Output

```
library(ranger)
```

```
Warning: package 'ranger' was built under R version 4.2.3
```

```
library(beepr)
library(ROSE)
```

```
Loaded ROSE 0.0-4
```

```
library(caret)
```

```
Loading required package: ggplot2
```

```
Warning: package 'ggplot2' was built under R version 4.2.3
```

```
Loading required package: lattice
```

```
library(ggplot2)
library(dplyr)
```

```
Warning: package 'dplyr' was built under R version 4.2.3
```

```

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(randomForest)
```

```
randomForest 4.7-1.1

Type rfNews() to see new features/changes/bug fixes.


Attaching package: 'randomForest'

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

    combine

The following object is masked from 'package:ggplot2':

    margin

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

    importance
```

```r
library(tree)



data <- read.csv("../Indicators_Of_Heart_Disease/2022/heart_2022_no_nans.csv")
summary(data)
```

```
   State               Sex            GeneralHealth      PhysicalHealthDays
Length:246022      Length:246022      Length:246022      Min.   : 0.000
Class :character   Class :character   Class :character   1st Qu.: 0.000
Mode  :character   Mode  :character   Mode  :character   Median : 0.000
                                                         Mean   : 4.119
                                                         3rd Qu.: 3.000
                                                         Max.   :30.000
MentalHealthDays LastCheckupTime    PhysicalActivities   SleepHours
Min.   : 0.000   Length:246022      Length:246022      Min.   : 1.000
1st Qu.: 0.000   Class :character   Class :character   1st Qu.: 6.000
Median : 0.000   Mode  :character   Mode  :character   Median : 7.000
Mean   : 4.167                                         Mean   : 7.021
3rd Qu.: 4.000                                         3rd Qu.: 8.000
Max.   :30.000                                         Max.   :24.000
RemovedTeeth       HadHeartAttack      HadAngina          HadStroke
Length:246022      Length:246022      Length:246022      Length:246022
Class :character   Class :character   Class :character   Class :character
Mode  :character   Mode  :character   Mode  :character   Mode  :character



 HadAsthma         HadSkinCancer       HadCOPD          HadDepressiveDisorder
Length:246022      Length:246022      Length:246022      Length:246022
Class :character   Class :character   Class :character   Class :character
Mode  :character   Mode  :character   Mode  :character   Mode  :character



HadKidneyDisease   HadArthritis       HadDiabetes        DeafOrHardOfHearing
Length:246022      Length:246022      Length:246022      Length:246022
Class :character   Class :character   Class :character   Class :character
Mode  :character   Mode  :character   Mode  :character   Mode  :character



BlindOrVisionDifficulty DifficultyConcentrating DifficultyWalking
Length:246022           Length:246022           Length:246022
Class :character        Class :character        Class :character
Mode  :character        Mode  :character        Mode  :character



DifficultyDressingBathing DifficultyErrands  SmokerStatus
```

```
Length:246022             Length:246022      Length:246022     
Class :character          Class :character   Class :character  
Mode  :character          Mode  :character   Mode  :character  
                                                               
                                                               
                                                               
ECigaretteUsage     ChestScan         RaceEthnicityCategory AgeCategory       
Length:246022      Length:246022      Length:246022         Length:246022     
Class :character   Class :character   Class :character      Class :character  
Mode  :character   Mode  :character   Mode  :character      Mode  :character  
                                                                              
                                                                              
                                                                              
HeightInMeters  WeightInKilograms      BMI        AlcoholDrinkers   
Min.   :0.910   Min.   : 28.12    Min.   :12.02   Length:246022     
1st Qu.:1.630   1st Qu.: 68.04    1st Qu.:24.27   Class :character  
Median :1.700   Median : 81.65    Median :27.46   Mode  :character  
Mean   :1.705   Mean   : 83.62    Mean   :28.67                     
3rd Qu.:1.780   3rd Qu.: 95.25    3rd Qu.:31.89                     
Max.   :2.410   Max.   :292.57    Max.   :97.65                     
 HIVTesting        FluVaxLast12       PneumoVaxEver      TetanusLast10Tdap 
Length:246022      Length:246022      Length:246022      Length:246022     
Class :character   Class :character   Class :character   Class :character  
Mode  :character   Mode  :character   Mode  :character   Mode  :character  
                                                                           
                                                                           
                                                                           
HighRiskLastYear     CovidPos        
Length:246022      Length:246022     
Class :character   Class :character  
Mode  :character   Mode  :character  
```

```r
suppressMessages({
  attach(data)
})


data <- data %>%
  filter(BMI <= 41, BMI >= 14,
         MentalHealthDays < 10,
         PhysicalHealthDays <= 8,
         SleepHours < 11, SleepHours > 3)

outliers <- boxplot.stats(WeightInKilograms)$out
data <- data %>%
  filter(!(WeightInKilograms %in% outliers))

dim(data)
```

```
[1] 171871     40
```

```
# Set seed for reproducibility
set.seed(4322)

# Sample data
num_row = nrow(data)
new_data = data[sample(num_row, num_row*0.5),]

# Function to convert categorical variables
check_and_convert_categorical <- function(test_data) {
  for (col_name in names(test_data)) {
    if (!is.factor(test_data[[col_name]]) && (is.character(test_data[[col_name]]) || lengt
      test_data[[col_name]] <- as.numeric(as.factor(test_data[[col_name]]))
    }
  }
  return(test_data)
}
rf_data <- check_and_convert_categorical(new_data)

# Force conversion to factor for AgeCategory
rf_data$AgeCategory = as.factor(rf_data$AgeCategory)

# Map states to regions
northeast <- c("Maine", "New Hampshire", "Vermont", "Massachusetts", "Rhode Island",
               "Connecticut", "New York", "New Jersey", "Pennsylvania")
midwest <- c("Ohio", "Michigan", "Indiana", "Illinois", "Wisconsin", "Minnesota",
             "Iowa", "Missouri", "North Dakota", "South Dakota", "Nebraska", "Kansas")
south <- c("Delaware", "Maryland", "District of Columbia", "Virginia", "West Virginia",
           "Kentucky", "North Carolina", "South Carolina", "Tennessee", "Georgia",
           "Florida", "Alabama", "Mississippi", "Arkansas", "Louisiana", "Texas", "Oklahom
```

```r
west <- c("Montana", "Idaho", "Wyoming", "Colorado", "New Mexico", "Arizona",
          "Utah", "Nevada", "California", "Oregon", "Washington", "Alaska", "Hawaii")
territories <- c("Puerto Rico", "Guam", "Virgin Islands")

data$Region <- with(data, factor(
  ifelse(State %in% northeast, "Northeast",
         ifelse(State %in% midwest, "Midwest",
                ifelse(State %in% south, "South",
                       ifelse(State %in% west, "West",
                              ifelse(State %in% territories, "Territories", "Other")
                       )
                )
         )
  )
))

data <- data[, !(names(data) %in% "State")]

if(any(is.na(data$Region))) {
  warning("Some states were not categorized into any region.")
}
```

```r
# Split data
n = nrow(rf_data)
p = ncol(rf_data)

set.seed(4322)
train = sample(n, 0.8*n)

rf_train = rf_data[train, ]
rf_test = rf_data[-train, ]
print(Sys.time())
```

```
[1] "2024-04-28 17:54:57 CDT"
```

```r
cat("Model 1 starting with 500 trees, mtry = sqrt p")
```

```
Model 1 starting with 500 trees, mtry = sqrt p
```

```
rf_model <- ranger(HadHeartAttack ~ .,
                   data = rf_train,
                   num.trees = 500, mtry =  sqrt(p),
                   num.threads = 8, importance = "impurity")

cat("Model 1 ending with 500 trees, mtry = sqrt p")
```

```
Model 1 ending with 500 trees, mtry = sqrt p
```

```
print(Sys.time())
```

```
[1] "2024-04-28 17:55:10 CDT"
```

```
train_predictions = predict(rf_model, data = rf_train)$predictions
train_accuracy = mean(train_predictions == rf_train$HadHeartAttack)
cat("Training Accuracy:", train_accuracy, "\n")
```

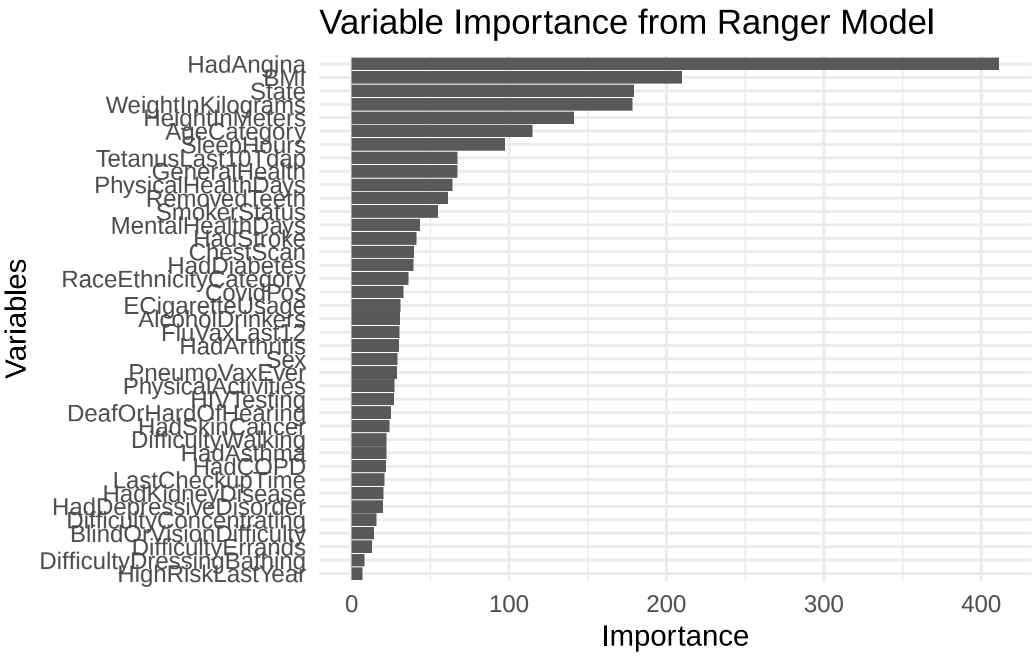
```
Training Accuracy: 0.1122796
```

```
test_predictions = predict(rf_model, data = rf_test)$predictions
test_accuracy = mean(test_predictions == rf_test$HadHeartAttack)
cat("Test Accuracy:", test_accuracy, "\n")
```

```
Test Accuracy: 0.08518066
```

```
importance_data <- as.data.frame(rf_model$variable.importance)
names(importance_data) <- c("Importance")
importance_data$Variable <- rownames(importance_data)
importance_data <- importance_data[order(importance_data$Importance, decreasing = TRUE),]


ggplot(importance_data, aes(x = reorder(Variable, Importance), y = Importance)) +
  geom_bar(stat = "identity") +
  theme_minimal() +
  labs(title = "Variable Importance from Ranger Model",
       x = "Variables",
       y = "Importance") +
  coord_flip()
```



Perform the train/test split and apply to the random forest model 10 times

```
test_error_table <- numeric(10)
for (i in 1:10)
{
  set.seed(4322)
  train = sample(n, 0.8 * n)
  rf_train = rf_data[train, ]
  rf_test = rf_data[-train, ]
  print(Sys.time())
  rf_model <- ranger(HadHeartAttack ~ .,
                     data = rf_train,
                     num.trees = 1000, mtry = sqrt(p),
                     num.threads = 8, importance = "impurity")

  test_predictions = predict(rf_model, data = rf_test)$predictions
  test_accuracy = mean(test_predictions == rf_test$HadHeartAttack)
  test_error_table[i] = test_accuracy
```

```r
}
```

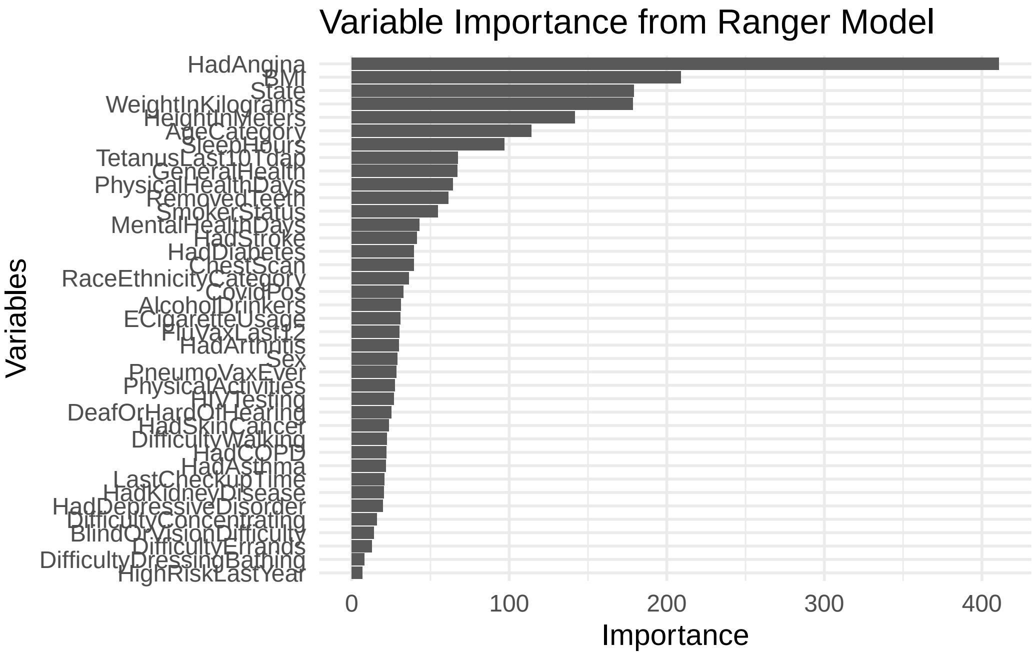
```
[1] "2024-04-28 17:55:13 CDT"
[1] "2024-04-28 17:55:39 CDT"
[1] "2024-04-28 17:56:06 CDT"
[1] "2024-04-28 17:56:32 CDT"
[1] "2024-04-28 17:56:58 CDT"
[1] "2024-04-28 17:57:27 CDT"
[1] "2024-04-28 17:57:53 CDT"
[1] "2024-04-28 17:58:19 CDT"
[1] "2024-04-28 17:58:44 CDT"
[1] "2024-04-28 17:59:10 CDT"
```

```r
# Print the mean of the test accuracy
test_acc_mean <- mean(test_error_table)
cat("Mean of the test accuracy", test_acc_mean)
```

```
Mean of the test accuracy 0.04928143
```

```r
# Importance
importance_data <- as.data.frame(rf_model$variable.importance)
names(importance_data) <- c("Importance")
importance_data$Variable <- rownames(importance_data)
importance_data <- importance_data[order(importance_data$Importance, decreasing = TRUE),]

# Plotting variable importance
ggplot(importance_data, aes(x = reorder(Variable, Importance), y = Importance)) +
  geom_bar(stat = "identity") +
  theme_minimal() +
  labs(title = "Variable Importance from Ranger Model",
       x = "Variables",
       y = "Importance") +
  coord_flip()
```



We can see that error rate does not improved (or rather stay the same as we increase# the number of tree). So in this case, we will let ntree = 500 when we perform the model# ten times in order to reduce the time.

```
print(Sys.time())
```

```
[1] "2024-04-28 17:59:36 CDT"
```

```
cat("Model 3 starting with 500 trees, mtry = sqrt p")
```

```
Model 3 starting with 500 trees, mtry = sqrt p
```

```
rf_model <- ranger(HadHeartAttack ~ HadAngina + HeightInMeters +
                     WeightInKilograms + AgeCategory +
                     BMI + Sex + SleepHours,
                   data = rf_train,
                   num.trees = 500, mtry =  sqrt(p),
                   num.threads = 8, importance = "impurity")
```

```r
cat("Model 3 ending with 500 trees, mtry = sqrt p")
```

```
Model 3 ending with 500 trees, mtry = sqrt p
```

```r
print(Sys.time())
```

```
[1] "2024-04-28 17:59:45 CDT"
```

```r
train_predictions = predict(rf_model, data = rf_train)$predictions
train_accuracy = mean(train_predictions == rf_train$HadHeartAttack)
cat("Training Accuracy:", train_accuracy, "\n")
```

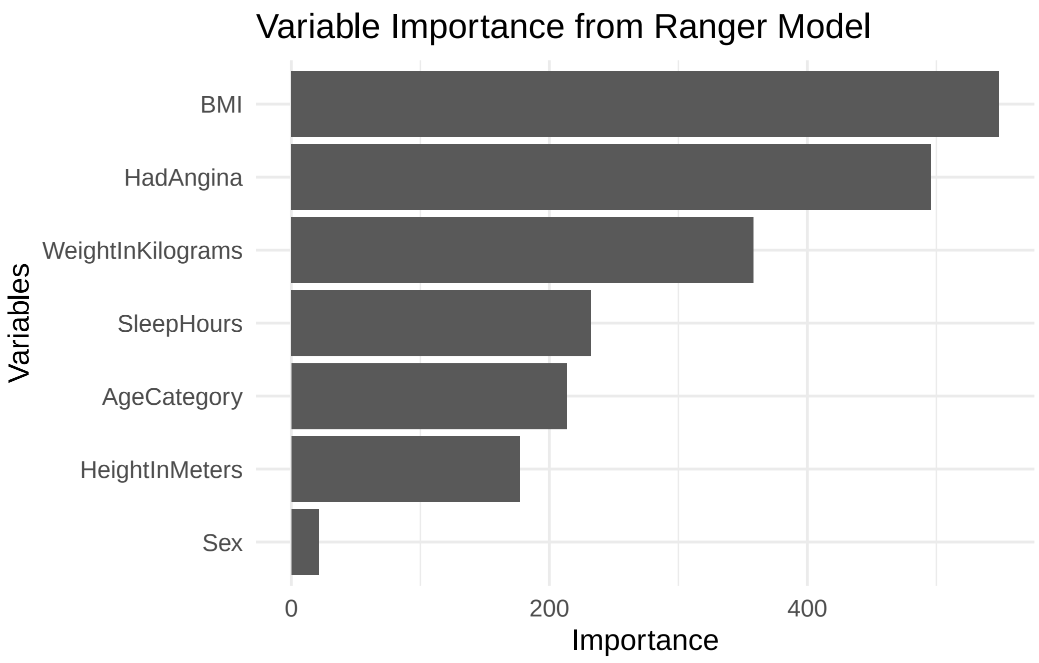
```
Training Accuracy: 0.4375836
```

```r
test_predictions = predict(rf_model, data = rf_test)$predictions
test_accuracy = mean(test_predictions == rf_test$HadHeartAttack)
cat("Test Accuracy:", test_accuracy, "\n")
```

```
Test Accuracy: 0.3884331
```

```r
importance_data <- as.data.frame(rf_model$variable.importance)
names(importance_data) <- c("Importance")
importance_data$Variable <- rownames(importance_data)
importance_data <- importance_data[order(importance_data$Importance, decreasing = TRUE),]


ggplot(importance_data, aes(x = reorder(Variable, Importance), y = Importance)) +
  geom_bar(stat = "identity") +
  theme_minimal() +
  labs(title = "Variable Importance from Ranger Model",
       x = "Variables",
       y = "Importance") +
  coord_flip()
```


Variable Importance from Ranger Model
Variables
BMI
HadAngina
WeightInKilograms
SleepHours
AgeCategory
HeightInMeters
Sex
0
200
400
Importance