Heart Disease - Hyper-parameter Tuning

Quyen Hoang

8/17/2022

## Introduction

In this project, I am going to use the [heart\_2020\_cleaned.csv](https://www.kaggle.com/code/parfarha/tuning-of-hyperparameters-for-rf-model/data) from Kaggle. Firstly, I am going to do some data wrangling where I will clean, transform it into a useable format, and get insights from the data. For example, I want to see how different the data is of those who have and do not have heart disease, the differences in data distribution between genders, the relationship between the predictors and target variable, etc. Secondly, I am going to solve the imbalanced data problem by using the sampling and weights methods with cross-validation on a random forest classification model. Lastly, after choosing the best method for the imbalance issues, I am going to tune the hyperparameter to find the most efficient one for our prediction.

## Loading libraries

library(readr)  
library(ggplot2)  
library(dplyr)

library(tidyr)  
library(purrr)  
library(corrplot)

library(skimr)  
library(gridExtra)

library(hrbrthemes)

library(caret)

library(ROSE)

library(tidymodels)

## Loading data

heartdata <-read\_csv("~/Northeastern/ALY 6015/Heart Disease/heart\_2020\_cleaned.csv")

## Rows: 319795 Columns: 18  
## -- Column specification --------------------------------------------------------

## Data Exploratory

***##Data Overview***  
summary(heartdata)

## HeartDisease BMI Smoking AlcoholDrinking   
## Length:319795 Min. :12.02 Length:319795 Length:319795   
## Class :character 1st Qu.:24.03 Class :character Class :character   
## Mode :character Median :27.34 Mode :character Mode :character   
## Mean :28.33   
## 3rd Qu.:31.42   
## Max. :94.85   
## Stroke PhysicalHealth MentalHealth DiffWalking   
## Length:319795 Min. : 0.000 Min. : 0.000 Length:319795   
## Class :character 1st Qu.: 0.000 1st Qu.: 0.000 Class :character   
## Mode :character Median : 0.000 Median : 0.000 Mode :character   
## Mean : 3.372 Mean : 3.898   
## 3rd Qu.: 2.000 3rd Qu.: 3.000   
## Max. :30.000 Max. :30.000   
## Sex AgeCategory Race Diabetic   
## Length:319795 Length:319795 Length:319795 Length:319795   
## Class :character Class :character Class :character Class :character   
## Mode :character Mode :character Mode :character Mode :character   
##   
##   
##   
## PhysicalActivity GenHealth SleepTime Asthma   
## Length:319795 Length:319795 Min. : 1.000 Length:319795   
## Class :character Class :character 1st Qu.: 6.000 Class :character   
## Mode :character Mode :character Median : 7.000 Mode :character   
## Mean : 7.097   
## 3rd Qu.: 8.000   
## Max. :24.000   
## KidneyDisease SkinCancer   
## Length:319795 Length:319795   
## Class :character Class :character   
## Mode :character Mode :character   
##   
##   
##

*#Dimmension*  
dim(heartdata)

## [1] 319795 18

*#Data types.*   
str(heartdata)

## spec\_tbl\_df [319,795 x 18] (S3: spec\_tbl\_df/tbl\_df/tbl/data.frame)  
## $ HeartDisease : chr [1:319795] "No" "No" "No" "No" ...  
## $ BMI : num [1:319795] 16.6 20.3 26.6 24.2 23.7 ...  
## $ Smoking : chr [1:319795] "Yes" "No" "Yes" "No" ...  
## $ AlcoholDrinking : chr [1:319795] "No" "No" "No" "No" ...  
## $ Stroke : chr [1:319795] "No" "Yes" "No" "No" ...  
## $ PhysicalHealth : num [1:319795] 3 0 20 0 28 6 15 5 0 0 ...  
## $ MentalHealth : num [1:319795] 30 0 30 0 0 0 0 0 0 0 ...  
## $ DiffWalking : chr [1:319795] "No" "No" "No" "No" ...  
## $ Sex : chr [1:319795] "Female" "Female" "Male" "Female" ...  
## $ AgeCategory : chr [1:319795] "55-59" "80 or older" "65-69" "75-79" ...  
## $ Race : chr [1:319795] "White" "White" "White" "White" ...  
## $ Diabetic : chr [1:319795] "Yes" "No" "Yes" "No" ...  
## $ PhysicalActivity: chr [1:319795] "Yes" "Yes" "Yes" "No" ...  
## $ GenHealth : chr [1:319795] "Very good" "Very good" "Fair" "Good" ...  
## $ SleepTime : num [1:319795] 5 7 8 6 8 12 4 9 5 10 ...  
## $ Asthma : chr [1:319795] "Yes" "No" "Yes" "No" ...  
## $ KidneyDisease : chr [1:319795] "No" "No" "No" "No" ...  
## $ SkinCancer : chr [1:319795] "Yes" "No" "No" "Yes" ...  
## - attr(\*, "spec")=  
## .. cols(  
## .. HeartDisease = col\_character(),  
## .. BMI = col\_double(),  
## .. Smoking = col\_character(),  
## .. AlcoholDrinking = col\_character(),  
## .. Stroke = col\_character(),  
## .. PhysicalHealth = col\_double(),  
## .. MentalHealth = col\_double(),  
## .. DiffWalking = col\_character(),  
## .. Sex = col\_character(),  
## .. AgeCategory = col\_character(),  
## .. Race = col\_character(),  
## .. Diabetic = col\_character(),  
## .. PhysicalActivity = col\_character(),  
## .. GenHealth = col\_character(),  
## .. SleepTime = col\_double(),  
## .. Asthma = col\_character(),  
## .. KidneyDisease = col\_character(),  
## .. SkinCancer = col\_character()  
## .. )  
## - attr(\*, "problems")=<externalptr>

*#list numeric columns*  
numColumns <- colnames(dplyr::select\_if(heartdata, is.numeric))  
numColumns

## [1] "BMI" "PhysicalHealth" "MentalHealth" "SleepTime"

*#list categorical columns*  
charColumns <- colnames(dplyr::select\_if(heartdata, is.character))  
charColumns

## [1] "HeartDisease" "Smoking" "AlcoholDrinking" "Stroke"   
## [5] "DiffWalking" "Sex" "AgeCategory" "Race"   
## [9] "Diabetic" "PhysicalActivity" "GenHealth" "Asthma"   
## [13] "KidneyDisease" "SkinCancer"

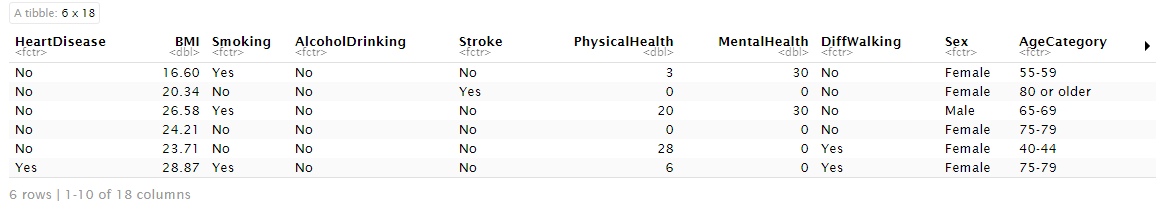
The data set has 18 variables and 319,795 observations.  
There are 4 numeric columns and 14 categorical columns.

*#Check for NUll values*  
colnames(heartdata)[colSums(is.na(heartdata)) > 0]

## character(0)

There is no NA value within it.

*#First 6 rows of dataset*

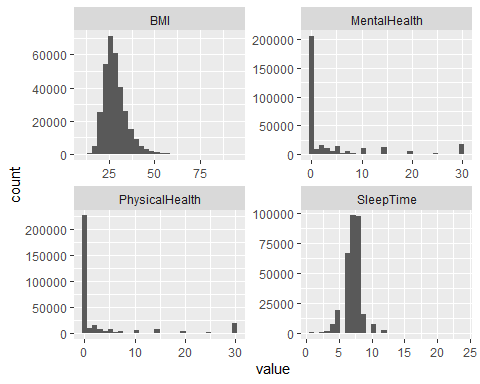


## Data Visualization

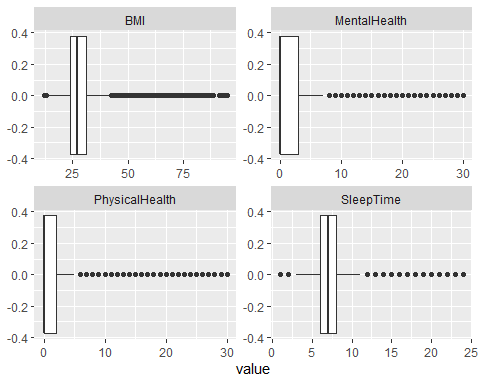
### Overall Distribution of All Variables.

*#Histogram for numeric data*  
heartdata %>%  
 keep(is.numeric) %>%   
 gather() %>%   
 ggplot(aes(value)) +  
 facet\_wrap(~ key, scales = "free") +  
 geom\_histogram()

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



*#Boxplot for numeric variables*  
heartdata %>%  
 keep(is.numeric) %>%   
 gather() %>%   
 ggplot(aes(value)) +  
 facet\_wrap(~ key, scales = "free") +  
 geom\_boxplot()

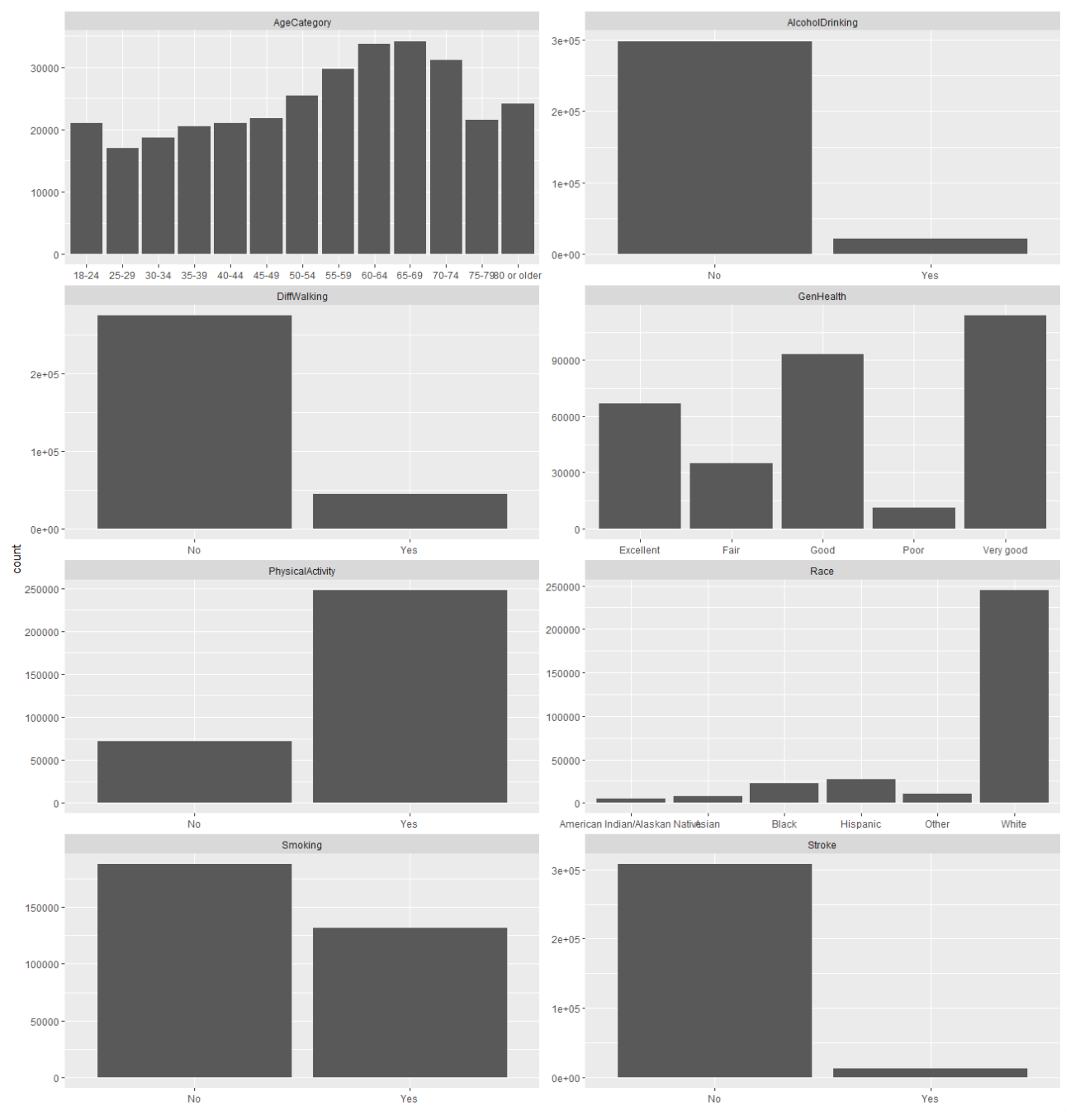


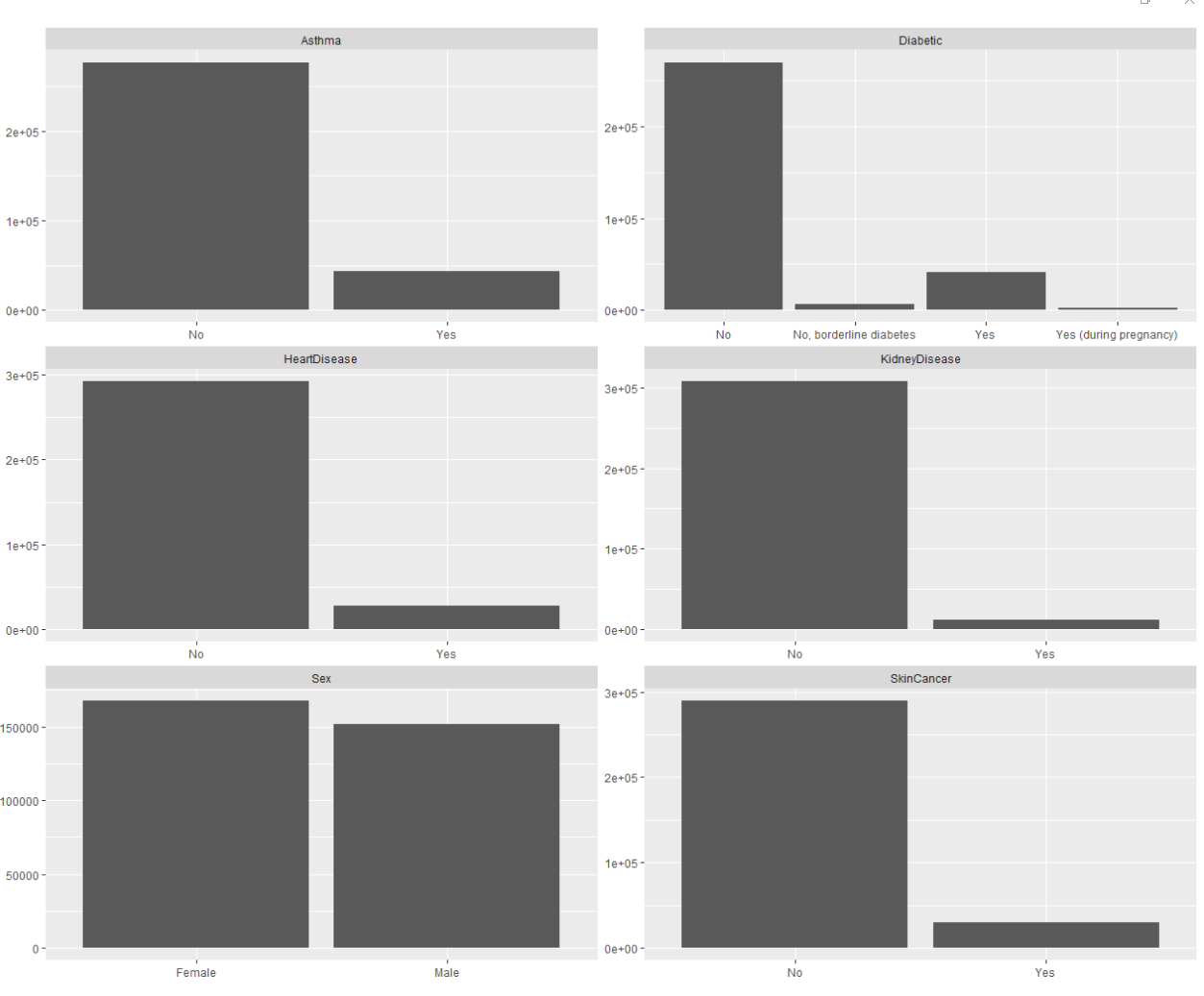
- Sleep time: The number of hours people sleep gathers mostly around 6-8 hours.

-BMI: Body Mass Index is right-skewed with a min BMI o 12.02, the most popular BMI is 25, and the maximum is 94.85.

-Physical and mental health: Their distributions are quite similar two each other. Their data distribute mostly around 0 (~70%), the rest of them scatter from 1-30.

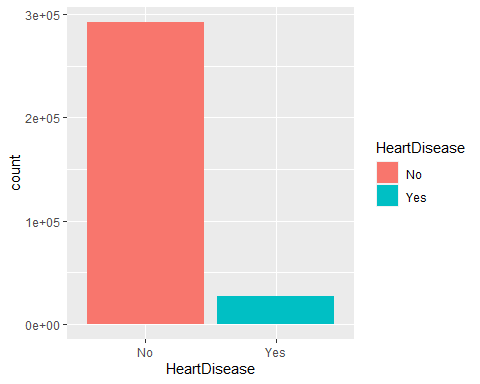
*#Barlot for categorical variables*  
heartdata %>%  
 keep(is.character) %>%   
 gather() %>%   
 ggplot(aes(value)) +  
 facet\_wrap(~ key, scales = "free") +  
 geom\_bar()





* Distribution of Heart Disease variable.

ggplot(heartdata, aes(x=HeartDisease, fill=HeartDisease))+geom\_bar()



**Interpretation:**

-First, the number of people who have heart disease in this data set is 1/10 of the number of those who do not.

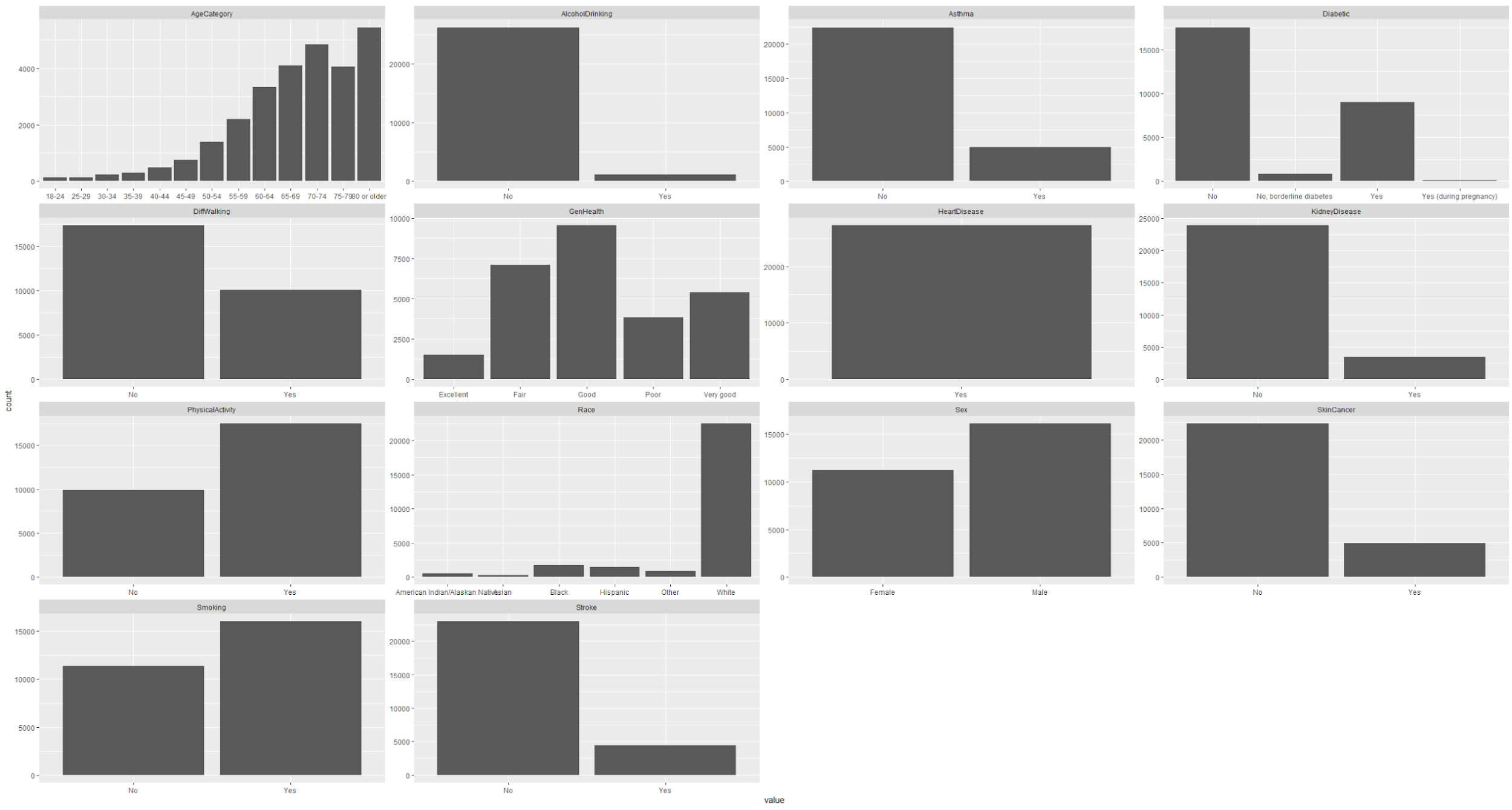
-Second, it looks like the ratio of heart disease is in direct proportion to the ratio of Skin Cancer, Stroke, Diabetic, ASthma, AlcoholDrinking, and DiffWalking.

-To see whether the people that have heart disease are also those who have these problems above I will run code to get the data of those who have heart disease only.

### Compare the distribution of Variables in the general data set and the heartdisease\_yes data set

* **Distribution of the categorical values**

*#Barplot for categorical values.*   
heartdisease\_yes<-heartdata %>% filter(HeartDisease=="Yes")  
heartdisease\_yes %>%  
 keep(is.character) %>%   
 gather() %>%   
 ggplot(aes(value)) +  
 facet\_wrap(~ key, scales = "free") +  
 geom\_bar()

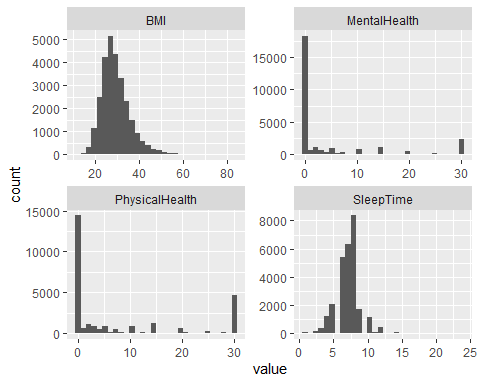


-From the result, it seems my assumption is not quite right. For the people who have Heart Disease, the number of those who drink Alcohol, have Asthma, Stroke, Skin Cancer, and Kidney Disease are still way lesser than people that do not. Only the percentage of diabetic people increases for those who have heart disease.

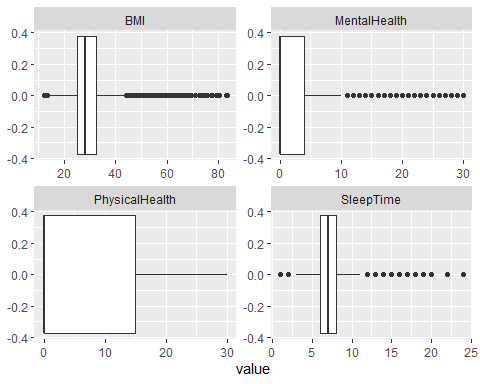
* **Distribution of the numeric variables**

*#Histogram - People who have heart disease*  
heartdisease\_yes %>%  
 keep(is.numeric) %>%   
 gather() %>%   
 ggplot(aes(value)) +  
 facet\_wrap(~ key, scales = "free") +  
 geom\_histogram()

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



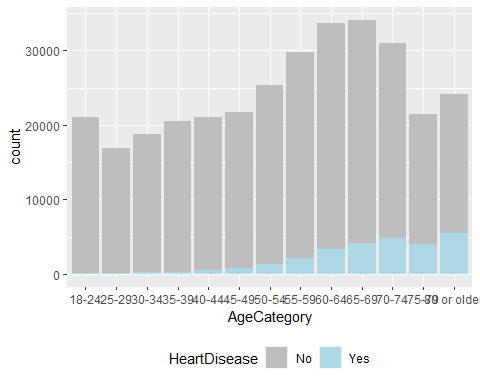
*#Boxplot for numeric variables - People who have heart disease.*   
heartdisease\_yes %>%  
 keep(is.numeric) %>%   
 gather() %>%   
 ggplot(aes(value)) +  
 facet\_wrap(~ key, scales = "free") +  
 geom\_boxplot()



**Comparison**: For people who have heart disease, the range of physical health (from 0-15) is larger than the one of the general dataset (~0-3). Although the percentage of people who have physical health=0 is still large, the number of people that have physical health from 0-15 increases significantly. There are not many differences in the distribution of Sleep Time, BMI, and Mental Health of people who have and do not have heart disease.

### Age Distribution

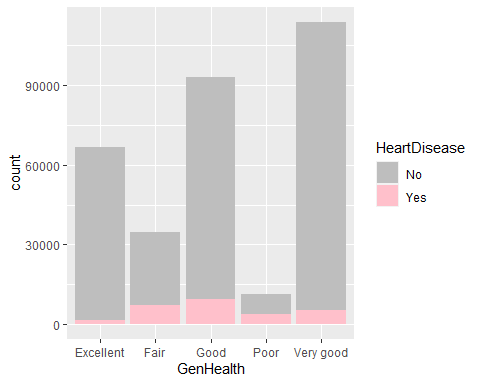
*#Distribution of Age by Heart Disease*  
ggplot(data=heartdata, aes(fill=HeartDisease, x=AgeCategory))+geom\_bar()+scale\_fill\_manual(values = c("Grey","LightBlue"))+theme(legend.position = "bottom")



**Comparison**: The number of people who have heart disease increases at an older age.

### General Health Distribution

*#Distribution of General Health by Heart Disease*  
ggplot(data=heartdata, aes(fill=HeartDisease, x=GenHealth))+geom\_bar()+scale\_fill\_manual(values = c("Grey","Pink"))



**Comparison**: There is a difference in the general health of people that have heart disease compared to the general data set: The general health of HeartDisease\_Yes people are not as good as those in the general data set

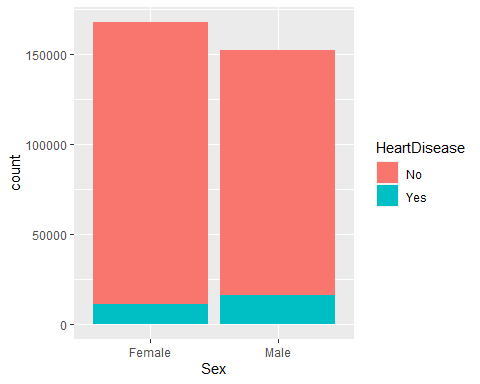
Arrange from the lowest to largest in quantity

- in the whole data set: Poor, Fair, Excellent, Good, Very Good.

-in the HeartDisease\_Yes data: Excellent, Poor, Fair, Very Good, Good.

### Sex Distribution

*#Barplot HeartDisease by Sex*  
ggplot(data=heartdata, aes(fill=HeartDisease, x=Sex))+geom\_bar()

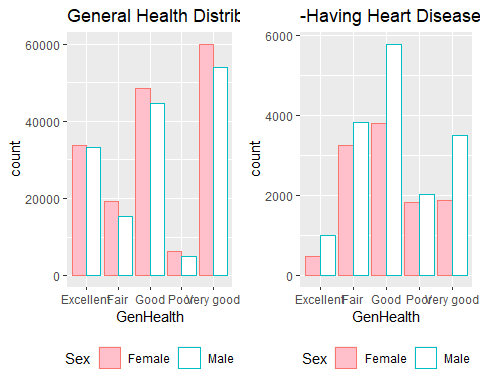


table(heartdisease\_yes$Sex)

##   
## Female Male   
## 11234 16139

**Comparison**: There are more males having heart disease than females.

### General Health Distribution by Sex



Males seem to have better health than females for those who have heart disease.

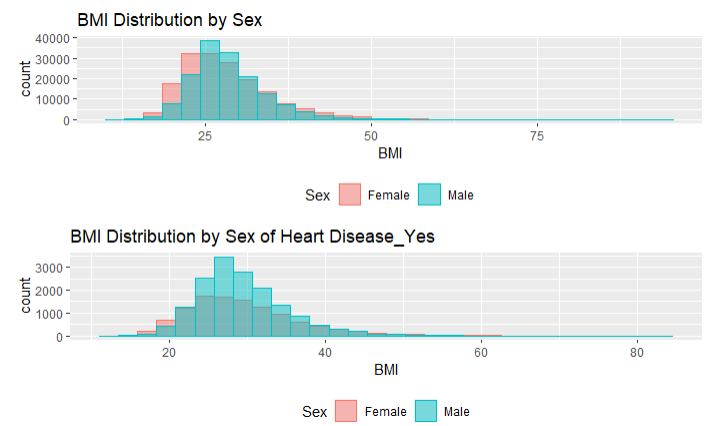
### BMI Distribution

*#Distribution of BMI by Heart Disease*  
ggplot(data=heartdata, aes(color=HeartDisease, x=BMI, fill=HeartDisease))+geom\_histogram(alpha=0.5, position="identity")+ggtitle("BMI Distribution")+theme(legend.position = "bottom")

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.

* BMI Distribution by Sex

b1<-ggplot(data=heartdata, aes(color=Sex, x=BMI, fill=Sex))+geom\_histogram(alpha=0.5, position="identity")+ggtitle("BMI Distribution by Sex")+theme(legend.position = "bottom")  
b2<-ggplot(data=heartdisease\_yes, aes(color=Sex, x=BMI, fill=Sex))+geom\_histogram(alpha=0.5, position="identity")+ggtitle("BMI of people have Heart Disease")+theme(legend.position = "bottom")  
grid.arrange(b1,b2, nrow=2)



**Comparison:**

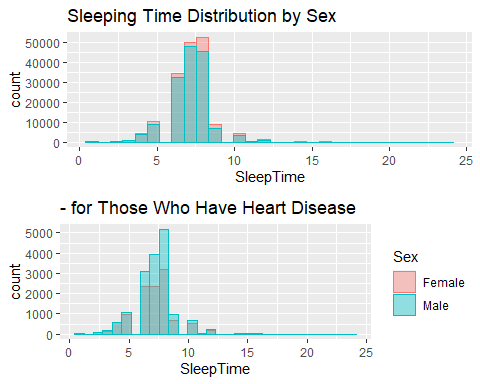
**Gender:** There is a difference between the distributions of Males and Females. As we can see Males tend to fall into the overweight range (25>29.9), while females are kind of lean to the left which is normal weights (18.5-24.9) and underweight (<18.4).

**Heart Disease**: There is no difference between distributions of BMI of HeartDisease\_Yes and the overall data set. The difference in the two plots caused by the number of males in the heartdisease\_yes data set is larger than females compare to the whole data set.

### Sleep time Distribution by Sex

***##Overall***  
s1<-ggplot(data=heartdata, aes(color=Sex, x=SleepTime, fill=Sex))+geom\_histogram(alpha=0.4, position="identity")+theme(legend.position = "none")+ggtitle("Sleeping Time Distribution by Sex")  
***##for whom who have heart disease***  
s2<-ggplot(data=heartdisease\_yes, aes(color=Sex, x=SleepTime, fill=Sex))+geom\_histogram(alpha=0.4, position="identity")+ggtitle("- for Those Who Have Heart Disease")  
  
grid.arrange(s1,s2, nrow=2)

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.  
## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



table(heartdata$SleepTime)

##   
## 1 2 3 4 5 6 7 8 9 10 11 12 13   
## 551 788 1992 7750 19184 66721 97751 97602 16041 7796 415 2205 97   
## 14 15 16 17 18 19 20 21 22 23 24   
## 243 189 236 21 102 3 64 2 9 3 30

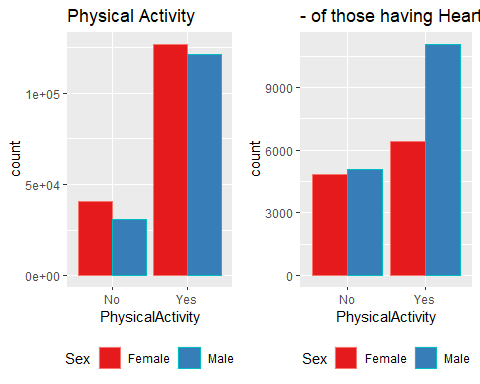
**Comparison:**

**Gender**: The amount of sleeping time for males and females is pretty equal. Both groups’ sleeping time gathers mostly around 7-8 hours per day. Female seems to sleep more than males represented by the sleeping time of female get peak at a higher time.

**Heart Disease**: Compare to the overall data, there is not much difference in the distribution in sleep time of Males and Females. The difference in the two plots is caused by the decrease in the number of females in the HeartDisease\_Yes data set.

### Physical Activity Distribution by Sex

p1<-ggplot(data=heartdata, aes(color=Sex, x=PhysicalActivity, fill=Sex))+geom\_bar(position="dodge")+scale\_fill\_brewer(palette = "Set1")+ggtitle("Physical Activity")+theme(legend.position = "bottom")  
p2<-ggplot(data=heartdisease\_yes, aes(color=Sex, x=PhysicalActivity, fill=Sex))+geom\_bar(position="dodge")+scale\_fill\_brewer(palette = "Set1")+ggtitle("- of those having Heart Disease")+theme(legend.position = "bottom")  
grid.arrange(p1,p2, ncol=2)



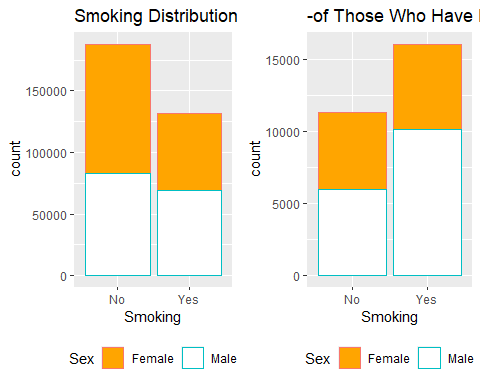
**Comparison**:

**Gender**: In the general data set, Males and females are quite equal in the number of doing Physical Activity.

**Heart Disease:** When it comes to those who get heart disease, Males are doing more Physical Activity than Females.

### Smoking Distribution by Sex

sm1<-ggplot(data=heartdata, aes(color=Sex, x=Smoking, fill=Sex))+geom\_bar()+scale\_fill\_manual(values = c("Orange","white"))+ggtitle("Smoking Distribution by Sex")+theme(legend.position = "bottom")  
sm2<-ggplot(data=heartdisease\_yes, aes(color=Sex, x=Smoking, fill=Sex))+geom\_bar()+scale\_fill\_manual(values = c("Orange","white"))+ggtitle("-of Those Who Have Heart Disease")+theme(legend.position = "bottom")  
grid.arrange(sm1,sm2, ncol=2)



**Comparison:**

**Gender:** Males and Females are equal in the number of Smoking.

**Heart Disease:** when it comes to those who got heart disease, Males are doing more Smoking than Females.

## Correlation matrix

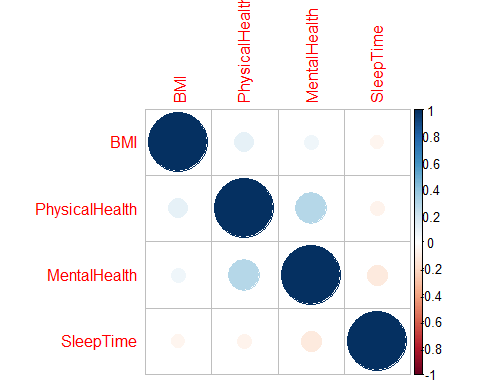
library(viridis)

## Loading required package: viridisLite

*#Correlation matrix*  
corrMatrix <- cor(subset( heartdata, select = numColumns))  
corrMatrix

## BMI PhysicalHealth MentalHealth SleepTime  
## BMI 1.00000000 0.10978754 0.06413057 -0.05182225  
## PhysicalHealth 0.10978754 1.00000000 0.28798667 -0.06138663  
## MentalHealth 0.06413057 0.28798667 1.00000000 -0.11971679  
## SleepTime -0.05182225 -0.06138663 -0.11971679 1.00000000

corrplot(corrMatrix)



Overall, there is no strong relationship between numeric variables, the highest correlation value is 0.29 which is between Mental and Physical health. Sleep time seems to have low negative relationships with all the numeric variables.

## Data Analysis

1. Data Splitting

*#Data Splitting*  
set.seed(145)

trainning.samples<-heartdata$HeartDisease %>% createDataPartition(p=0.7, list=FALSE)

train<-heartdata[trainning.samples,]

test<- heartdata[-trainning.samples,]

1. Data Training

*#Modeling the original unbalanced data*  
et.seed(145)

grid<-expand.grid(mtry=2)

ctrl <- trainControl(method = "cv",

number = 5,

verboseIter = TRUE,

savePredictions = "final")

model\_rf <- caret::train(HeartDisease ~ .,

data = train,

method = "rf",

ntree= 400,

preProcess = c("scale", "center"),

trControl = ctrl,

tuneGrid=grid)

*#Confusion matrix*  
final <- data.frame(actual = test$HeartDisease,

predict(model\_rf, newdata = test, type = "prob"))

final$predict <-as.factor(ifelse(final$Yes > 0.5, "Yes", "No"))

cm\_original <- confusionMatrix(final$predict, test$HeartDisease, positive = 'Yes')

> cm\_original

Confusion Matrix and Statistics

Reference

Prediction No Yes

No 87723 8192

Yes 3 19

Accuracy : 0.9146

95% CI : (0.9128, 0.9163)

No Information Rate : 0.9144

P-Value [Acc > NIR] : 0.4296

Kappa : 0.0042

Mcnemar's Test P-Value : <2e-16

Sensitivity : 0.0023140

Specificity : 0.9999658

Pos Pred Value : 0.8636364

Neg Pred Value : 0.9145910

Prevalence : 0.0855874

Detection Rate : 0.0001980

Detection Prevalence : 0.0002293

Balanced Accuracy : 0.5011399

'Positive' Class : Yes

> cm\_original$byClass

Sensitivity Specificity Pos Pred Value Neg Pred Value Precision Recall F1

0.0023139691 0.9999658026 0.8636363636 0.9145910442 0.8636363636 0.0023139691 0.0046155715

Prevalence Detection Rate Detection Prevalence Balanced Accuracy

0.0855874167 0.0001980466 0.0002293172 0.5011398858

Recall: 0.0023, Precision: 0.86, F1: 0.0046

High Precision +Low Recall: the model can’t detect the class well but is highly trustable when it does.

Because our data set is imbalanced- the number of No is 10 times compared to the Yes so when we train the data, there are not enough Yes observations for it to learn for an accurate prediction later.

We need to solve the imbalanced data problem. In this project I am going to use 4 methods: under-sampling, over-sampling, smote, weighted-fit; and confusion metric to measure the accuracy.

The F1 score of a class is given by the harmonic mean of precision and recall (2×precision×recall / (precision + recall)), it combines the precision and recall of a class in one metric.

1. Dealing with unbalanced data
2. Under-sampling method

*# Modeling on the under-sampling data set*

ctrl <- trainControl(method = "cv",

number = 5,

verboseIter = TRUE,

savePredictions = "final",

sampling = 'down')

set.seed(145)

grid<-expand.grid(mtry=2)

model\_rf\_under<- caret::train(HeartDisease ~ .,

data = train,

method = "rf",

ntree= 400,

preProcess = c("scale", "center"),

trControl = ctrl,

tuneGrid=grid

*# Confusion matrix*

final\_under <- data.frame(actual = test$HeartDisease,

predict(model\_rf\_under, newdata = test, type = "prob"))

final\_under$predict <- as.factor(ifelse(final\_under$Yes > 0.5, "Yes", "No"))

cm\_under <- confusionMatrix(final\_under$predict, test$HeartDisease, positive = 'Yes')

> cm\_under

Confusion Matrix and Statistics

Reference

Prediction No Yes

No 64392 1841

Yes 23334 6370

Accuracy : 0.7376

95% CI : (0.7348, 0.7404)

No Information Rate : 0.9144

P-Value [Acc > NIR] : 1

Kappa : 0.2332

Mcnemar's Test P-Value : <2e-16

Sensitivity : 0.77579

Specificity : 0.73401

Pos Pred Value : 0.21445

Neg Pred Value : 0.97220

Prevalence : 0.08559

Detection Rate : 0.06640

Detection Prevalence : 0.30962

Balanced Accuracy : 0.75490

'Positive' Class : Yes

> cm\_under$byClass

Sensitivity Specificity Pos Pred Value Neg Pred Value Precision Recall F1

0.77578858 0.73401272 0.21444923 0.97220419 0.21444923 0.77578858 0.33601477

Prevalence Detection Rate Detection Prevalence Balanced Accuracy

0.08558742 0.06639774 0.30961985 0.75490065

Recall: 0.776, Precision: 0.214, F1: 0.336

The precision metric of the under-sampling dataset is improved 10 times compare to the original one and so does the F1

1. Over-sampling method

*# Modeling on the over-sampling data set*

ctrl$sampling<-"up"

set.seed(145)

grid<-expand.grid(mtry=2)

model\_rf\_over <- caret::train(HeartDisease ~ .,

data = train,

method = "rf",

ntree= 400,

preProcess = c("scale", "center"),

trControl = ctrl,

tuneGrid=grid)

*# Confusion matrix*

final\_over <- data.frame(actual = test$HeartDisease,

predict(model\_rf\_over, newdata = test, type = "prob"))

final\_over$predict <- as.factor(ifelse(final\_over$Yes > 0.5, "Yes", "No"))

cm\_over <- confusionMatrix(final\_over$predict, test$HeartDisease, positive = 'Yes')

> cm\_over

Confusion Matrix and Statistics

Reference

Prediction No Yes

No 65446 1964

Yes 22280 6247

Accuracy : 0.7473

95% CI : (0.7445, 0.75)

No Information Rate : 0.9144

P-Value [Acc > NIR] : 1

Kappa : 0.2389

Mcnemar's Test P-Value : <2e-16

Sensitivity : 0.76081

Specificity : 0.74603

Pos Pred Value : 0.21899

Neg Pred Value : 0.97086

Prevalence : 0.08559

Detection Rate : 0.06512

Detection Prevalence : 0.29735

Balanced Accuracy : 0.75342

'Positive' Class : Yes

> cm\_over$byClasscm\_over$byClass

Sensitivity Specificity Pos Pred Value Neg Pred Value Precision Recall F1

0.76080867 0.74602740 0.21898552 0.97086486 0.21898552 0.76080867 0.34008384

Prevalence Detection Rate Detection Prevalence Balanced Accuracy

0.08558742 0.06511565 0.29735139 0.75341804

Recall: 0.76, Precision: 0.219, F1: 0.34

The metrics are quite similar to those of the under-sampling method.

1. Smote

*#Modeling on the smote data set*  
ctrl <- trainControl(method = "cv",

number = 5,

verboseIter = TRUE,

savePredictions = "final",

sampling = "smote")

set.seed(145)

model\_rf\_smote<- caret::train(HeartDisease ~ .,

data = train,

method = "rf",

ntree= 400,

preProcess = c("scale", "center"),

trControl = ctrl,

tuneGrid=grid)

*# Confusion matrix*

final\_smote <- data.frame(actual = test$HeartDisease,

predict(model\_rf\_smote, newdata = test, type = "prob"))

final\_smote$predict <- as.factor(ifelse(final\_smote$Yes > 0.5, "Yes", "No"))

cm\_smote <- confusionMatrix(final\_smote$predict, test$HeartDisease, positive = 'Yes')

> cm\_smote

Confusion Matrix and Statistics

Reference

Prediction No Yes

No 79855 4639

Yes 7871 3572

Accuracy : 0.8696

95% CI : (0.8675, 0.8717)

No Information Rate : 0.9144

P-Value [Acc > NIR] : 1

Kappa : 0.293

Mcnemar's Test P-Value : <2e-16

Sensitivity : 0.43503

Specificity : 0.91028

Pos Pred Value : 0.31216

Neg Pred Value : 0.94510

Prevalence : 0.08559

Detection Rate : 0.03723

Detection Prevalence : 0.11928

Balanced Accuracy : 0.67265

'Positive' Class : Yes

> cm\_smote$byClass

Sensitivity Specificity Pos Pred Value Neg Pred Value Precision Recall F1

0.43502618 0.91027745 0.31215590 0.94509669 0.31215590 0.43502618 0.36348835

Prevalence Detection Rate Detection Prevalence Balanced Accuracy

0.08558742 0.03723277 0.11927619 0.67265182

Precision: 0.312, Recall: 0.435, F1: 0.363

The Precision decreases but the recall is improved 2 times. F1 is quite the same.

1. Weighted-fit

*#Modeling on weighted-fit data set.*

ctrl <- trainControl(method = "cv",

number = 5,

verboseIter = TRUE,

savePredictions = "final")

model\_weights <- ifelse(train$HeartDisease == "Yes",

(1/table(train$HeartDisease)['Yes']) \* 0.5,

(1/table(train$HeartDisease)['No']) \* 0.5)

grid<-expand.grid(mtry=2)

set.seed(145)

weighted\_fit <- train(HeartDisease ~ .,

data = train,

method = "rf",

verbose = TRUE,

weights = model\_weights,

tuneGrid=grid,

trControl = ctrl)

*#MConfusion matrix*

final\_weighted\_fit <- data.frame(actual = test$HeartDisease,

predict(weighted\_fit, newdata = test, type = "prob"))

final\_weighted\_fit$predict <- as.factor(ifelse(final\_weighted\_fit$Yes > 0.5, "Yes", "No"))

cm\_weighted\_fit <- confusionMatrix(final\_weighted\_fit$predict, test$HeartDisease, positive='Yes')

> cm\_weighted\_fit

Confusion Matrix and Statistics

Reference

Prediction No Yes

No 87723 8193

Yes 3 18

Accuracy : 0.9146

95% CI : (0.9128, 0.9163)

No Information Rate : 0.9144

P-Value [Acc > NIR] : 0.4342

Kappa : 0.0039

Mcnemar's Test P-Value : <2e-16

Sensitivity : 0.0021922

Specificity : 0.9999658

Pos Pred Value : 0.8571429

Neg Pred Value : 0.9145815

Prevalence : 0.0855874

Detection Rate : 0.0001876

Detection Prevalence : 0.0002189

Balanced Accuracy : 0.5010790

'Positive' Class : Yes

> cm\_weighted\_fit$byClass

Sensitivity Specificity Pos Pred Value Neg Pred Value Precision Recall F1

0.0021921812 0.9999658026 0.8571428571 0.9145815088 0.8571428571 0.0021921812 0.0043731778

Prevalence Detection Rate Detection Prevalence Balanced Accuracy

0.0855874167 0.0001876231 0.0002188936 0.5010789919

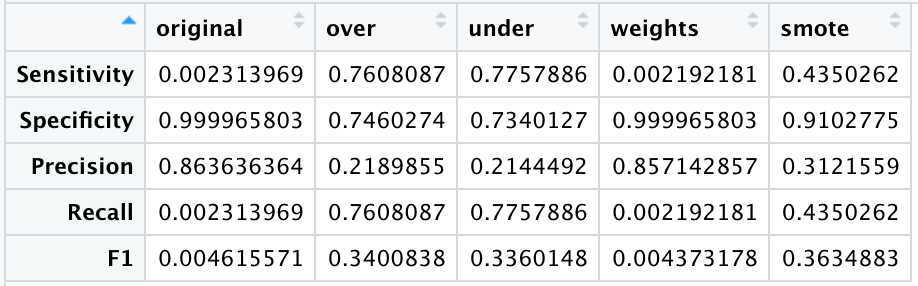
This method yields the same result as the original one.

1. Compare 5 models

### compare in traditional way

*#Compare the predictions of 5 models*  
compare<-cbind(as.data.frame(cm\_original$byClass),as.data.frame(cm\_over$byClass),as.data.frame(cm\_under$byClass), as.data.frame(cm\_weighted\_fit$byClass), as.data.frame(cm\_smote$byClass))[c(1,2,5,6,7),]

colnames(compare)<-c('original','over','under','weights', 'smote')



I decided to choose the under-sampling method, which yields relatively balanced metrics for the prediction.

## Tuning Hyperparameter

*#Set tunegrid*  
tuneGrid <- expand.grid(.mtry = c(1,5,17,30))

ntrees <- c(300, 600)

nodesize <- c(2,20,40)

ctrl <- trainControl(method = "cv",

number = 5,

verboseIter = TRUE,

savePredictions = "final",

classProbs = TRUE,

summaryFunction = twoClassSummary,

sampling = "down")

set.seed(145)

params <- expand.grid(ntrees = ntrees,

nodesize = nodesize)

store\_maxnode <- vector("list", nrow(params))

*#Hyper Parameter Tuning*

for(i in 1:nrow(params)){

nodesize <- params[i,2]

ntree <- params[i,1]

set.seed(145)

rf\_model <- train(HeartDisease~.,

data = train,

method = "rf",

importance=TRUE,

metric="ROC",

tuneGrid = tuneGrid,

trControl = ctrl,

ntree = ntree,

nodesize = nodesize)

store\_maxnode[[i]] <- rf\_model

}

> rf\_model

Random Forest

223858 samples

17 predictor

2 classes: 'No', 'Yes'

No pre-processing

Resampling: Cross-Validated (5 fold)

Summary of sample sizes: 179085, 179086, 179087, 179087, 179087

Addtional sampling using down-sampling

Resampling results across tuning parameters:

mtry Accuracy Kappa

1 0.7887992 0.2612409

5 0.7240081 0.2292526

17 0.7258396 0.2279900

30 0.7231280 0.2242624

Accuracy was used to select the optimal model using the largest value.

The final value used for the model was mtry = 1.

*#Get the best model*   
names(store\_maxnode) <- paste("ntrees:", params$ntrees,

"nodesize:", params$nodesize)

results\_mtry <- resamples(store\_maxnode)

summary(results\_mtry)

> rf\_modell$finalModel$param

$importance

[1] TRUE

$ntree

[1] 600

$nodesize

[1] 40

The final model has ntree = 600, nodesize=40, mtry=1

*#Test 6 models of tuning to get the prediction metrics.*

*##Model 1*  
model <- store\_maxnode$`ntrees: 600 nodesize: 40`

final1 <- data.frame(actual = test$HeartDisease,

predict(model, newdata = test, type = "prob"))

final1$predict <- as.factor(ifelse(final1$Yes > 0.5, "Yes", "No"))

cm\_final1 <- confusionMatrix(final1$predict, test$HeartDisease, positive = 'Yes')

*##Model 2*

model2 <- store\_maxnode$`ntrees: 300 nodesize: 5`

final2 <- data.frame(actual = test$HeartDisease,

predict(model2, newdata = test, type = "prob"))

final2$predict <- as.factor(ifelse(final2$Yes > 0.5, "Yes", "No"))

cm\_final2 <- confusionMatrix(final2$predict, test$HeartDisease, positive = 'Yes')

*##Model 3*

model3 <- store\_maxnode$`ntrees: 600 nodesize: 5`

final3 <- data.frame(actual = test$HeartDisease,

predict(model3, newdata = test, type = "prob"))

final3$predict <- as.factor(ifelse(final3$Yes > 0.5, "Yes", "No"))

cm\_final3 <- confusionMatrix(final3$predict, test$HeartDisease, positive = 'Yes')

*##Model 4*

model4 <- store\_maxnode$`ntrees: 300 nodesize: 20`

final4 <- data.frame(actual = test$HeartDisease,

predict(model4, newdata = test, type = "prob"))

final4$predict <- as.factor(ifelse(final4$Yes > 0.5, "Yes", "No"))

cm\_final4 <- confusionMatrix(final4$predict, test$HeartDisease, positive = 'Yes')

*##Model 5*

model5 <- store\_maxnode$`ntrees: 600 nodesize: 20`

final5 <- data.frame(actual = test$HeartDisease,

predict(model5, newdata = test, type = "prob"))

final5$predict <- as.factor(ifelse(final5$Yes > 0.5, "Yes", "No"))

cm\_final5 <- confusionMatrix(final5$predict, test$HeartDisease, positive = 'Yes')

*##Model 6*

model6 <- store\_maxnode$`ntrees: 300 nodesize: 40`

final6 <- data.frame(actual = test$HeartDisease,

predict(model6, newdata = test, type = "prob"))

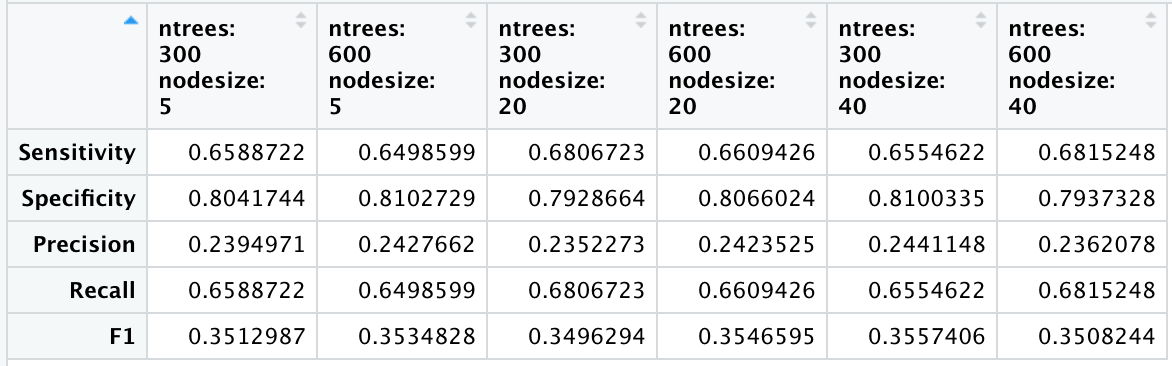
final6$predict <- as.factor(ifelse(final6$Yes > 0.5, "Yes", "No"))

cm\_final6 <- confusionMatrix(final6$predict, test$HeartDisease, positive = 'Yes')

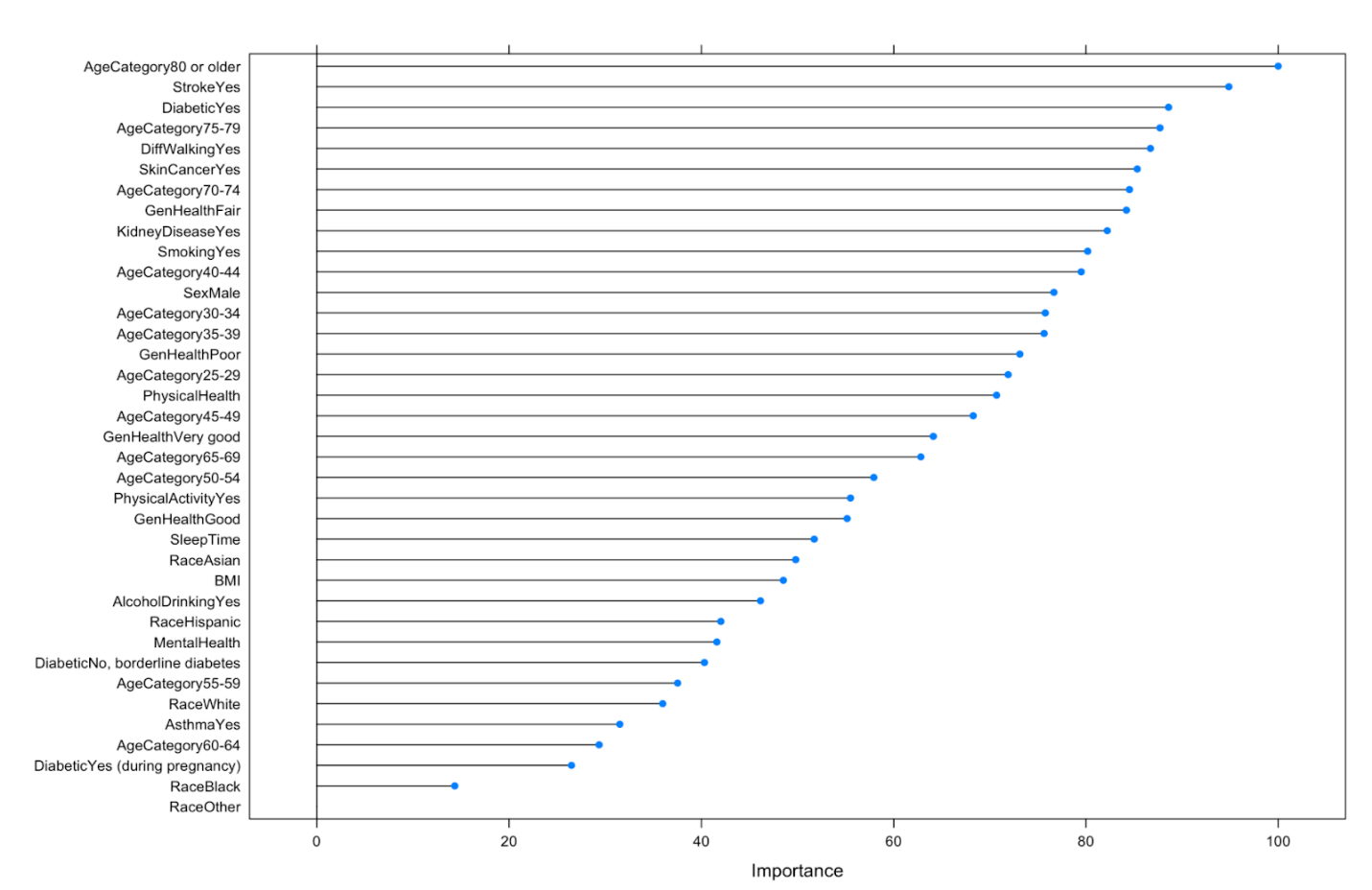
*#Compare predictions of 6 models*

compareTuning<-cbind(as.data.frame(cm\_final2$byClass),as.data.frame(cm\_final3$byClass), as.data.frame(cm\_final4$byClass), as.data.frame(cm\_final5$byClass), as.data.frame(cm\_final6$byClass), as.data.frame(cm\_final1$byClass))[c(1,2,5,6,7),]

colnames(compareTuning)<-c('ntrees: 300 nodesize: 5','ntrees: 600 nodesize: 5','ntrees: 300 nodesize: 20','ntrees: 600 nodesize: 20', 'ntrees: 300 nodesize: 40', 'ntrees: 600 nodesize: 40')



1. Feature Importance



Feature Importance assign assigns a score base on how useful a variable is at predicting a target variable.

In this case, age Category 80 or Older has the highest score in Feature Importance, meaning that it is the most useful variable for predicting Heart Disease.

## Conclusion

The under-sampling method with cross-validation was used to solve the imbalanced data problem which improves the precision, recall, and F1 metrics 10 times.

Tuning Hyperparameter on Random Forest model found that the model with mtry=1, ntree=600, and nodesize = 40 yields the best results of precision = 0.236, recall= 0.682, and F1=0.351.

Age Category 80andolder is the most important feature, the features StrokeYes and Diabetic yes are recorded as the next important features.

References

<https://www.datanovia.com/en/blog/ggplot-title-subtitle-and-caption/#:~:text=The%20default%20ggplot%20title%20alignment%20is%20not%20centered.,%2B%20theme%20%28plot.title%20%3D%20element_text%20%28hjust%20%3D%200.5%29%29>.

<https://www.cancer.org/cancer/cancer-causes/diet-physical-activity/body-weight-and-cancer-risk/adult-bmi.html> <https://r-charts.com/part-whole/stacked-bar-chart-ggplot2/>

<https://towardsdatascience.com/handling-imbalanced-datasets-in-machine-learning-7a0e84220f28>

<https://www.r-bloggers.com/2016/12/handling-class-imbalance-with-r-and-caret-an-introduction/>

<https://www.r-bloggers.com/2017/04/dealing-with-unbalanced-data-in-machine-learning/>

<https://www.r-bloggers.com/2020/03/tuning-random-forest-hyperparameters-with-tidytuesday-trees-data/>

<https://www.r-bloggers.com/2021/05/class-imbalance-handling-imbalanced-data-in-r/>

<https://machinelearningmastery.com/tune-machine-learning-algorithms-in-r/>