Predicting Chest Discomfort and Blood Pressure Categories using NHANES 2017-2018

Here's a place for a subtitle if you use one

Ritesh KC

2023-05-09

Table of contents

R Packages and Setup

```
knitr::opts_chunk$set(comment = NA)

library(janitor)
```

```
Attaching package: 'janitor'

The following objects are masked from 'package:stats':
```

```
    chisq.test, fisher.test
```

```
library(broom)
library(GGally)
```

```
Loading required package: ggplot2
```

```
Registered S3 method overwritten by 'GGally':
  method from
  +.gg   ggplot2
```

```
library(gtsummary)
```

```
Warning: package 'gtsummary' was built under R version 4.3.3
```

```
library(haven)
library(knitr)
library(nhanesA)
```

```
Warning: package 'nhanesA' was built under R version 4.3.3
```

```
library(naniar)
library(patchwork)
library(ROCR)
library(brant)
```

```
Warning: package 'brant' was built under R version 4.3.3
```

```
library(glue)
```

```
Warning: package 'glue' was built under R version 4.3.3
```

```
library(mice)
```

```
Warning: package 'mice' was built under R version 4.3.3

Attaching package: 'mice'

The following object is masked from 'package:stats':

    filter

The following objects are masked from 'package:base':

    cbind, rbind
```

```
library(rstanarm)
```

```
Loading required package: Rcpp

This is rstanarm version 2.32.1

- See https://mc-stan.org/rstanarm/articles/priors for changes to default priors!

- Default priors may change, so it's safest to specify priors, even if equivalent to the defa

- For execution on a local, multicore CPU with excess RAM we recommend calling

  options(mc.cores = parallel::detectCores())
```

```
library(tidymodels)
```

```
-- Attaching packages ------------------------------------ tidymodels 1.1.1 --

v dials        1.2.1      v rsample      1.2.0
v dplyr        1.1.4      v tibble       3.2.1
v infer        1.0.6      v tidyr        1.3.1
v modeldata    1.3.0      v tune         1.1.2
v parsnip      1.2.0      v workflows    1.1.4
v purrr        1.0.2      v workflowsets 1.0.1
v recipes      1.0.10     v yardstick    1.3.0
```

```
-- Conflicts --------------------------------------- tidymodels_conflicts() --
x purrr::discard()    masks scales::discard()
x dplyr::filter()     masks mice::filter(), stats::filter()
x dplyr::lag()        masks stats::lag()
x rsample::populate() masks Rcpp::populate()
x recipes::step()     masks stats::step()
* Dig deeper into tidy modeling with R at https://www.tmwr.org
```

```
library(caret)
```

```
Warning: package 'caret' was built under R version 4.3.3
```

```
Loading required package: lattice
```

```
Attaching package: 'caret'
```

```
The following objects are masked from 'package:yardstick':

    precision, recall, sensitivity, specificity
```

```
The following object is masked from 'package:purrr':

    lift
```

```
The following objects are masked from 'package:rstanarm':

    compare_models, R2
```

```
library(MASS)
```

```
Attaching package: 'MASS'
```

```
The following object is masked from 'package:dplyr':

    select
```

```
The following object is masked from 'package:patchwork':

    area

The following object is masked from 'package:gtsummary':

    select
```

```
library(gmodels)
```

```
Warning: package 'gmodels' was built under R version 4.3.3
```

```
library(nnet)
library(rsample)
library(simputation)
```

```
Warning: package 'simputation' was built under R version 4.3.3

Attaching package: 'simputation'

The following object is masked from 'package:naniar':

    impute_median
```

```
library(rms)
```

```
Loading required package: Hmisc

Attaching package: 'Hmisc'

The following object is masked from 'package:simputation':

    impute

The following object is masked from 'package:parsnip':

    translate
```

```
The following objects are masked from 'package:dplyr':

    src, summarize

The following objects are masked from 'package:base':

    format.pval, units
```

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v forcats   1.0.0     v readr     2.1.5
v lubridate 1.9.3     v stringr   1.5.1

-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x readr::col_factor() masks scales::col_factor()
x purrr::discard()    masks scales::discard()
x dplyr::filter()     masks mice::filter(), stats::filter()
x stringr::fixed()    masks recipes::fixed()
x dplyr::lag()        masks stats::lag()
x caret::lift()       masks purrr::lift()
x MASS::select()      masks dplyr::select(), gtsummary::select()
x readr::spec()       masks yardstick::spec()
x Hmisc::src()        masks dplyr::src()
x Hmisc::summarize()  masks dplyr::summarize()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becor
```

```
theme_set(theme_bw(base_size = 15))
```

1 Background

Cardiovascular diseases (CVDs) are the leading cause of death in the United States. In the year 2020, 697,000 people in the United States died from heart disease (2). Among many CVDs, more than four out of five CVD deaths are due to heart attacks and strokes. The risk factors of heart disease and stroke are unhealthy diet, physical inactivity, tobacco use and harmful use of alcohol among others. The effects of behavioral risk factors may show up in individuals as raised blood pressure and other symptoms such as discomfort or pain in the chest and indicate an increased risk for CVDs. Therefore, predicting the risk of CVDs is of great significance for disease management, including timely intervention and rational drug use.

2 Data Source

The National Health and Nutrition Examination Survey (NHANES) is a program of studies designed to assess the health and nutritional status of adults and children in the United States. The survey combines interviews and physical examinations and the findings from these surveys are used to determine the prevalence of major diseases and risk factors for diseases. Data from these surveys also help to develop public health policies, programs and services. 2017-2018 survey data is described in detail here: https://wwwn.cdc.gov/nchs/nhanes/continuousnhanes/default.aspx?BeginYear=2017

The reason I picked 2017-2018 data is because NHANES program suspended field operations in March 2020 due to the corona virus disease 2019 (COVID-19) pandemic. Therefore, the data collection for the latest NHANES 2019-2020 cycle was not completed and the collected data are not nationally representative. Further it might have additional biases in surveys due to the pandemic itself.

3 Loading and Tidying the Raw Data

3.1 NHANES data we will collect

NHANES data contain several data documents which are broadly divided into several categories such as demographic data, examination data, dietary data, laboratory data, questionnaire data. The variables I picked are under following categories.

1. Demographic Data

- SEQN = Respondent identifying code
- RIAGENDER = Gender of respondents at the time of the screening interview, 1 Male, 2 Female
- RIDAGEYR = Age in years, 0-79 range of values, 80 and above=80

2. Examination Data:

- BPXPULS = Pulse regular or irregular, 1 Regular, 2 Irregular
- BPXSY1 = Systolic blood pressure in mm of Hg, first reading, range of values

- BPXDI1 = Diastolic blood pressure in mm of Hg, first reading, range of values
- BMXBMI = Body mass index (Kg/m^2), range of values, . missing
- SLD012 = Sleep hours weekend or workdays

3. Laboratory Data

- LBXTC = Total cholesterol (mg/dL), range of values

- LBDHDD = Direct HDL-Cholesterol (mg/dL), range of values

4. Questionnaire Data

- BPQ080 = Doctor told you high cholesterol level, 1 Yes, 2 No
- CDQ001 = SP ever had pain or discomfort in chest, 1 Yes, 2 No
- PAQ605 = Vigorous work activity for 10mins in a week, 1 Yes, 2 No
- PAQ620 = Moderate work activity for 10mins in a week, 1 Yes, 2 No
- PAD680 = Minutes of sedentary activity, range of values, 7777 refused, 9999 don't know
- DIQ010 = Doctor told you if you have diabetes, 1 = Yes, 2 = No, 3 = borderline
- SMQ040 = Do you now smoke cigarette, 1 = Everyday, 2 = Someday, 3 = No

4 Data Ingest and Management

First I pulled in all the required data from the NHANES 2017-2018. Then, I joined all the data by using left join function and using SEQN as the reference and converted into tibble.

```
bpq_1 <- read_xpt("BPQ_J.XPT") |> dplyr::select(SEQN, BPQ080)
bpm_1 <- read_xpt("BPX_J.XPT") |> dplyr::select(SEQN, BPXPULS, BPXSY1, BPXDI1)
bmi_1 <- read_xpt("BMX_J.XPT") |> dplyr::select(SEQN, BMXBMI)
sldq_1 <- read_xpt("SLQ_J.XPT") |> dplyr::select(SEQN, SLD012)
tchl_1 <- read_xpt("TCHOL_J.XPT") |> dplyr::select(SEQN, LBXTC)
hdl_1 <- read_xpt("HDL_J.XPT") |> dplyr::select(SEQN, LBDHDD)
demo_1 <- read_xpt("DEMO_J.XPT") |> dplyr::select(SEQN, RIAGENDR, RIDAGEYR)
card_h <- read_xpt("CDQ_J.XPT") |>  dplyr::select(SEQN, CDQ001)
phy_a <- read_xpt("PAQ_J.XPT") |> dplyr::select(SEQN, PAQ605, PAQ620, PAD680)
dia_a <- read_xpt("DIQ_J.XPT") |> dplyr::select(SEQN, DIQ010)
smq_1 <- read_xpt("SMQ_J.XPT") |> dplyr::select(SEQN, SMQ040)

df1 <- demo_1 |>  left_join(bpq_1, by="SEQN") |> left_join(bpm_1, by="SEQN") |> left_join(
```

I further sorted subjects that are between the age of 45 and 75. This age group is selected on the basis of their association with risk of cardiovascular diseases.

```
df1 <- df1 |> filter(RIDAGEYR > 45 & RIDAGEYR < 75)
```

4.1 Checking Complete Cases on Output Variable

I am planning to use blood pressure categories, which include both systolic and diastolic and chest discomfort as my outcome variable. Therefore, I only picked data that has complete cases in our outcome variable.

```r
df1 <- df1 |> drop_na(c(BPXSY1,BPXDI1,CDQ001)) #complete cases for output variable

miss_var_summary(df1) |> filter(n_miss > 0)
```

```
# A tibble: 6 x 3
  variable n_miss pct_miss
  <chr>     <int>    <num>
1 SMQ040     1226  53.7
2 LBXTC       131   5.74
3 LBDHDD      131   5.74
4 BMXBMI       21   0.921
5 SLD012       19   0.833
6 PAD680        1   0.0438
```

Next, I dropped all cases that has "NA" is smoking variable. The motivation of dropping "NA" on smoking variable is that smoking could be a risk factor for blood pressure increase and chest pain and the it has too many missing data (~54%). Therefore, by excluding NA the data would be more manageable and interpretable.

```r
df1 <- df1 |> drop_na(SMQ040)

miss_var_summary(df1) |> filter(n_miss > 0)
```

```
# A tibble: 5 x 3
  variable n_miss pct_miss
  <chr>     <int>    <num>
1 LBXTC        62   5.88
2 LBDHDD       62   5.88
3 SLD012       13   1.23
4 BMXBMI        9   0.853
5 PAD680        1   0.0948
```

4.2 Creating bp_cat Variable

Further, I divided blood pressure into four groups based on systolic and diastolic blood pressure values and categories used by American Heart Association. Systolic less than 120 and diastolic less than 80 = Normal Systolic 120-129 and diastolic less than 80 = Elevated Systolic 130-139 or diastolic 80-89 = Hypertension stage 1 Systolic over 140 or diastolic over 90 = Hypertension stage 2 Blood pressure categories are added as a new variable (bp_cat) in the data.

```
#Group bp into 4 groups based on systolic and diastolic values.
df1 <- df1 |> mutate(bp_cat = factor(case_when(BPXSY1 < 120 & BPXDI1 < 80 ~"1",
                                               BPXSY1 >= 120 & BPXSY1 < 130 & BPXDI1 < 80
                                               BPXSY1 >= 130 & BPXSY1 < 139 | BPXDI1 >= 80
                                               BPXSY1 >= 140 | BPXDI1 >= 90 ~"4")))
```

5 Cleaning the Data

5.1 Select Variables

Variables are selected and named accordingly. The variables that needed to be changed to factor are also converted accordingly.

```
df2 <- df1 |> mutate (id = as.character(SEQN),
          age = RIDAGEYR,
          sex = as.factor(RIAGENDR),
          bmi = BMXBMI,
          sleep = SLD012,
          chol = LBXTC,
          hdl = LBDHDD,
          inact = PAD680,
          chst_pain = as.factor(CDQ001),
          smoke = as.factor(SMQ040),
          bp_cat = as.factor(bp_cat))

df2 <-  df2 |> dplyr:: select(id, age, sex, bmi, sleep, chol, hdl,inact, smoke, chst_pain,
```

5.2 Recoding Factor Variables

Suggestion from Dr. Love after presentation, I am re-coding all factor variable here.

```
df2 <- df2 |> mutate(sex = fct_recode(sex, "M"="1", "F"="2"),
                     smoke = fct_recode(smoke, "Everyday"="1", "Sometimes"="2", "Never"="3
                     chst_pain = fct_recode(chst_pain, "CP_Yes" = "1", "CP_No" = "2"),
                     bp_cat = fct_recode(bp_cat, "Normal" = "1", "Elevated" = "2", "Hypert
```

5.3 Checking Quantitative Variables

```
df2 |> dplyr::select(age, bmi, sleep, chol, hdl, inact) |> summary()
```

```
      age              bmi             sleep              chol      
 Min.   :46.00   Min.   :14.90   Min.   : 2.000   Min.   : 76.0  
 1st Qu.:54.00   1st Qu.:25.40   1st Qu.: 6.500   1st Qu.:164.0  
 Median :61.00   Median :29.10   Median : 7.500   Median :189.0  
 Mean   :60.45   Mean   :30.11   Mean   : 7.523   Mean   :193.2  
 3rd Qu.:67.00   3rd Qu.:33.80   3rd Qu.: 8.500   3rd Qu.:221.0  
 Max.   :74.00   Max.   :74.80   Max.   :13.000   Max.   :446.0  
                 NA's   :9       NA's   :13       NA's   :62     
      hdl              inact       
 Min.   : 18.00   Min.   :   0  
 1st Qu.: 41.00   1st Qu.: 180  
 Median : 49.00   Median : 300  
 Mean   : 51.96   Mean   : 428  
 3rd Qu.: 60.00   3rd Qu.: 480  
 Max.   :178.00   Max.   :9999  
 NA's   :62       NA's   :1     
```

Looking at the summary of our quantitative variable ranges for age and sleep look plausible. However, the maximum for bmi, chol, hdl looks a bit too high. I will see if they will show up as outlier later. The max range for inact is 9999 which is due to the respondent answer of "don't know", which I will remove next.

5.4 Remove 9999 values from Inact

The max range for inact was 9999, which corresponds to the respondent's answer of "don't know". I removed them from inact variable.

```
df2 <- df2 |> filter(inact != '9999')
dim(df2)
```

```
[1] 1045   11
```

Now my data has 1045 rows and 11 columns.

5.5 Checking Categorical Variables

```
df2 |> tabyl(sex)
```

```
sex   n   percent
  M 660 0.6315789
  F 385 0.3684211
```

```
df2 |> tabyl(chst_pain)
```

```
chst_pain   n   percent
   CP_Yes 372 0.3559809
    CP_No 673 0.6440191
```

```
df2 |> tabyl(smoke)
```

```
    smoke   n    percent
 Everyday 345 0.33014354
Sometimes  84 0.08038278
    Never 616 0.58947368
```

```
df2 |> tabyl(bp_cat)
```

```
              bp_cat   n   percent
              Normal 261 0.2497608
           Elevated 189 0.1808612
 Hypertension_Stage_1 351 0.3358852
 Hypertension_Stage_2 244 0.2334928
```

The dataset doesn't seem to have any odd observations in any of the categorical variable.

5.6 Checking for Missingness

```
summary(complete.cases(df2))
```

```
   Mode   FALSE    TRUE
logical      76     969
```

```
miss_var_summary(df2)|> filter(n_miss > 0)
```

```
# A tibble: 4 x 3
  variable n_miss pct_miss
  <chr>     <int>    <num>
1 chol         61    5.84
2 hdl          61    5.84
3 sleep        11    1.05
4 bmi           8    0.766
```

In my data df2, I have a total of 969 complete cases. I have 61 missing values each for chol and hdl. In addition 11 values are missing from variable sleep and 8 values are missing from variable bmi.

6 The Tidy Tibble

6.1 Listing the Tibble

```
df2
```

```
# A tibble: 1,045 x 11
   id      age sex     bmi sleep  chol   hdl inact smoke    chst_pain bp_cat
   <chr> <dbl> <fct> <dbl> <dbl> <dbl> <dbl> <dbl> <fct>    <fct>     <fct>
 1 93713    67 M      23.5   5.5   184    48   120 Everyday CP_No     Normal
 2 93715    71 M      22.5   5     180    57   180 Everyday CP_Yes    Normal
 3 93716    61 M      30.7   7     225    39   300 Never    CP_No     Elevated
 4 93726    67 F      31.1  10     176    35    60 Never    CP_No     Hypertens~
 5 93732    72 M      21.3   6      NA    NA   300 Never    CP_Yes    Hypertens~
 6 93740    72 M      30.6   8      NA    NA   300 Everyday CP_Yes    Hypertens~
 7 93742    72 M      33.9   9.5   160    48   180 Never    CP_Yes    Hypertens~
 8 93743    61 M      22.5   5.5   146    41   420 Everyday CP_No     Hypertens~
 9 93752    73 F      25.5   5     262    60   180 Everyday CP_No     Elevated
10 93758    55 F      30.8   5.5   446    49   240 Everyday CP_Yes    Hypertens~
# i 1,035 more rows
```

6.2 Size and Identifiers

```
dim(df2)
```

```
[1] 1045   11
```

```
n_distinct(df2$id)
```

```
[1] 1045
```

My table called df2 has now 1045 rows and 11 columns corresponding to observations and variables respectively. My indicator variable is id, which is unique for each row shown by the distinct number of rows above.

6.3 Saving the Tibble

```
saveRDS(df2, "projectBportfolio_df2.riteshkc.Rds")
```

7 Code Book and Description

1. Sample Size: The sample of my data consists of 1045 subjects between the age of 45 and 75 from NHANES 2017-2018 for whom the outcome variable is chst_pain and bp_cat.
2. Missingness: There are a total of 969 complete cases. chol and hdl are missing 61 values each, sleep is missing 11 values, and bmi is missing 8.
3. Outcome : My outcome variable is chst_pain, which is whether the respondents said "yes" or "no" to the question if they have any pain or discomfort in the chest. Another outcome variable is the blood pressure groups that I created on the basis of American Heart Association categorization. Both of our outcome variables do not have any missing data.
4. Predictors: Candidate predictors for my outcome includes age, sex, bmi, sleep ,inact, smoke that are common for both logistic and multicategorical prediction. While chol is included for multicategorical model and hdl for logistic model.
5. Id: The variable id my tibble is the subject identifying code.

7.1 Defining the Variables

```
tbl_summary(dplyr::select(df2, -id),
            label = list(
            age = "Age (in years)",
            sex = "sex (Male/ Female)",
            bmi = "Body Mass Index (in Kg/m^2)",
            sleep = "sleep (in sleep hours per day)?",
            chol = "Total Cholesterol (in mg/dL)",
            hdl = "High Density Lipid (im mg/dL)",
            inact = "Sedenatry Status (hours per day)",
            smoke = "smoking Status (Everyday/ Sometimes/ Never)",
            chst_pain = "Chest Pain (CP_Yes/ CP_No)",
            bp_cat = "Blood Pressure Groups (Normal/ Elevated/ Hypertension Stage 1/ Hyper
        stat = list(all_continuous() ~
                "{median} [{min} to {max}]" ))
```

7.2 Numeric Descripton

```
describe(df2) |> html()
```

n	missing	distinct
1045	0	1045

lowest : 100020 100037 100046 100051 100055 , highest: 99969 99984 99987 99991 99996

---

age: Age in years at screening

n	missing	distinct	Info	Mean	Gmd	.05	.10	.25	.50	.75	.90	.95
1045	0	29	0.999	60.43	8.951	47	49	54	61	67	71	72

lowest : 46 47 48 49 50 , highest: 70 71 72 73 74

Characteristic	**N = 1,**
Age (in years)	61 [46 t
sex (Male/ Female)	
M	660 (6
F	385 (3
Body Mass Index (in Kg/m^2)	29 [15 t
Unknown	8
sleep (in sleep hours per day)?	7.50 [2.00 t
Unknown	11
Total Cholesterol (in mg/dL)	189 [76 t
Unknown	61
High Density Lipid (im mg/dL)	49 [18 t
Unknown	61
Sedenatry Status (hours per day)	300 [0 to
smoking Status (Everyday/ Sometimes/ Never)	
Everyday	345 (3
Sometimes	84 (8.
Never	616 (5
Chest Pain (CP_Yes/ CP_No)	
CP_Yes	372 (3
CP_No	673 (6
Blood Pressure Groups (Normal/ Elevated/ Hypertension Stage 1/ Hypertension Stage 2)	
Normal	261 (2
Elevated	189 (1
Hypertension_Stage_1	351 (3
Hypertension_Stage_2	244 (2

[1]Median [Min to Max]; n (%)

sex

n	missing	distinct
1045	0	2

```
Value          M     F
Frequency    660   385
Proportion 0.632 0.368
```

---

bmi: Body Mass Index (kg/m**2)

n	missing	distinct	Info	Mean	Gmd	.05	.10	.25	.50	.75	.90	.95
1037	8	285	1	30.11	7.726	20.38	22.10	25.40	29.00	33.80	39.34	43.30

lowest : 14.9 15.5 15.7 16.7 16.8 , highest: 57.2 58.8 61.6 63.4 74.8

---

sleep: Sleep hours - weekdays or workdays

n	missing	distinct	Info	Mean	Gmd	.05	.10	.25	.50	.75	.90	.95
1034	11	22	0.987	7.524	1.883	4.5	5.5	6.5	7.5	8.5	9.5	10.0

lowest : 2 3 3.5 4 4.5 , highest: 11 11.5 12 12.5 13

---

chol: Total Cholesterol (mg/dL)

n	missing	distinct	Info	Mean	Gmd	.05	.10	.25	.50	.75	.90	.95
984	61	204	1	193.2	50.45	124.0	137.3	164.0	189.0	221.0	251.0	273.0

lowest : 76 79 81 84 94 , highest: 352 354 365 431 446

hdl: Direct HDL-Cholesterol (mg/dL)

n	missing	distinct	Info	Mean	Gmd	.05	.10	.25	.50	.75	.90	.95
984	61	83	0.999	51.97	17.07	32	35	41	49	60	72	80

lowest : 18 22 23 24 26 , highest: 121 122 139 147 178

---

inact: Minutes sedentary activity

n	missing	distinct	Info	Mean	Gmd	.05	.10	.25	.50	.75	.90	.95
1045	0	33	0.99	345.6	235.9	60	120	180	300	480	660	720

lowest : 0 3 15 20 30 , highest: 960 1020 1080 1200 1320

---

smoke

n	missing	distinct
1045	0	3

Value	Everyday	Sometimes	Never
Frequency	345	84	616
Proportion	0.330	0.080	0.589

---

chst_pain

n	missing	distinct
1045	0	2

```
Value      CP_Yes  CP_No
Frequency     372    673
Proportion  0.356  0.644
```

---

bp_cat

n	missing	distinct
1045	0	4

8 Analysis

8.1 My First Research Question

How well can we predict blood pressure groups using age, sex, bmi, sleep hour, total cholesterol level, sedentary minutes, and smoking status in a sample of 1045 NHANES participants ages 45-75?

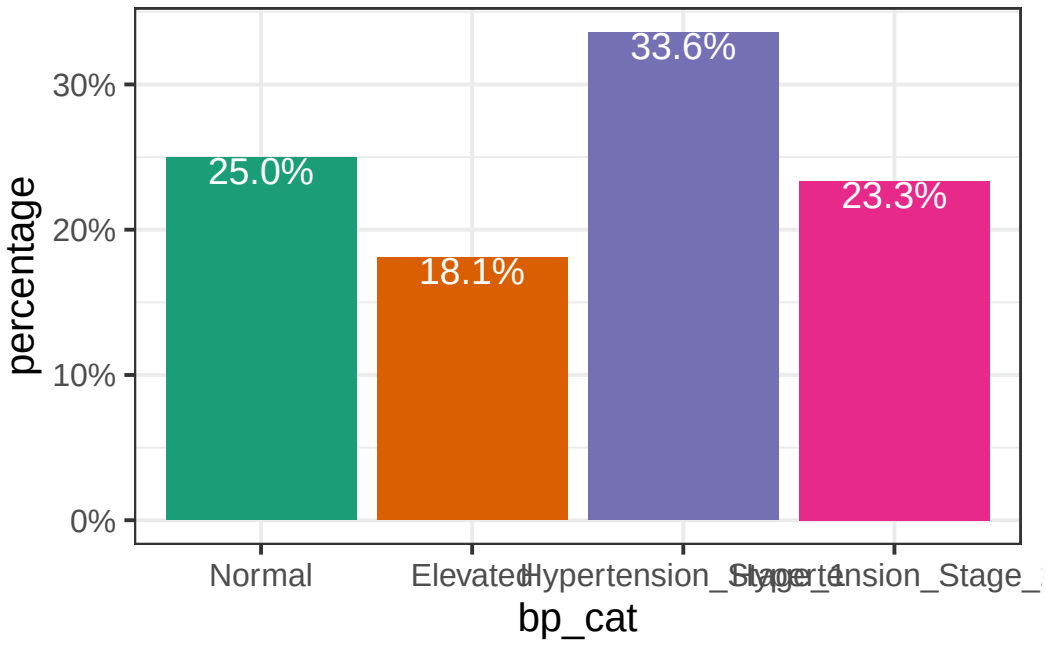
8.1.1 My Categorical Outcome

- My categorical outcome is bp_cat and I am predicting this value using other demographic and risk factors.
- I have a complete data in bp_cat for all 1045 of my subjects.

Lets check the distribution of samples across my bp_group categories.

```
ggplot(df2, aes(x = bp_cat, fill = bp_cat)) +
  geom_bar(aes(y = (after_stat(count))/sum(after_stat(count)))) +
  geom_text(aes(y = (after_stat(count))/sum(after_stat(count)),
label = scales::percent((after_stat(count)) / sum(after_stat(count)))),
              stat = "count", vjust = 1,
color = "white", size = 5) + scale_y_continuous(labels = scales::percent) + scale_fill_bre
labs(y = "percentage")
```

```
Warning: The `<scale>` argument of `guides()` cannot be `FALSE`. Use "none" instead as
of ggplot2 3.3.4.
```



The histogram shows that we have highest percentage of subjects in category 3 (hypertension stage 1) and lowest percentage of subjects in category 2 (elevated). The actual number of samples in group 1-4 are 261, 189, 351, 244 respectively. I have enough samples for each group. Therefore, merging of categories is not necessary.

8.1.2 My Planned Predictors (Categorical Outcome)

- age has 29 distinct values, and is measured in years.
- sex has two distinct values 1 for male 2 for female.
- bmi has 285 distinct values, measured in kg/m^2.
- sleep has 22 distinct values, measured in hours per day.
- chol has 204 distinct values, measured in mg/dL.
- inact has 33 distinct values, measured in minutes per day
- smoke has three distinct categories 1 for smoke everyday, 2 for smoke sometimes , 3 for never.

8.1.3 My Anticipated Outcome

I expect that the odds of being in lower blood pressure group is associated with younger age, with being female, with lower bmi, with more sleeping hours, with low cholesterol, with low inactive minutes, and with no smoking.

8.2 Ordinal Logistic Regression Model

8.2.1 Missingness

Lets check the complete cases and missingness in the data

```
n_case_complete(df2)
```

```
[1] 969
```

```
miss_var_summary(df2) |> filter(n_miss > 0)
```

```
# A tibble: 4 x 3
  variable n_miss pct_miss
  <chr>     <int>    <num>
1 chol         61    5.84
2 hdl          61    5.84
3 sleep        11    1.05
4 bmi           8    0.766
```

8.2.2 Single Imputation Approach

I assume data are missing at random. I used simple imputation approach using mice package and the method of predictive mean matching. I further checked the summary of missing variable, which shows there are no missing values.

```
set.seed(4325)

df2_imp <- complete(mice(df2 , m = 1, method = "pmm"))     # Predictive mean matching impu
```

```
 iter imp variable
  1   1  bmi  sleep  chol  hdl
  2   1  bmi  sleep  chol  hdl
  3   1  bmi  sleep  chol  hdl
  4   1  bmi  sleep  chol  hdl
  5   1  bmi  sleep  chol  hdl

Warning: Number of logged events: 1
```

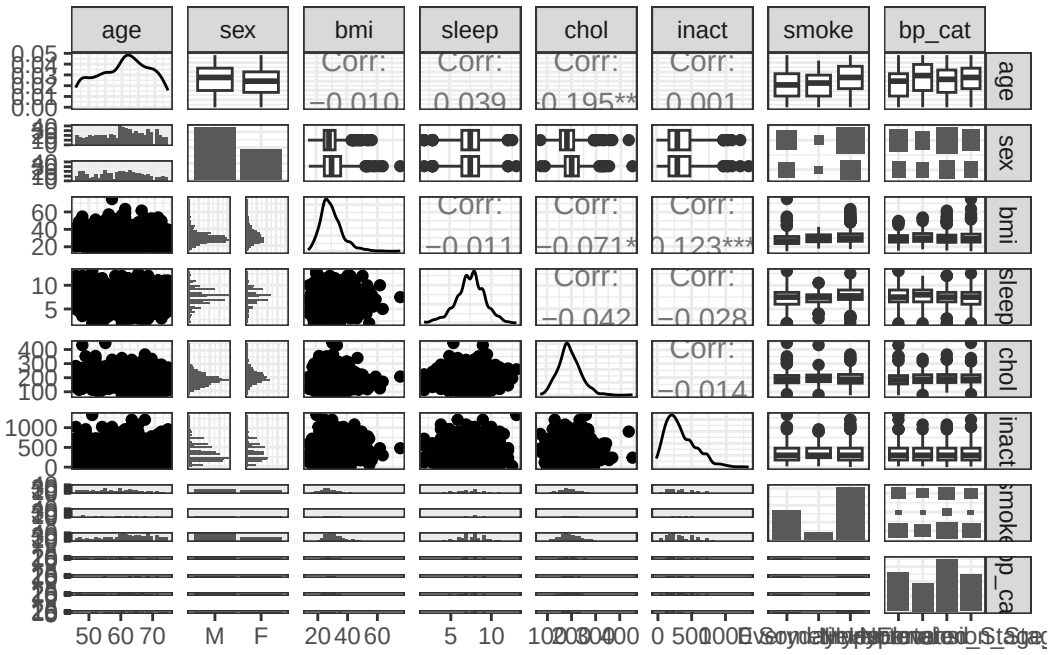
```r
miss_var_summary(df2_imp) |> filter(n_miss > 0)
```

```
# A tibble: 0 x 3
# i 3 variables: variable <chr>, n_miss <int>, pct_miss <num>
```

8.2.3 Scatterplot Matrix and Collinearity

```r
GGally::ggpairs(df2_imp |>
dplyr::select(age, sex, bmi, sleep, chol, inact, smoke, bp_cat))+
theme_bw()
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



The data may look little chaotic here. However, few things to note. Young people seem to be in normal bp category. There seems to be low correlation especially between bmi and inactivity and cholesterol and age. However, nothing too concerning.

In order to make sure my ordinal categorical outcome variables are ordered, I reordered them.

```
str(df2_imp$bp_cat) # to check the bp_cat variable
```

```
Factor w/ 4 levels "Normal","Elevated",..: 1 1 2 4 3 4 4 3 2 4 ...
```

```
df2_imp$bp_cat <- factor(df2_imp$bp_cat, ordered = T, levels = c('Normal', 'Elevated', 'Hy
# define reference by ensuring it is the first level of the factor

str(df2_imp$bp_cat) #ordinal factor check
```

```
Ord.factor w/ 4 levels "Normal"<"Elevated"<..: 1 1 2 4 3 4 4 3 2 4 ...
```

8.3 Splitting Data into Train and Test

I will split sample into training (70%) and testing (30%) using function from dplyr package

```r
set.seed(43223)

split_samples <- df2_imp$bp_cat |> createDataPartition(p = 0.7, list = FALSE)

df2_imp_train <- df2_imp[split_samples,]
df2_imp_test <- df2_imp[-split_samples,]

dim(df2_imp_train) #Check the dimension of splitted data.
```

```
[1] 733  11
```

```r
dim(df2_imp_test)
```

```
[1] 312  11
```

8.4 Fitting Polr Model Using Train Sample

I am running the ordinal regression model using the polr function in the MASS package on training sample. Further, the coefficients are converted into interpretable odds ratios using the exp() command.

```r
mod_polr <- polr(bp_cat ~ age + sex + bmi + sleep + chol + inact + smoke , data = df2_imp_

exp(coef(mod_polr))
```

```
      age          sexF           bmi         sleep          chol 
1.0337048     0.9414813     1.0304859     0.9162681     1.0027494 
    inact smokeSometimes     smokeNever 
0.9993735     1.3635694     0.9433968 
```

```r
exp(confint(mod_polr))
```

```
Waiting for profiling to be done...
```

```
                2.5 %    97.5 %
age         1.0158193 1.0520373
sexF        0.7076244 1.2522865
```

```
bmi             1.0106328 1.0508101
sleep           0.8460527 0.9918415
chol            0.9998036 1.0057204
inact           0.9987704 0.9999735
smokeSometimes 0.8297047 2.2453493
smokeNever      0.6996228 1.2715149
```

8.5 Tidy for Polr Model

```
tidy(mod_polr, exponentiate = TRUE, conf.int = TRUE) |> kable(digits = 3)
```

```
Warning: 'xfun::attr()' is deprecated.
Use 'xfun::attr2()' instead.
See help("Deprecated")

Warning: 'xfun::attr()' is deprecated.
Use 'xfun::attr2()' instead.
See help("Deprecated")
```

term	estimate	std.error	statistic	conf.low	conf.high	coef.type
age	1.034	0.009	3.710	1.016	1.052	coefficient
sexF	0.941	0.146	-0.414	0.708	1.252	coefficient
bmi	1.030	0.010	3.024	1.011	1.051	coefficient
sleep	0.916	0.041	-2.158	0.846	0.992	coefficient
chol	1.003	0.002	1.821	1.000	1.006	coefficient
inact	0.999	0.000	-2.008	0.999	1.000	coefficient
smokeSometimes	1.364	0.254	1.223	0.830	2.245	coefficient
smokeNever	0.943	0.152	-0.382	0.700	1.272	coefficient
Normal|Elevated	4.076	0.797	1.764	NA	NA	scale
Elevated|Hypertension_Stage_1	9.507	0.799	2.820	NA	NA	scale
Hypertension_Stage_1|Hypertension_Stage_2	23.844	0.805	4.669	NA	NA	scale

My model predicts that other variables remaining constant, if Harry is one year older than sally, he will have 1.03 (95% CI 1.02, 1.05) the odds of sally to be in elevated blood pressure categories. Therefore an increase in age is associated with poor blood pressure categories (higher order). My model predicts that other variables remaining constant, if Harry sleeps one hour longer than sally, he will have 0.92 (95% CI 0.85, 0.99) the odds of sally to be in elevated blood pressure categories. Therefore an increase in sleeping hour is associated with improved blood pressure categories.

The usability of a proportional odds logistic regression model depends on the assumption that each input variable has a similar effect on the different levels of the ordinal outcome variable. To test the proportional odds assumption, I used brant package.

```
brant(mod_polr)
```

```
------------------------------------------
Test for    X2  df  probability
------------------------------------------
Omnibus     20.83   16  0.19
age     11.62   2   0
sexF        2.91    2   0.23
bmi     3.99    2   0.14
sleep       0.74    2   0.69
chol        1.36    2   0.51
inact       0.05    2   0.97
smokeSometimes  1   2   0.61
smokeNever  1.89    2   0.39
------------------------------------------

H0: Parallel Regression Assumption holds
```

A low p-value in a Brant-Wald test is an indicator that the coefficient does not satisfy the proportional odds assumption. Here my p-value (0.19) is greater than 0.05 which suggests that there is some evidence that the assumption of proportional odds is satisfied by the model. Lets see now how the multinational model fits.

8.6 Running Multinomial Model

Since my output variables are already ordered, I do not have to relevel.

```
mod_mno <- multinom(bp_cat ~ age + sex + bmi + sleep + chol + inact + smoke , data = df2_i
```

```
# weights:  40 (27 variable)
initial  value 1016.153767
iter  10 value 996.390777
iter  20 value 980.276934
iter  30 value 972.309509
final  value 972.308645
converged
```

```
mod_mno
```

```
Call:
multinom(formula = bp_cat ~ age + sex + bmi + sleep + chol +
    inact + smoke, data = df2_imp_train)

Coefficients:
                     (Intercept)        age        sexF        bmi       sleep
Elevated               -5.586786 0.05795039  0.02438716 0.04697949 -0.01881318
Hypertension_Stage_1   -2.556055 0.02899383 -0.28028697 0.03385670 -0.04299094
Hypertension_Stage_2   -5.212750 0.06721436  0.02058364 0.05529590 -0.13859147
                            chol         inact smokeSometimes  smokeNever
Elevated              0.003957480 -0.0001737078      0.3622167 -0.35416423
Hypertension_Stage_1 0.003984759 -0.0005591388      0.6072883 -0.09985433
Hypertension_Stage_2 0.004459329 -0.0009126791      0.4598295 -0.19856287

Residual Deviance: 1944.617
AIC: 1998.617
```

```
exp(coef(mod_mno))
```

```
                     (Intercept)      age      sexF      bmi     sleep     chol
Elevated              0.003747051 1.059662 1.0246870 1.048101 0.9813627 1.003965
Hypertension_Stage_1 0.077610283 1.029418 0.7555669 1.034436 0.9579201 1.003993
Hypertension_Stage_2 0.005446674 1.069525 1.0207969 1.056853 0.8705836 1.004469
                         inact smokeSometimes smokeNever
Elevated              0.9998263       1.436510  0.7017597
Hypertension_Stage_1 0.9994410       1.835447  0.9049692
Hypertension_Stage_2 0.9990877       1.583804  0.8199082
```

My multinomial model predicts that for one year increase in age, the odds of being in elevated blood pressure increases by 1.06 (95% CI 1.04, 1.09) vs being in normal blood pressure if other variables remain constant.

8.7 Tidy for Multinomial Model

```
tidy(mod_mno, exponentiate = TRUE, conf.int = TRUE) |> kable(digits = 3)
```

```
Warning: 'xfun::attr()' is deprecated.
Use 'xfun::attr2()' instead.
See help("Deprecated")

Warning: 'xfun::attr()' is deprecated.
Use 'xfun::attr2()' instead.
See help("Deprecated")
```

y.level	term	estimate	std.error	statistic	p.value	conf.low	conf.high
Elevated	(Intercept)	0.004	0.763	-7.324	0.000	0.001	0.017
Elevated	age	1.060	0.012	4.666	0.000	1.034	1.086
Elevated	sexF	1.025	0.249	0.098	0.922	0.629	1.669
Elevated	bmi	1.048	0.016	2.884	0.004	1.015	1.082
Elevated	sleep	0.981	0.066	-0.285	0.776	0.862	1.117
Elevated	chol	1.004	0.002	1.688	0.091	0.999	1.009
Elevated	inact	1.000	0.001	-0.341	0.733	0.999	1.001
Elevated	smokeSometimes	1.437	0.467	0.776	0.438	0.576	3.584
Elevated	smokeNever	0.702	0.258	-1.371	0.171	0.423	1.165
Hypertension_Stage1	(Intercept)	0.078	0.729	-3.507	0.000	0.019	0.324
Hypertension_Stage1	age	1.029	0.011	2.652	0.008	1.008	1.052
Hypertension_Stage1	sexF	0.756	0.215	-1.301	0.193	0.495	1.152
Hypertension_Stage1	bmi	1.034	0.015	2.288	0.022	1.005	1.065
Hypertension_Stage1	sleep	0.958	0.057	-0.750	0.453	0.856	1.072
Hypertension_Stage1	chol	1.004	0.002	1.938	0.053	1.000	1.008
Hypertension_Stage1	inact	0.999	0.000	-1.274	0.203	0.999	1.000
Hypertension_Stage1	smokeSometimes	1.835	0.403	1.505	0.132	0.832	4.047
Hypertension_Stage1	smokeNever	0.905	0.222	-0.449	0.653	0.585	1.399
Hypertension_Stage2	(Intercept)	0.005	0.767	-6.794	0.000	0.001	0.025
Hypertension_Stage2	age	1.070	0.012	5.583	0.000	1.045	1.095
Hypertension_Stage2	sexF	1.021	0.236	0.087	0.931	0.642	1.622
Hypertension_Stage2	bmi	1.057	0.016	3.566	0.000	1.025	1.089
Hypertension_Stage2	sleep	0.871	0.063	-2.200	0.028	0.769	0.985
Hypertension_Stage2	chol	1.004	0.002	1.992	0.046	1.000	1.009
Hypertension_Stage2	inact	0.999	0.001	-1.809	0.070	0.998	1.000
Hypertension_Stage2	smokeSometimes	1.584	0.449	1.024	0.306	0.657	3.819
Hypertension_Stage2	smokeNever	0.820	0.247	-0.804	0.422	0.505	1.331

8.8 Comparing AIC and BIC of Proportional Odd or Multinomial logit models

```
AIC(mod_polr)
```

```
[1] 1986.778
```

```
AIC(mod_mno)
```

```
[1] 1998.617
```

```
BIC(mod_polr)
```

```
[1] 2037.346
```

```
BIC(mod_mno)
```

```
[1] 2122.74
```

```
compare <- data.frame(Model = c("Proportional Odds", "Multinomial"),
  AIC = c(1986.778, 1998.617),
  BIC = c(2037.346, 2122.74))

compare |> kable(digits = 2)
```

```
Warning: 'xfun::attr()' is deprecated.
Use 'xfun::attr2()' instead.
See help("Deprecated")

Warning: 'xfun::attr()' is deprecated.
Use 'xfun::attr2()' instead.
See help("Deprecated")
```

Model	AIC	BIC
Proportional Odds	1986.78	2037.35
Multinomial	1998.62	2122.74

Since AIC and BIC of proportional odd model is smaller than the multinomial model, proportional odd model is our preferred model. This is consistent with meeting the assumption of proportional odds shown by Brant Test.

Now my preferred model is ordinal logistic model.

8.9 Evaluating the ordinal logistic model Model

8.9.1 Prediction Accuracy of the Model Using Train Data

```
pred_train <- predict(mod_polr, df2_imp_train)
```

8.9.2 Confusion Matrix and Accuracy of Train Data

```
con_mat_train <- table(pred_train, df2_imp_train$bp_cat)
con_mat_train
```

```
pred_train             Normal Elevated Hypertension_Stage_1
  Normal                   64       19                    43
  Elevated                  0        0                     0
  Hypertension_Stage_1    117      104                   194
  Hypertension_Stage_2      2       10                     9

pred_train             Hypertension_Stage_2
  Normal                                 21
  Elevated                                0
  Hypertension_Stage_1                  134
  Hypertension_Stage_2                   16
```

```
sum(diag(con_mat_train))/sum(con_mat_train)
```

```
[1] 0.3738063
```

8.9.3 Prediction Accuracy of the Model Using Test Data

```
pred_test <- predict(mod_polr, df2_imp_test)
```

8.9.4 Confusion Matrix and Accuracy of Test Data

```
con_mat_test <- table(pred_test, df2_imp_test$bp_cat)
con_mat_test
```

```
pred_test              Normal Elevated Hypertension_Stage_1
  Normal                   19        5                   15
  Elevated                  0        0                    0
  Hypertension_Stage_1     55       48                   83
  Hypertension_Stage_2      4        3                    7

pred_test              Hypertension_Stage_2
  Normal                                 14
  Elevated                                0
  Hypertension_Stage_1                   57
  Hypertension_Stage_2                    2
```

```
sum(diag(con_mat_test))/sum(con_mat_test)
```

```
[1] 0.3333333
```

The prediction accuracy of the training sample is 37% and test sample is 33%. The model seemed to be poorly fitting here and doesn't seem to predict elevated blood pressure group well.

8.10 Using Lrm for Proportional Odds Logistic Regression on Train Sample

```
d <- datadist(df2_imp_train)
options(datadist = "d")
mod_lrm <- lrm(bp_cat ~ age + sex + bmi + sleep + chol + inact + smoke , data = df2_imp_tr
```

8.10.1 Output of Lrm Model

```
mod_lrm
```

```
Logistic Regression Model

lrm(formula = bp_cat ~ age + sex + bmi + sleep + chol + inact +
    smoke, data = df2_imp_train, x = T, y = T)


Frequencies of Responses

              Normal             Elevated Hypertension_Stage_1
                 183                  133                   246
Hypertension_Stage_2
                 171

                      Model Likelihood      Discrimination    Rank Discrim.
                            Ratio Test             Indexes          Indexes
Obs           733    LR chi2      32.06      R2       0.046    C       0.584
max |deriv| 3e-11    d.f.             8      R2(8,733)0.032    Dxy     0.169
                     Pr(> chi2) <0.0001    R2(8,680.2)0.035    gamma   0.169
                                           Brier    0.241    tau-a   0.125

                        Coef    S.E.   Wald Z Pr(>|Z|)
y>=Elevated             -1.4052 0.7963 -1.76  0.0776
y>=Hypertension_Stage_1 -2.2521 0.7984 -2.82  0.0048
y>=Hypertension_Stage_2 -3.7576 0.8044 -4.67  <0.0001
age                      0.0331 0.0089  3.71  0.0002
sex=F                   -0.0603 0.1456 -0.41  0.6787
bmi                      0.0300 0.0099  3.02  0.0025
sleep                   -0.0874 0.0405 -2.16  0.0310
chol                     0.0027 0.0015  1.83  0.0679
inact                   -0.0006 0.0003 -2.04  0.0411
smoke=Sometimes          0.3101 0.2536  1.22  0.2214
smoke=Never             -0.0583 0.1523 -0.38  0.7021
```

My model has pretty poor C-statistics (0.58) and Somer's Dxy (0.17), which suggest very low predictive performance. From the Wald test, it appears that age, bmi, sleep, inact adds significantly detectable value to the model.

8.10.2 Effect size of the Lrm Model

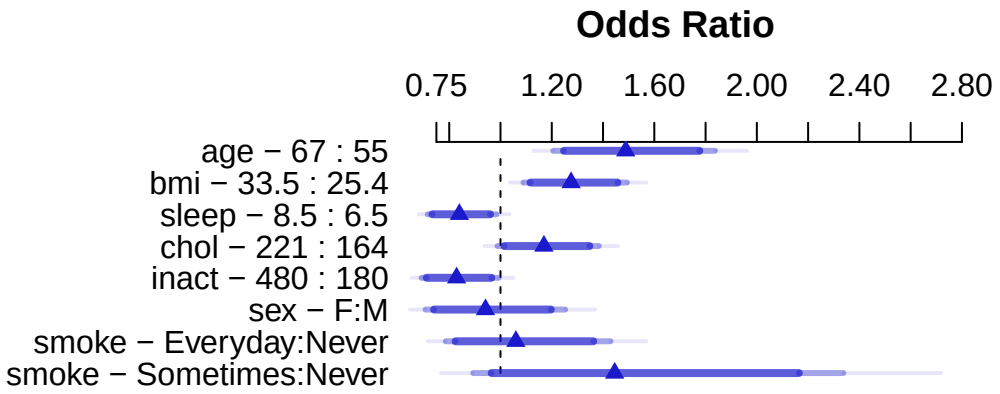
```
summary(mod_lrm)
```

```
            Effects              Response : bp_cat

Factor                  Low   High  Diff. Effect    S.E.     Lower 0.95
age                      55.0  67.0  12.0  0.397800 0.107190  0.187710
 Odds Ratio              55.0  67.0  12.0  1.488500       NA  1.206500
bmi                      25.4  33.5   8.1  0.243250 0.080422  0.085622
 Odds Ratio              25.4  33.5   8.1  1.275400       NA  1.089400
sleep                     6.5   8.5   2.0 -0.174890 0.081055 -0.333750
 Odds Ratio               6.5   8.5   2.0  0.839550       NA  0.716230
chol                    164.0 221.0  57.0  0.156500 0.085710 -0.011491
 Odds Ratio             164.0 221.0  57.0  1.169400       NA  0.988580
inact                   180.0 480.0 300.0 -0.187990 0.092054 -0.368410
 Odds Ratio             180.0 480.0 300.0  0.828620       NA  0.691830
sex - F:M                 1.0   2.0    NA -0.060302 0.145560 -0.345590
 Odds Ratio               1.0   2.0    NA  0.941480       NA  0.707810
smoke - Everyday:Never    3.0   1.0    NA  0.058272 0.152340 -0.240300
 Odds Ratio               3.0   1.0    NA  1.060000       NA  0.786390
smoke - Sometimes:Never   3.0   2.0    NA  0.368380 0.245280 -0.112360
 Odds Ratio               3.0   2.0    NA  1.445400       NA  0.893720
Upper 0.95
 0.607890
 1.836500
 0.400870
 1.493100
-0.016026
 0.984100
 0.324490
 1.383300
-0.007567
 0.992460
 0.224980
 1.252300
 0.356850
 1.428800
 0.849130
 2.337600
```

8.10.3 Effect size plot of the LRM model

```
plot(summary(mod_lrm))
```



Interpretation for the age variable: Summary plot suggest that an increase in age from 55 to 67 is 1.49 (95% CI 1.21, 1.83) times the odds of being in elevated blood pressure category compared to normal blood pressure category if other variables in the model remain constant.

8.10.4 Validation of the Lrm Model

I used bootstrap validation method using default parameters

```
set.seed(4325); validate(mod_lrm)
```

```
          index.orig training    test optimism index.corrected  n
Dxy           0.1689   0.2031  0.1537   0.0494          0.1195 40
R2            0.0458   0.0654  0.0377   0.0276          0.0182 40
Intercept     0.0000   0.0000  0.0629  -0.0629          0.0629 40
Slope         1.0000   1.0000  0.7557   0.2443          0.7557 40
Emax          0.0000   0.0000  0.0732   0.0732          0.0732 40
```

```
D              0.0424   0.0618  0.0346   0.0272          0.0152 40
U             -0.0027  -0.0027 -1.3279   1.3252         -1.3279 40
Q              0.0451   0.0645  1.3625  -1.2980          1.3431 40
B              0.2408   0.2377  0.2438  -0.0061          0.2469 40
g              0.4362   0.5221  0.3918   0.1303          0.3059 40
gp             0.1040   0.1225  0.0938   0.0287          0.0753 40
```

```
C_statiscic <-  print(0.5+.1195/2)
```

```
[1] 0.55975
```

My validated proportional odds model using LRM has Nagelskerke (R^2) of 0.018 and C-statistics of 0.559 with Somer's D value of 0.119. The model is fitting very poorly.

9 Analysis 2

9.1 My Second Research Question

How well can we can we predict chest pain or discomfort in chest using age, sex, bmi, sleep hour, hdl level, sedentary minutes, and smoking status in a sample of 1045 NHANES participants ages 45-75?

9.2 My Categorical Outcome

- My categorical outcome is chst_pain and I am predicting this value using other demographic and risk factors.
- I have a complete data in bp_cat for all 1045 of my subjects.

9.3 My Planned Predictors (Categorical Outcome)

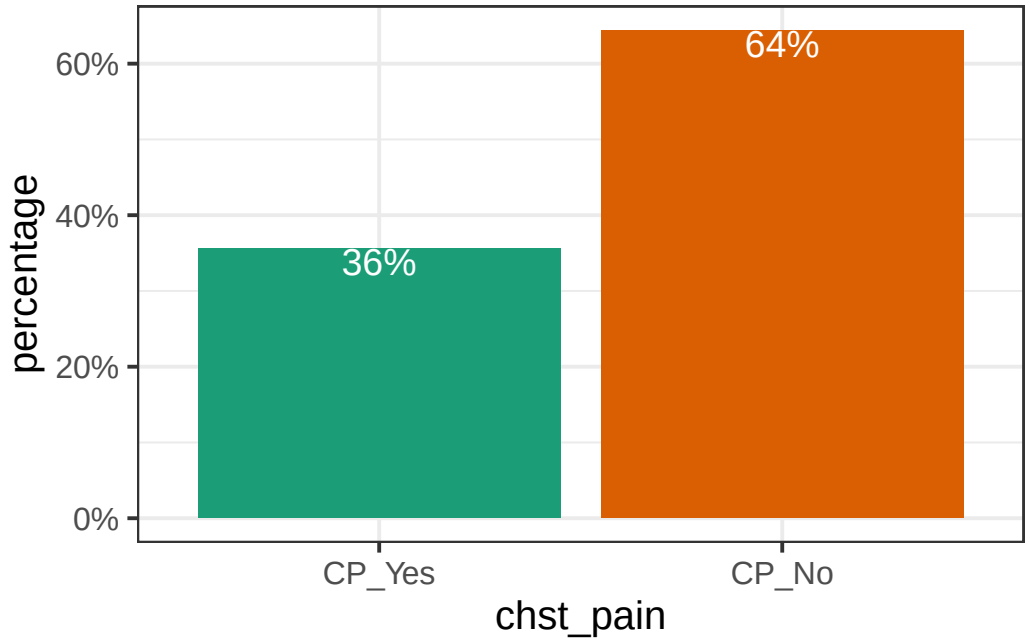
- age has 29 distinct values, and is measured in years.
- sex has two distinct values 1 for male 2 for female.
- bmi has 285 distinct values, measured in kg/m^2.
- sleep has 22 distinct values, measured in hours per day.
- hdl has 83 distinct values, measured in mg/dL.
- inact has 33 distinct values, measured in minutes per day
- smoke has three distinct categories 1 for smoke everyday, 2 for smoke sometimes , 3 for never.

9.4 My Anticipated Outcome

I expect that the odds of chest pain is associated with older age, with being male, with higher bmi, with less sleeping hours, with low hdl, with high inactive minutes, and with smoking.

Lets check the distribution of samples across my chst_pain categories.

```
ggplot(df2, aes(x = chst_pain, fill = chst_pain)) +
  geom_bar(aes(y = (after_stat(count))/sum(after_stat(count)))) +
  geom_text(aes(y = (after_stat(count))/sum(after_stat(count)),
label = scales::percent((after_stat(count)) / sum(after_stat(count)))),
              stat = "count", vjust = 1,
color = "white", size = 5) + scale_y_continuous(labels = scales::percent) + scale_fill_bre
labs(y = "percentage")
```



The histogram shows that we have ~36 percent of subjects (372) who have chest pain and ~64 (673) percent of subjects who didn't have any chest pain.

9.5 Prepare My Outcome

we want our binary outcome to be a factor variable.

```
str(df2$chst_pain)
```

```
Factor w/ 2 levels "CP_Yes","CP_No": 2 1 2 2 1 1 1 2 2 1 ...
```

```
df2 |> tabyl(chst_pain)
```

```
chst_pain   n   percent
   CP_Yes 372 0.3559809
    CP_No 673 0.6440191
```

We have ~36% in chest pain categories and ~64% in no chest pain categories in both testing and training samples.

9.6 Checking Proper Order of Outcome Variable

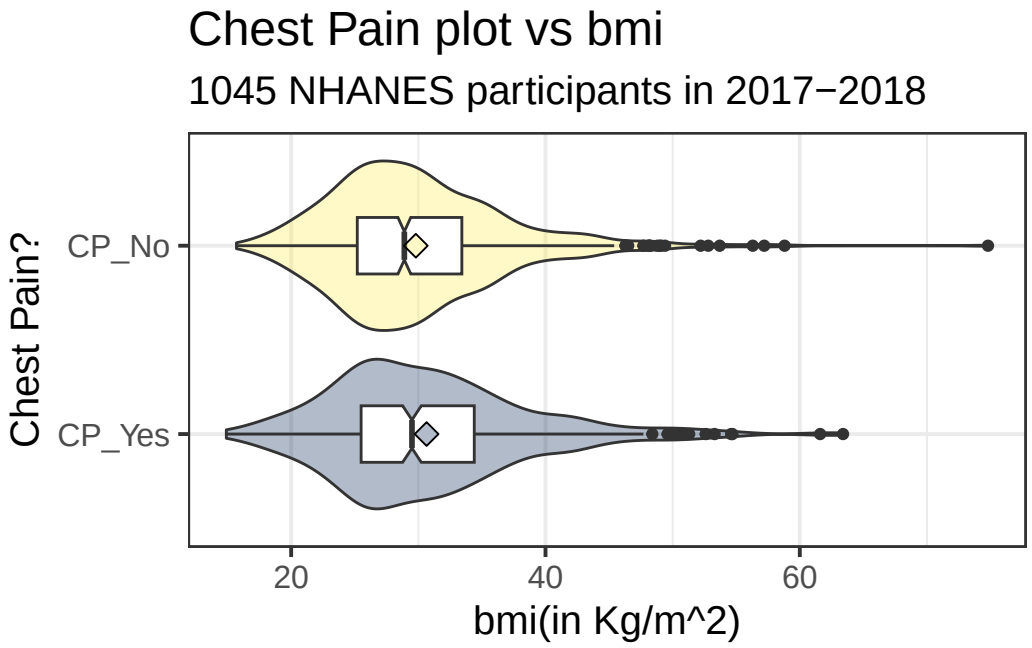
Proper re leveling of outcome variable is necessary for stan modeling. Let's check bmi values across chst_pain categories.

```
dat <- df2 |> dplyr::select(bmi, chst_pain)
  ggplot(dat, aes(x = factor(chst_pain), y = bmi)) +
    geom_violin(aes(fill = factor(chst_pain))) +
    geom_boxplot(width = 0.3, notch = TRUE) +
    stat_summary(aes(fill = factor(chst_pain)), fun = "mean", geom = "point",
                 shape = 23, size = 3) +
    guides(fill = "none", col = "none") +
    scale_fill_viridis_d(option = "cividis", alpha = 0.3) +
    coord_flip() +
    labs(x = "Chest Pain?",
         y = "bmi(in Kg/m^2)",
         title = "Chest Pain plot vs bmi",
         subtitle = glue(nrow(dat), " NHANES participants in 2017-2018"))
```

```
Warning: Removed 8 rows containing non-finite outside the scale range
(`stat_ydensity()`).
```

```
Warning: Removed 8 rows containing non-finite outside the scale range
(`stat_boxplot()`).
```

```
Warning: Removed 8 rows containing non-finite outside the scale range
(`stat_summary()`).
```



bmi is higher in chest pain group and lower in no chest pain group. This suggests that increase in bmi is associated with the increased odds of chest pain or odds should be greater than one.

Lets look at the chest pain prediction using only bmi.

```
mage_1 <- glm(chst_pain ~ bmi, family = binomial,
              data = df2)
tidy(mage_1) |> kable(digits = 3)
```

```
Warning: 'xfun::attr()' is deprecated.
Use 'xfun::attr2()' instead.
See help("Deprecated")

Warning: 'xfun::attr()' is deprecated.
Use 'xfun::attr2()' instead.
See help("Deprecated")
```

term	estimate	std.error	statistic	p.value
(Intercept)	1.082	0.279	3.886	0.00
bmi	-0.016	0.009	-1.811	0.07

```
tidy(mage_1, exponentiate = TRUE) |> kable(digits = 3)
```

```
Warning: 'xfun::attr()' is deprecated.
Use 'xfun::attr2()' instead.
See help("Deprecated")

Warning: 'xfun::attr()' is deprecated.
Use 'xfun::attr2()' instead.
See help("Deprecated")
```

term	estimate	std.error	statistic	p.value
(Intercept)	2.952	0.279	3.886	0.00
bmi	0.984	0.009	-1.811	0.07

The model is predicting that the odds of chest pain is lower than one with the increase in bmi. However, based on the violin plot above, it should be higher. Therefore, I have to relevel the chst_pain outcome variable.

I will create df3 with releveled chst_pain. Let's check the level of chst_pain first.

```
str(df2$chst_pain) #Check for levels
```

```
Factor w/ 2 levels "CP_Yes","CP_No": 2 1 2 2 1 1 1 2 2 1 ...
```

```
df3 <- df2 |> mutate(chst_pain = fct_relevel(chst_pain, "CP_No", "CP_Yes")) #Relevel
str(df3$chst_pain) #Check for relevel
```

```
Factor w/ 2 levels "CP_No","CP_Yes": 1 2 1 1 2 2 2 1 1 2 ...
```

9.7 Split df3 into Train and Test

I will split df3 based on chst_pain reference.

```r
set.seed(4321)
df3_splits <- initial_split(df3, prop = 0.7, strata = chst_pain)
df3_train <- training(df3_splits)
df3_test <- testing(df3_splits)
```

9.8 Check Stratification

Lets check if the splitting of the data worked.

```r
df3_train |> tabyl(chst_pain)
```

```
chst_pain   n   percent
    CP_No 471 0.6443228
   CP_Yes 260 0.3556772
```

```r
df3_test |> tabyl(chst_pain)
```

```
chst_pain   n   percent
    CP_No 202 0.6433121
   CP_Yes 112 0.3566879
```

9.9 Build a Recipe for My Model

```r
df3_rec <- recipe(chst_pain ~ age + sex + bmi + sleep + hdl + inact + smoke, data = df3) |
    step_impute_bag(all_predictors()) |>
    step_dummy(all_nominal(), -all_outcomes()) |>
    step_normalize(all_predictors())
```

While building a recipe, I specified an output variable, imputed all variables, and created dummy variable and normalized all predictors.

9.10 Specify the Engine for My fit

```r
df3_glm_model <- logistic_reg() |> set_engine("glm")

prior_dist <- rstanarm::normal(0, 3)
```

```
df3_stan_model <- logistic_reg() |> set_engine("stan", prior_intercept = prior_dist, prior
```

9.11 Creating Workflow to Fit Models

```
df3_glm_wf <- workflow() |>
    add_model(df3_glm_model) |>
    add_recipe(df3_rec)
df3_stan_wf <- workflow() |>
    add_model(df3_stan_model) |>
    add_recipe(df3_rec)
```

9.12 Fit Glm and Stan Model

```
fit_glm <- fit(df3_glm_wf, df3_train)
set.seed(432)
fit_stan <- fit(df3_stan_wf, df3_train)
```

9.13 Tied Coefficeint in Log Odds Scale for Glm Model

```
glm_tidy <- tidy(fit_glm, conf.int = T) |>
    mutate(modname = "glm")
stan_tidy <- broom.mixed::tidy(fit_stan, conf.int = T) |>
    mutate(modname = "stan")
coefs_comp <- bind_rows(glm_tidy, stan_tidy)
coefs_comp
```

```
# A tibble: 18 x 8
   term        estimate std.error statistic   p.value conf.low conf.high modname
   <chr>          <dbl>     <dbl>     <dbl>     <dbl>    <dbl>     <dbl> <chr>
 1 (Intercept)  -0.608     0.0784    -7.75   8.99e-15  -0.763   -0.455   glm
 2 age           0.0240    0.0798     0.301  7.63e- 1  -0.132    0.181   glm
 3 bmi           0.146     0.0847     1.72   8.54e- 2  -0.0206   0.312   glm
 4 sleep         0.0930    0.0790     1.18   2.39e- 1  -0.0614   0.249   glm
 5 hdl          -0.111     0.0907    -1.22   2.22e- 1  -0.293    0.0634  glm
 6 inact        -0.0375    0.0802    -0.468  6.40e- 1  -0.196    0.119   glm
 7 sex_F         0.114     0.0861     1.32   1.86e- 1  -0.0553   0.283   glm
```

```
 8 smoke_Some~  -0.218     0.0913    -2.38   1.72e- 2  -0.406   -0.0450  glm
 9 smoke_Never  -0.172     0.0864    -1.99   4.67e- 2  -0.341   -0.00250 glm
10 (Intercept)  -0.614     0.0786    NA      NA         -0.750   -0.486   stan
11 age           0.0244    0.0806    NA      NA         -0.109    0.161   stan
12 bmi           0.147     0.0877    NA      NA          0.0106   0.288   stan
13 sleep         0.0963    0.0845    NA      NA         -0.0397   0.228   stan
14 hdl          -0.110     0.0901    NA      NA         -0.269    0.0345  stan
15 inact        -0.0400    0.0795    NA      NA         -0.172    0.0914  stan
16 sex_F         0.115     0.0883    NA      NA         -0.0282   0.255   stan
17 smoke_Some~  -0.222     0.0917    NA      NA         -0.380   -0.0711  stan
18 smoke_Never  -0.170     0.0890    NA      NA         -0.314   -0.0301  stan
```

9.14 Tied Coefficeint of Glm Model in Odds Scale

```
glm_odds <- glm_tidy |>
  mutate(odds = exp(estimate),
  odds_low = exp(conf.low),
  odds_high = exp(conf.high)) |>
  filter(term != "(Intercept)") |>
  dplyr::select(modname, term, odds, odds_low, odds_high)
glm_odds
```

```
# A tibble: 8 x 5
  modname term             odds odds_low odds_high
  <chr>   <chr>           <dbl>    <dbl>     <dbl>
1 glm     age             1.02     0.876     1.20
2 glm     bmi             1.16     0.980     1.37
3 glm     sleep           1.10     0.940     1.28
4 glm     hdl             0.895    0.746     1.07
5 glm     inact           0.963    0.822     1.13
6 glm     sex_F           1.12     0.946     1.33
7 glm     smoke_Sometimes 0.804    0.667     0.956
8 glm     smoke_Never     0.842    0.711     0.998
```

9.15 Tied Coefficeint of Stan Model in Odds Scale

```
stan_odds <- stan_tidy |>
  mutate(odds = exp(estimate),
  odds_low = exp(conf.low),
```

```
    odds_high = exp(conf.high)) |>
    filter(term != "(Intercept)") |>
    dplyr::select(modname, term, odds, odds_low, odds_high)
  glm_odds
```

```
# A tibble: 8 x 5
  modname term             odds odds_low odds_high
  <chr>   <chr>           <dbl>    <dbl>     <dbl>
1 glm     age             1.02     0.876     1.20
2 glm     bmi             1.16     0.980     1.37
3 glm     sleep           1.10     0.940     1.28
4 glm     hdl             0.895    0.746     1.07
5 glm     inact           0.963    0.822     1.13
6 glm     sex_F           1.12     0.946     1.33
7 glm     smoke_Sometimes 0.804    0.667     0.956
8 glm     smoke_Never     0.842    0.711     0.998
```

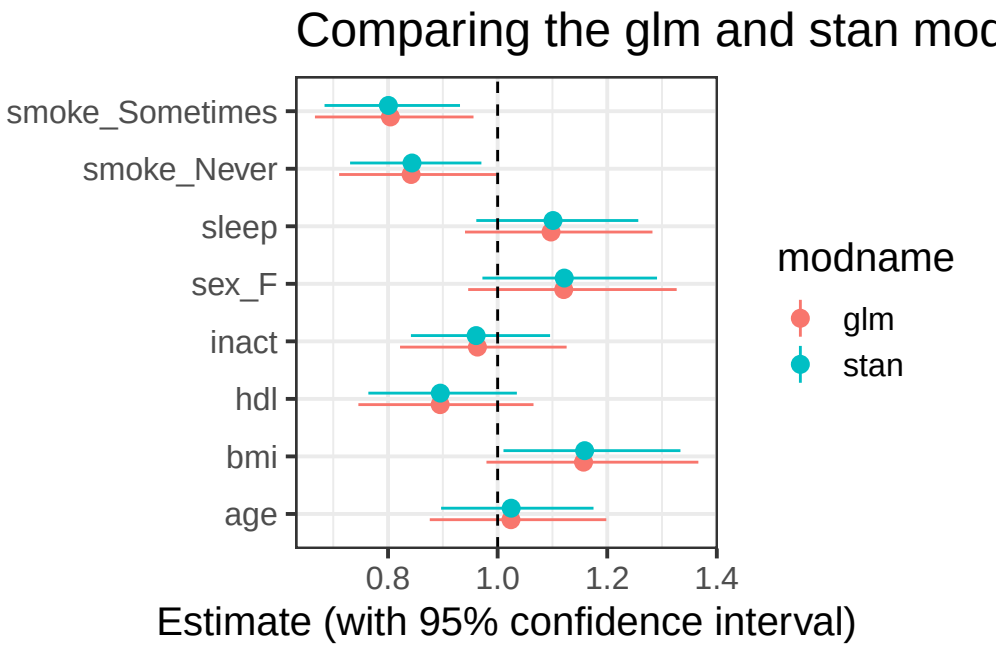
9.16 Comparison of Coefficients of Glm and Stan Model

```
  coefs_comp <- bind_rows(glm_odds, stan_odds)
  coefs_comp
```

```
# A tibble: 16 x 5
   modname term             odds odds_low odds_high
   <chr>   <chr>           <dbl>    <dbl>     <dbl>
 1 glm     age             1.02     0.876     1.20
 2 glm     bmi             1.16     0.980     1.37
 3 glm     sleep           1.10     0.940     1.28
 4 glm     hdl             0.895    0.746     1.07
 5 glm     inact           0.963    0.822     1.13
 6 glm     sex_F           1.12     0.946     1.33
 7 glm     smoke_Sometimes 0.804    0.667     0.956
 8 glm     smoke_Never     0.842    0.711     0.998
 9 stan    age             1.02     0.897     1.17
10 stan    bmi             1.16     1.01      1.33
11 stan    sleep           1.10     0.961     1.26
12 stan    hdl             0.895    0.764     1.04
13 stan    inact           0.961    0.842     1.10
14 stan    sex_F           1.12     0.972     1.29
```

```
15 stan    smoke_Sometimes 0.801    0.684     0.931
16 stan    smoke_Never     0.844    0.731     0.970
```

```r
ggplot(coefs_comp, aes(x = term, y = odds, col = modname,
                       ymin = odds_low, ymax = odds_high)) +
  geom_point(position = position_dodge2(width = 0.4)) +
  geom_pointrange(position = position_dodge2(width = 0.4)) +
  geom_hline(yintercept = 1, lty = "dashed") +
  coord_flip() +
  labs(x = "", y = "Estimate (with 95% confidence interval)",
   title = "Comparing the glm and stan model coefficients")
```



The point estimates look fairly similar between my glm and stan model, however, the glm model seem to have wider confidence interval. The odds of chest pain decreases with less smoking, increase in hdl level and increase in sedentary minutes. While the increase in the odds of chest pain is associated with older age, increase in bmi, being female and increase in sleep hours, based on point estimates.

9.17 Evaluating Train Sample Performance

9.17.1 Making Prediction with Glm Fit

```
glm_probs <- predict(fit_glm, df3_train, type = "prob") |>
    bind_cols(df3_train |> dplyr::select(chst_pain))
head(glm_probs, 5)
```

```
# A tibble: 5 x 3
  .pred_CP_No .pred_CP_Yes chst_pain
        <dbl>        <dbl> <fct>
1       0.642        0.358 CP_No
2       0.548        0.452 CP_No
3       0.652        0.348 CP_No
4       0.600        0.400 CP_No
5       0.684        0.316 CP_No
```

Next, we'll use roc_auc from yardstick. This assumes that the first level of df2_train is the thing we're trying to predict. Is that true in our case?

```
df3_train |> tabyl(chst_pain)
```

```
 chst_pain   n   percent
    CP_No 471 0.6443228
   CP_Yes 260 0.3556772
```

This is not correct. I am going to predict CP_Yes which the second level in chst_pain variable. So, I need to switch event level to second.

```
glm_probs |> roc_auc(chst_pain, .pred_CP_Yes, event_level = "second") |>
    kable(dig = 5)
```

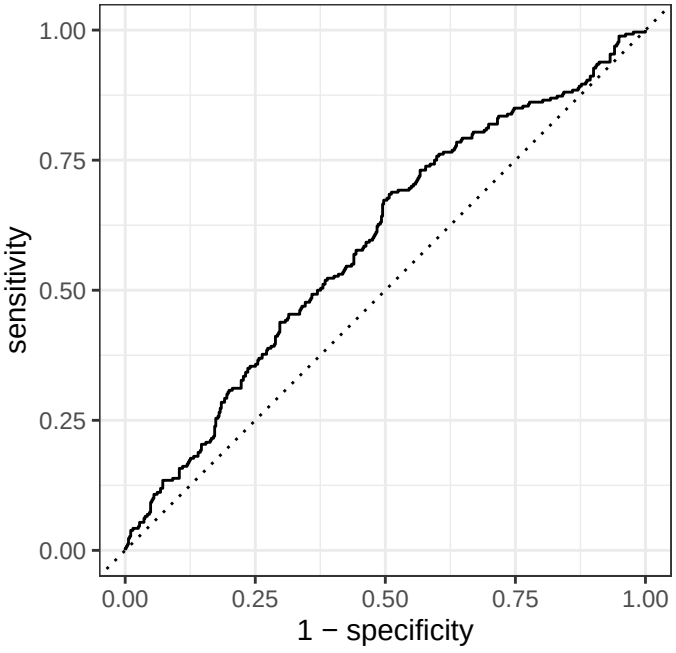
```
Warning: 'xfun::attr()' is deprecated.
Use 'xfun::attr2()' instead.
See help("Deprecated")

Warning: 'xfun::attr()' is deprecated.
Use 'xfun::attr2()' instead.
See help("Deprecated")
```

.metric	.estimator	.estimate
roc_auc	binary	0.58982

9.18 ROC curve for Glm Fit

```
glm_roc <- glm_probs |> roc_curve(chst_pain, .pred_CP_Yes, event_level = "second")
autoplot(glm_roc)
```



9.18.1 Making Prediction with Stan Fit in Train Sample

```
stan_probs <- predict(fit_stan, df3_train, type = "prob") |>
  bind_cols(df3_train |> dplyr::select(chst_pain))
  head(stan_probs, 5)
```

```
# A tibble: 5 x 3
  .pred_CP_No .pred_CP_Yes chst_pain
        <dbl>        <dbl> <fct>
1       0.644        0.356 CP_No
```

```
2       0.546        0.454 CP_No
3       0.655        0.345 CP_No
4       0.602        0.398 CP_No
5       0.686        0.314 CP_No
```

```
stan_probs |> roc_auc(chst_pain, .pred_CP_Yes, event_level = "second" ) |>
    kable(dig = 5)
```

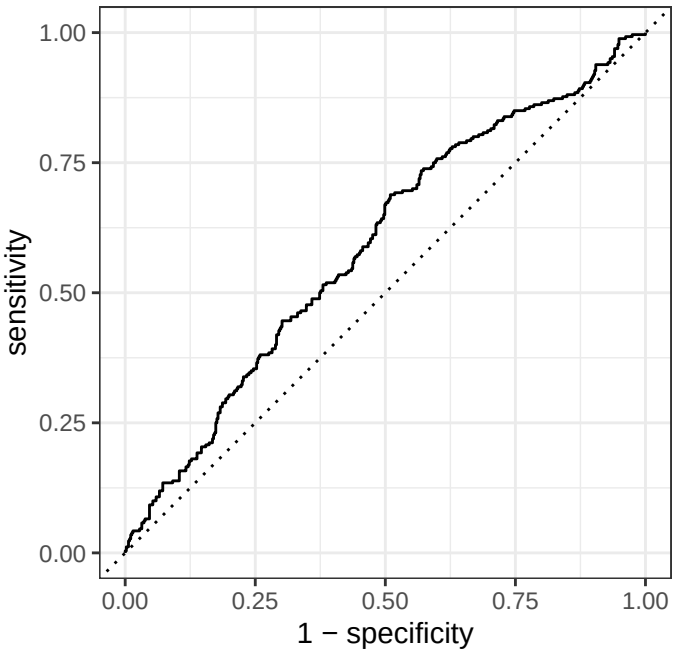
```
Warning: 'xfun::attr()' is deprecated.
Use 'xfun::attr2()' instead.
See help("Deprecated")

Warning: 'xfun::attr()' is deprecated.
Use 'xfun::attr2()' instead.
See help("Deprecated")
```

.metric	.estimator	.estimate
roc_auc	binary	0.58986

9.19 ROC curve for Stan Fit

```
stan_roc <- stan_probs |> roc_curve(chst_pain, .pred_CP_Yes, event_level = "second")
autoplot(stan_roc)
```



My C statistic for both Glm and Stan fit is also 0.589

9.20 Establishing a Decision Rule for the Glm Fit

Let's use .pred_CP_Yes > 0.35 for now to indicate a prediction of chst_pain.

```
glm_probs <- predict(fit_glm, df3_train, type = "prob") |>
    bind_cols(df3_train |> dplyr::select(chst_pain)) |>
    mutate(chst_pain_pre = ifelse(.pred_CP_Yes > 0.35, "CP_Yes", "CP_No")) |>
    mutate(chst_pain_pre = fct_relevel(factor(chst_pain_pre), "CP_No"))

glm_probs |> tabyl(chst_pain_pre, chst_pain)
```

```
chst_pain_pre CP_No CP_Yes
        CP_No   250    105
       CP_Yes   221    155
```

9.21 Confusion Matrix and Accuracy for Glm Fit

```
  conf_mat(glm_probs, truth = chst_pain, estimate = chst_pain_pre)
```

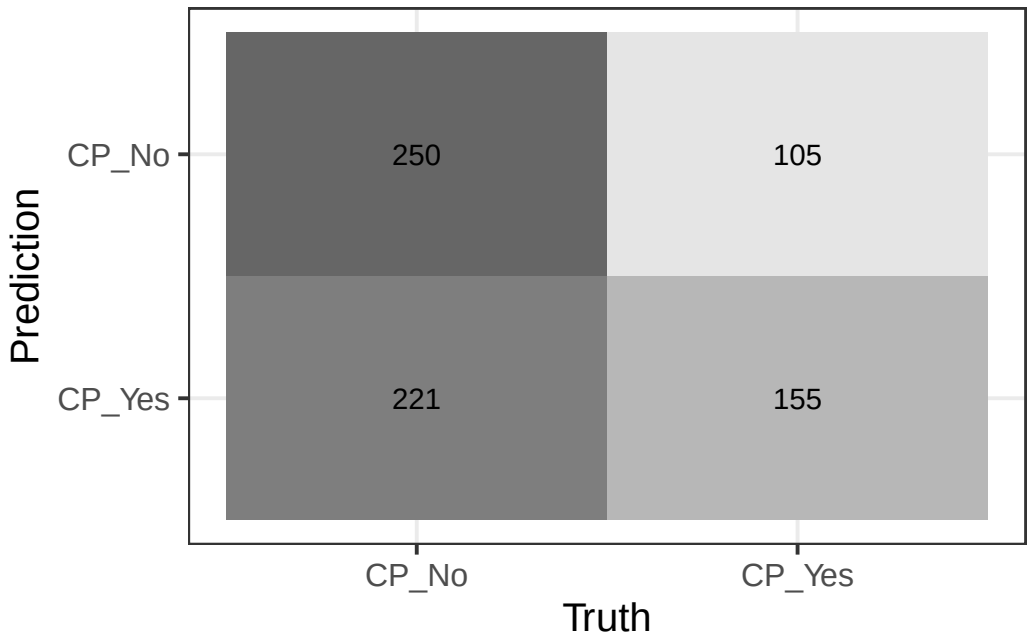
```
          Truth
Prediction CP_No CP_Yes
    CP_No    250    105
    CP_Yes   221    155
```

```
  metrics(glm_probs, truth = chst_pain, estimate = chst_pain_pre)
```

```
# A tibble: 2 x 3
  .metric  .estimator .estimate
  <chr>    <chr>          <dbl>
1 accuracy binary         0.554
2 kap      binary         0.115
```

9.22 Plot Confusion Matrix for Glm Fit

```
  conf_mat(glm_probs, truth = chst_pain, estimate = chst_pain_pre) |>
      autoplot(type = "heatmap")
```



9.23 Establishing a Decision Rule for the Stan Fit

Let's use .pred_1 > 0.35 for now to indicate a prediction of chst_pain.

```
stan_probs <- predict(fit_stan, df3_train, type = "prob") |>
    bind_cols(df3_train |> dplyr::select(chst_pain)) |>
    mutate(chst_pain_pre = ifelse(.pred_CP_Yes > 0.35, "CP_Yes", "CP_No")) |>
    mutate(chst_pain_pre = fct_relevel(factor(chst_pain_pre), "CP_No"))

stan_probs |> tabyl(chst_pain_pre, chst_pain)
```

```
chst_pain_pre CP_No CP_Yes
        CP_No   256    108
       CP_Yes   215    152
```

9.24 Confusion Matrix and Accuracy for Stan Fit

```
conf_mat(stan_probs, truth = chst_pain, estimate = chst_pain_pre)
```

```
          Truth
Prediction CP_No CP_Yes
    CP_No    256    108
    CP_Yes   215    152
```

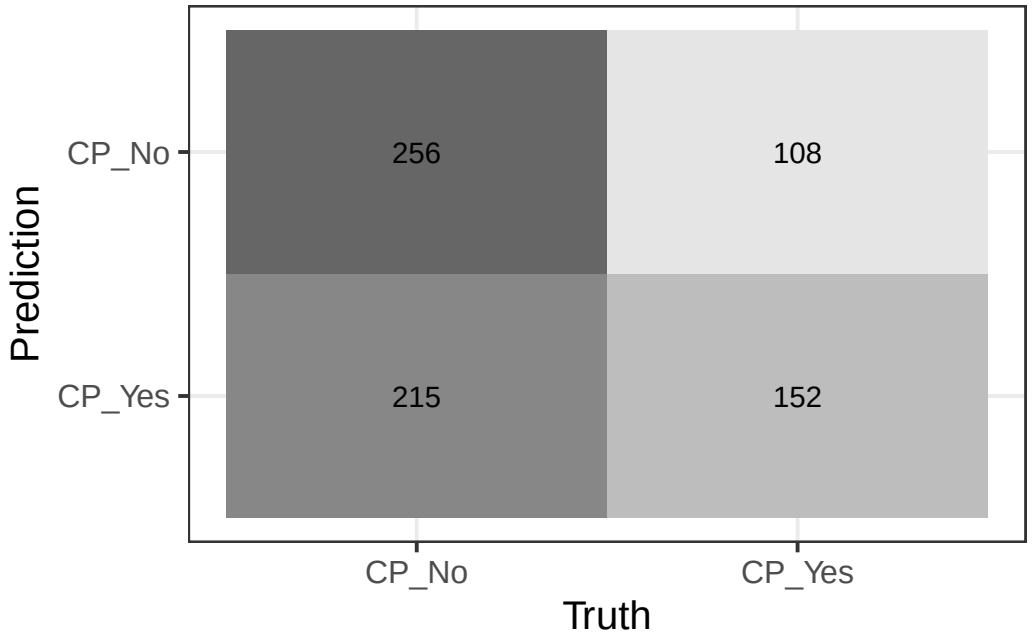
```
metrics(stan_probs, truth = chst_pain, estimate = chst_pain_pre)
```

```
# A tibble: 2 x 3
  .metric  .estimator .estimate
  <chr>    <chr>          <dbl>
1 accuracy binary         0.558
2 kap      binary         0.117
```

The accuracy of stan model does not seem to be any better than glm model in training sample (0.558 vs 0.554).

9.25 Plot Confusion Matrix for Stan Fit

```
conf_mat(stan_probs, truth = chst_pain, estimate = chst_pain_pre) |>
    autoplot(type = "heatmap")
```



9.26 Assess Test Sample Performance.

```
glm_test <- predict(fit_glm, df3_test, type = "prob") |>
    bind_cols(df3_test |> dplyr::select(chst_pain))

stan_test <- predict(fit_stan, df3_test, type = "prob") |>
  bind_cols(df3_test |> dplyr::select(chst_pain))
```

9.26.1 Test Sample C statistic comparison

```
glm_test |> roc_auc(chst_pain, .pred_CP_Yes, event_level = "second" ) |>
    kable(dig = 5)
```

```
Warning: 'xfun::attr()' is deprecated.
Use 'xfun::attr2()' instead.
See help("Deprecated")

Warning: 'xfun::attr()' is deprecated.
Use 'xfun::attr2()' instead.
See help("Deprecated")
```

.metric	.estimator	.estimate
roc_auc	binary	0.57306

```
stan_test |> roc_auc(chst_pain, .pred_CP_Yes, event_level = "second" ) |>
    kable(dig = 5)
```

```
Warning: 'xfun::attr()' is deprecated.
Use 'xfun::attr2()' instead.
See help("Deprecated")

Warning: 'xfun::attr()' is deprecated.
Use 'xfun::attr2()' instead.
See help("Deprecated")
```

.metric	.estimator	.estimate
roc_auc	binary	0.5732

C-statistics from glm fit is similar to the the C-statistics from stan fit in test sample.

9.27 Confusion Matrix and Model Accuracy for glm test sample

```
glm_test <- predict(fit_glm, df3_test, type = "prob") |>
    bind_cols(df3_test |> dplyr::select(chst_pain)) |>
    mutate(chst_pain_pre = ifelse(.pred_CP_Yes > 0.35, "CP_Yes", "CP_No")) |>
    mutate(chst_pain_pre = fct_relevel(factor(chst_pain_pre), "CP_No"))

glm_test |> tabyl(chst_pain_pre, chst_pain)
```

```
 chst_pain_pre CP_No CP_Yes
         CP_No   105     43
        CP_Yes    97     69
```

```
conf_mat(glm_test, truth = chst_pain, estimate = chst_pain_pre)
```

```
          Truth
Prediction CP_No CP_Yes
    CP_No    105     43
    CP_Yes    97     69
```

```
metrics(glm_test, truth = chst_pain, estimate = chst_pain_pre)
```

```
# A tibble: 2 x 3
  .metric  .estimator .estimate
  <chr>    <chr>          <dbl>
1 accuracy binary         0.554
2 kap      binary         0.123
```

9.28 Confusion Matrix and Model Accuracy for stan test sample

```
stan_test <- predict(fit_glm, df3_test, type = "prob") |>
    bind_cols(df3_test |> dplyr::select(chst_pain)) |>
    mutate(chst_pain_pre = ifelse(.pred_CP_Yes > 0.35, "CP_Yes", "CP_No")) |>
    mutate(chst_pain_pre = fct_relevel(factor(chst_pain_pre), "CP_No"))

stan_test |> tabyl(chst_pain_pre, chst_pain)
```

```
 chst_pain_pre CP_No CP_Yes
         CP_No   105     43
        CP_Yes    97     69
```

```
conf_mat(stan_test, truth = chst_pain, estimate = chst_pain_pre)
```

```
          Truth
Prediction CP_No CP_Yes
    CP_No    105     43
    CP_Yes    97     69
```

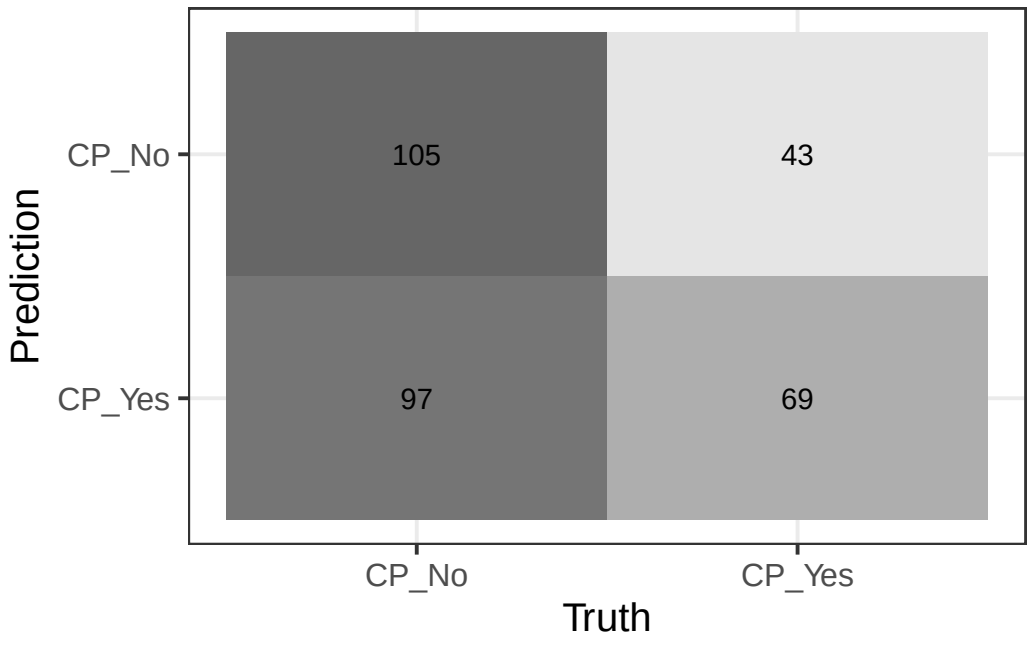
```
metrics(glm_test, truth = chst_pain, estimate = chst_pain_pre)
```

```
# A tibble: 2 x 3
  .metric  .estimator .estimate
  <chr>    <chr>          <dbl>
1 accuracy binary         0.554
2 kap      binary         0.123
```

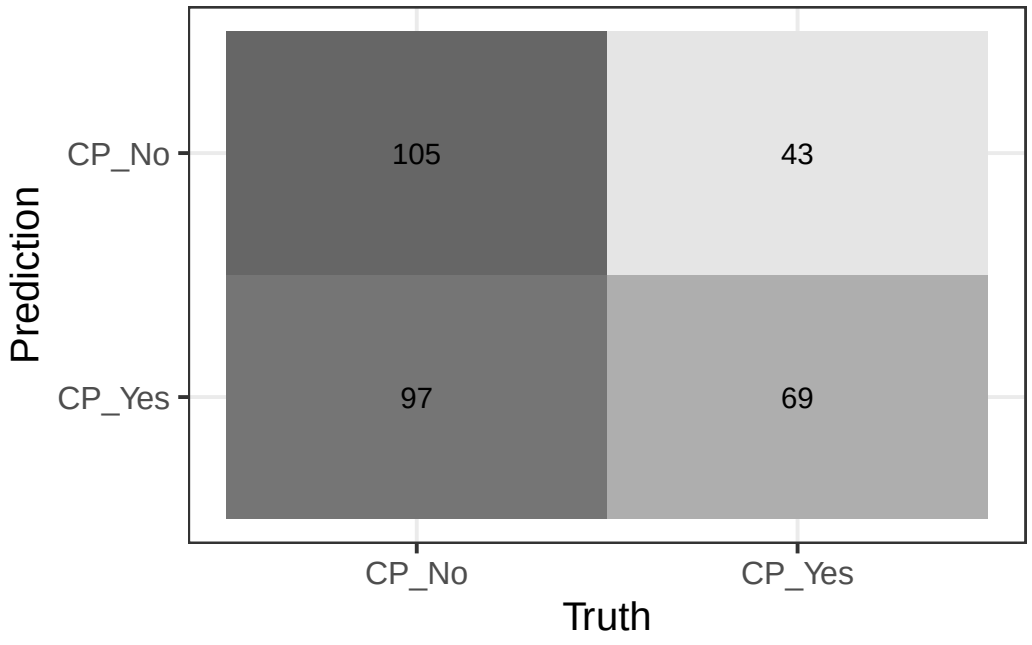
The accuracy of stan model does not seem to be any better than glm model in training sample (0.554 vs 0.554)

9.29 Plot Confusion Matrix

```
conf_mat(glm_test, truth = chst_pain, estimate = chst_pain_pre) |>
    autoplot(type = "heatmap")
```




```
conf_mat(stan_test, truth = chst_pain, estimate = chst_pain_pre) |>
    autoplot(type = "heatmap")
```



10 Conclusions and Discussion

I used proportional odds logistic model to predict blood pressure groups on NHANES 2017-2018 age 45-75 based on the given predictors age, sex, bmi, cholesterol, sleep, sedentary minute, smoking categories. My model has validated C-statistics of 0.56 and Somer's Dxy of 0.119 with Nagelkerke R2 of 0.018, which suggest very poor fitting model, slightly better than random prediction probability. My proportional odd model estimated the odds of being in poor blood pressure categories is associated with increase in age, bmi, and cholesterol. However, the odds is decreased with increase in sleep hour and sedentary minutes. Interestingly, smoking status showed decreased association with the odds of being in poor blood pressure categories with effect size including zero, meaning no difference. Further, I used Bayesian (stan) and glm model to predict chest pain outcome on NHANES 2017-2018 age 45-75 using the given predictors age, sex, bmi, hdl , sleep, sedentary minute, smoking categories. The point estimates look fairly similar between my glm and stan model, however, the glm model seem to have wider confidence interval. The odds of chest pain decreases with less smoking, increase in hdl level and increase in sedentary minutes. While the increase in the odds of chest pain is associated with older age, increase in bmi, being female and increase in sleep hours, based on point estimates. Both of my models have similar C-statistics of 0.589 with accuracy of 0.55 in both train and test sample. For the models I generated, I used main effects only. The models could benefit if I add nonlinear terms or interactions. For multicategorical prediction would have been better if I had merged elevated blood pressure with another blood pressure category as the sample size was comparatively lower in elevated blood pressure category. The model seem to fail predicting elevated blood pressure category. Addition of better predictors, for example in the case of sedentary minutes, it would have been better if I had added active hours instead. It is possible that people that are highly active can stay sedentary for longer time.

10.1 Answering My Research Questions

10.1.1 Answering My First Research Question

The increase in age, bmi, and cholesterol increases the odds of being in higher blood pressure category (poor blood pressure category) and increase in sleep and sedentary minutes decreases the odds of being in high blood pressure categories if other variables remain constant. Smoking does not seem to show much of a difference in predicting the odds of being in any blood pressure categories.

10.1.2 Answering My Second Research Question

The odds of chest pain decreases with less smoking, increase in hdl level, and increase in sedentary minutes. While the increase in the odds of chest pain is associated with older age, increase in bmi, being female and increase in sleep hours, based on point estimates.

11 References and Acknowledgments

11.1 References

1. Data Source description https://wwwn.cdc.gov/nchs/nhanes/continuousnhanes/default.aspx?BeginYear=

2. Tsao CW, Aday AW, Almarzooq ZI, Beaton AZ, Bittencourt MS, Boehme AK, et al. Heart Disease and Stroke Statistics—2022 Update: A Report From the American Heart Association. Circulation. 2022;145(8):e153–e639.

12 Session Information

```
xfun::session_info()
```

```
R version 4.3.1 (2023-06-16)
Platform: aarch64-apple-darwin20 (64-bit)
Running under: macOS Ventura 13.0

Locale: en_US.UTF-8 / en_US.UTF-8 / en_US.UTF-8 / C / en_US.UTF-8 / en_US.UTF-8

Package version:
  abind_1.4-5          askpass_1.2.0        backports_1.4.1
  base64enc_0.1-3      bayesplot_1.11.1     BH_1.84.0.0
  bigD_0.3.1           bit_4.0.5            bit64_4.0.5
  bitops_1.0.7         blob_1.2.4           boot_1.3-30
  brant_0.3-0          brio_1.1.4           broom_1.0.5
  broom.mixed_0.2.9.4  bslib_0.6.1          cachem_1.0.8
  callr_3.7.5          car_3.1.2            carData_3.0.5
  cards_0.6.0          caret_7.0-1          caTools_1.18.2
  cellranger_1.1.0     checkmate_2.3.1      class_7.3-22
  cli_3.6.5            clipr_0.8.0          clock_0.7.0
  cluster_2.1.6        coda_0.19.4.1        codetools_0.2-19
  colorspace_2.1-0     colourpicker_1.3.0   commonmark_1.9.5
  compiler_4.3.1       conflicted_1.2.0     cpp11_0.4.7
  crayon_1.5.2         crosstalk_1.2.1      curl_5.2.1
  data.table_1.15.2    DBI_1.2.2            dbplyr_2.4.0
  DEoptimR_1.1.3       desc_1.4.3           diagram_1.6.5
  dials_1.2.1          DiceDesign_1.10      diffobj_0.3.5
  digest_0.6.34        distributional_0.4.0 doRNG_1.8.6.2
  dplyr_1.1.4          DT_0.32              dtplyr_1.3.1
```

dygraphs_1.1.1.6	e1071_1.7.16	ellipsis_0.3.2
evaluate_0.23	fansi_1.0.6	farver_2.1.1
fastmap_1.1.1	fontawesome_0.5.2	forcats_1.0.0
foreach_1.5.2	foreign_0.8-86	Formula_1.2-5
fs_1.6.6	furrr_0.3.1	future_1.33.1
future.apply_1.11.1	gargle_1.5.2	gdata_3.0.1
generics_0.1.3	GGally_2.2.1	ggplot2_3.5.0
ggridges_0.5.6	ggstats_0.9.0	glmnet_4.1-8
globals_0.16.2	glue_1.8.0	gmodels_2.19.1
googledrive_2.1.1	googlesheets4_1.1.1	gower_1.0.1
GPfit_1.0-8	gplots_3.1.3.1	graphics_4.3.1
grDevices_4.3.1	grid_4.3.1	gridExtra_2.3
gt_1.0.0	gtable_0.3.4	gtools_3.9.5
gtsummary_2.2.0	hardhat_1.3.1	haven_2.5.4
highr_0.10	Hmisc_5.1-1	hms_1.1.3
htmlTable_2.4.2	htmltools_0.5.8.1	htmlwidgets_1.6.4
httpuv_1.6.14	httr_1.4.7	ids_1.0.1
igraph_2.0.2	infer_1.0.6	inline_0.3.19
ipred_0.9-14	isoband_0.2.7	iterators_1.0.14
itertools_0.1.3	janitor_2.2.0	jomo_2.7-6
jquerylib_0.1.4	jsonlite_1.8.8	juicyjuice_0.1.0
KernSmooth_2.23.22	knitr_1.45	labeling_0.4.3
laeken_0.5.3	later_1.3.2	lattice_0.22-5
lava_1.8.0	lazyeval_0.2.2	lhs_1.1.6
lifecycle_1.0.4	listenv_0.9.1	litedown_0.7
lme4_1.1-35.1	lmtest_0.9.40	loo_2.7.0
lubridate_1.9.3	magrittr_2.0.3	markdown_2.0
MASS_7.3-60.0.1	Matrix_1.6-5	MatrixModels_0.5-3
matrixStats_1.2.0	memoise_2.0.1	methods_4.3.1
mgcv_1.9.1	mice_3.18.0	mime_0.12
miniUI_0.1.1.1	minqa_1.2.6	missForest_1.5
mitml_0.4-5	modeldata_1.3.0	modelenv_0.1.1
ModelMetrics_1.2.2.2	modelr_0.1.11	multcomp_1.4-25
munsell_0.5.0	mvtnorm_1.2-4	naniar_1.1.0
nhanesA_1.3	nlme_3.1-164	nloptr_2.0.3
nnet_7.3-19	norm_1.0.11.1	numDeriv_2016.8.1.1
openssl_2.1.1	ordinal_2023.12.4.1	pan_1.9
parallel_4.3.1	parallelly_1.37.1	parsnip_1.2.0
patchwork_1.2.0	pbkrtest_0.5.2	pillar_1.9.0
pkgbuild_1.4.3	pkgconfig_2.0.3	pkgload_1.3.4
plyr_1.8.9	polspline_1.1.24	posterior_1.5.0
praise_1.0.0	prettyunits_1.2.0	pROC_1.18.5
processx_3.8.3	prodlim_2023.08.28	progress_1.2.3

progressr_0.14.0 promises_1.2.1 proxy_0.4.27
ps_1.7.6 purrr_1.0.2 quantreg_5.97
QuickJSR_1.1.3 R6_2.5.1 ragg_1.2.7
randomForest_4.7.1.2 ranger_0.17.0 rappdirs_0.3.3
RColorBrewer_1.1-3 Rcpp_1.0.12 RcppEigen_0.3.4.0.0
RcppParallel_5.1.7 reactable_0.4.4 reactR_0.6.1
readr_2.1.5 readxl_1.4.3 recipes_1.0.10
rematch_2.0.0 rematch2_2.1.2 reprex_2.1.0
reshape2_1.4.4 rlang_1.1.6 rmarkdown_2.26
rms_6.7-1 rngtools_1.5.2 robustbase_0.99.2
ROCR_1.0-11 rpart_4.1.23 rprojroot_2.0.4
rsample_1.2.0 rstan_2.32.6 rstanarm_2.32.1
rstantools_2.4.0 rstudioapi_0.15.0 rvest_1.0.4
sandwich_3.1-0 sass_0.4.10 scales_1.3.0
selectr_0.4.2 shape_1.4.6.1 shiny_1.8.0
shinyjs_2.1.0 shinystan_2.6.0 shinythemes_1.2.0
simputation_0.2.9 slider_0.3.1 snakecase_0.11.1
sourcetools_0.1.7.1 sp_2.1.3 SparseM_1.81
splines_4.3.1 SQUAREM_2021.1 StanHeaders_2.32.6
stats_4.3.1 stats4_4.3.1 stringi_1.8.3
stringr_1.5.1 survival_3.5-8 sys_3.4.2
systemfonts_1.0.5 tensorA_0.36.2.1 testthat_3.2.1
textshaping_0.3.7 TH.data_1.1-2 threejs_0.3.3
tibble_3.2.1 tidymodels_1.1.1 tidyr_1.3.1
tidyselect_1.2.1 tidyverse_2.0.0 timechange_0.3.0
timeDate_4032.109 tinytex_0.49 tools_4.3.1
tune_1.1.2 tzdb_0.4.0 ucminf_1.2.2
UpSetR_1.4.0 utf8_1.2.4 utils_4.3.1
uuid_1.2.0 V8_6.0.3 vcd_1.4.13
vctrs_0.6.5 VIM_6.2.2 viridis_0.6.5
viridisLite_0.4.2 visdat_0.6.0 vroom_1.6.5
waldo_0.5.2 warp_0.2.1 withr_3.0.0
workflows_1.1.4 workflowsets_1.0.1 xfun_0.52
xml2_1.3.6 xtable_1.8-4 xts_0.13.2
yaml_2.3.8 yardstick_1.3.0 zoo_1.8-12