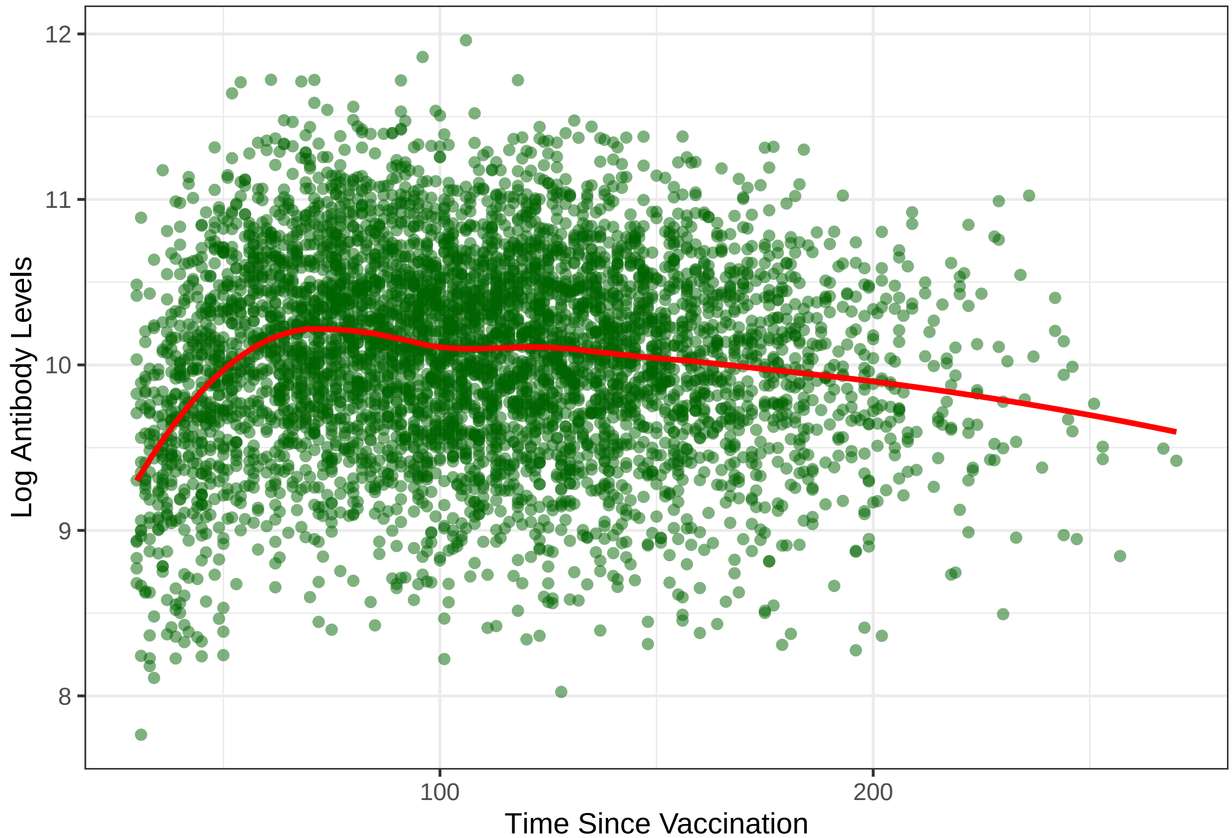

```r
#Distribution of Log Antibody Levels by LDL
LDL_antibody <- ggplot(table_data, aes_string(x = "LDL", y = "log_antibody")) +
  geom_point(color = "darkgreen", alpha = 0.5) +
  geom_smooth(method = "loess", span = 0.5, color = "red", se = FALSE) +
  theme_bw() +
  labs(x = "LDL", y = "Log Antibody Levels")

LDL_antibody
```

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

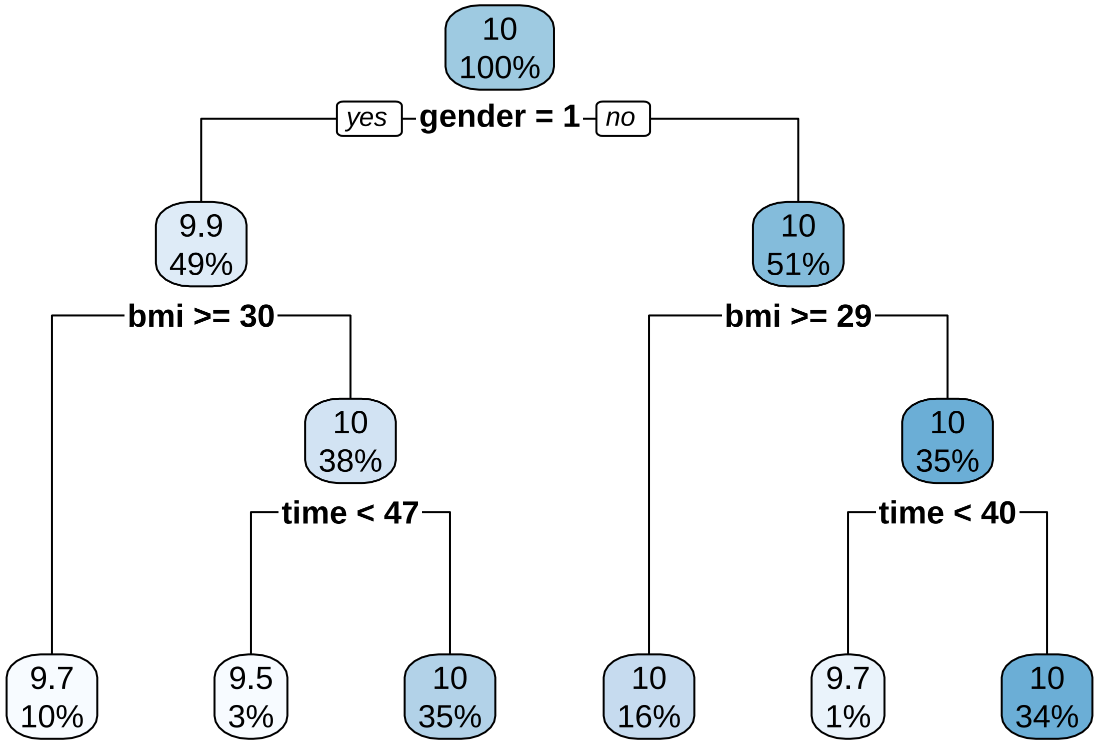



```
#Distribution of Log Antibody Levels by Time Since Vaccination
time_antibody <- ggplot(table_data, aes_string(x = "time", y = "log_antibody")) +
  geom_point(color = "darkgreen", alpha = 0.5) +
  geom_smooth(method = "loess", span = 0.5, color = "red", se = FALSE) +
  theme_bw() +
  labs(x = "Time Since Vaccination", y = "Log Antibody Levels")

time_antibody
```

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



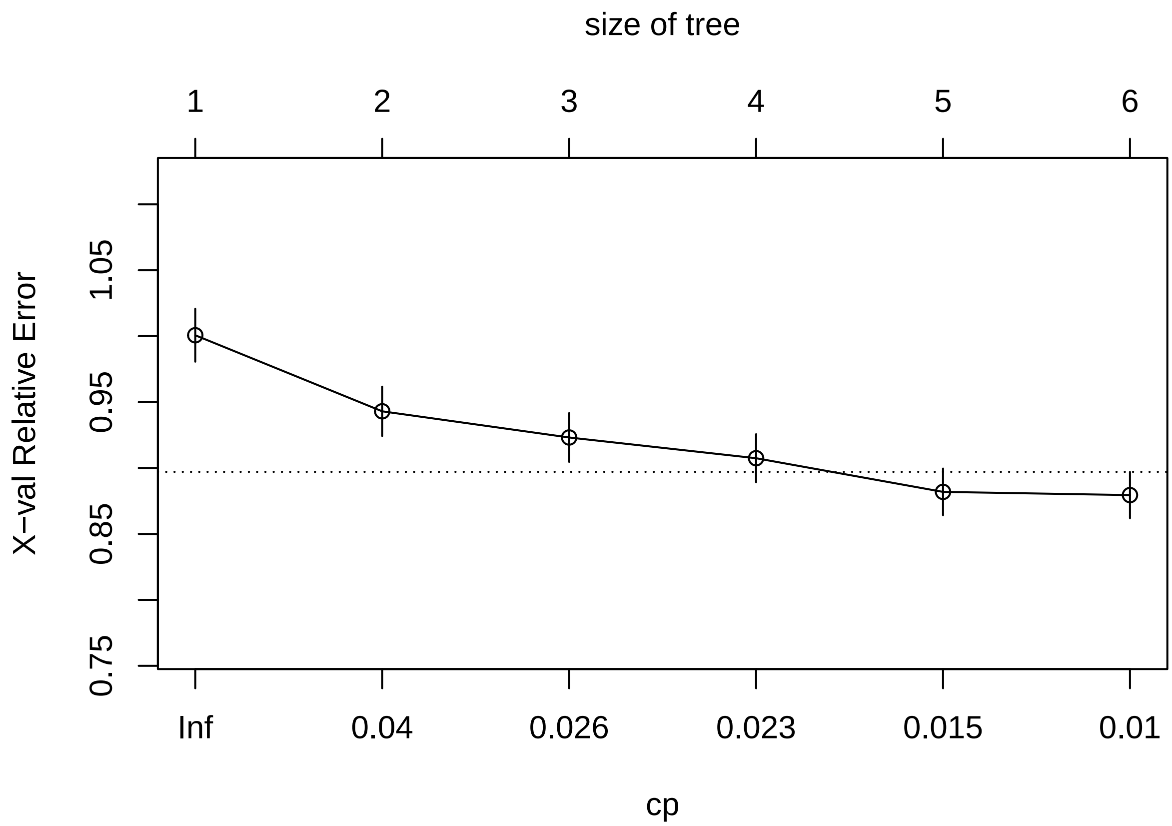

```
#setting seed for reproducibility
set.seed(1)

#applying the regression tree method to the data using a complexity parameter of 0.01
tree1 = rpart(formula = log_antibody~.,
              data = dat1,
              control = rpart.control(cp=0.01))

rpart.plot(tree1)
```



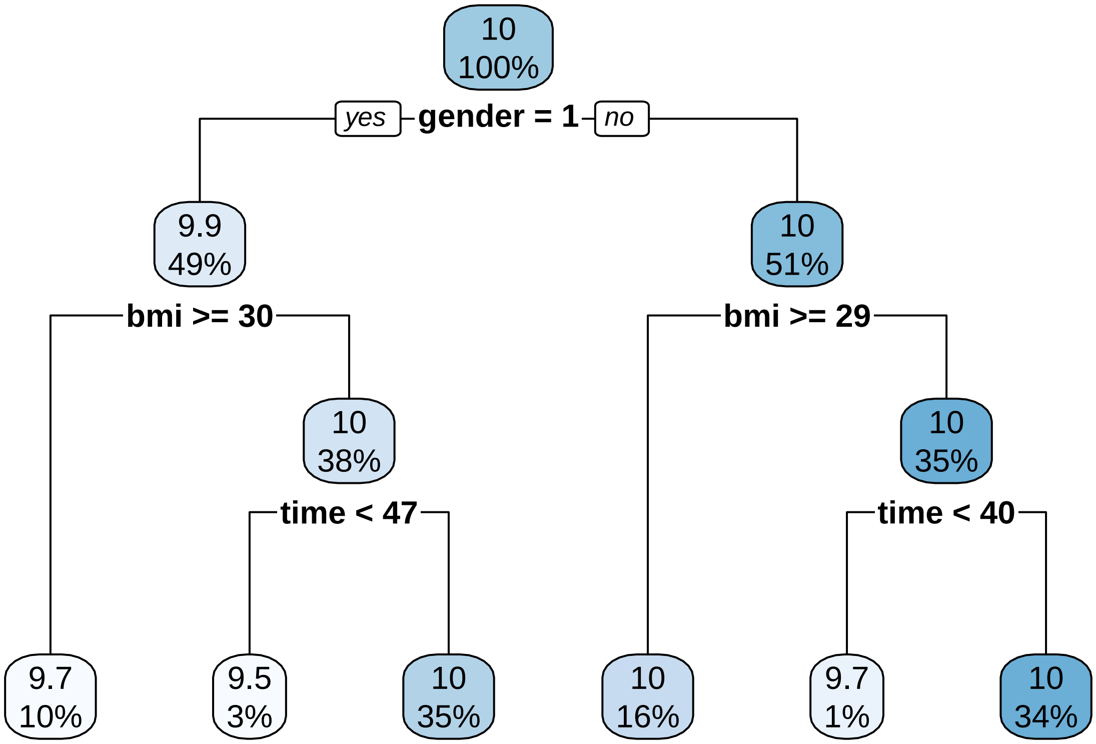

```r
#applying cost complexity pruning to obtain a tree with the right size
printcp(tree1)
```

```
##
## Regression tree:
## rpart(formula = log_antibody ~ ., data = dat1, control = rpart.control(cp = 0.01))
##
## Variables actually used in tree construction:
## [1] bmi    gender time
##
## Root node error: 1778.6/5000 = 0.35572
##
## n= 5000
##
##          CP nsplit rel error  xerror     xstd
## 1 0.057918      0   1.00000 1.00067 0.020039
## 2 0.027294      1   0.94208 0.94301 0.018701
## 3 0.025172      2   0.91479 0.92315 0.018498
## 4 0.020759      3   0.88962 0.90743 0.018234
## 5 0.010560      4   0.86886 0.88190 0.017687
## 6 0.010000      5   0.85830 0.87946 0.017565
```

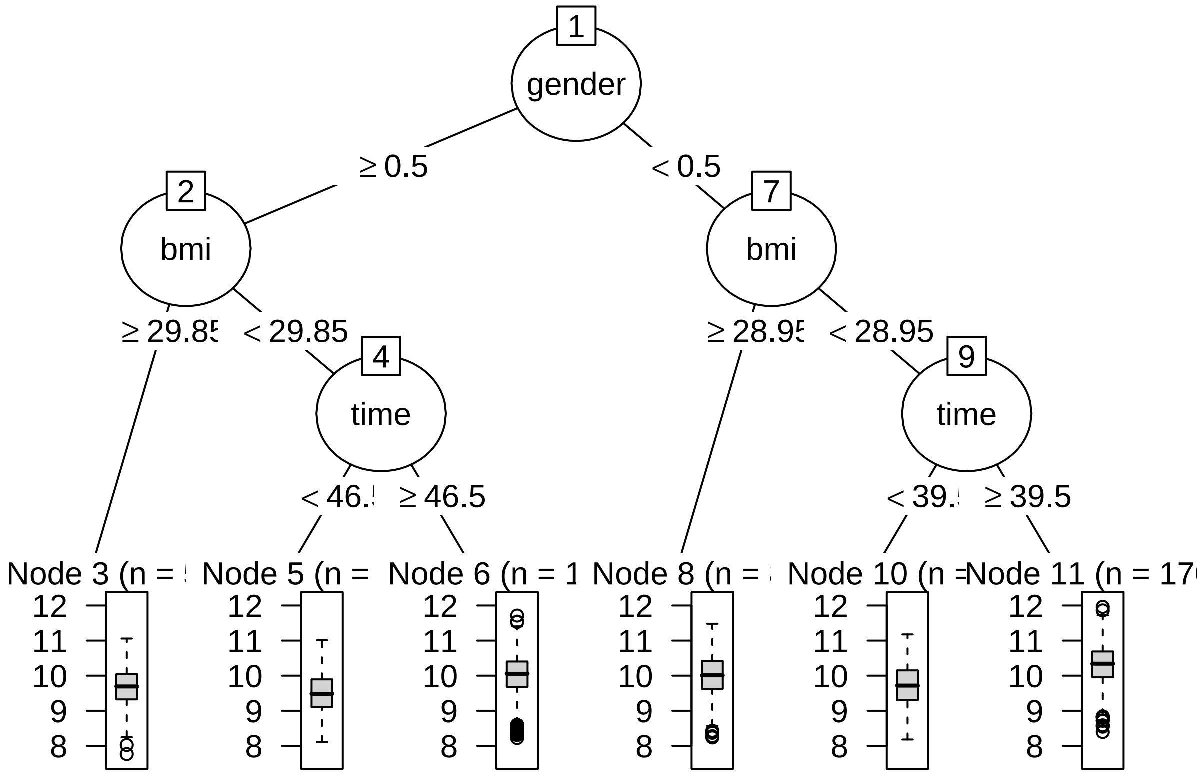
```r
cpTable = tree1$cptable
plotcp(tree1)
```




```
#picking the cp that yields the minimum cross-validation error
minErr = which.min(cpTable[,4])
tree3 = rpart::prune(tree1, cp = cpTable[minErr,1])
rpart.plot(tree3)
```




```
plot(as.party(tree3))
```




```
#predictions on the test data set
head(predict(tree3, newdata=dat2))
```

```
##     5001     5002     5003     5004     5005     5006
## 10.31597 10.00291 10.31597 10.02662 10.02662 10.31597
```

```
#computing the RMSE on the test set
RMSE(predict(tree3, newdata=dat2),dat2$log_antibody)
```

```
## [1] 0.5873775
```

The RMSE for the regression tree model is 0.5873775.

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

x2 <- model.matrix(log_antibody ~ ., dat2) [, -1]
y2 <- dat2$log_antibody

ctrl1 = trainControl(method = "cv", number = 10)
#GAM
gam.fit <- train(x, y,
method = "gam",
trControl = ctrl1)
```

```
## Loading required package: mgcv

## Loading required package: nlme

##
## Attaching package: 'nlme'

## The following object is masked from 'package:dplyr':
##
##     collapse

## This is mgcv 1.9-1. For overview type 'help("mgcv-package")'.
```

```
gam.fit$bestTune
```

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

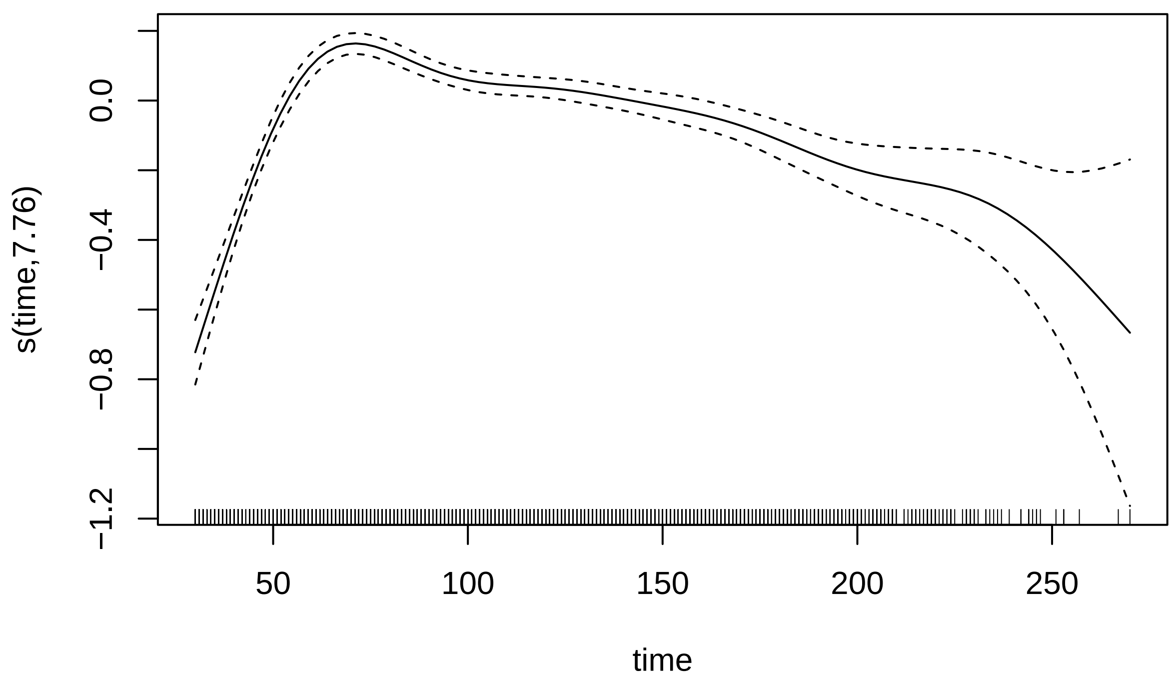
```
gam.fit$finalModel
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## .outcome ~ gender + race2 + race3 + race4 + smoking1 + smoking2 +
##     diabetes + hypertension + s(age) + s(SBP) + s(LDL) + s(bmi) +
##     s(time) + s(height) + s(weight) + s(id)
##
## Estimated degrees of freedom:
## 0.991 0.000 0.000 4.661 7.846 1.216 0.000
## 0.000  total = 23.71
##
## GCV score: 0.2786709
```

```
gam.m1 <- gam(log_antibody ~ time + age + gender + race + smoking + bmi + diabetes + hypertension +
    SBP + LDL, data = dat1)

gam.m2 <- gam(log_antibody ~ s(time) + age + gender + race + smoking + bmi + diabetes + hypertension +
    SBP + LDL, data = dat1)

anova(gam.m1, gam.m2, test = "F")
```

```
## Analysis of Deviance Table
##
## Model 1: log_antibody ~ time + age + gender + race + smoking + bmi + diabetes +
##     hypertension + SBP + LDL
## Model 2: log_antibody ~ s(time) + age + gender + race + smoking + bmi +
##     diabetes + hypertension + SBP + LDL
##   Resid. Df Resid. Dev     Df Deviance      F    Pr(>F)
## 1    4986.0     1520.6
## 2    4978.5     1407.0 7.4964   113.57 53.614 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(gam.m2)
```




```
#Summary of the model
summary(gam.fit)
```

```
##
## 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) + s(id)
##
## Parametric coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  10.228204   0.015328 667.284  < 2e-16 ***
## gender       -0.297827   0.014933 -19.944  < 2e-16 ***
## race2        -0.002962   0.033010  -0.090    0.928
## race3        -0.010567   0.018838  -0.561    0.575
## race4        -0.037352   0.026175  -1.427    0.154
## smoking1      0.022213   0.016659   1.333    0.182
## smoking2     -0.193179   0.025834  -7.478 8.88e-14 ***
## diabetes      0.014216   0.020639   0.689    0.491
## hypertension -0.007766   0.015995  -0.486    0.627
```

```
## ---
## 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)     9.910e-01      9 13.737  <2e-16 ***
## s(SBP)     5.214e-07      9  0.000   0.758
## s(LDL)     5.607e-07      9  0.000   0.634
## s(bmi)     4.661e+00      9 42.117  <2e-16 ***
## s(time)    7.846e+00      9 44.897  <2e-16 ***
## s(height) 1.216e+00      9  0.277   0.119
## s(weight) 1.746e-06      9  0.000   0.612
## s(id)      5.058e-07      9  0.000   0.731
## ---
## 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. = 0.27735   n = 5000
```

```
predy2_gam2 = predict(gam.fit, newdata=x2)
mean((y2 - predy2_gam2)^2)
```

```
## [1] 0.3233601
```

```
#MARS
mars_grid <- expand.grid(degree = 1:3,
                         nprune = 2:24)
set.seed(2)

mars.fit <- train(x, y,
                  method = "earth",
                  tuneGrid = mars_grid,
                  trControl = ctrl1)
```

```
## Loading required package: earth

## Loading required package: Formula

## Loading required package: plotmo

## Loading required package: plotrix

## Loading required package: TeachingDemos
```

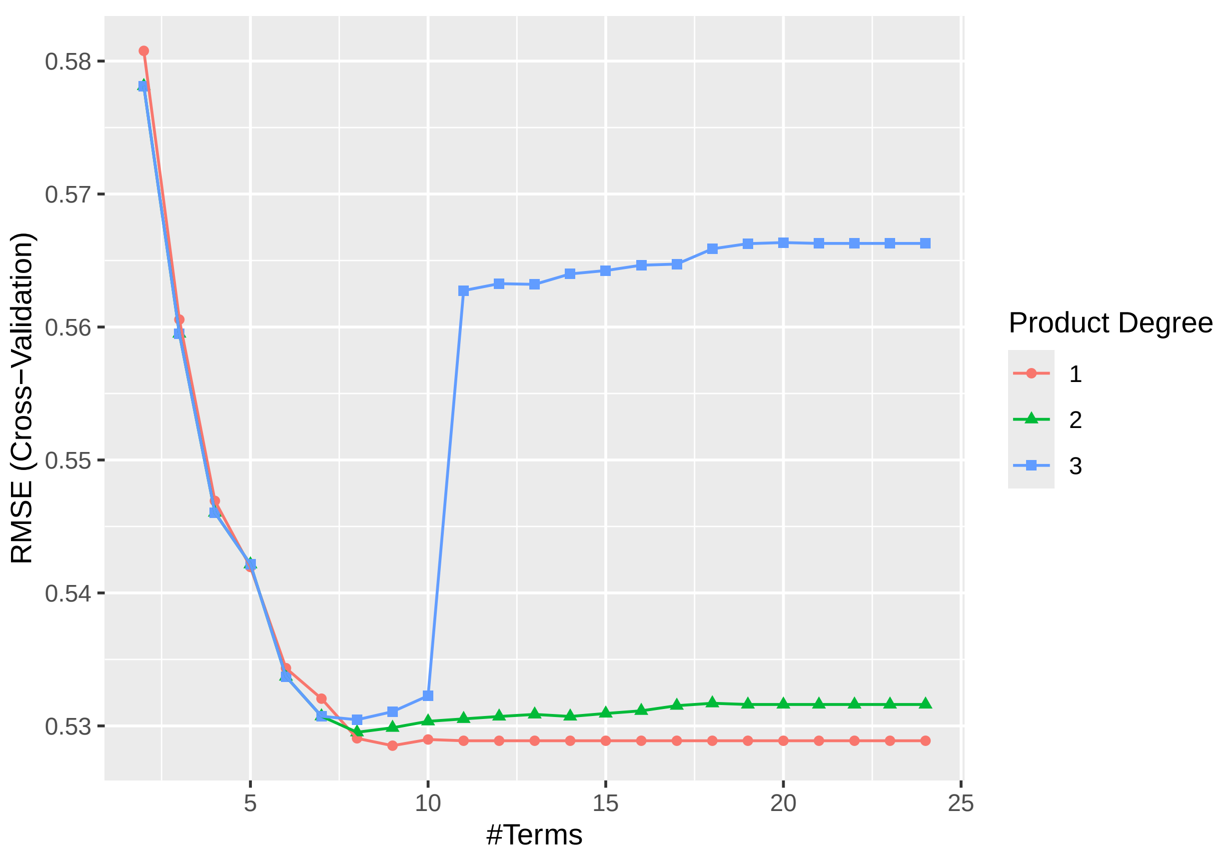
```
print(mars.fit)
```

```
## Multivariate Adaptive Regression Spline
##
## 5000 samples
##   16 predictor
##
```

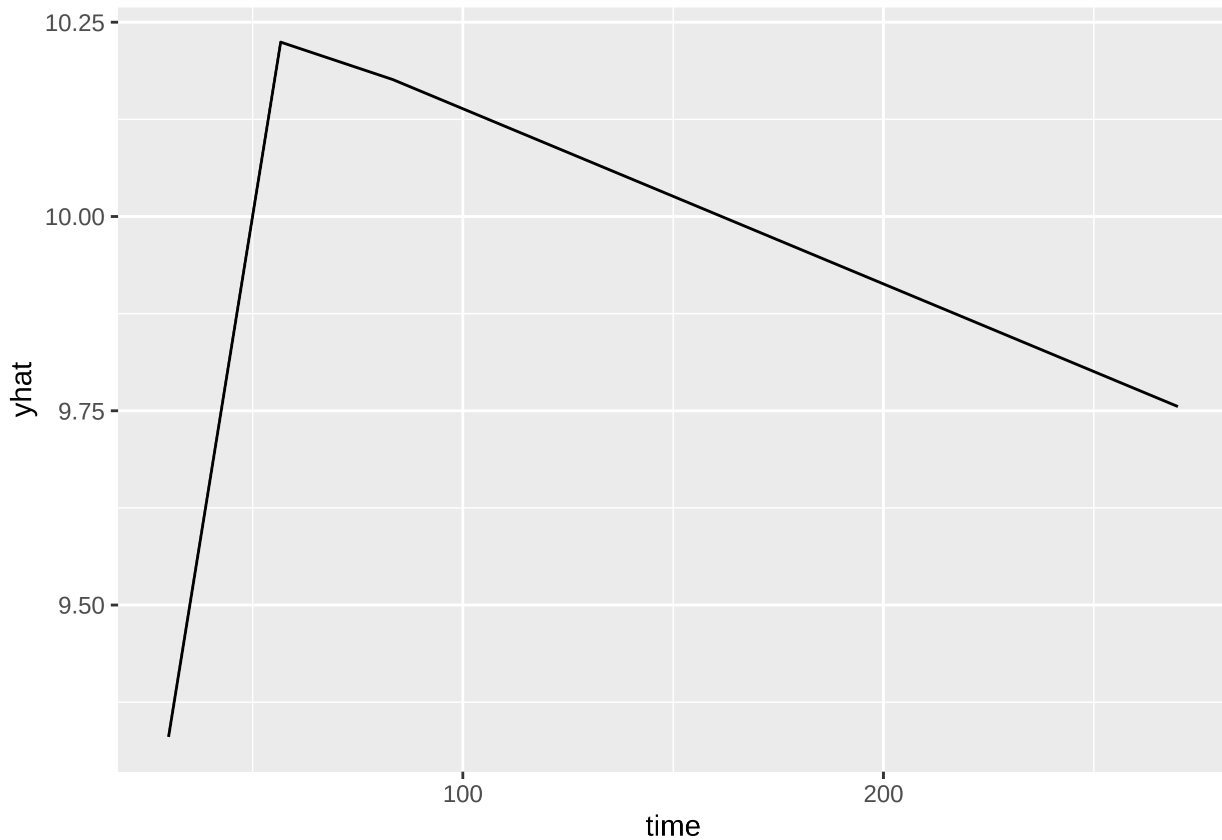
```
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 4500, 4500, 4500, 4500, 4500, 4500, ...
## Resampling results across tuning parameters:
##
##   degree  nprune  RMSE       Rsquared    MAE
##   1        2      0.5807692  0.05283061  0.4635960
##   1        3      0.5605605  0.11745151  0.4466383
##   1        4      0.5469228  0.16052460  0.4368910
##   1        5      0.5419521  0.17557467  0.4337872
##   1        6      0.5343428  0.19820980  0.4275795
##   1        7      0.5320492  0.20474285  0.4249636
##   1        8      0.5290659  0.21363786  0.4231253
##   1        9      0.5285140  0.21525203  0.4225896
##   1       10      0.5289781  0.21395019  0.4229606
##   1       11      0.5288834  0.21420645  0.4228466
##   1       12      0.5288834  0.21420645  0.4228466
##   1       13      0.5288834  0.21420645  0.4228466
##   1       14      0.5288834  0.21420645  0.4228466
##   1       15      0.5288834  0.21420645  0.4228466
##   1       16      0.5288834  0.21420645  0.4228466
##   1       17      0.5288834  0.21420645  0.4228466
##   1       18      0.5288834  0.21420645  0.4228466
##   1       19      0.5288834  0.21420645  0.4228466
##   1       20      0.5288834  0.21420645  0.4228466
##   1       21      0.5288834  0.21420645  0.4228466
##   1       22      0.5288834  0.21420645  0.4228466
##   1       23      0.5288834  0.21420645  0.4228466
##   1       24      0.5288834  0.21420645  0.4228466
##   2        2      0.5781251  0.06107058  0.4603579
##   2        3      0.5595074  0.12073713  0.4459426
##   2        4      0.5460432  0.16286797  0.4359953
##   2        5      0.5421573  0.17494284  0.4339437
##   2        6      0.5337112  0.20001565  0.4268039
##   2        7      0.5307271  0.20876169  0.4242725
##   2        8      0.5295219  0.21231107  0.4235161
##   2        9      0.5298622  0.21139347  0.4237756
##   2       10      0.5303457  0.21003540  0.4242311
##   2       11      0.5305282  0.20968669  0.4245326
##   2       12      0.5307113  0.20917149  0.4245285
##   2       13      0.5308646  0.20871531  0.4245775
##   2       14      0.5307109  0.20911254  0.4243655
##   2       15      0.5309348  0.20847865  0.4244027
##   2       16      0.5311350  0.20792582  0.4245865
##   2       17      0.5315224  0.20683739  0.4251212
##   2       18      0.5317025  0.20638212  0.4252511
##   2       19      0.5316107  0.20661726  0.4251365
##   2       20      0.5316107  0.20661726  0.4251365
##   2       21      0.5316107  0.20661726  0.4251365
##   2       22      0.5316107  0.20661726  0.4251365
##   2       23      0.5316107  0.20661726  0.4251365
##   2       24      0.5316107  0.20661726  0.4251365
##   3        2      0.5781251  0.06107058  0.4603579
##   3        3      0.5595074  0.12073713  0.4459426
```

```
##   3        4      0.5460432  0.16286797  0.4359953
##   3        5      0.5421573  0.17494284  0.4339437
##   3        6      0.5337112  0.20001565  0.4268039
##   3        7      0.5307271  0.20876169  0.4242725
##   3        8      0.5304631  0.21036958  0.4239092
##   3        9      0.5310714  0.20881385  0.4241905
##   3       10      0.5322644  0.20561321  0.4252479
##   3       11      0.5627406  0.19453970  0.4284630
##   3       12      0.5632603  0.19309239  0.4289979
##   3       13      0.5632076  0.19333725  0.4290152
##   3       14      0.5639942  0.19113227  0.4295424
##   3       15      0.5642453  0.19043259  0.4297209
##   3       16      0.5646472  0.18928015  0.4298023
##   3       17      0.5647326  0.18904363  0.4298871
##   3       18      0.5658774  0.18613820  0.4307447
##   3       19      0.5662693  0.18511710  0.4311256
##   3       20      0.5663512  0.18490003  0.4311733
##   3       21      0.5662872  0.18507242  0.4311410
##   3       22      0.5662872  0.18507242  0.4311410
##   3       23      0.5662872  0.18507242  0.4311410
##   3       24      0.5662872  0.18507242  0.4311410
##
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were nprune = 9 and degree = 1.
```

```
ggplot(mars.fit)
```



```
pdp::partial(mars.fit, pred.var =c("time"), grid.resolution=10) |> autoplot()
```




```
mars.fit$bestTune
```

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

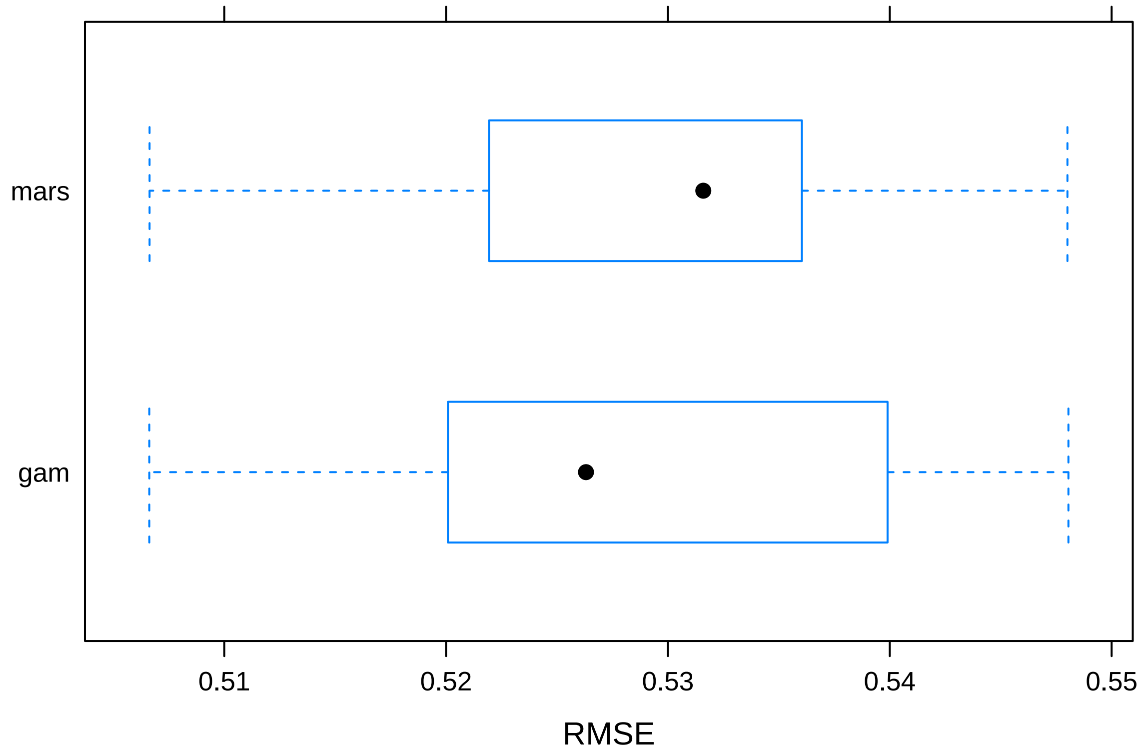
```
coef(mars.fit$finalModel)
```

```
##  (Intercept)  h(27.8-bmi)   h(time-57)   h(57-time)       gender    h(age-59)
## 10.847446930 -0.061997354 -0.002254182 -0.033529326 -0.296290451 -0.022957648
##    h(59-age)     smoking2  h(bmi-23.7)
##  0.016138468 -0.205126851 -0.084380175
```

```
predy2_mars2 = predict(mars.fit, newdata=x2)
mean((y2 - predy2_mars2)^2)
```

```
## [1] 0.2838458
```

```
#comparing models based on resampling results
resamp = resamples(list(gam = gam.fit,
                        mars = mars.fit))
bwplot(resamp, metric = "RMSE")
```



The RMSE for the MARS model is 0.2838458. The RMSE for the GAM model is 0.3233601. Resampling results provide additional evidence that the strongest of the two models compared above is the MARS model.

```
ctrl1 = trainControl(method = "repeatedcv",
                     number = 10,
                     repeats = 5,
                     selectionFunction = "best")

#pcr using caret
set.seed(2)

pcr_fit = train(x,y,
                method = "pcr",
                tuneGrid = data.frame(ncomp=1:5),
                trControl = ctrl1,
                preProcess = c("center","scale"))

predy2_pcr2 = predict(pcr_fit, newdata=x2)
mean((y2 - predy2_pcr2)^2)
```

```
## [1] 0.3675697
```

```
#pls using caret
pls_fit = train(x,y,
                method="pls",
```

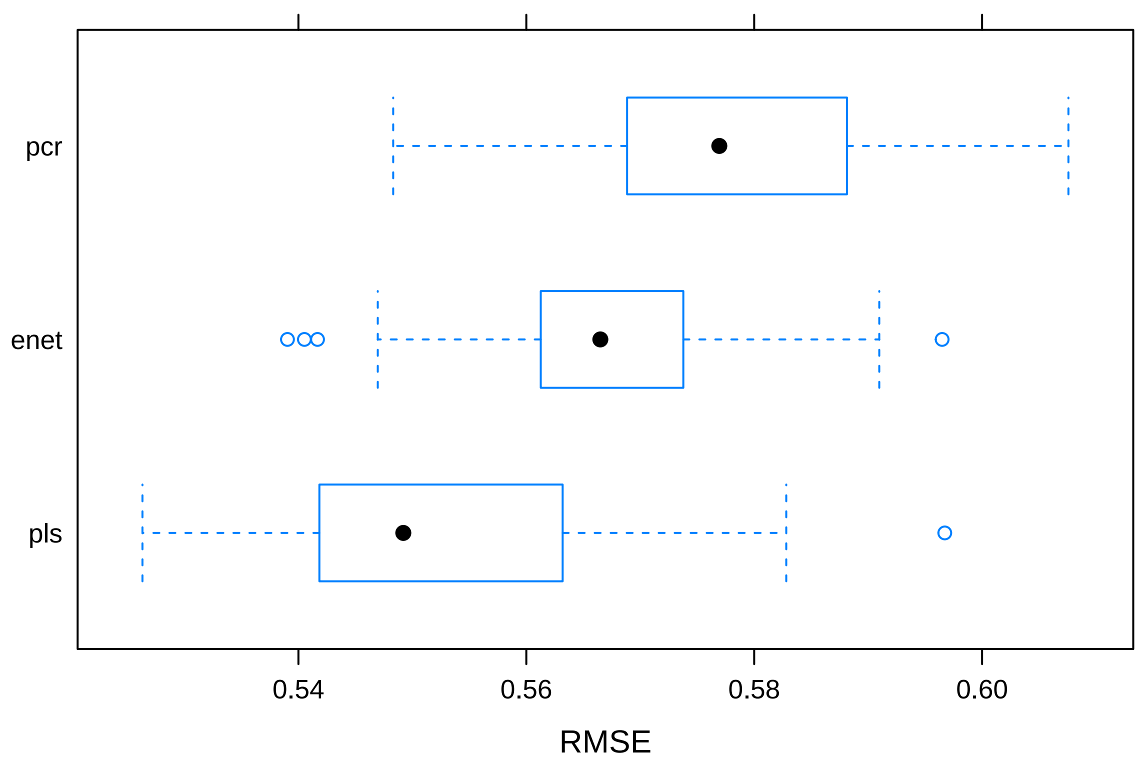
```r
                 tunegrid = data.frame(ncomp=1:5),
                 trControl = ctrl1,
                 preProcess = c("center","scale"))

predy2_pls2 = predict(pls_fit, newdata=x2)
mean((y2 - predy2_pls2)^2)
```

```
## [1] 0.324535
```

```r
#enet using caret
enet_fit = train(log_antibody ~.,
                  data = dat1,
                  method = "glmnet",
                  tuneGrid = expand.grid(alpha=seq(0,1, length = 21),
                                         lambda = exp(seq(6,0,length = 100))),
                  trControl = ctrl1)
```

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

```r
#comparing models based on resampling results
resamp = resamples(list(pcr = pcr_fit,
                         pls = pls_fit,
                         enet = enet_fit))
bwplot(resamp, metric = "RMSE")
```



The RMSE for the PCR model is 0.3675697. The RMSE for the PLS model is 0.324535. Resampling results provide additional evidence that the strongest of the three models compared above is the PLS model.

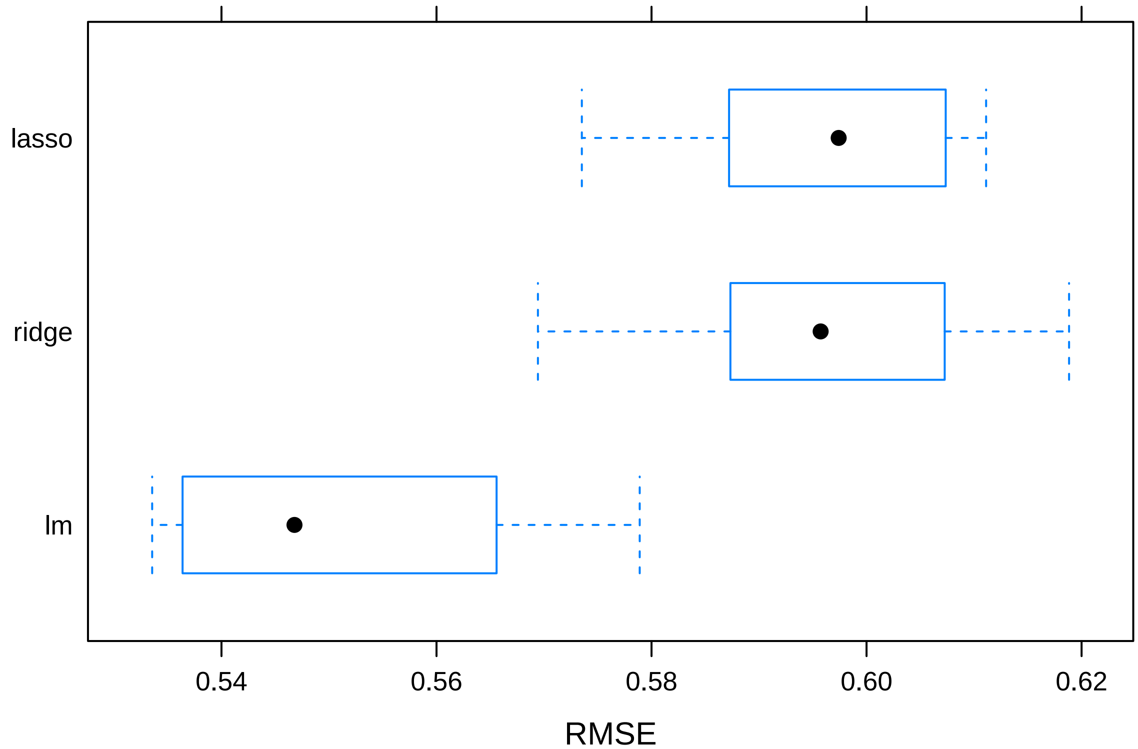
```
ctrl1 = trainControl(method = "cv", number = 10)

#lasso using caret
set.seed(2)
lasso.fit = train(log_antibody ~.,
                  data = dat1,
                  method = "glmnet",
                  tuneGrid = expand.grid(alpha=1,
                                         lambda = exp(seq(6,0,length=100))),
                  trControl = ctrl1)
```

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

```
#lm using caret
lm.fit = train(log_antibody ~.,
               data = dat1,
               method = "lm",
               trControl = ctrl1)

#ridge using caret
ridge.fit = train(log_antibody ~.,
                  data = dat1,
                  method = "glmnet",
                  tuneGrid = expand.grid(alpha=1,
                                         lambda = exp(seq(6,0,length=100))),
                  trControl = ctrl1)
```

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

```
#comparing models based on resampling results
resamp = resamples(list(lasso = lasso.fit,
                        lm = lm.fit,
                        ridge = ridge.fit))
bwplot(resamp, metric = "RMSE")
```



Among the three models compared above, the lm model appears to be the 'best' based on resampling results (as it has the lowest RMSE).