Final Project

Caterina Ponti and Disha Khati

2024-11-06

Write Up:

Question/Topic

Can we build a predictive model to identify individuals at high risk of heart disease using patient data, and what are the most significant predictors of heart disease in this data set?

Motivation

Heart disease is a leading global health concern, responsible for significant mortality and morbidity. This data set offers an opportunity to address real-world problems by analyzing patient information to predict heart disease risk. By building an effective predictive model, we can enable early detection and prevention and personalized care.

The Data set

We analyzed the Data of Patients (For Medical Field) data set in Kaggle https://www.kaggle.com/datasets/tarekmuhammed/patients-data-for-medical-field (https://www.kaggle.com/datasets/tarekmuhammed/patients-data-for-medical-field).

Challenges

1. The dataset exhibits a significant imbalance, with 224,429 observations for “No Heart Attack” compared to only 13,201 for “Had Heart Attack.” This imbalance can hinder the performance of predictive models, as they may become biased toward the majority class, leading to poor sensitivity for detecting heart attack cases.
2. When trying to predict whether a person had a heart attack, we found that the variable ‘Had Angina’ had a high predictive accuracy of 0.73. This suggests that the feature is potentially providing redundant, which could influence the model if not handled properly.
3. The dataset has a large number of rows (237,630) and 35 variables (columns), leading to challenges with computational efficiency, overfitting, and the potential for irrelevant or noisy features to dilute the model’s predictive power.

How did you overcome the challenges?

1. To address the class imbalance, we employed downsampling techniques. This involves reducing the number of samples from the majority class (“No Heart Attack”) to match the minority class (“Had Heart Attack”), thereby balancing the class distribution. This approach helps prevent the model from being biased toward the majority class and improves its ability to detect heart attack cases.
2. To avoid overfitting and ensure the model is not overly influenced by ‘Had Angina’, we performed feature selection and examined the correlation between variables.
3. To handle the large number of features and avoid overfitting, we applied techniques such as feature selection and sampling techniques to create a more manageable subset of the data for training.

What packages were needed for your analysis?

- `readxl` : Used for reading and importing Excel files into R.
- `ggplot2` :Essential for creating data visualizations such as histograms, scatter plots, and bar charts.
- `pheatmap` : Utilized for creating heatmaps, especially useful for visualizing correlation matrices.
- `dplyr` : For data manipulation (filtering and grouping).
- `reshape2` : To reshape data.
- `randomForest` : To implement Random Forest Regression for Variable Importance.
- `varImp` : To assess variable importance.
- `caret` : To build predictive models, data splitting, training and validation.
- `cluster` : To perform hierarchical clustering.
- `dendextend` - Used for visualizing dendograms created during hierarchical clustering.

Discussion: What did you learn from this experience?

From this experience, we learned the importance of performing Random Forest Regression for variable importance before fitting a Logistic Regression Model. The Random Forest method helped us identify which variables contributed most to predicting heart attacks, providing valuable insights. Additionally, hierarchical clustering allowed us to better understand the relationships between features and how they grouped in the data.

Regarding the dataset, we observed that fitting the model with all the variables gave us an accuracy of 0.79. However, the model's performance did not change significantly when we fitted it with only the most 5 important variables.

- Accuracy with 'HadAngina': 0.73
- Accuracy without 'HadAngina': 0.75
- Accuracy of Top 5 variables without 'HadAngina'(AgeCategory, ChestScan, HadStroke, HadDiabetes, GeneralHealth): 0.72

This demonstrates that while 'HadAngina' is a strong predictor, other variables still contribute meaningfully to the model's overall predictive power.

Additionally, after performing hierchical clustering, we noticed that Cluster 4 had the highest proportion of heart attack cases and features in the cluster included: most individual had ChestScan, majority did not have a stroke but experienced difficulty in walking, and HadAngina and AgeCategory were key features.

What more could you do with this project in the future?

In the future, there are two directions to explore:

- We could apply more advanced clustering techniques or adjust the number of clusters to see if any additional insights can be gained about the different patient groups.
- Experimenting with deep learning techniques could improve prediction accuracy by using complex relationships between variables.

Group Members Contributions:

Disha Khati: Exploratory Data Analysis (plots and data preparation), presentation slides, project's Write-Up.

Caterina Ponti: Model Building and Evaluation (Logistic Model Regression, Random Regression for Feature Importance and Hierarchical Clustering) and presentation slides.

```
library(readxl)
file.exists("/Users/caterinaponti/Desktop/BSDS100/Patients Data ( Used for Heart Disease Prediction ) 2.xlsx")
```

```
## [1] TRUE
```

```
data <- read_excel("/Users/caterinaponti/Desktop/BSDS100/Patients Data ( Used for Heart
Disease Prediction ) 2.xlsx")

head(data)
```

```
## # A tibble: 6 × 35
##   PatientID State   Sex    GeneralHealth AgeCategory  HeightInMeters
##       <dbl> <chr>   <chr>  <chr>         <chr>                 <dbl>
## 1         1 Alabama Female Fair          Age 75 to 79           1.63
## 2         2 Alabama Female Very good     Age 65 to 69           1.60
## 3         3 Alabama Male   Excellent     Age 60 to 64           1.78
## 4         4 Alabama Male   Very good     Age 70 to 74           1.78
## 5         5 Alabama Female Good          Age 50 to 54           1.68
## 6         6 Alabama Male   Very good     Age 75 to 79           1.85
## # ℹ 29 more variables: WeightInKilograms <dbl>, BMI <dbl>,
## #   HadHeartAttack <dbl>, HadAngina <dbl>, HadStroke <dbl>, HadAsthma <dbl>,
## #   HadSkinCancer <dbl>, HadCOPD <dbl>, HadDepressiveDisorder <dbl>,
## #   HadKidneyDisease <dbl>, HadArthritis <dbl>, HadDiabetes <chr>,
## #   DeafOrHardOfHearing <dbl>, BlindOrVisionDifficulty <dbl>,
## #   DifficultyConcentrating <dbl>, DifficultyWalking <dbl>,
## #   DifficultyDressingBathing <dbl>, DifficultyErrands <dbl>, …
```

```
names(data)
```

```
##  [1] "PatientID"                "State"
##  [3] "Sex"                      "GeneralHealth"
##  [5] "AgeCategory"              "HeightInMeters"
##  [7] "WeightInKilograms"        "BMI"
##  [9] "HadHeartAttack"           "HadAngina"
## [11] "HadStroke"                "HadAsthma"
## [13] "HadSkinCancer"            "HadCOPD"
## [15] "HadDepressiveDisorder"    "HadKidneyDisease"
## [17] "HadArthritis"             "HadDiabetes"
## [19] "DeafOrHardOfHearing"      "BlindOrVisionDifficulty"
## [21] "DifficultyConcentrating"  "DifficultyWalking"
## [23] "DifficultyDressingBathing" "DifficultyErrands"
## [25] "SmokerStatus"             "ECigaretteUsage"
## [27] "ChestScan"                "RaceEthnicityCategory"
## [29] "AlcoholDrinkers"          "HIVTesting"
## [31] "FluVaxLast12"             "PneumoVaxEver"
## [33] "TetanusLast10Tdap"        "HighRiskLastYear"
## [35] "CovidPos"
```

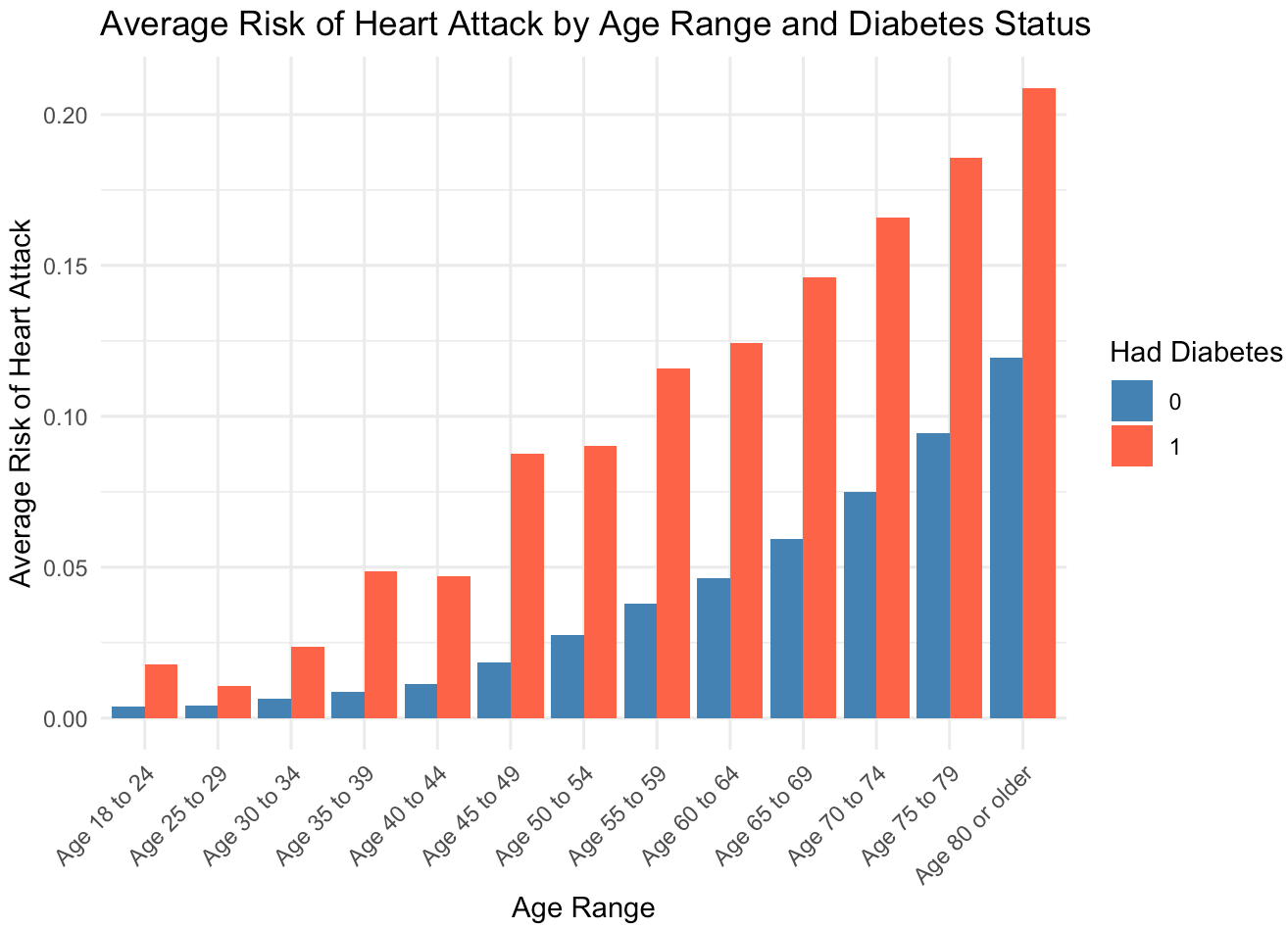
```
library(ggplot2)
library(dplyr)
```

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

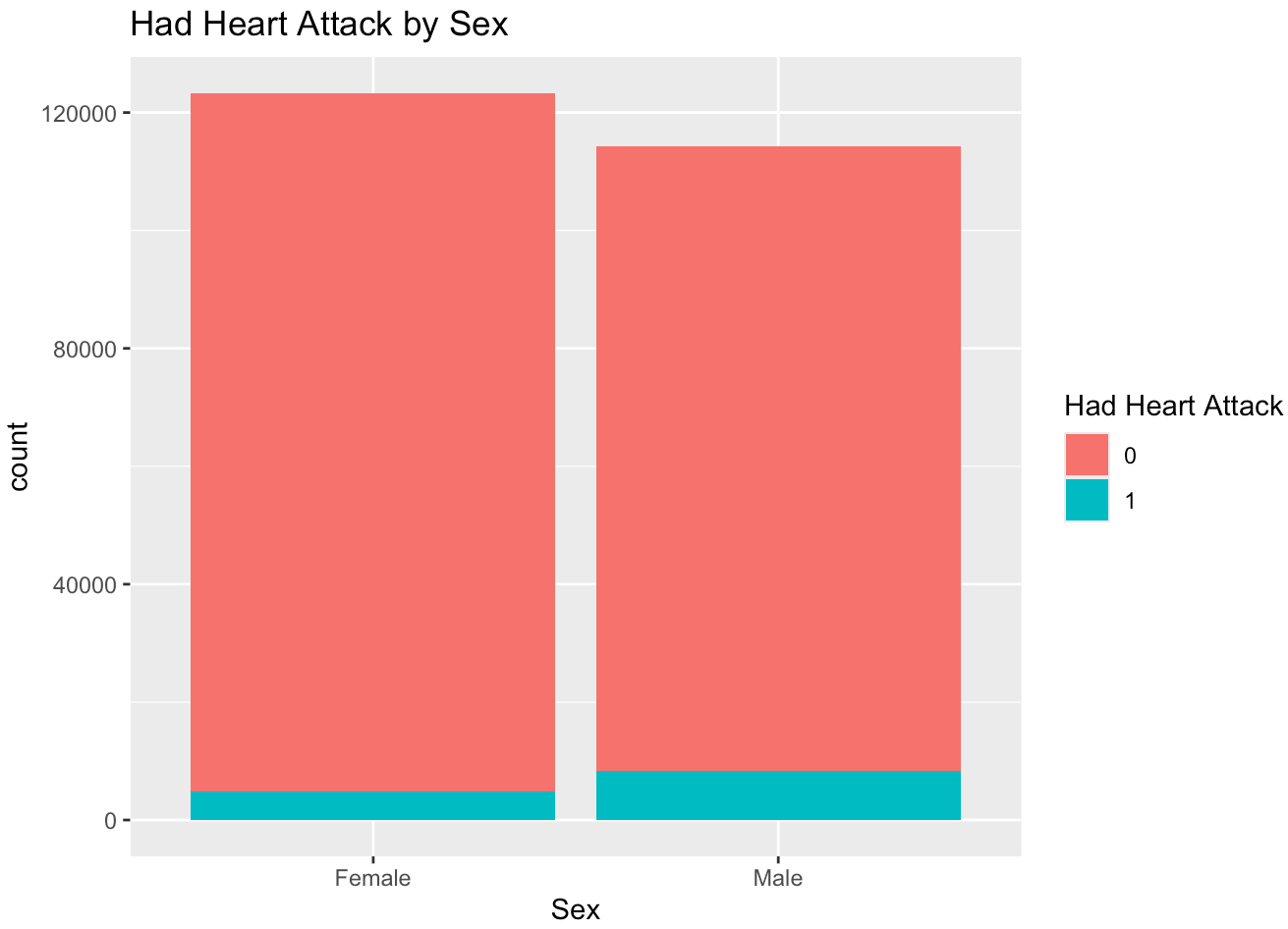
```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

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

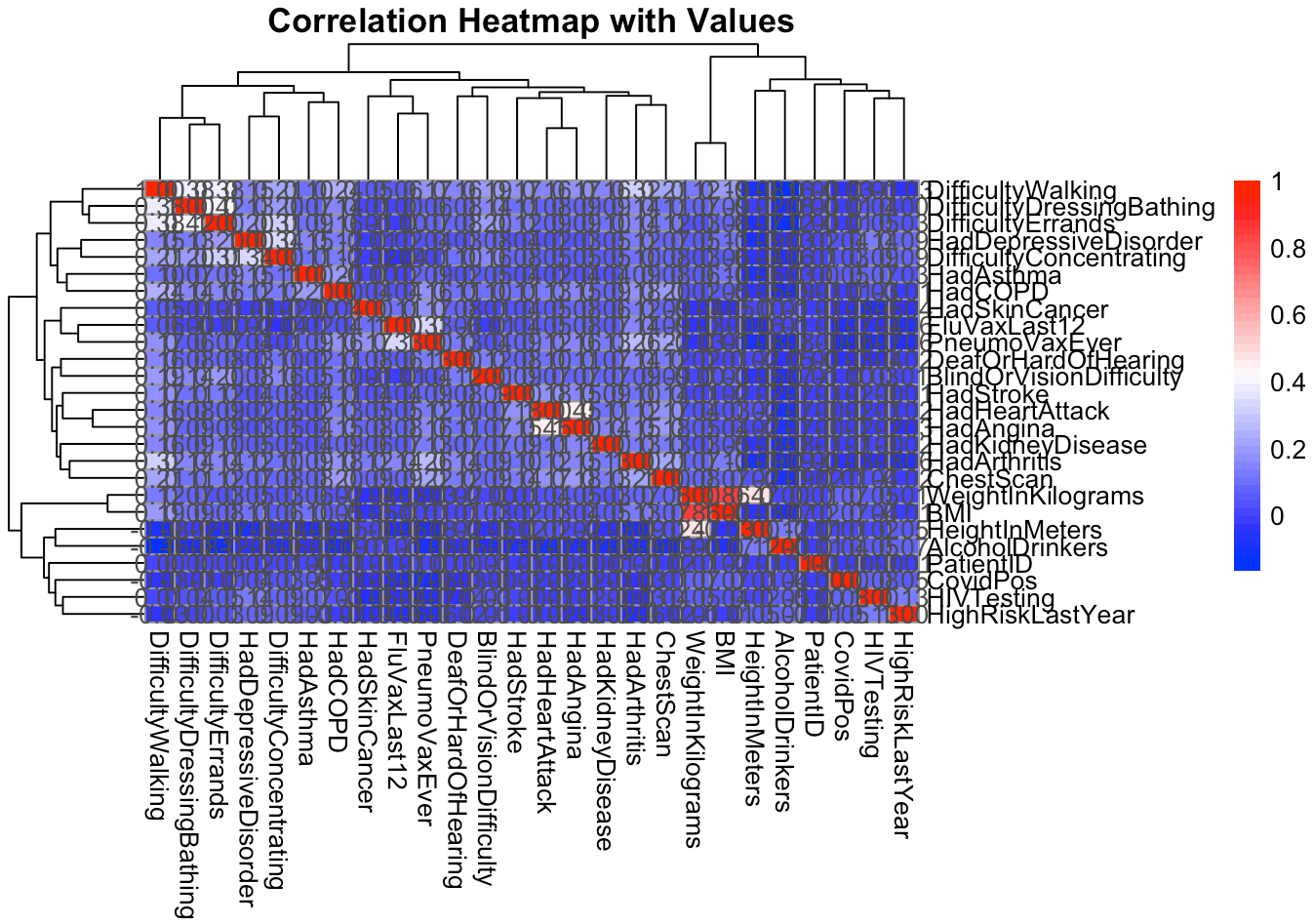
```
patients_df <- data %>%
  mutate(HadDiabetes = ifelse(HadDiabetes == "Yes", 1, 0))

age_diabetes_risk <- aggregate(HadHeartAttack ~ AgeCategory + HadDiabetes, data = patien
ts_df, FUN = mean)

# Plot the average risk by age category and diabetes status
ggplot(age_diabetes_risk, aes(x = factor(AgeCategory), y = HadHeartAttack, fill = factor
(HadDiabetes))) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(
    title = "Average Risk of Heart Attack by Age Range and Diabetes Status",
    x = "Age Range",
    y = "Average Risk of Heart Attack",
    fill = "Had Diabetes"
  ) +
  scale_fill_manual(values = c("steelblue", "tomato")) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```
#Stacked Bar Plot for Health Conditions by Sex
ggplot(data, aes(x=factor(Sex), fill = factor(HadHeartAttack))) + geom_bar(position="sta
ck") + labs(title= "Had Heart Attack by Sex", x = "Sex", fill= "Had Heart Attack")
```




```
library(pheatmap) # For melting the correlation matrix
library(dplyr)
library(reshape2)

df_numeric <- data[sapply(data, is.numeric)]

corr <- cor(df_numeric)

corr_melted <- melt(corr)

# Plot the heatmap
pheatmap(corr,
         color = colorRampPalette(c("blue", "white", "red"))(50),
         display_numbers = TRUE,
         fontsize_number = 10,
         main = "Correlation Heatmap with Values")
```




```
#who had a heart attack
class_counts <- table(data$HadHeartAttack)
print(class_counts)
```

```
##
##      0      1
## 224429  13201
```

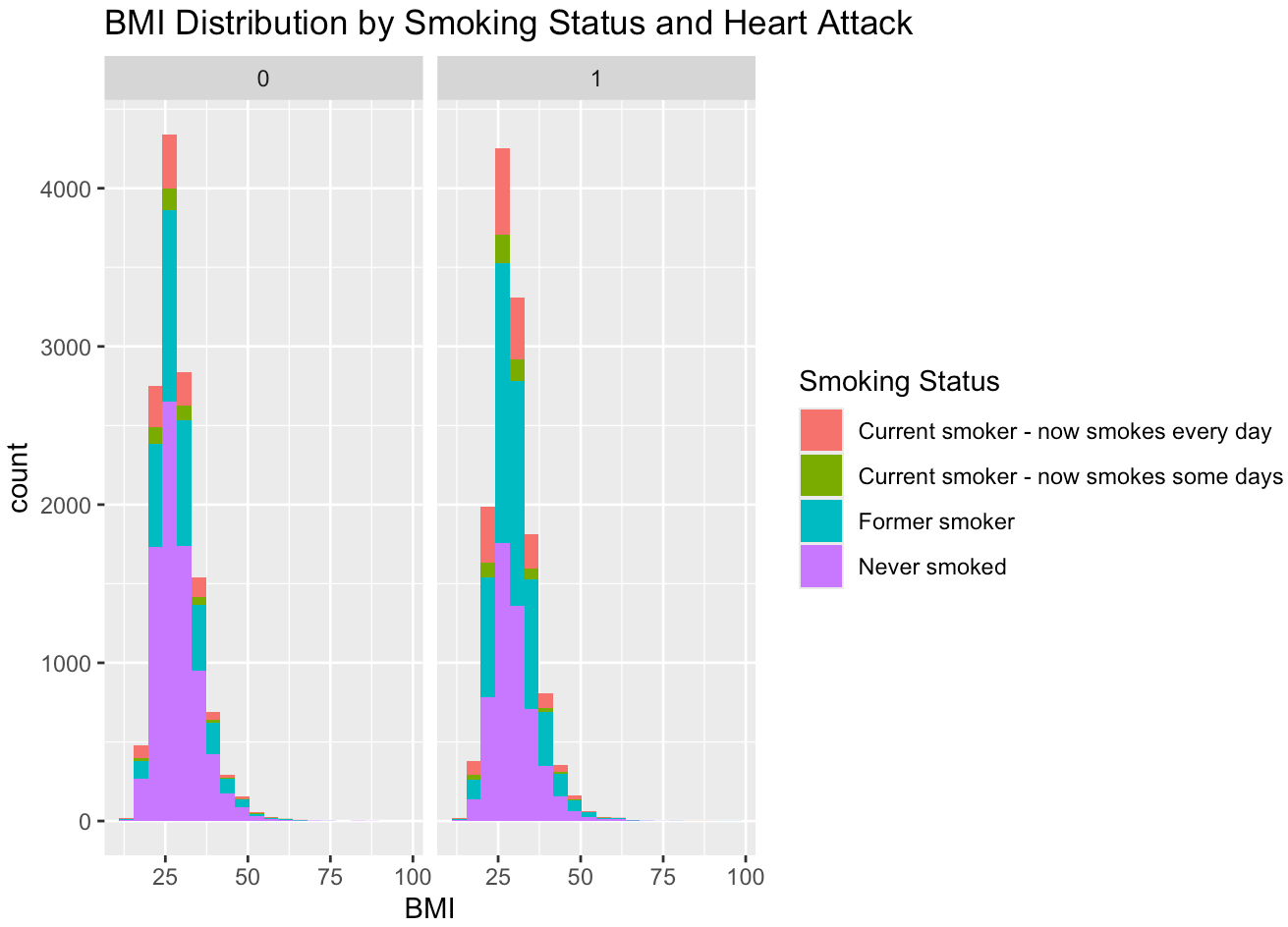
```
class_0 <- data %>% filter(HadHeartAttack == 0)
class_1 <- data %>% filter(HadHeartAttack == 1)
```

```
class_0_downsampled <- class_0[sample(nrow(class_0), size = nrow(class_1)), ]

balanced_data <- bind_rows(class_0_downsampled, class_1)

table(balanced_data$HadHeartAttack)
```

```
##
##     0     1
## 13201 13201
```

#Facet Grid for BMI by Smoking Status and Health Conditions

```
ggplot(balanced_data, aes(x=BMI, fill=factor(SmokerStatus))) + geom_histogram(bins=20) +
facet_wrap(~ factor(HadHeartAttack)) + labs(
  title="BMI Distribution by Smoking Status and Heart Attack",
  x = "BMI",
  fill = "Smoking Status"
)
```




```
library(randomForest)
```

```
## randomForest 4.7-1.2
```

```
## 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
```

```
library(varImp)
```

```
## Loading required package: measures
```

```
## Loading required package: party
```

```
## Loading required package: grid
```

```
## Loading required package: mvtnorm
```

```
## Loading required package: modeltools
```

```
## Loading required package: stats4
```

```
## Loading required package: strucchange
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

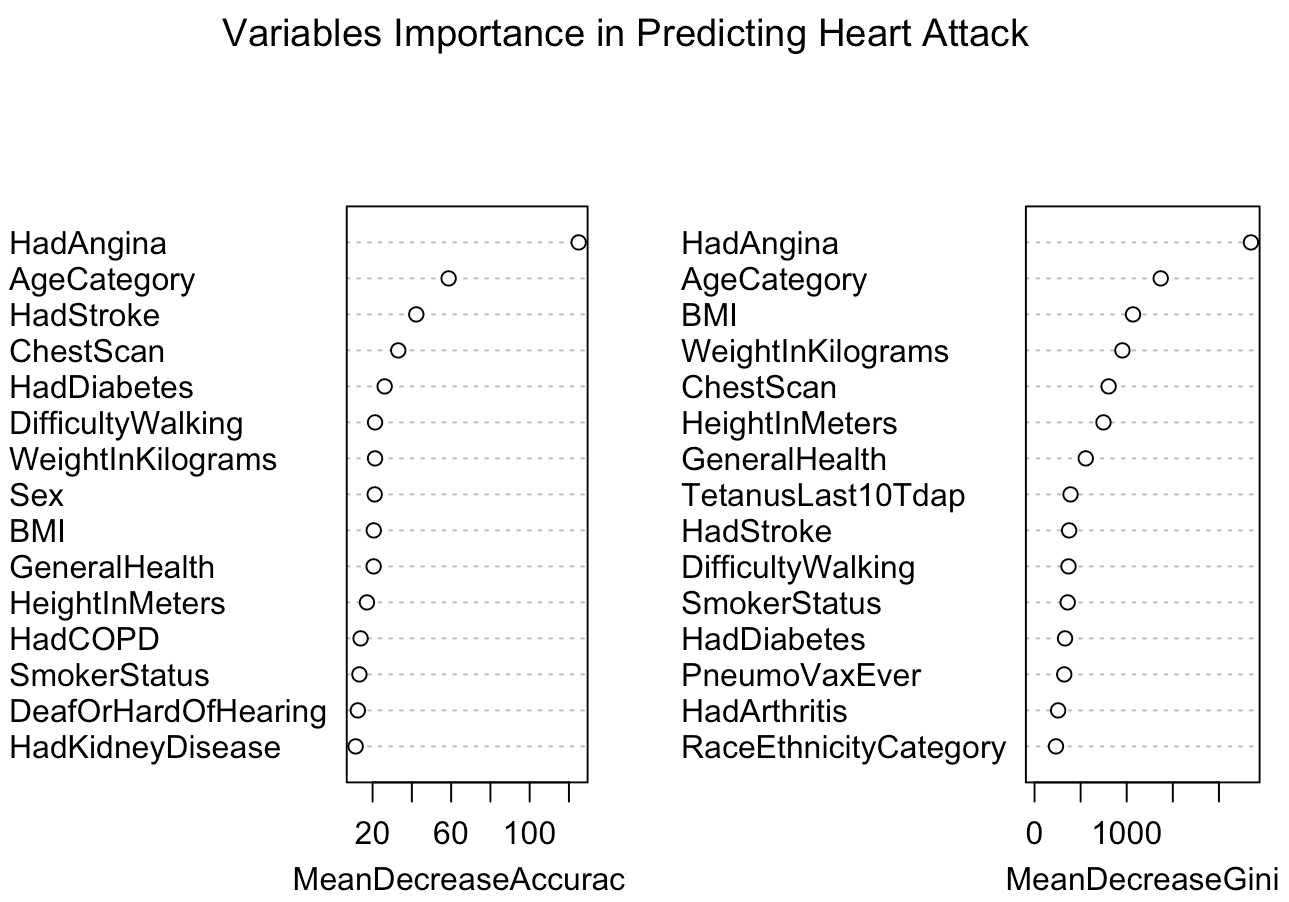
```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Loading required package: sandwich
```

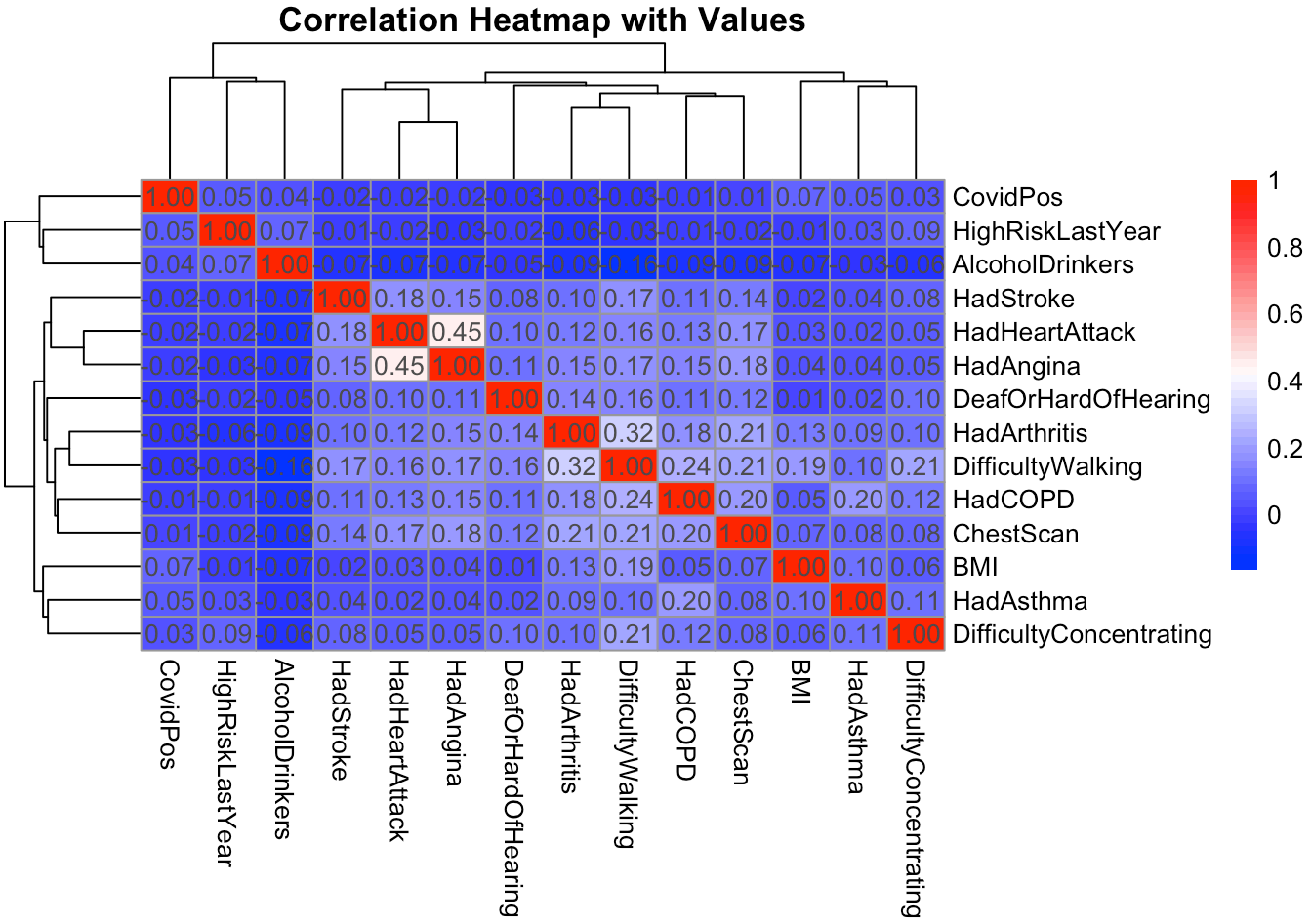
```
##
## Attaching package: 'party'
```

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

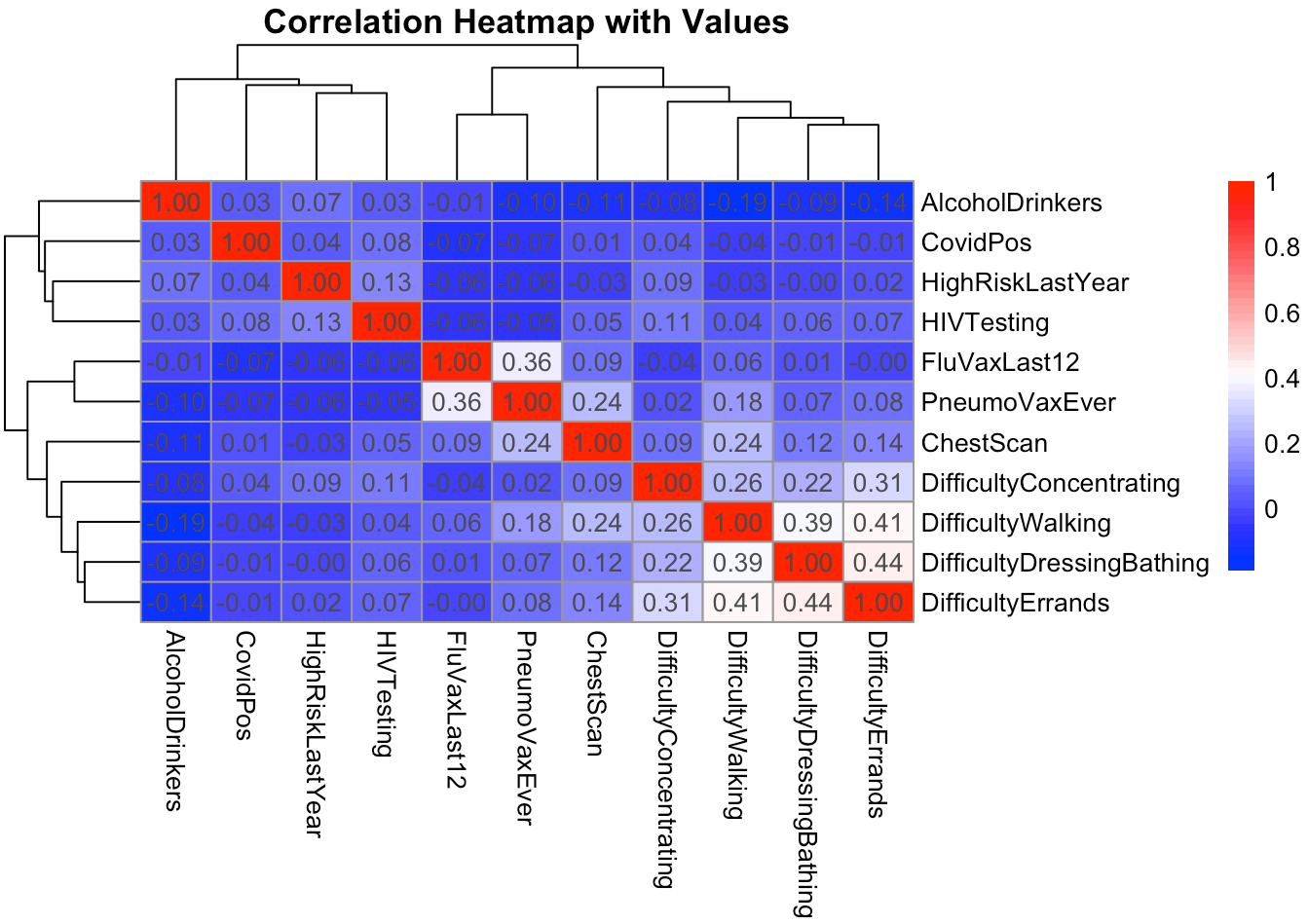
```
balanced_data$HadHeartAttack <- as.factor(balanced_data$HadHeartAttack)
balanced_data <- subset(balanced_data, select=-c(PatientID, State))

rf_model <- randomForest(HadHeartAttack ~ ., data = balanced_data, importance = TRUE, mt
ry=6, ntree=100)

importance_values <- importance(rf_model)

sorted_by_accuracy <- importance_values[order(-importance_values[, "MeanDecreaseAccurac
y"]), ]

varImpPlot(rf_model, n.var = 15, main="Variables Importance in Predicting Heart Attack")
```



```
subset_data <- df_numeric[,c( 'HadHeartAttack', 'HadArthritis', 'DeafOrHardOfHearing',
'BMI','HadAngina', 'HadStroke', 'HadAsthma', 'HadCOPD','DifficultyConcentrating', 'Diffi
cultyWalking', 'ChestScan', 'HighRiskLastYear', 'AlcoholDrinkers','CovidPos' )]

corr <- cor(subset_data)

corr_melted <- melt(corr)

# Plot the heatmap
pheatmap(corr,
         color = colorRampPalette(c("blue", "white", "red"))(50),
         display_numbers = TRUE,
         fontsize_number = 10,
         main = "Correlation Heatmap with Values")
```



```
subset_data2 <- balanced_data[,c('DifficultyConcentrating', 'DifficultyWalking',
       'DifficultyDressingBathing', 'DifficultyErrands','ChestScan','HighRiskLastYear',
'CovidPos','AlcoholDrinkers', 'HIVTesting', 'FluVaxLast12', 'PneumoVaxEver')]

corr <- cor(subset_data2)

corr_melted <- melt(corr)

# Plot the heatmap
pheatmap(corr,
         color = colorRampPalette(c("blue", "white", "red"))(50),
         display_numbers = TRUE,
         fontsize_number = 10,
         main = "Correlation Heatmap with Values")
```




```
#select categorical columns
categorical_cols <- names(balanced_data)[sapply(balanced_data, is.character)]

# Apply encoding
for (col in categorical_cols) {
  balanced_data[[col]] <- as.integer(factor(balanced_data[[col]]))
}
```

```
#Define Target Variables and Features
X <- subset(balanced_data, select = -HadHeartAttack)
y <- balanced_data$HadHeartAttack
```

```
library(caret)
```

```
## Loading required package: lattice
```

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

```
## The following object is masked from 'package:varImp':
##
##     varImp
```

```
## The following objects are masked from 'package:measures':
##
##     MAE, RMSE
```

```
set.seed(42)

train_indices <- createDataPartition(y, p = 0.8, list = FALSE)
X_train <- X[train_indices, ]
X_test <- X[-train_indices, ]
y_train <- y[train_indices]
y_test <- y[-train_indices]
```

Fitting a Logistic Regression Model against all variables

```
suppressWarnings({model <- train(x = X_train, y = y_train, method = "glm", family = "bin
omial")})

predictions <- predict(model, newdata = X_test)

accuracy <- mean(predictions == y_test)
print(paste("Accuracy: ", accuracy))
```

```
## [1] "Accuracy:  0.794318181818182"
```

```
#Accuracy:  0.789204545454545

conf_matrix <- confusionMatrix(predictions, y_test)
print(conf_matrix)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 2227  673
##          1  413 1967
##
##                Accuracy : 0.7943
##                  95% CI : (0.7832, 0.8052)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5886
##
##  Mcnemar's Test P-Value : 3.862e-15
##
##             Sensitivity : 0.8436
##             Specificity : 0.7451
##          Pos Pred Value : 0.7679
##          Neg Pred Value : 0.8265
##              Prevalence : 0.5000
##          Detection Rate : 0.4218
##    Detection Prevalence : 0.5492
##       Balanced Accuracy : 0.7943
##
##        'Positive' Class : 0
##
```

```
#top 10 for variable importance
X_2 <- balanced_data[,c('HadAngina', 'AgeCategory', 'ChestScan', 'HadStroke', 'Difficult
yWalking', 'Sex', 'HadDiabetes', 'WeightInKilograms', 'GeneralHealth', 'HeightInMeter
s')]

y <- balanced_data$HadHeartAttack

set.seed(43)

train_indices <- createDataPartition(y, p = 0.8, list = FALSE)
X_train2 <- X_2[train_indices, ]
X_test2 <- X_2[-train_indices, ]
y_train2 <- y[train_indices]
y_test2 <- y[-train_indices]
```

Fitting a Logistic Regression Model with top 10 variables for importance

```
suppressWarnings({model2 <- train(x = X_train2, y = y_train2, method = "glm", family =
"binomial")})

predictions2 <- predict(model2, newdata = X_test2)

accuracy2 <- mean(predictions2 == y_test2)
print(paste("Accuracy: ", accuracy2))
```

```
## [1] "Accuracy:  0.796022727272727"
```

```
#Accuracy:  0.790530303030303

conf_matrix2 <- confusionMatrix(predictions2, y_test2)
print(conf_matrix2)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 2232  669
##          1  408 1971
##
##                Accuracy : 0.796
##                  95% CI : (0.7849, 0.8068)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.592
##
##  Mcnemar's Test P-Value : 2.327e-15
##
##             Sensitivity : 0.8455
##             Specificity : 0.7466
##          Pos Pred Value : 0.7694
##          Neg Pred Value : 0.8285
##              Prevalence : 0.5000
##          Detection Rate : 0.4227
##    Detection Prevalence : 0.5494
##       Balanced Accuracy : 0.7960
##
##        'Positive' Class : 0
##
```

```
#top 5 for variable importance
X_3 <- balanced_data[,c('HadAngina', 'AgeCategory', 'ChestScan', 'HadStroke', 'Difficult
yWalking')]

y <- balanced_data$HadHeartAttack

set.seed(43)

train_indices <- createDataPartition(y, p = 0.8, list = FALSE)
X_train3 <- X_3[train_indices, ]
X_test3 <- X_3[-train_indices, ]
y_train3 <- y[train_indices]
y_test3 <- y[-train_indices]
```

Fitting a Logistic Regression Model with top 5 variables for importance

```
suppressWarnings({model3 <- train(x = X_train3, y = y_train3, method = "glm", family =
"binomial")})

predictions3 <- predict(model3, newdata = X_test3)

accuracy3 <- mean(predictions3 == y_test3)
print(paste("Accuracy: ", accuracy3))
```

```
## [1] "Accuracy:  0.787121212121212"
```

```
#Accuracy:  0.78030303030303

conf_matrix3 <- confusionMatrix(predictions3, y_test3)
print(conf_matrix3)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 2261  745
##          1  379 1895
##
##                Accuracy : 0.7871
##                  95% CI : (0.7758, 0.7981)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5742
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.8564
##             Specificity : 0.7178
##          Pos Pred Value : 0.7522
##          Neg Pred Value : 0.8333
##              Prevalence : 0.5000
##          Detection Rate : 0.4282
##    Detection Prevalence : 0.5693
##       Balanced Accuracy : 0.7871
##
##        'Positive' Class : 0
##
```

Fitted a Logistic Regression Model with only HadAngina to predict HadHeartAttack

```
#Only HadAngina
X_4 <- balanced_data[,c('HadAngina')]

y <- balanced_data$HadHeartAttack

set.seed(43)

train_indices <- createDataPartition(y, p = 0.8, list = FALSE)
X_train4 <- X_4[train_indices, ]
X_test4 <- X_4[-train_indices, ]
y_train4 <- y[train_indices]
y_test4 <- y[-train_indices]

suppressWarnings({model4 <- train(x = X_train4, y = y_train4, method = "glm", family =
"binomial")})

predictions4 <- predict(model4, newdata = X_test4)

accuracy4 <- mean(predictions4 == y_test4)
print(paste("Accuracy: ", accuracy4))
```

```
## [1] "Accuracy:  0.734848484848485"
```

```
#Accuracy:  0.734469696969697

conf_matrix4 <- confusionMatrix(predictions4, y_test4)
print(conf_matrix4)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 2561 1321
##          1   79 1319
##
##                Accuracy : 0.7348
##                  95% CI : (0.7227, 0.7467)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4697
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9701
##             Specificity : 0.4996
##          Pos Pred Value : 0.6597
##          Neg Pred Value : 0.9435
##              Prevalence : 0.5000
##          Detection Rate : 0.4850
##    Detection Prevalence : 0.7352
##       Balanced Accuracy : 0.7348
##
##        'Positive' Class : 0
##
```

Fitted a Logistic Regression model with all the variables except HadAngina to predict HadHeartAttack

```
#Fit the model without 'HadAngina'
X_5 <- subset(balanced_data, select=-c(HadAngina, HadHeartAttack))

y <- balanced_data$HadHeartAttack

set.seed(43)

train_indices <- createDataPartition(y, p = 0.8, list = FALSE)
X_train5 <- X_5[train_indices, ]
X_test5 <- X_5[-train_indices, ]
y_train5 <- y[train_indices]
y_test5 <- y[-train_indices]

suppressWarnings({model5 <- train(x = X_train5, y = y_train5, method = "glm", family =
"binomial")})

predictions5 <- predict(model5, newdata = X_test5)

accuracy5 <- mean(predictions5 == y_test5)
print(paste("Accuracy: ", accuracy5))
```

```
## [1] "Accuracy:  0.758143939393939"
```

```
#Accuracy:  0.750568181818182

conf_matrix5 <- confusionMatrix(predictions5, y_test5)
print(conf_matrix4)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 2561 1321
##          1   79 1319
##
##                Accuracy : 0.7348
##                  95% CI : (0.7227, 0.7467)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4697
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9701
##             Specificity : 0.4996
##          Pos Pred Value : 0.6597
##          Neg Pred Value : 0.9435
##              Prevalence : 0.5000
##          Detection Rate : 0.4850
##    Detection Prevalence : 0.7352
##       Balanced Accuracy : 0.7348
##
##        'Positive' Class : 0
##
```

Fitted a Logistic Regression model with top 5 variables except HadAngina to predict HadHeartAttack

```
X_6 <- balanced_data[,c('AgeCategory', 'ChestScan', 'HadStroke', 'HadDiabetes', 'General
Health')]

y <- balanced_data$HadHeartAttack

set.seed(43)

train_indices <- createDataPartition(y, p = 0.8, list = FALSE)
X_train6 <- X_6[train_indices, ]
X_test6 <- X_6[-train_indices, ]
y_train6 <- y[train_indices]
y_test6 <- y[-train_indices]

suppressWarnings({model6 <- train(x = X_train6, y = y_train6, method = "glm", family =
"binomial")})

predictions6 <- predict(model6, newdata = X_test6)

accuracy6 <- mean(predictions6 == y_test6)
print(paste("Accuracy: ", accuracy6))
```

```
## [1] "Accuracy:  0.725757575757576"
```

```
#Accuracy:  0.723106060606061

conf_matrix6 <- confusionMatrix(predictions6, y_test6)
print(conf_matrix6)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1896  704
##          1  744 1936
##
##                Accuracy : 0.7258
##                  95% CI : (0.7135, 0.7378)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.4515
##
##  Mcnemar's Test P-Value : 0.3054
##
##             Sensitivity : 0.7182
##             Specificity : 0.7333
##          Pos Pred Value : 0.7292
##          Neg Pred Value : 0.7224
##              Prevalence : 0.5000
##          Detection Rate : 0.3591
##    Detection Prevalence : 0.4924
##       Balanced Accuracy : 0.7258
##
##        'Positive' Class : 0
##
```

```
#splitting the data set in male and females
#Female = 1, Male = 2
female = balanced_data[balanced_data$Sex == 1, ]
male = balanced_data[balanced_data$Sex == 2, ]
dim(female)
```

```
## [1] 11811    33
```

```
dim(male)
```

```
## [1] 14591    33
```

Fitting Logistic Regression for Female Dataset

```
X_female <-  subset(female, select = -HadHeartAttack)

y_female <- female$HadHeartAttack

set.seed(43)

train_indices <- createDataPartition(y_female, p = 0.8, list = FALSE)
X_train <- X_female[train_indices, ]
X_test <- X_female[-train_indices, ]
y_train <- y_female[train_indices]
y_test <- y_female[-train_indices]

suppressWarnings({model_female <- train(x = X_train, y = y_train, method = "glm", family
= "binomial")})

predictions_female <- predict(model_female, newdata = X_test)

accuracy_female <- mean(predictions_female == y_test)
print(paste("Accuracy: ", accuracy_female))
```

```
## [1] "Accuracy:  0.793395427603726"
```

```
# Accuracy:  0.802013422818792

conf_matrix_female <- confusionMatrix(predictions_female, y_test)
print(conf_matrix_female)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1213  310
##          1  178  661
##
##                Accuracy : 0.7934
##                  95% CI : (0.7765, 0.8096)
##     No Information Rate : 0.5889
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5644
##
##  Mcnemar's Test P-Value : 3.028e-09
##
##             Sensitivity : 0.8720
##             Specificity : 0.6807
##          Pos Pred Value : 0.7965
##          Neg Pred Value : 0.7878
##              Prevalence : 0.5889
##          Detection Rate : 0.5135
##    Detection Prevalence : 0.6448
##       Balanced Accuracy : 0.7764
##
##        'Positive' Class : 0
##
```

Fitting Logistic Regression for Male Dataset

```
X_male <-  subset(male, select = -HadHeartAttack)

y_male <- male$HadHeartAttack

set.seed(43)

train_indices <- createDataPartition(y_male, p = 0.8, list = FALSE)
X_train <- X_male[train_indices, ]
X_test <- X_male[-train_indices, ]
y_train <- y_male[train_indices]
y_test <- y_male[-train_indices]

suppressWarnings({model_male <- train(x = X_train, y = y_train, method = "glm", family =
"binomial")})

predictions_male <- predict(model_male, newdata = X_test)

accuracy_male <- mean(predictions_male == y_test)
print(paste("Accuracy: ", accuracy_male))
```

```
## [1] "Accuracy:  0.79061000685401"
```

```
#Accuracy:  0.795854922279793

conf_matrix_male <- confusionMatrix(predictions_male, y_test)
print(conf_matrix_male)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0  980  342
##          1  269 1327
##
##                Accuracy : 0.7906
##                  95% CI : (0.7754, 0.8052)
##     No Information Rate : 0.572
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5755
##
##  Mcnemar's Test P-Value : 0.003582
##
##             Sensitivity : 0.7846
##             Specificity : 0.7951
##          Pos Pred Value : 0.7413
##          Neg Pred Value : 0.8315
##              Prevalence : 0.4280
##          Detection Rate : 0.3358
##    Detection Prevalence : 0.4531
##       Balanced Accuracy : 0.7899
##
##        'Positive' Class : 0
##
```

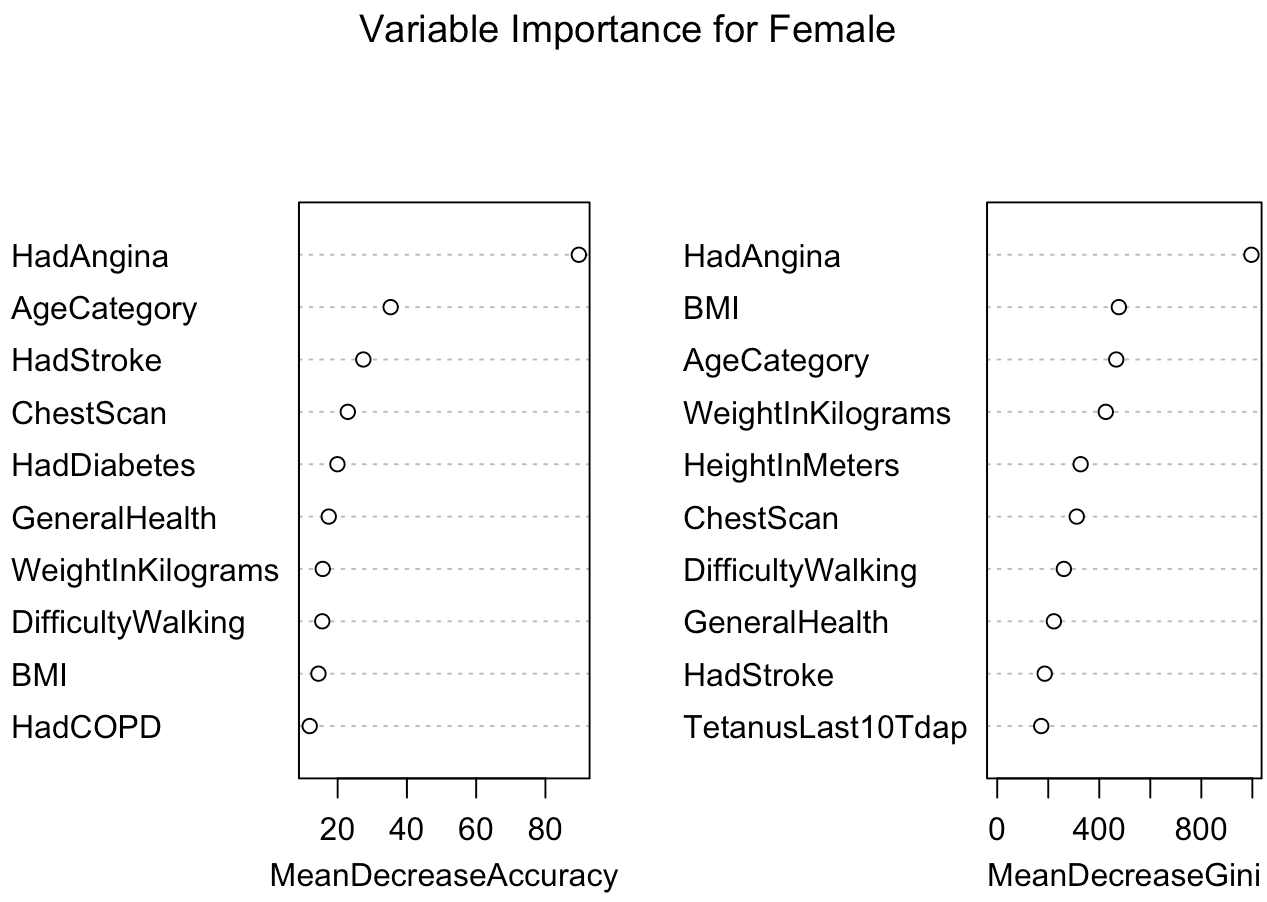
Running Random Forest Regression for Variable Importance for Female

```
rf_model_female <- randomForest(HadHeartAttack ~ ., data = female, importance = TRUE, mt
ry=6, ntree=100)

importance_values_female <- importance(rf_model_female)
importance_values_female
```

```
##                                   0          1 MeanDecreaseAccuracy
## Sex                         0.0000000  0.0000000            0.0000000
## GeneralHealth              10.0819938  9.9272340           17.4319024
## AgeCategory                23.4500762 19.4215393           35.3050139
## HeightInMeters              3.7196697  2.0372525            4.7536560
## WeightInKilograms          13.6099153  5.5325972           15.6996762
## BMI                        13.8697842  4.1193940           14.4517083
## HadAngina                  82.7485417 62.6525338           89.6506610
## HadStroke                  25.8630247 12.7701343           27.4140855
## HadAsthma                   3.3196803 -1.7724790            1.0548366
## HadSkinCancer               4.1897644 -1.2410711            2.4230295
## HadCOPD                    12.8141472 -0.0858590           11.9324759
## HadDepressiveDisorder       0.2507545  2.9094760            2.1835404
## HadKidneyDisease            9.0778751  3.9132440           10.1962565
## HadArthritis                7.8684894  3.8096607            7.9078280
## HadDiabetes                18.0174239  7.2701311           19.9329156
## DeafOrHardOfHearing         8.4595438  0.4956038            7.8464328
## BlindOrVisionDifficulty     5.9075391  1.4583049            5.4295368
## DifficultyConcentrating     8.3900875 -3.7153816            6.8254624
## DifficultyWalking          14.2208300  7.1612596           15.5576977
## DifficultyDressingBathing   6.7026392 -0.5032825            5.9297772
## DifficultyErrands           9.9022591 -0.5556427            9.0735176
## SmokerStatus                3.7608421  4.1428189            5.4953584
## ECigaretteUsage             5.3122258  0.3718109            4.7384638
## ChestScan                  16.4737486 16.3737828           22.9318000
## RaceEthnicityCategory       1.9513704 -0.6365869            1.1012804
## AlcoholDrinkers             3.6523272  6.0279784            6.8232788
## HIVTesting                 -1.2569954  2.2063209            0.5719777
## FluVaxLast12                4.4995954  0.1063186            3.8326757
## PneumoVaxEver               6.9049071  0.2938596            6.0484882
## TetanusLast10Tdap          -0.7946153  1.8196050            0.6475852
## HighRiskLastYear            0.3665717  1.6198547            1.4442966
## CovidPos                    2.0496857 -1.8313825            0.4720946
##                           MeanDecreaseGini
## Sex                                0.00000
## GeneralHealth                    222.18813
## AgeCategory                      466.54796
## HeightInMeters                   327.03739
## WeightInKilograms                425.56401
## BMI                              476.79551
## HadAngina                        996.42031
## HadStroke                        186.17065
## HadAsthma                         63.68828
## HadSkinCancer                     57.71625
## HadCOPD                          109.27490
## HadDepressiveDisorder             71.94845
## HadKidneyDisease                  64.84526
## HadArthritis                     147.94220
## HadDiabetes                      156.92432
## DeafOrHardOfHearing               56.60971
## BlindOrVisionDifficulty           48.59072
## DifficultyConcentrating           56.55480
```

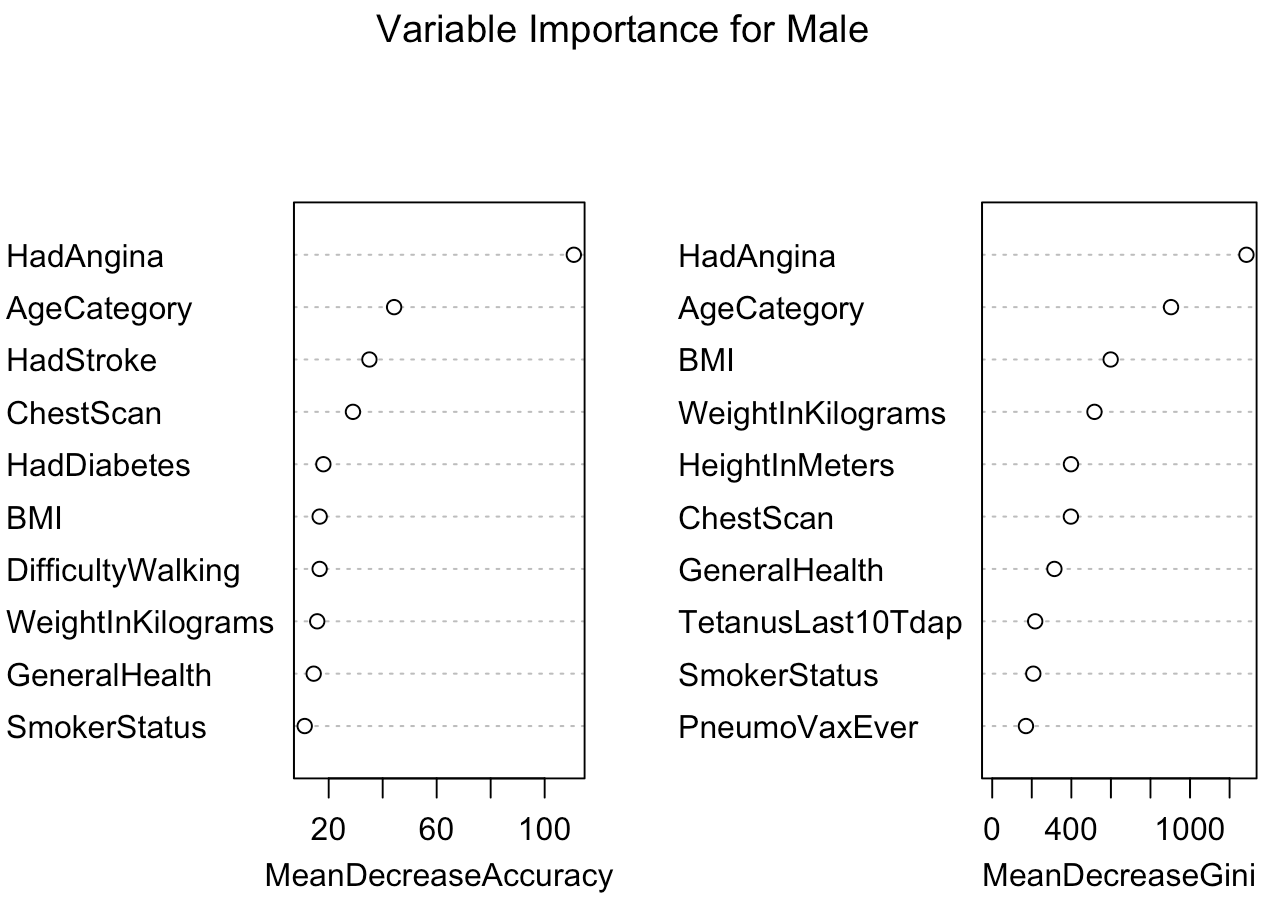
```
## DifficultyWalking               261.21299
## DifficultyDressingBathing        34.93943
## DifficultyErrands                62.99225
## SmokerStatus                    141.45361
## ECigaretteUsage                  81.57095
## ChestScan                       311.68859
## RaceEthnicityCategory           109.30330
## AlcoholDrinkers                  88.81396
## HIVTesting                       71.32102
## FluVaxLast12                     87.76635
## PneumoVaxEver                   103.56480
## TetanusLast10Tdap               172.54227
## HighRiskLastYear                 13.99906
## CovidPos                         74.64484
```

```
sorted_by_accuracy_female <- importance_values_female[order(-importance_values_female[,
"MeanDecreaseAccuracy"]), ]

varImpPlot(rf_model_female, n.var = 10, main="Variable Importance for Female")
```

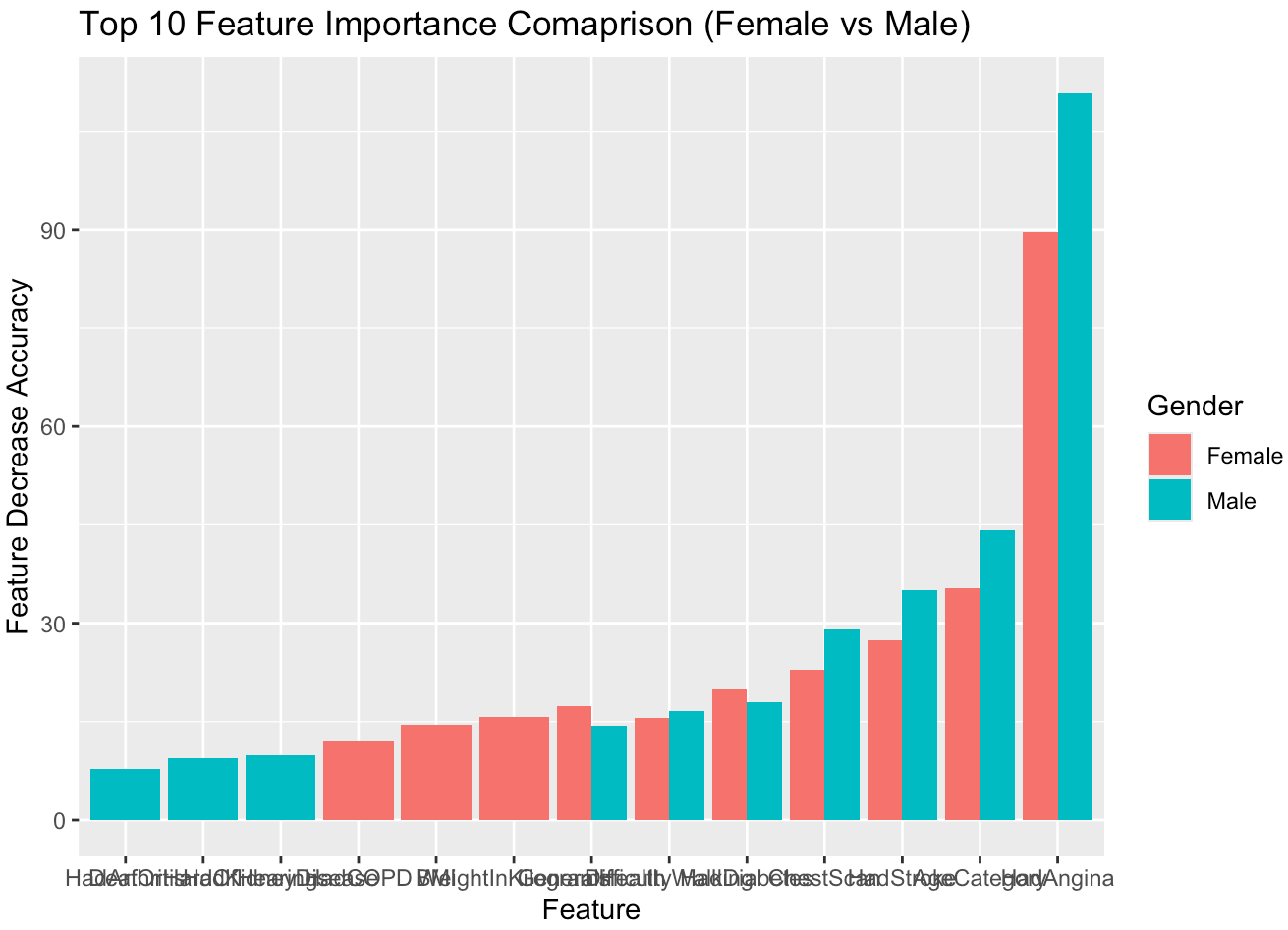


Running Random Forest Regression for Variable Importance for Male

```
male$HadHeartAttack <-as.factor(male$HadHeartAttack)
rf_model_male <- randomForest(HadHeartAttack ~ ., data = male, importance = TRUE, mtry=
6, ntree=100)

importance_values_male <- importance(rf_model_male)
sorted_by_accuracy_male <- importance_values_male[order(-importance_values_male[, 1]), ]


varImpPlot(rf_model_male, n.var=10, main="Variable Importance for Male")
```



```
#Bar Plot for male and female importance comparison

top_male <- head(rownames(sorted_by_accuracy_male), 10)

top_female <- head(rownames(sorted_by_accuracy_female), 10)

male_importance <- sorted_by_accuracy_male[seq(1, 10), "MeanDecreaseAccuracy"]

female_importance <- sorted_by_accuracy_female[seq(1, 10), "MeanDecreaseAccuracy"]

importance_data <- data.frame(
  Feature = rep(c(top_male, top_female), each = 1),
  Importance = c(male_importance, female_importance),
  Gender = rep(c("Male", "Female"), each=10)
)

# Plot
ggplot(importance_data, aes(x = reorder(Feature, Importance), y = Importance, fill = Gen
der)) + geom_bar(stat = "identity", position="dodge")+
  labs(title="Top 10 Feature Importance Comaprison (Female vs Male)", x = "Feature", y =
"Feature Decrease Accuracy")
```



Hierarchical Clustering

```
#Load required libraries for clustering
library(cluster)
```

```
balanced_subset <- balanced_data[,c('HadAngina', 'AgeCategory', 'ChestScan', 'HadStrok
e', 'DifficultyWalking', 'HadHeartAttack')]

balanced_subset <- balanced_subset[ sample(1:nrow(balanced_subset), size = 75),]
df_feature <- balanced_subset[, sapply(balanced_subset, is.numeric)]
# Standardize the data (z-score normalization)
df_feature <- scale(df_feature)

#distance matrix
distance_matrix <- dist(df_feature, method = "euclidean")

# Perform hierarchical clustering
set.seed(123)
hc <- hclust(distance_matrix, method = "ward.D2")

# Cut tree into clusters
k <- 5
clusters <- cutree(hc, k = k)

balanced_subset$Cluster <- clusters
```

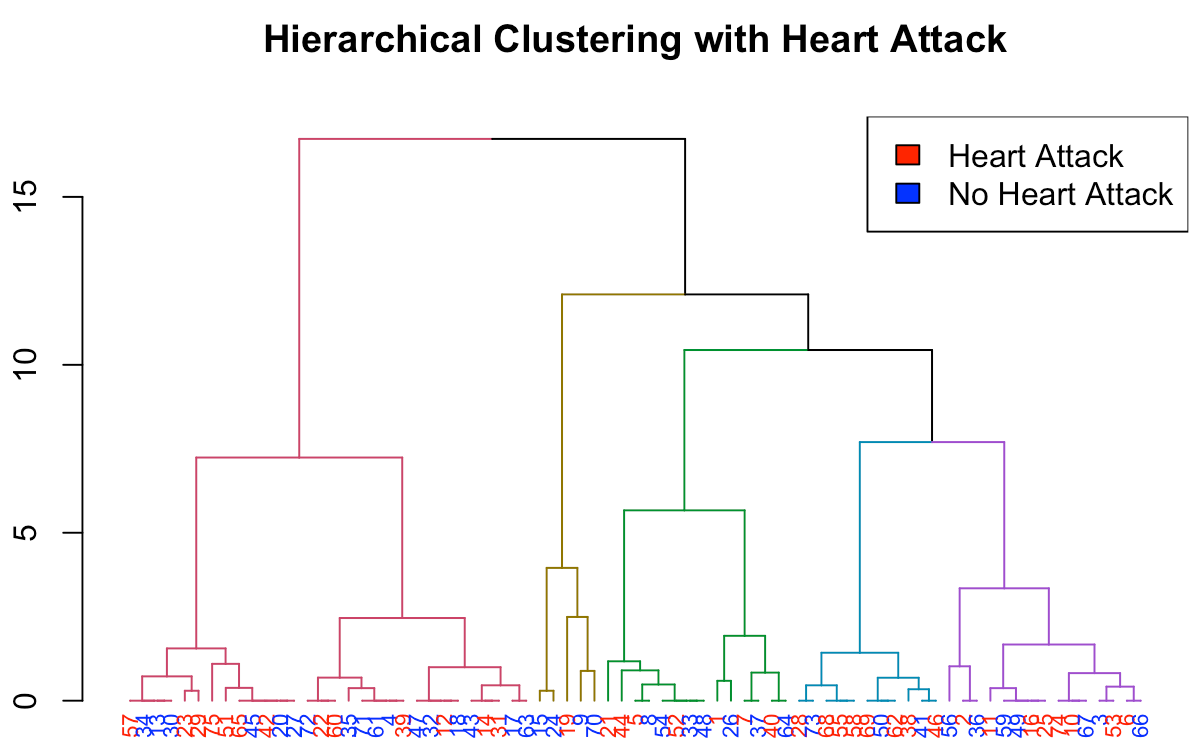
```
library(dendextend)
```

```
##
## ---------------------
## Welcome to dendextend version 1.19.0
## Type citation('dendextend') for how to cite the package.
##
## Type browseVignettes(package = 'dendextend') for the package vignette.
## The github page is: https://github.com/talgalili/dendextend/
##
## Suggestions and bug-reports can be submitted at: https://github.com/talgalili/dendext
end/issues
## You may ask questions at stackoverflow, use the r and dendextend tags:
##   https://stackoverflow.com/questions/tagged/dendextend
##
##  To suppress this message use:  suppressPackageStartupMessages(library(dendextend))
## ---------------------
```

```
##
## Attaching package: 'dendextend'
```

```
## The following object is masked from 'package:stats':
##
##     cutree
```

```
dend <- as.dendrogram(hc)

has_heart_attack <- ifelse(balanced_subset$HadHeartAttack == 1, "red", "blue")

dend <- dend %>% set("branches_k_color", k = k) %>%
  set("labels_colors", has_heart_attack) %>%
  set("labels_cex", 0.7)

plot(dend, main= "Hierarchical Clustering with Heart Attack")

legend("topright", legend=c("Heart Attack", "No Heart Attack"), fill = c("red", "blue"))
```




```
#Analyze proportion of HadHeartAttack within each cluster to understand how well the clu
sters separate individuals with and without heart attacks.
table(balanced_subset$Cluster, balanced_subset$HadHeartAttack)
```

```
##
##      0  1
##   1  2 12
##   2  8  7
##   3 27  3
##   4  1  4
##   5  0 11
```

```
#Summary Statistic of Clusters

cluster_summary <- balanced_subset %>%
  mutate(HadHeartAttack = as.numeric(HadHeartAttack)) %>%
  group_by(Cluster) %>%
  summarize(across(c(HadHeartAttack, HadAngina, ChestScan, HadStroke, DifficultyWalking,
AgeCategory), mean))

print(cluster_summary)
```

```
## # A tibble: 5 × 7
##   Cluster HadHeartAttack HadAngina ChestScan HadStroke DifficultyWalking
##     <int>          <dbl>     <dbl>     <dbl>     <dbl>              <dbl>
## 1       1           1.86     0.571         1         0                1
## 2       2           1.47     0             1         0                0
## 3       3           1.1      0             0         0                0
## 4       4           1.8      0.8           1         1                0.4
## 5       5           2        1             1         0                0
## # ℹ 1 more variable: AgeCategory <dbl>
```

Cluster 4 has the highest rate of “HadHeartAttack” = 1.933333.