Classification of Heart Disease

Youssef Donia

3/27/2022

This document is for the classification section of the R project in CS 4375. I will be doing the classification based on the heart disease dataset below.

The dataset was found through this link: https://www.kaggle.com/datasets/kamilpytlak/personal-key-indicators-of-heart-disease

Reading from the Data Set

Let's read from the data set and output the first few rows

```
df <- read.csv("heart_2020.csv")
head(df)
```

```
##   HeartDisease   BMI Smoking AlcoholDrinking Stroke PhysicalHealth MentalHealth
## 1           No 16.60     Yes              No     No              3           30
## 2           No 20.34      No              No    Yes              0            0
## 3           No 26.58     Yes              No     No             20           30
## 4           No 24.21      No              No     No              0            0
## 5           No 23.71      No              No     No             28            0
## 6          Yes 28.87     Yes              No     No              6            0
##   DiffWalking    Sex AgeCategory  Race Diabetic PhysicalActivity GenHealth
## 1          No Female       55-59 White      Yes              Yes Very good
## 2          No Female 80 or older White       No              Yes Very good
## 3          No   Male       65-69 White      Yes              Yes      Fair
## 4          No Female       75-79 White       No               No      Good
## 5         Yes Female       40-44 White       No              Yes Very good
## 6         Yes Female       75-79 Black       No               No      Fair
##   SleepTime Asthma KidneyDisease SkinCancer
## 1         5    Yes            No        Yes
## 2         7     No            No         No
## 3         8    Yes            No         No
## 4         6     No            No        Yes
## 5         8     No            No         No
## 6        12     No            No         No
```

Data Cleaning

The "HeartDisease" column is heavily unbalanced, so to fix this I can downsample the "Yes" class in HeartDisease to balance.

```
# output list structure for the dataset
str(df)
```

```
## 'data.frame':    319795 obs. of  18 variables:
##  $ HeartDisease    : chr  "No" "No" "No" "No" ...
##  $ BMI             : num  16.6 20.3 26.6 24.2 23.7 ...
##  $ Smoking         : chr  "Yes" "No" "Yes" "No" ...
##  $ AlcoholDrinking : chr  "No" "No" "No" "No" ...
##  $ Stroke          : chr  "No" "Yes" "No" "No" ...
##  $ PhysicalHealth  : num  3 0 20 0 28 6 15 5 0 0 ...
##  $ MentalHealth    : num  30 0 30 0 0 0 0 0 0 0 ...
##  $ DiffWalking     : chr  "No" "No" "No" "No" ...
##  $ Sex             : chr  "Female" "Female" "Male" "Female" ...
##  $ AgeCategory     : chr  "55-59" "80 or older" "65-69" "75-79" ...
##  $ Race            : chr  "White" "White" "White" "White" ...
##  $ Diabetic        : chr  "Yes" "No" "Yes" "No" ...
##  $ PhysicalActivity: chr  "Yes" "Yes" "Yes" "No" ...
##  $ GenHealth       : chr  "Very good" "Very good" "Fair" "Good" ...
##  $ SleepTime       : num  5 7 8 6 8 12 4 9 5 10 ...
##  $ Asthma          : chr  "Yes" "No" "Yes" "No" ...
##  $ KidneyDisease   : chr  "No" "No" "No" "No" ...
##  $ SkinCancer      : chr  "Yes" "No" "No" "Yes" ...
```

```
# data cleaning
df <- na.omit(df)
table(df$HeartDisease)
```

```
##
##     No    Yes
## 292422  27373
```

```
# down sampling the "No" class in the Heart Disease column to balance the data set
yes <- which(df$HeartDisease == "Yes")
no <- which(df$HeartDisease == "No")

length(yes) # this should be about 27373
```

```
## [1] 27373
```

```
length(no) # this should be about 292422
```

```
## [1] 292422
```

```
no_downsample <- sample(no, length(yes)) # down sampling the No to have the same length as Yes
df <- df[c(no_downsample, yes),] # create a new data frame with the changed column

str(df) # the length now should be much smaller, and the number of No's and Yes' should be the same
```

```
## 'data.frame':    54746 obs. of  18 variables:
##  $ HeartDisease    : chr  "No" "No" "No" "No" ...
##  $ BMI             : num  23.6 27.9 31.9 28.3 34 ...
##  $ Smoking         : chr  "Yes" "Yes" "No" "No" ...
##  $ AlcoholDrinking : chr  "No" "No" "No" "No" ...
##  $ Stroke          : chr  "No" "No" "No" "No" ...
##  $ PhysicalHealth  : num  0 0 0 0 30 0 0 0 0 28 ...
##  $ MentalHealth    : num  29 0 15 0 30 0 3 0 5 30 ...
##  $ DiffWalking     : chr  "No" "No" "No" "No" ...
##  $ Sex             : chr  "Male" "Male" "Female" "Female" ...
##  $ AgeCategory     : chr  "45-49" "25-29" "70-74" "70-74" ...
##  $ Race            : chr  "White" "Black" "White" "Hispanic" ...
##  $ Diabetic        : chr  "No" "No" "No" "No" ...
##  $ PhysicalActivity: chr  "Yes" "Yes" "No" "Yes" ...
##  $ GenHealth       : chr  "Very good" "Excellent" "Very good" "Good" ...
##  $ SleepTime       : num  7 8 6 8 8 6 8 6 6 4 ...
##  $ Asthma          : chr  "No" "Yes" "Yes" "No" ...
##  $ KidneyDisease   : chr  "No" "No" "No" "No" ...
##  $ SkinCancer      : chr  "No" "No" "No" "No" ...
```

```
yes <- which(df$HeartDisease == "Yes")
no <- which(df$HeartDisease == "No")
length(yes) # should be 27373
```

```
## [1] 27373
```

```
length(no) # should be 27373
```

```
## [1] 27373
```

Now we're going to want to clean the rest of the data frame. To do this, let's remove some columns that we will not need and make columns that have limited outputs into factors.Some columns must also be removed, as they are still unbalanced and could give the model a false accuracy.

```
# convert variables into factors, delete variables that are too unbalanced
df$AlcoholDrinking <- NULL
df$Stroke <- NULL
df$Race <- NULL
df$Asthma <- NULL
df$KidneyDisease <- NULL
df$SkinCancer <- NULL
df$MentalHealth <- NULL

# for the KNN algorithm, switch columns to numeric/ factor depending on qualities
df$Smoking[df$Smoking == "Yes"] <- TRUE
df$Smoking[df$Smoking == "No"] <- FALSE
df$Smoking <- as.factor(df$Smoking) # seems good

df$DiffWalking[df$DiffWalking == "Yes"] <- TRUE
df$DiffWalking[df$DiffWalking == "No"] <- FALSE
df$DiffWalking <- as.factor(df$DiffWalking)

df$PhysicalActivity[df$PhysicalActivity == "Yes"] <- TRUE
df$PhysicalActivity[df$PhysicalActivity == "No"] <- FALSE
```

```
df$PhysicalActivity <- as.factor(df$PhysicalActivity)

df$Sex[df$Sex == "Male"] <- 0
df$Sex[df$Sex == "Female"] <- 1
df$Sex <- as.factor(df$Sex) # seems good

df$Diabetic[df$Diabetic == "Yes"] <- TRUE
df$Diabetic[df$Diabetic == "No"] <- FALSE
df$Diabetic[df$Diabetic == "Yes (during pregnancy)"] <- FALSE
df$Diabetic[df$Diabetic == "No, borderline diabetes"] <- TRUE
df$Diabetic <- as.factor(df$Diabetic)

df$GenHealth[df$GenHealth == "Poor"] <- 0
df$GenHealth[df$GenHealth == "Fair"] <- 1
df$GenHealth[df$GenHealth == "Good"] <- 2
df$GenHealth[df$GenHealth == "Very good"] <- 3
df$GenHealth[df$GenHealth == "Excellent"] <- 4
df$GenHealth <- as.factor(df$GenHealth)
df$GenHealth <- as.factor(df$GenHealth)

df$AgeCategory[df$AgeCategory == "18-24"] <- 0
df$AgeCategory[df$AgeCategory == "25-29"] <- 1
df$AgeCategory[df$AgeCategory == "30-34"] <- 2
df$AgeCategory[df$AgeCategory == "35-39"] <- 3
df$AgeCategory[df$AgeCategory == "40-44"] <- 4
df$AgeCategory[df$AgeCategory == "45-49"] <- 5
df$AgeCategory[df$AgeCategory == "50-54"] <- 6
df$AgeCategory[df$AgeCategory == "55-59"] <- 7
df$AgeCategory[df$AgeCategory == "60=64"] <- 8
df$AgeCategory[df$AgeCategory == "65-69"] <- 9
df$AgeCategory[df$AgeCategory == "70-74"] <- 10
df$AgeCategory[df$AgeCategory == "75-79"] <- 11
df$AgeCategory[df$AgeCategory == "80 or older"] <- 12
df$AgeCategory <- as.factor(df$AgeCategory)


# rename the "physical health" variable to "injury rate" as that seems more accurate
colnames(df)[which(names(df) == "PhysicalHealth")] <- "InjuryRate"

names(df) # check to see the updated columns
```

```
##  [1] "HeartDisease"     "BMI"              "Smoking"          "InjuryRate"
##  [5] "DiffWalking"      "Sex"              "AgeCategory"      "Diabetic"
##  [9] "PhysicalActivity" "GenHealth"        "SleepTime"
```

Data Visualization

After the data is cleaned, we are going to use R functions to visualize our data to help us understand the data we are working with better, and find good predictors for our Heart Disease variable.

First, let's print out a couple summaries for our data frame to check that everything is the way we want it.

```r
summary(df)
```

```
##  HeartDisease            BMI         Smoking        InjuryRate    
##  Length:54746       Min.   :12.02   FALSE:27759   Min.   : 0.000  
##  Class :character   1st Qu.:24.39   TRUE :26987   1st Qu.: 0.000  
##  Mode  :character   Median :27.71                 Median : 0.000  
##                     Mean   :28.79                 Mean   : 5.407  
##                     3rd Qu.:32.08                 3rd Qu.: 5.000  
##                     Max.   :91.82                 Max.   :30.000  
##                                                                   
##  DiffWalking   Sex        AgeCategory     Diabetic     PhysicalActivity
##  FALSE:41414   0:28797   10     : 7290   FALSE:41417   FALSE:15720     
##  TRUE :13332   1:25949   12     : 7219   TRUE :13329   TRUE :39026     
##                          9      : 6933                                 
##                          60-64  : 6203                                 
##                          11     : 5686                                 
##                          7      : 4747                                 
##                          (Other):16668                                 
##  GenHealth   SleepTime     
##  0: 4604   Min.   : 1.000  
##  1: 9737   1st Qu.: 6.000  
##  2:17182   Median : 7.000  
##  3:15552   Mean   : 7.113  
##  4: 7671   3rd Qu.: 8.000  
##            Max.   :24.000  
## 
```

```r
str(df)
```

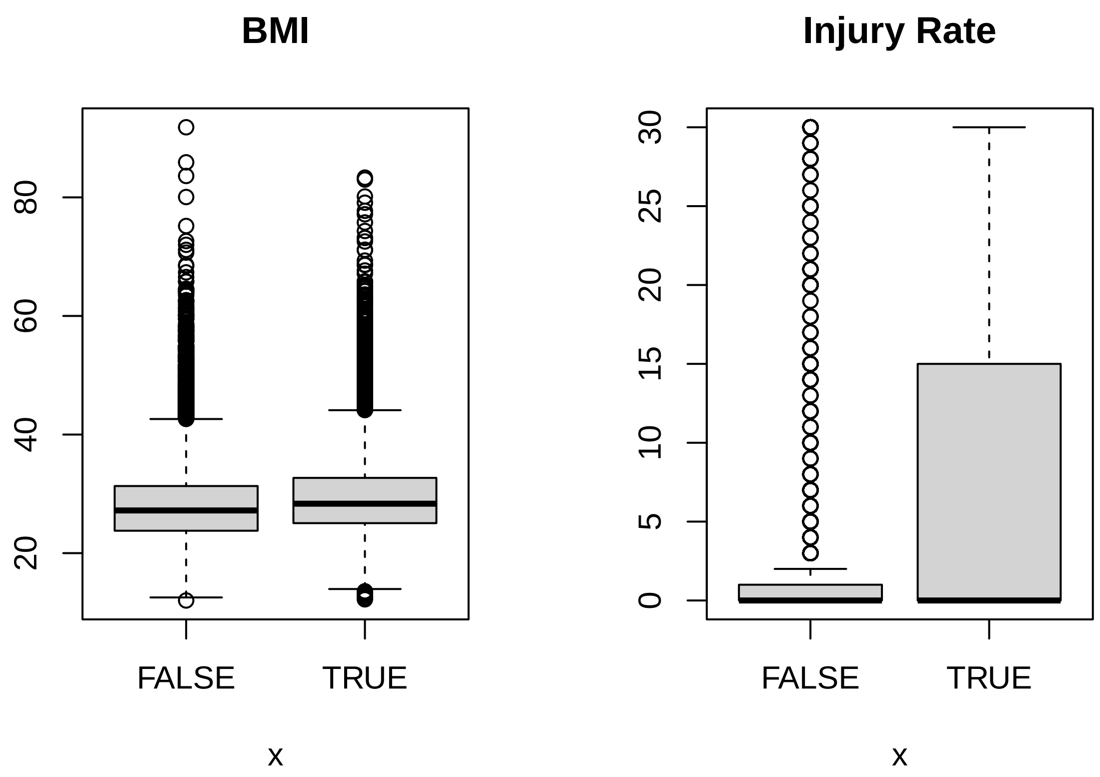
```
## 'data.frame':    54746 obs. of  11 variables:
##  $ HeartDisease    : chr  "No" "No" "No" "No" ...
##  $ BMI             : num  23.6 27.9 31.9 28.3 34 ...
##  $ Smoking         : Factor w/ 2 levels "FALSE","TRUE": 2 2 1 1 1 2 1 1 1 1 ...
##  $ InjuryRate      : num  0 0 0 0 30 0 0 0 0 28 ...
##  $ DiffWalking     : Factor w/ 2 levels "FALSE","TRUE": 1 1 1 1 2 1 1 1 1 1 ...
##  $ Sex             : Factor w/ 2 levels "0","1": 1 1 2 2 2 1 1 2 2 2 ...
##  $ AgeCategory     : Factor w/ 13 levels "0","1","10","11",..: 9 2 3 3 13 6 1 13 9 13 ...
##  $ Diabetic        : Factor w/ 2 levels "FALSE","TRUE": 1 1 1 1 1 1 1 1 1 2 ...
##  $ PhysicalActivity: Factor w/ 2 levels "FALSE","TRUE": 2 2 1 2 1 2 2 2 2 1 ...
##  $ GenHealth       : Factor w/ 5 levels "0","1","2","3",..: 4 5 4 3 3 4 4 3 4 1 ...
##  $ SleepTime       : num  7 8 6 8 8 6 8 6 6 4 ...
```

We can see that the rows we edited are now factors and the unbalanced variables have been deleted, so everything seems to be working well.

Let's plot some of our variables with the Heart Disease variable to get a good view of whether they would be good predictor variables or not. I wil start with the relationship between Heart Disease vs. BMI and Heart Disease vs. Injury Rate.

```r
# must change the Heart Disease column first from string to TRUE or FALSE to be numeric
df$HeartDisease[df$HeartDisease == "Yes"] <- TRUE
df$HeartDisease[df$HeartDisease == "No"] <- FALSE
df$HeartDisease <- as.factor(df$HeartDisease)
```

```r
# plot
par(mfrow=c(1,2))
plot(df$HeartDisease,df$BMI, main="BMI", ylab="", varwidth=TRUE)
plot(df$HeartDisease,df$InjuryRate, main="Injury Rate", ylab="", varwidth=TRUE)
```



Looking at these graphs, BMI and Injury Rate do not seem like good predictors, as the data is not very diverse. We can tell because in each graph, the medians are very close together. This means that we should not use these variables alone as predictors as they will not be very helpful.

Models

Let's create some models using three different algorithms to predict our response variable, which is Heart Disease. I will be using the Logistic Regression, Naive Bayes, and KNN algorithms to perform classification on the data.

Train and Test

First, we need to divide the train and test data.

```r
set.seed(1234)

i <- sample(1:nrow(df), nrow(df)*0.75, replace=FALSE)
train <- df[i,]
```

```
test <- df[-i,]
nrow(train) # size of train data
```

```
## [1] 41059
```

```
nrow(test) # size of test data
```

```
## [1] 13687
```

Logistic Regression

Let's start with a Logistic Regression Model. With this model, we will predict Heart Disease from all the other predictors. I am starting with this to see how well the model performs, and from there I can decide if there is any way to improve the accuracy.

```
glm1 <- glm(HeartDisease~., data=train, family=binomial)
summary(glm1)
```

```
##
## Call:
## glm(formula = HeartDisease ~ ., family = binomial, data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7598  -0.8037   0.2263   0.8254   2.8874
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)             -1.270113   0.147709  -8.599  < 2e-16 ***
## BMI                      0.009987   0.002025   4.931 8.17e-07 ***
## SmokingTRUE              0.390726   0.024512  15.940  < 2e-16 ***
## InjuryRate               0.008762   0.001633   5.366 8.04e-08 ***
## DiffWalkingTRUE          0.321098   0.033765   9.510  < 2e-16 ***
## Sex1                    -0.724499   0.024796 -29.218  < 2e-16 ***
## AgeCategory1             0.107842   0.153107   0.704 0.481211
## AgeCategory10            2.830108   0.112918  25.063  < 2e-16 ***
## AgeCategory11            3.104508   0.114445  27.127  < 2e-16 ***
## AgeCategory12            3.407551   0.113903  29.916  < 2e-16 ***
## AgeCategory2             0.457369   0.138371   3.305 0.000948 ***
## AgeCategory3             0.629334   0.132439   4.752 2.02e-06 ***
## AgeCategory4             0.962042   0.125597   7.660 1.86e-14 ***
## AgeCategory5             1.277901   0.121858  10.487  < 2e-16 ***
## AgeCategory6             1.706647   0.117163  14.566  < 2e-16 ***
## AgeCategory60-64         2.255769   0.113353  19.900  < 2e-16 ***
## AgeCategory7             1.955652   0.114939  17.015  < 2e-16 ***
## AgeCategory9             2.519510   0.112860  22.324  < 2e-16 ***
## DiabeticTRUE             0.515427   0.030032  17.163  < 2e-16 ***
## PhysicalActivityTRUE     0.010356   0.028807   0.359 0.719227
## GenHealth1              -0.419528   0.060403  -6.946 3.77e-12 ***
## GenHealth2              -0.962532   0.062215 -15.471  < 2e-16 ***
## GenHealth3              -1.619349   0.065018 -24.906  < 2e-16 ***
## GenHealth4           -2.109612   0.072303 -29.177  < 2e-16 ***
## SleepTime            -0.030744   0.007823  -3.930 8.49e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 56920  on 41058  degrees of freedom
## Residual deviance: 41571  on 41034  degrees of freedom
## AIC: 41621
##
## Number of Fisher Scoring iterations: 5
```

This model seems a little bit too cluttered, so let's try to make another mode with less predictors. Let's use smoking, BMI, Injury Rate, Diabetic, and GenHealth as predictors. I ommited the others because they were either too vague or in general are not associated as highly with heart disease.

```
glm2 <- glm(HeartDisease~Smoking+BMI+InjuryRate+Diabetic+GenHealth,data=train, family="binomial")
summary(glm2)
```

```
##
## Call:
## glm(formula = HeartDisease ~ Smoking + BMI + InjuryRate + Diabetic +
##     GenHealth, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2778  -0.9839   0.3984   1.0067   2.0443
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.904773   0.078588  11.513  < 2e-16 ***
## SmokingTRUE   0.527710   0.022117  23.860  < 2e-16 ***
## BMI          -0.006315   0.001784  -3.539 0.000402 ***
## InjuryRate    0.010251   0.001466   6.991 2.72e-12 ***
## DiabeticTRUE  0.883733   0.027849  31.733  < 2e-16 ***
## GenHealth1   -0.459177   0.056826  -8.080 6.45e-16 ***
## GenHealth2   -1.019213   0.057657 -17.677  < 2e-16 ***
## GenHealth3   -1.743637   0.059631 -29.240  < 2e-16 ***
## GenHealth4   -2.433867   0.066290 -36.715  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 56920  on 41058  degrees of freedom
## Residual deviance: 48295  on 41050  degrees of freedom
## AIC: 48313
##
## Number of Fisher Scoring iterations: 4
```

This model turned out to be worse than the first, so we will try again. Removing any more predictors than this will make the model fit worse, so we will remove only very selective predictors from the original model now, and see if we can improve it any more. The AgeCategory and the Physical activity coefficients were pretty low in some cases, and had high std. errors, so we will remove those.

```
glm3 <- glm(HeartDisease~.-AgeCategory-PhysicalActivity, data=train, family="binomial")
summary(glm3)
```

```
##
## Call:
## glm(formula = HeartDisease ~ . - AgeCategory - PhysicalActivity,
##     family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5064  -0.9068   0.2873   0.9672   2.2876
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)       0.709864   0.096885   7.327 2.36e-13 ***
## BMI              -0.011949   0.001835  -6.513 7.37e-11 ***
## SmokingTRUE       0.445520   0.022611  19.704  < 2e-16 ***
## InjuryRate        0.005233   0.001535   3.408 0.000653 ***
## DiffWalkingTRUE   0.714612   0.031310  22.824  < 2e-16 ***
## Sex1             -0.647424   0.022930 -28.235  < 2e-16 ***
## DiabeticTRUE      0.810374   0.028385  28.549  < 2e-16 ***
## GenHealth1       -0.380932   0.057819  -6.588 4.45e-11 ***
## GenHealth2       -0.868944   0.059172 -14.685  < 2e-16 ***
## GenHealth3       -1.559996   0.061482 -25.373  < 2e-16 ***
## GenHealth4       -2.260974   0.068197 -33.153  < 2e-16 ***
## SleepTime         0.062109   0.007379   8.417  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 56920  on 41058  degrees of freedom
## Residual deviance: 46991  on 41047  degrees of freedom
## AIC: 47015
##
## Number of Fisher Scoring iterations: 4
```

While this model is better than the one before it, it seems to still not be as good as the first, so we will use the first one.

Now, let's predict the probabilites, make binary predictions, and test the accuracy of the model.

```
# predict probabilities
glmprobs <- predict(glm1, newdata=test, type="response")

# make binary predictions
glmpred <- rep(TRUE, nrow(test))
glmpred[glmprobs<0.5] <- FALSE # if probability is less than 0.5, then we will predict that they do not
```

Now, let's test the accuracy of the model.

```r
# accuracy of the model
glmacc <- mean(glmpred == test$HeartDisease)
print(glmacc)
```

```
## [1] 0.7578724
```

```r
table(Predicted = glmpred, Actual = test$HeartDisease)
```

```
##          Actual
## Predicted FALSE TRUE
##     FALSE  5067 1513
##     TRUE   1801 5306
```

As we can see from the output for this code, the accuracy was about 76%. This is a moderately accurate model, but a good model for this type of prediction should probably have a higher accuracy. The diagonals in the table are where the predictions were correct. The model predicted that a person didn't have heart disease correctly 5116 times, and predicted that they did correctly 5288 times. It predicted a person didn't have heart disease wrongly 1531 times, and that they did wrongly 1752 times.

Naive Bayes

Let's use the Naive Bayes algorithm on this same dataset to see if we get better results that way.

```r
library(e1071)
nb1 <- naiveBayes(HeartDisease~., data=train) # use all predictors as that worked best with logistic re
nb1
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##     FALSE      TRUE
## 0.4994033 0.5005967
##
## Conditional probabilities:
##        BMI
## Y           [,1]     [,2]
##   FALSE 28.16727 6.307117
##   TRUE  29.39794 6.587281
##
##        Smoking
## Y           FALSE      TRUE
##   FALSE 0.6011704 0.3988296
##   TRUE  0.4128637 0.5871363
##
##        InjuryRate
## Y           [,1]      [,2]
##   FALSE 3.016776  7.514215
##   TRUE  7.823051 11.504520
##
##        DiffWalking
## Y           FALSE      TRUE
##   FALSE 0.8774445 0.1225555
##   TRUE  0.6329668 0.3670332
##
##        Sex
## Y               0         1
##   FALSE 0.4642770 0.5357230
##   TRUE  0.5941909 0.4058091
##
##        AgeCategory
## Y                 0           1          10          11          12           2
##   FALSE 0.071299683 0.057888320 0.090319434 0.060668130 0.064325774 0.064033163
##   TRUE  0.004670624 0.004719276 0.177143135 0.149362654 0.197090591 0.008076287
##        AgeCategory
## Y                 3           4           5           6       60-64           7
##   FALSE 0.068032187 0.070568154 0.070909534 0.081394782 0.104218483 0.092904170
##   TRUE  0.010898122 0.017904058 0.027342610 0.050987642 0.122555220 0.079011385
##        AgeCategory
## Y                 9
##   FALSE 0.103438186
##   TRUE  0.150238396
##
##        Diabetic
## Y           FALSE      TRUE
##   FALSE 0.8691051 0.1308949
##   TRUE  0.6420648 0.3579352
##
##        PhysicalActivity
## Y           FALSE      TRUE
##   FALSE 0.2133626 0.7866374
##   TRUE  0.3628004 0.6371996
##
##        GenHealth
## Y                0          1          2          3          4
##   FALSE 0.02774933 0.09695196 0.27841990 0.37151914 0.22535967
##   TRUE  0.14216211 0.25985210 0.34776686 0.19606889 0.05415004
##
##        SleepTime
## Y           [,1]     [,2]
##   FALSE 7.088856 1.393755
##   TRUE  7.132772 1.767472
```

Let's evaluate this model on the test data.

```
# predict off the test data
nb.pred <- predict(nb1, newdata=test, type="class")

# evaluate model
table(nb.pred, test$HeartDisease)
```

```
##
## nb.pred FALSE TRUE
##   FALSE  5410 2412
##   TRUE   1458 4407
```

```
nb.acc <- mean(nb.pred == test$HeartDisease)
print(paste("Accuracy: ", nb.acc))
```

```
## [1] "Accuracy:  0.717249945203478"
```

As we can see, the Naive Bayes model was slightly less accurate than the Logistic Regression model, with an accuracy of about 72%. It predicted more accurate FALSE's, but less accurate TRUE's.

Let's try one more algorithm to see if we can make a more accurate model than the one we had for logistic regression. For this model, I will use the KNN algorithm. Let's create a model that will hopefully surpass the other two in accuracy.

KNN

First, let's clean the data so that it is suitable for the KNN algorithm

```
library(class)

# convert all columns to numeric
for (i in 1:ncol(df)){
    if(!is.numeric(df[1,i])) {
      df[,i] <- as.integer(df[,i])
    }
}

predictors <- c("BMI", "Smoking", "InjuryRate", "DiffWalking", "Sex", "AgeCategory", "Diabetic", "Physi

# run normalization on the dataset to improve the performance of knn
normalize <- function(x) { (x - min(x))/(max(x) - min(x))}

df_normalized <- as.data.frame(lapply(df[,predictors], normalize))
summary(df_normalized)
```

```
##       BMI            Smoking         InjuryRate      DiffWalking
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.1550   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :0.1966   Median :0.0000   Median :0.0000   Median :0.0000
##  Mean   :0.2102   Mean   :0.4929   Mean   :0.1802   Mean   :0.2435
##  3rd Qu.:0.2514   3rd Qu.:1.0000   3rd Qu.:0.1667   3rd Qu.:0.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##       Sex          AgeCategory        Diabetic      PhysicalActivity
##  Min.   :0.000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.000   1st Qu.:0.2500   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :0.000   Median :0.5000   Median :0.0000   Median :1.0000
##  Mean   :0.474   Mean   :0.5386   Mean   :0.2435   Mean   :0.7129
##  3rd Qu.:1.000   3rd Qu.:0.8333   3rd Qu.:0.0000   3rd Qu.:1.0000
##  Max.   :1.000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##    GenHealth        SleepTime
##  Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.2500   1st Qu.:0.2174
##  Median :0.5000   Median :0.2609
##  Mean   :0.5546   Mean   :0.2658
##  3rd Qu.:0.7500   3rd Qu.:0.3043
##  Max.   :1.0000   Max.   :1.0000
```

Let's divide into train and test once again for KNN specifically

```
set.seed(1234)
i <- sample(1:nrow(df_normalized), nrow(df_normalized)*0.75, replace=FALSE)

train <- df_normalized[i,]
test <- df_normalized[-i,]

# put the response variable into variables to be put in the cl parameter in the knn
train.labels <- df[i,"HeartDisease"] # HeartDisease column
test.labels <- df[-i,"HeartDisease"]
```

Now, let's finally create our model

```
knn.pred <- knn(train, test, cl=train.labels, k=9) # keep k odd for classification

results <- knn.pred == test.labels
knn.acc <- length(which(results == TRUE)) / length(results)
print(paste("Accuracy: ", knn.acc))
```

```
## [1] "Accuracy:  0.726090450792723"
```

```
table(results, knn.pred)
```

```
##        knn.pred
## results    1    2
##   FALSE 1599 2150
##   TRUE  4718 5220
```

Results Analysis

Looking at the three algorithms I implemented, Logistic Regression definitely performed the best. This is probably because Naives Bayes assumes independent variables, which might not have been accurate for this data. While the KNN algorithm performed slightly better than the Naive Bayes one, it was still less accurate than the Logistic Regression. Some of the predictor variables were heavily correlated with Heart Disease, such as Diabetes, Smoking, and Difficulty Walking. This would lead to LogReg performing better, as it does under collinearity. I would rank the Logistic Regression algorithm the highest in this case, followed by Naive Bayes, and finally the KNN. The reason I ranked KNN third even though it performed better than the Naive Bayes model is because of the difficulty of implementation. I had to make many changes to the data, including completely converting all of my columns to numbers, which involved finding the unique values of columns that were factors and assigning them a number. This made the data less readable. The Naive Bayes model was much easier to implement and was only 1% less accurate than the KNN.

What was learned from the data?

Looking at which predictors were the most useful in the models, I could tell that Smoking, Difficulty Walking, and Diabetes were definitely correlated with Heart Disease. They were the most useful predictors, and proven to be correlated with heart disease. Sleep time was negatively correlated with heart disease, which showed that insomnia is associated with heart diseases. This could be useful in finding other variables that could also be correlated or negatively correlated with heart disease to further educate us on how to live our lives to avoid getting such horrible illnesses.