Maternal-Risk-Analysis-Project.R

tomaz

2024-05-25

```
## MATERNAL HEALTH RISK ANALYSIS USING RANDOM FOREST & CLASSIFICATION TREE
# load the necessary libraries
library(mice)
```

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

```
##
## Attaching package: 'mice'
```

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

```
## The following objects are masked from 'package:base':
##
##     cbind, rbind
```

```
library(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
```

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(caTools)
library(caret)
```

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

```
## Zorunlu paket yükleniyor: lattice
```

```
library(randomForest)
```

```
## randomForest 4.7-1.1
```

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

```
##
## Attaching package: 'randomForest'
```

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

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(vip)
```

```
## Warning: package 'vip' was built under R version 4.3.3
```

```
##
## Attaching package: 'vip'
```

```
## The following object is masked from 'package:utils':
##
##     vi
```

```
library(rpart)
library(rpart.plot)

# load the data set
df <- read.csv("maternal.health.risk.csv", sep = ",")

# dim of the data set
dim(df)
```

```
## [1] 1014    7
```

```
# structure of the df
str(df)
```

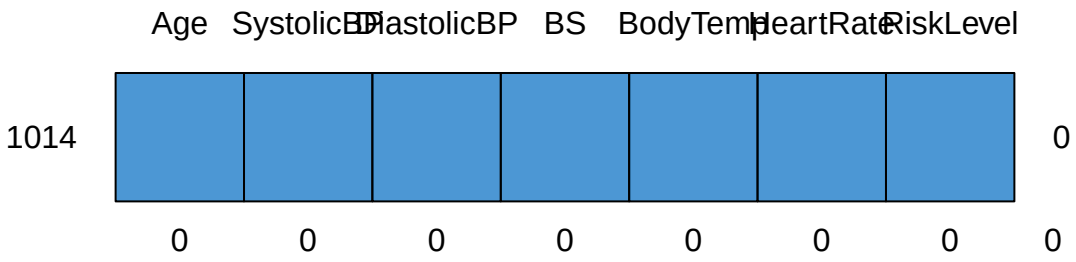
```
## 'data.frame':    1014 obs. of  7 variables:
##  $ Age        : int  25 35 29 30 35 23 23 35 32 42 ...
##  $ SystolicBP : int  130 140 90 140 120 140 130 85 120 130 ...
##  $ DiastolicBP: int  80 90 70 85 60 80 70 60 90 80 ...
##  $ BS         : num  15 13 8 7 6.1 7.01 7.01 11 6.9 18 ...
##  $ BodyTemp   : num  98 98 100 98 98 98 98 102 98 98 ...
##  $ HeartRate  : int  86 70 80 70 76 70 78 86 70 70 ...
##  $ RiskLevel  : chr  "high risk" "high risk" "high risk" "high risk" ...
```

```r
# changing the class of the RiskLevel col
df$RiskLevel <- as.factor(df$RiskLevel)

# number of patients belonging each risk group
table(df$RiskLevel)
```

```
##
## high risk  low risk  mid risk
##       272       406       336
```

```r
# missing data
md.pattern(df) # completely observed
```

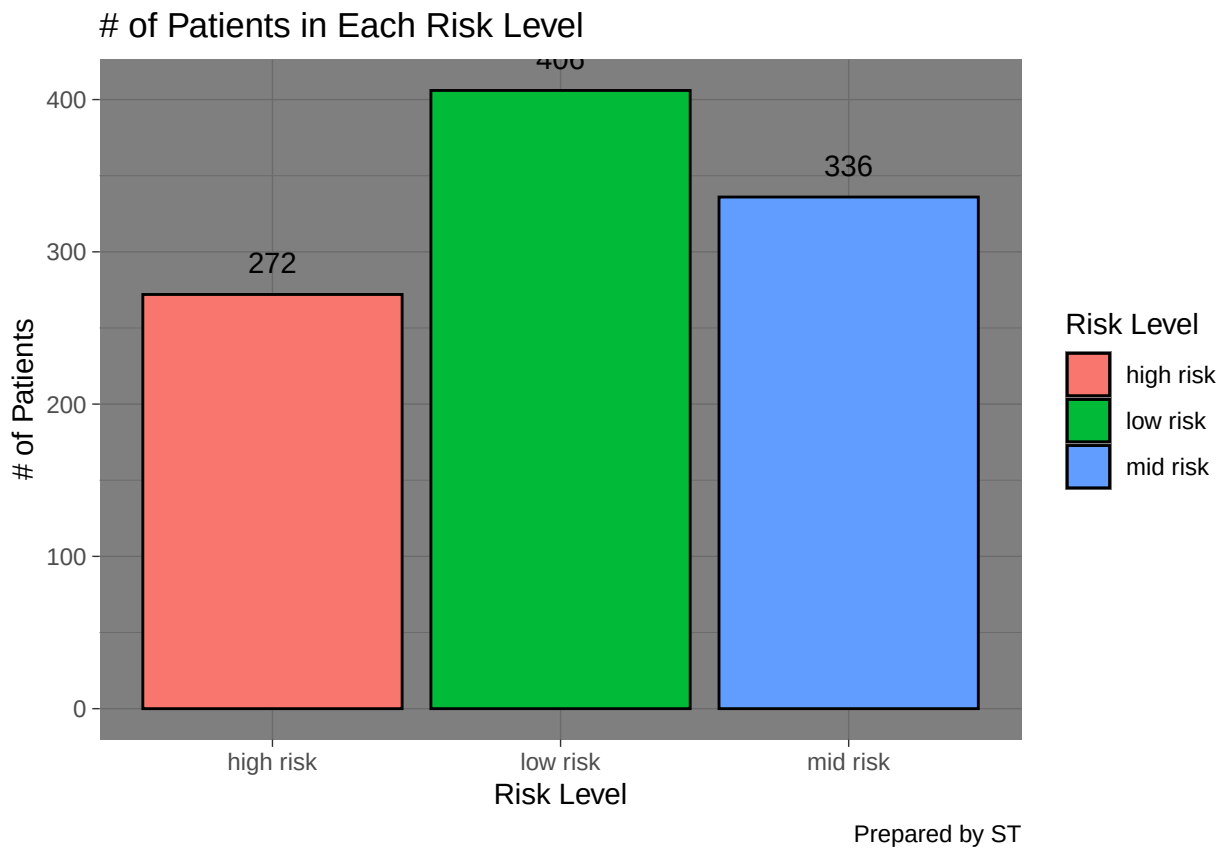
```
##  /\     /\
## {  `---'  }
## {  O   O  }
## ==>  V <==  No need for mice. This data set is completely observed.
##  \  \|/  /
##   `-----'
```




```
##      Age SystolicBP DiastolicBP BS BodyTemp HeartRate RiskLevel
## 1014   1          1           1  1        1         1         1 0
##        0          0           0  0        0         0         0 0
```

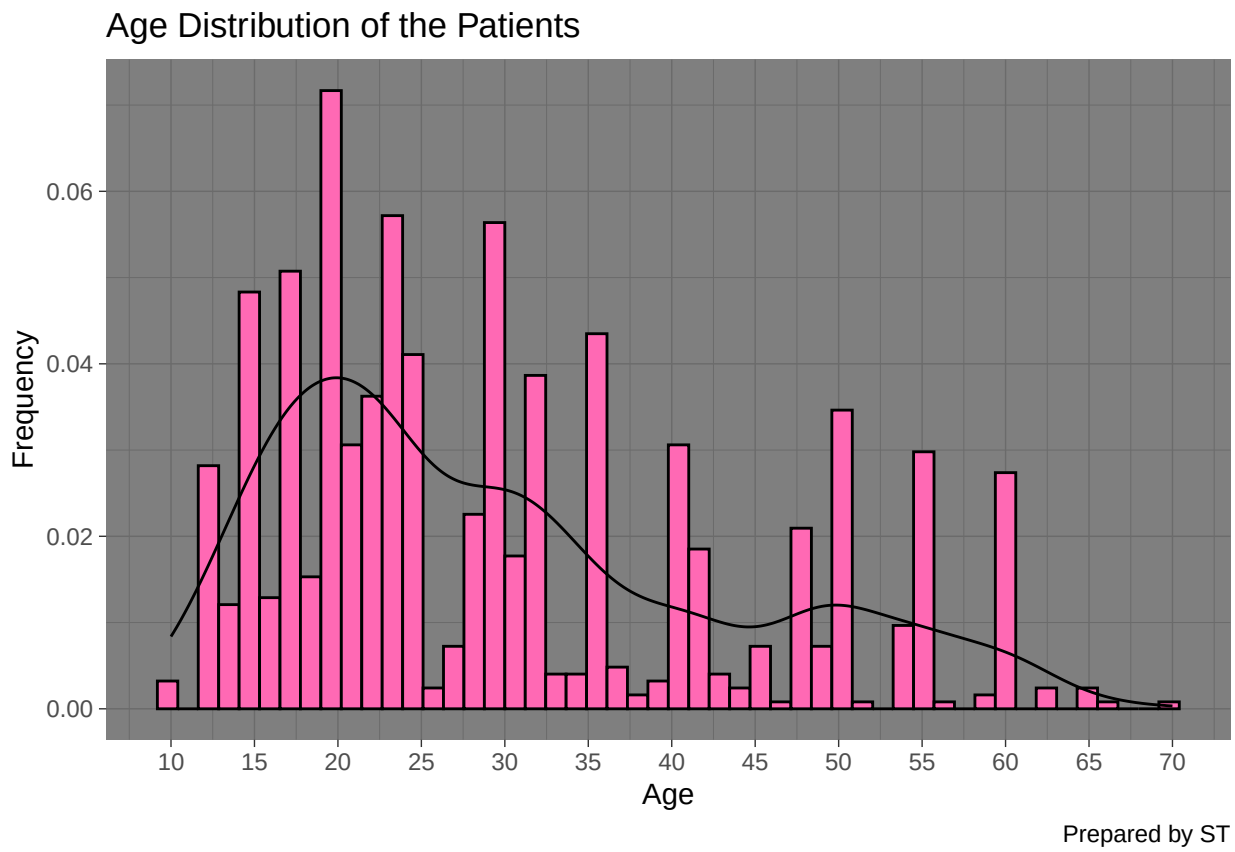
```
## EDA
# number of patients in each risk level
ggplot(df, aes(x = RiskLevel)) +
  geom_bar(aes(fill = RiskLevel), color = "black") +
  labs(x = "Risk Level",
       y = "# of Patients",
       caption = "Prepared by ST",
       fill = "Risk Level") +
  ggtitle("# of Patients in Each Risk Level") +
  theme_dark() +
  geom_text(aes(label = ..count..),
            stat = "count",
            vjust = -1)
```

```
## Warning: The dot-dot notation (‘..count..‘) was deprecated in ggplot2 3.4.0.
## i Please use ‘after_stat(count)‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```




```
# age histogram of the patients
ggplot(df, aes(x = Age)) +
  geom_histogram(aes(y = ..density..),
                 bins = 50,
                 color = "black",
```

```
                fill = "hotpink") +
  scale_x_continuous(breaks = seq(0, 80, by = 5)) +
  geom_density() +
  theme_dark() +
  ggtitle("Age Distribution of the Patients") +
  labs(x = "Age",
       y = "Frequency",
       caption = "Prepared by ST")
```

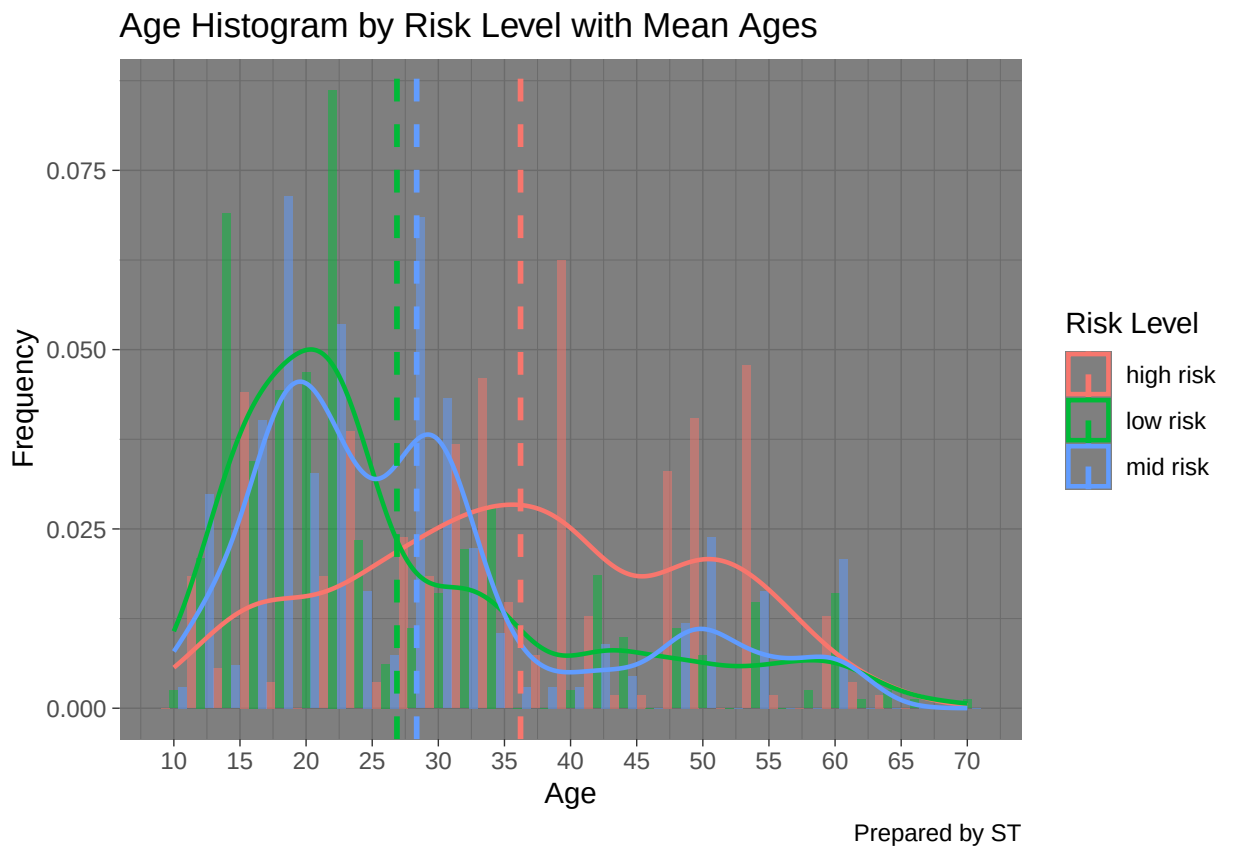



```
# mean ages for each risk level
mean.ages.df <- df %>%
  group_by(RiskLevel) %>%
  summarise(mean.ages = mean(Age))

# Density plot of age by risk level with mean ages for each risk level
ggplot(df, aes(x = Age)) +
  geom_histogram(aes(fill = RiskLevel,
                     y = ..density..),
                 position = "dodge",
                 bins = 30,
                 binwidth = 2,
                 alpha = 0.5) +
  geom_density(aes(color = RiskLevel),
               alpha = 0.4,
               size = 0.8) +
```

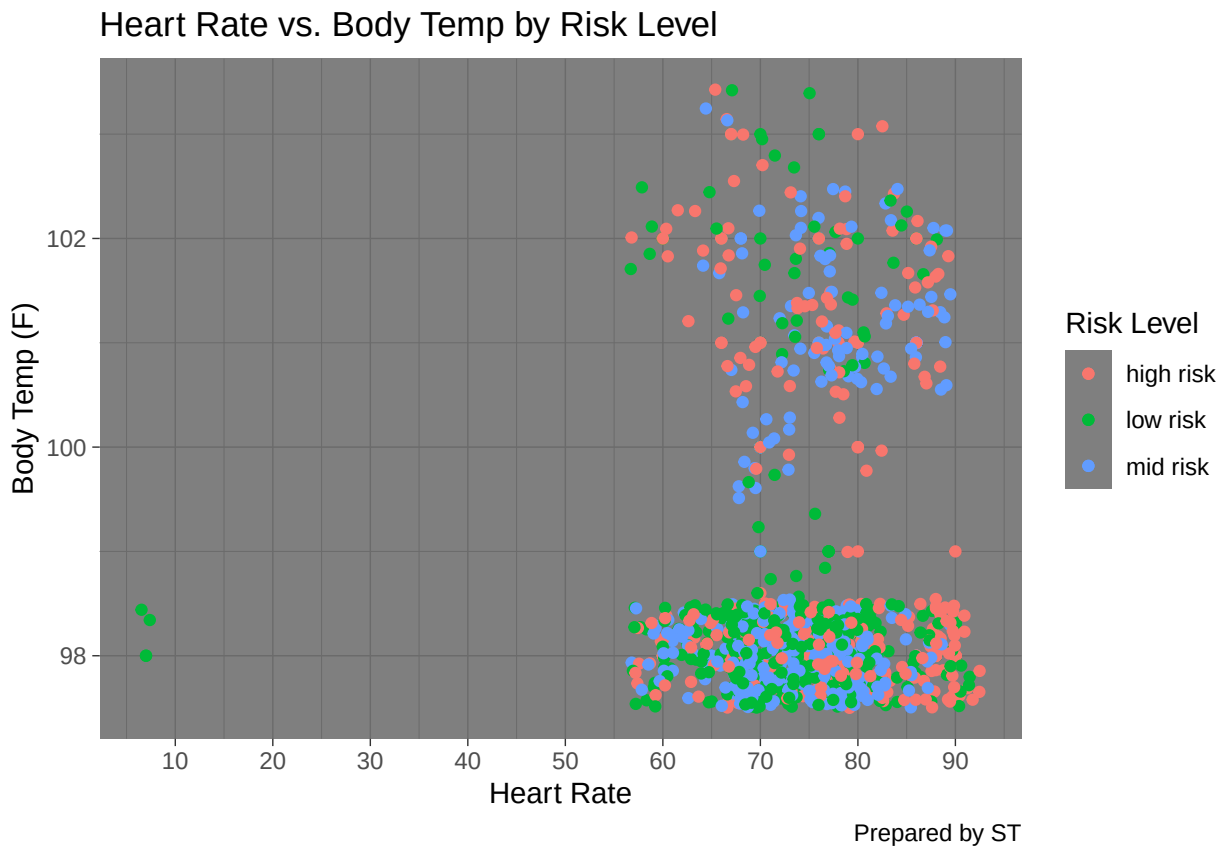
```
  scale_x_continuous(breaks = seq(10, 70, by = 5)) +
  geom_vline(data = mean.ages.df,
             aes(xintercept = mean.ages, color = RiskLevel),
             linetype = "dashed",
             size = 1) +
  theme_dark() +
  ggtitle("Age Histogram by Risk Level with Mean Ages") +
  labs(x = "Age",
       y = "Frequency",
       caption = "Prepared by ST",
       color = "Risk Level") +
  guides(fill = FALSE)
```

```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

```
## Warning: The ‘<scale>‘ argument of ‘guides()‘ cannot be ‘FALSE‘. Use "none" instead as
## of ggplot2 3.3.4.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

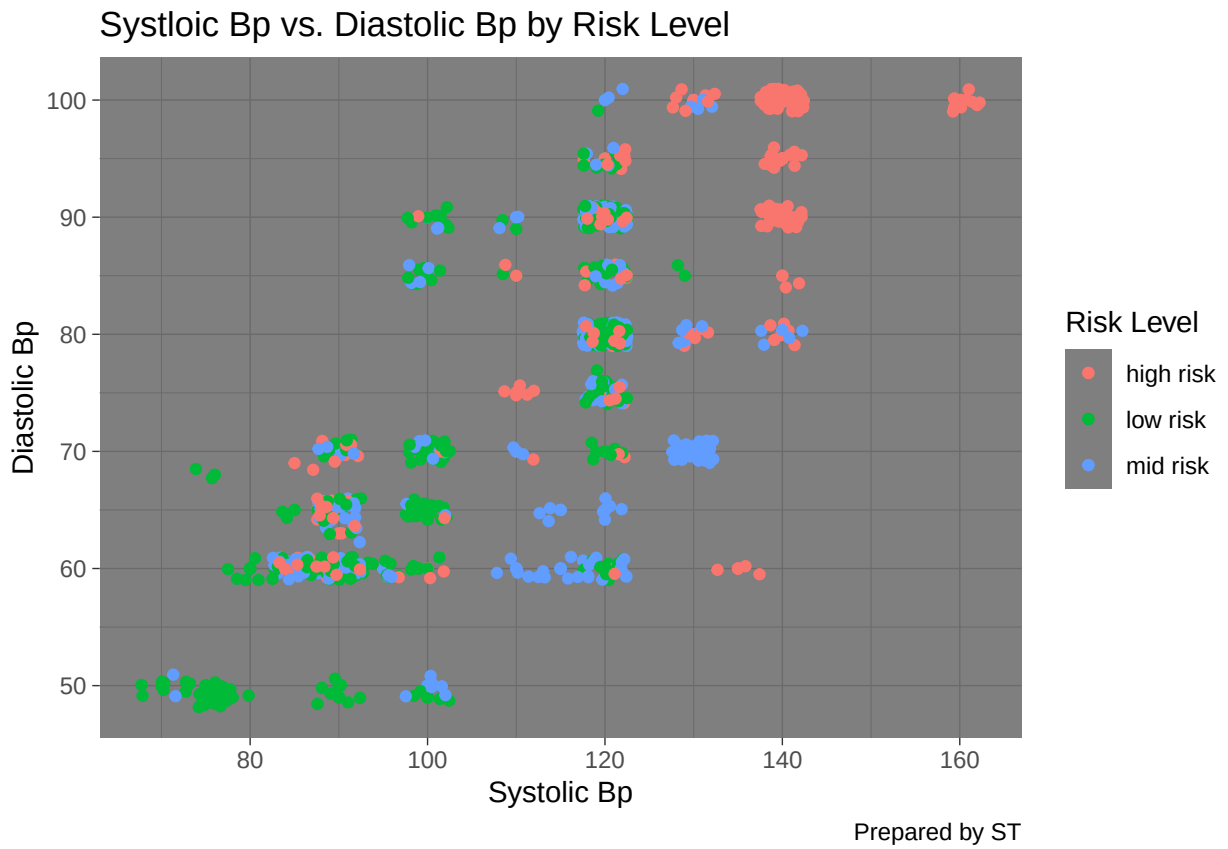


```r
# scatter plot of heart rate vs. body temp by risk level
# ~ 0 correlation between them
ggplot(df, aes(x = HeartRate,
               y = BodyTemp,
               color = RiskLevel)) +
  geom_point() +
  scale_x_continuous(breaks = seq(0, 90, by = 10)) +
  geom_jitter(width = 3.5, height = 0.5) +
  ggtitle("Heart Rate vs. Body Temp by Risk Level") +
  labs(x = "Heart Rate",
       y = "Body Temp (F)",
       caption = "Prepared by ST",
       color = "Risk Level") +
  theme_dark()
```

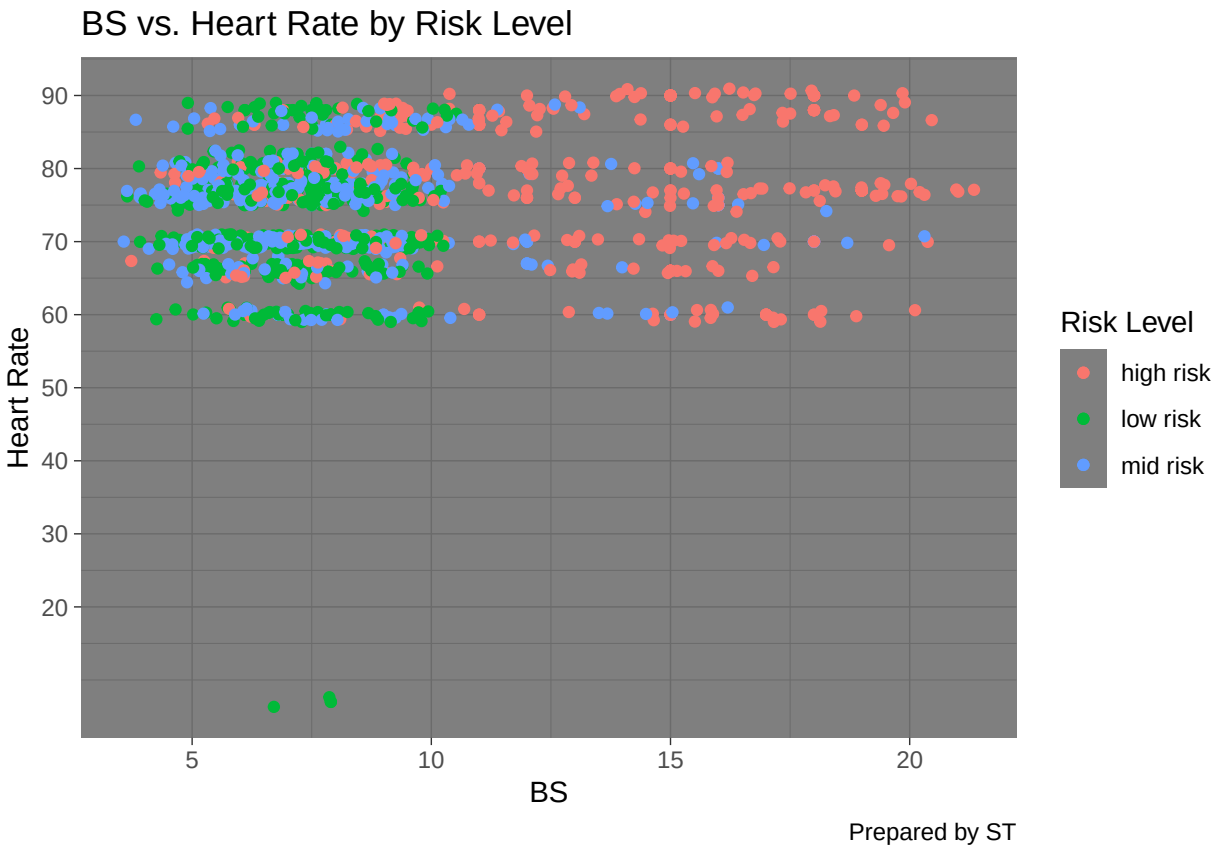



```r
# scatter plot of systolic bp vs. diastolic bp by risk level
# positive correlation
ggplot(df, aes(x = SystolicBP,
               y = DiastolicBP,
               color = RiskLevel)) +
  geom_point() +
  geom_jitter(width = 2.5,
              height = 1) +
  ggtitle("Systloic Bp vs. Diastolic Bp by Risk Level") +
  labs(x = "Systolic Bp",
```

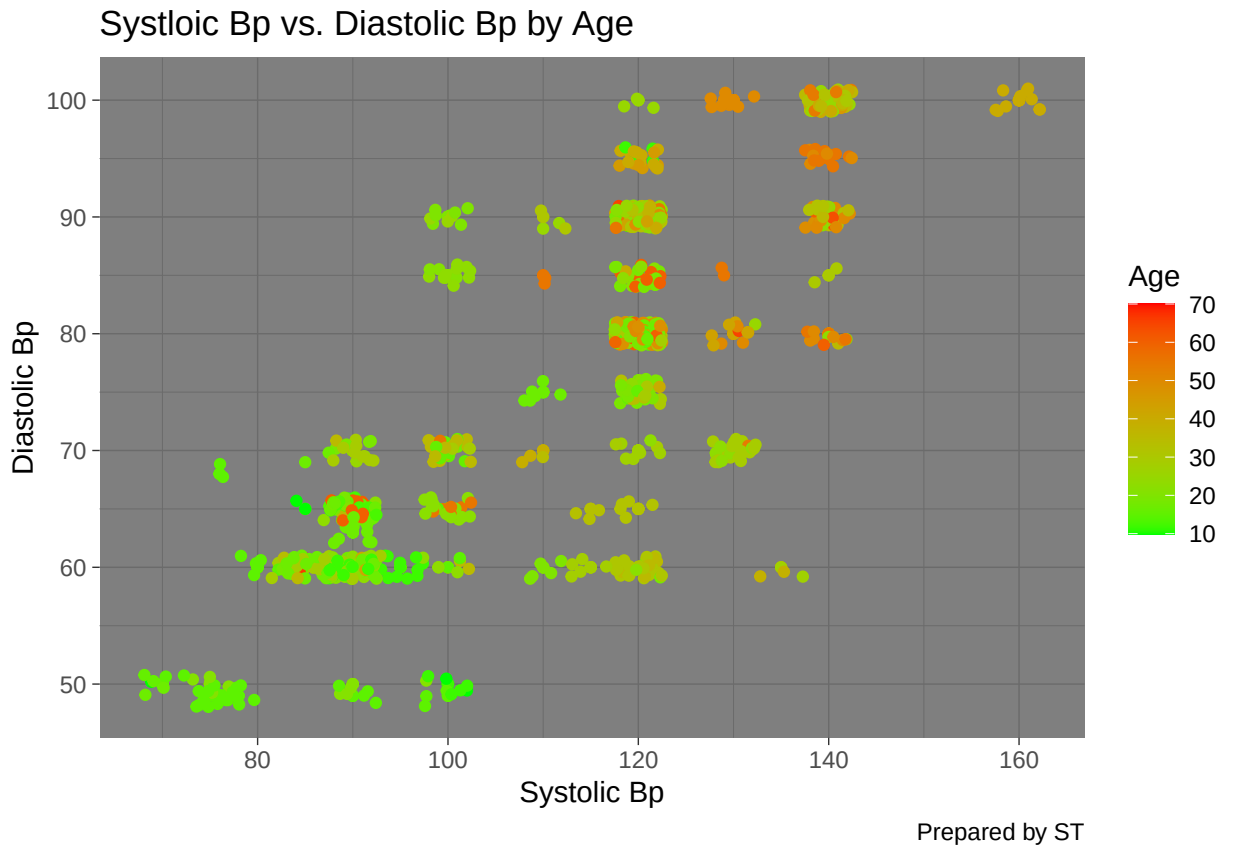
```
      y = "Diastolic Bp",
      caption = "Prepared by ST",
      color = "Risk Level") +
  theme_dark()
```



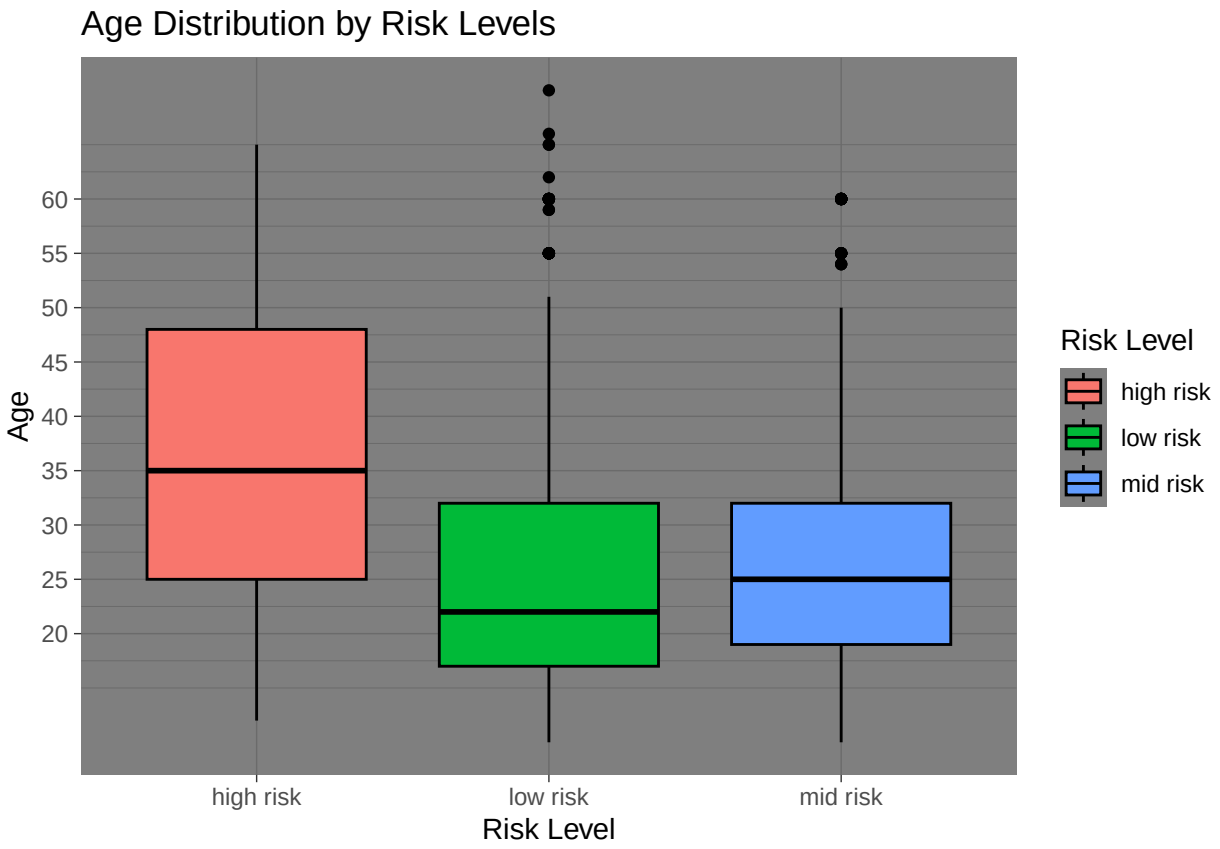

```
# scatter plot of bs vs. heart rate by risk level
# almost no correlation between them
ggplot(df, aes(x = BS, y = HeartRate, color = RiskLevel)) +
  geom_point() +
  geom_jitter(width = 2.5, height = 1) +
  scale_y_continuous(breaks = seq(20, 100, by = 10)) +
  ggtitle("BS vs. Heart Rate by Risk Level") +
  labs(x = "BS",
      y = "Heart Rate",
      caption = "Prepared by ST",
      color = "Risk Level") +
  theme_dark()
```



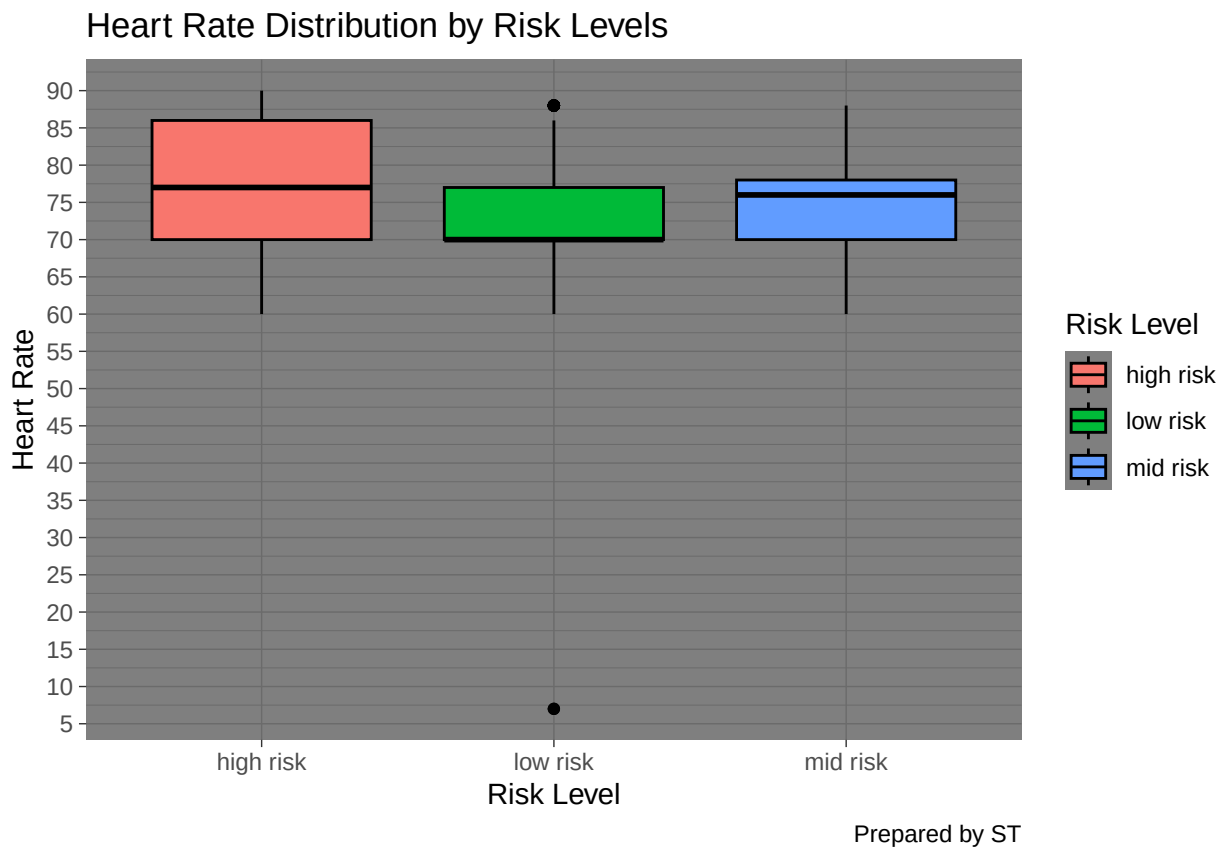

```r
# scatter plot of systolic bp vs. diastolic bp by age
ggplot(df, aes(x = SystolicBP,
               y = DiastolicBP,
               color = Age)) +
  geom_point() +
  geom_jitter(width = 2.5,
              height = 1) +
  ggtitle("Systloic Bp vs. Diastolic Bp by Age") +
  labs(x = "Systolic Bp",
       y = "Diastolic Bp",
       caption = "Prepared by ST",
       color = "Age") +
  theme_dark() +
  scale_color_gradient(low = "green", high = "red")
```



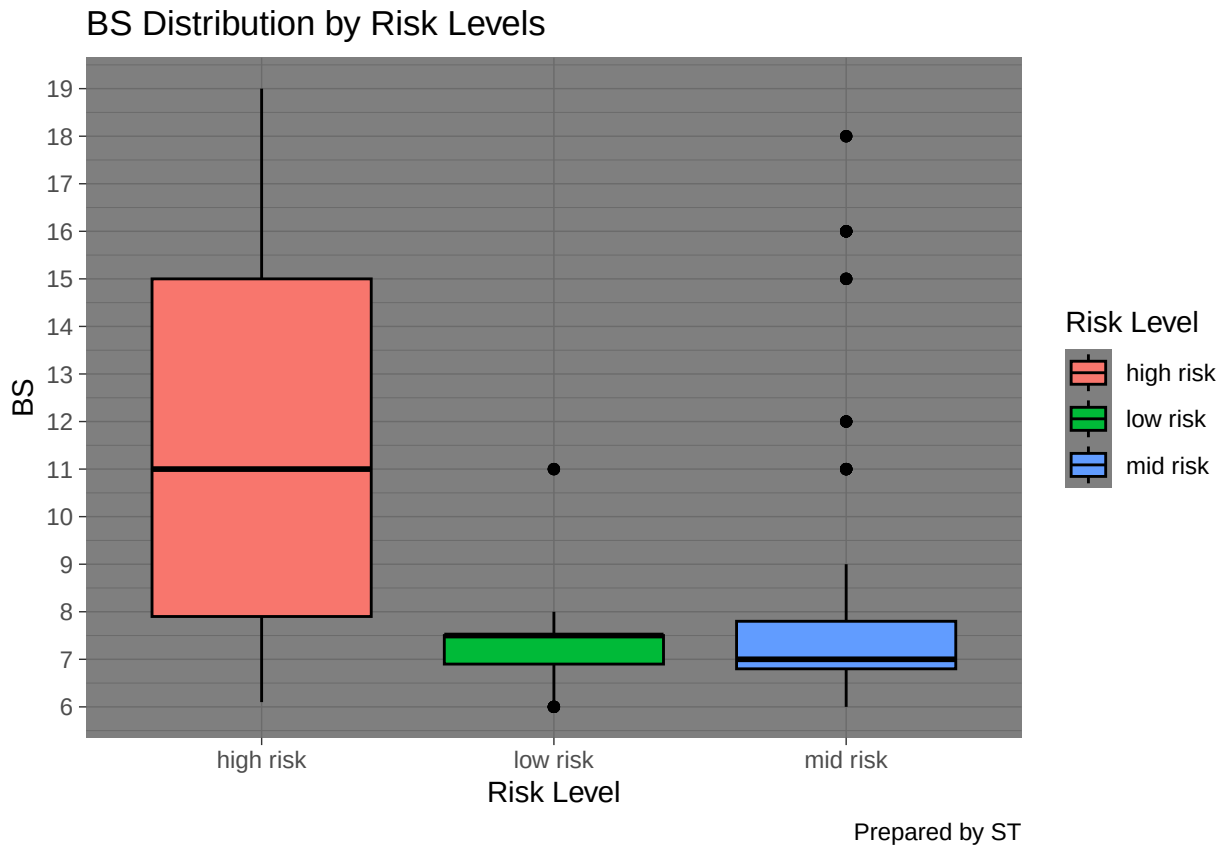

```r
# box plot of age - risk level
ggplot(df, aes(x = RiskLevel, y = Age)) +
  geom_boxplot(aes(fill = RiskLevel), color = "black") +
  ggtitle("Age Distribution by Risk Levels") +
  labs(x = "Risk Level",
       y = "Age",
       fill = "Risk Level") +
  scale_y_continuous(breaks = seq(20, 60, by = 5)) +
  theme_dark()
```



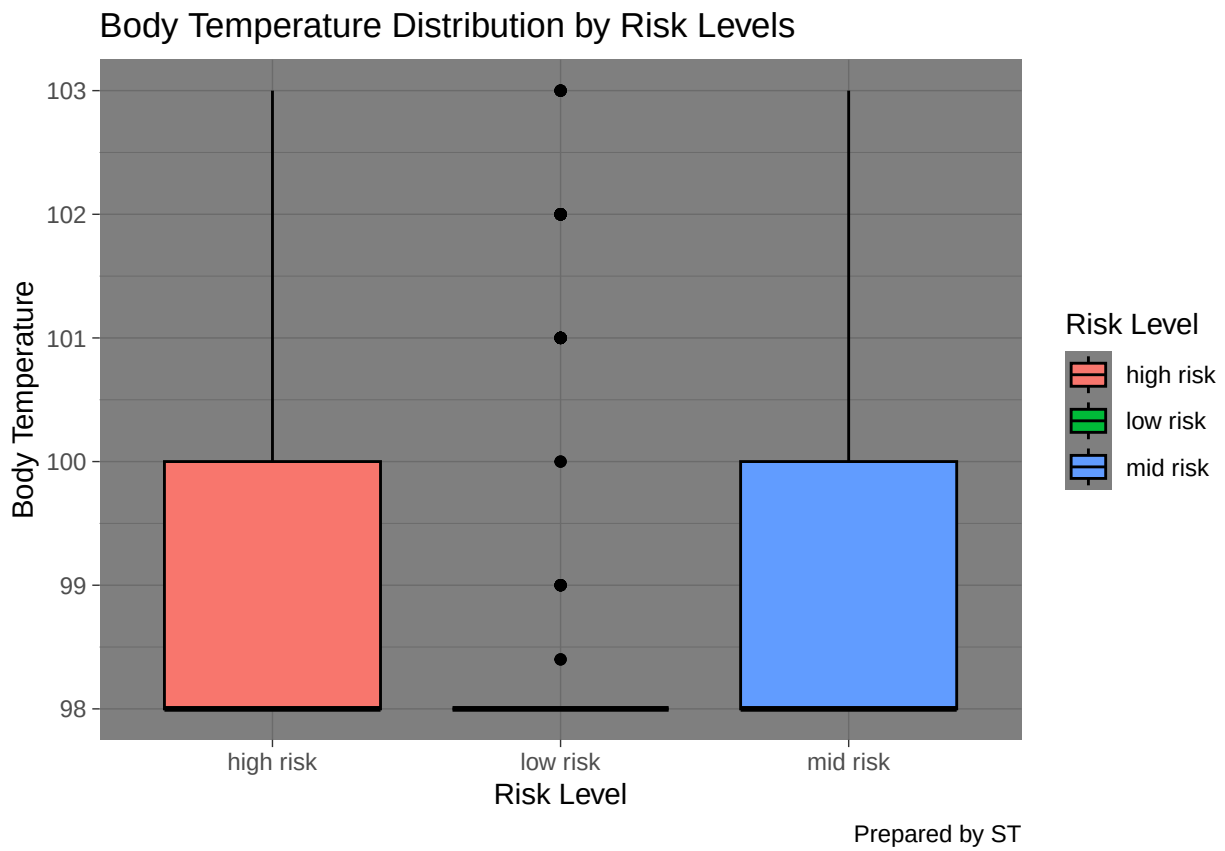

```r
# box plot of heart rate - risk level
ggplot(df, aes(x = RiskLevel, y = HeartRate)) +
  geom_boxplot(aes(fill = RiskLevel), color = "black") +
  scale_y_continuous(breaks = seq(0, 90, by = 5)) +
  ggtitle("Heart Rate Distribution by Risk Levels") +
  labs(x = "Risk Level",
       y = "Heart Rate",
       caption = "Prepared by ST",
       fill = "Risk Level") +
  theme_dark()
```



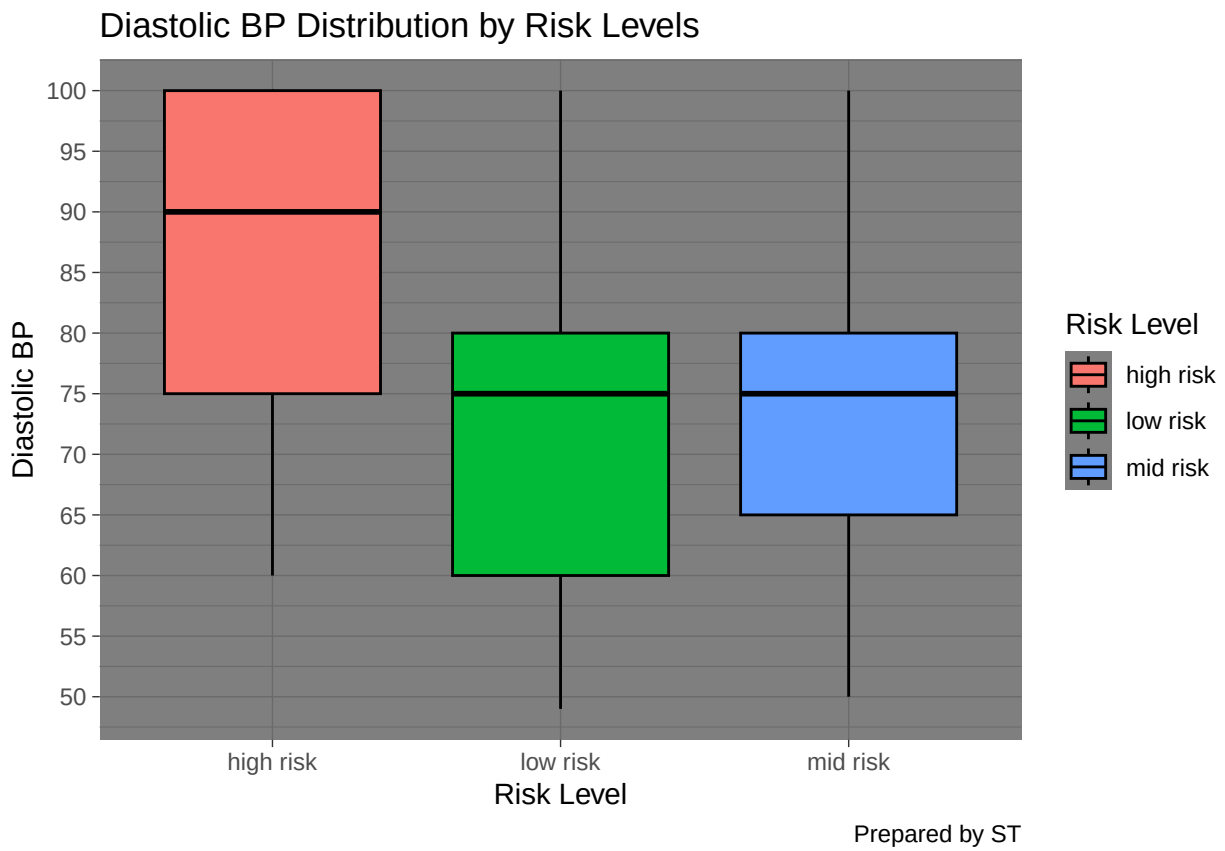

```r
# box plot of BS - risk level
ggplot(df, aes(x = RiskLevel, y = BS)) +
  geom_boxplot(aes(fill = RiskLevel), color = "black") +
  scale_y_continuous(breaks = seq(0, 20, by = 1)) +
  ggtitle("BS Distribution by Risk Levels") +
  labs(x = "Risk Level",
       y = "BS",
       fill = "Risk Level",
       caption = "Prepared by ST") +
  theme_dark()
```



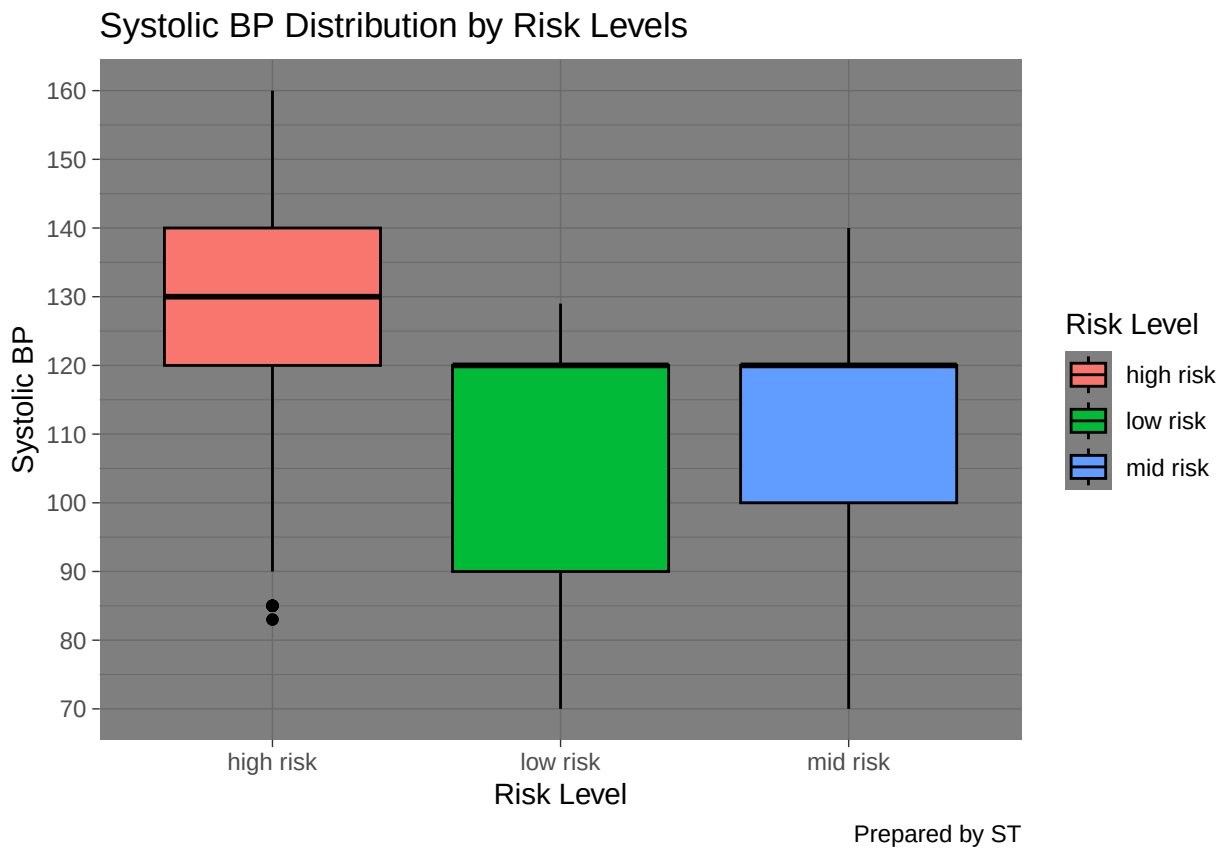

```r
# box plot of body temp - risk level
ggplot(df, aes(x = RiskLevel, y = BodyTemp)) +
  geom_boxplot(aes(fill = RiskLevel), color = "black") +
  scale_y_continuous(breaks = seq(90, 110, by = 1)) +
  ggtitle("Body Temperature Distribution by Risk Levels") +
  labs(x = "Risk Level",
       y = "Body Temperature",
       caption = "Prepared by ST",
       fill = "Risk Level") +
  theme_dark()
```



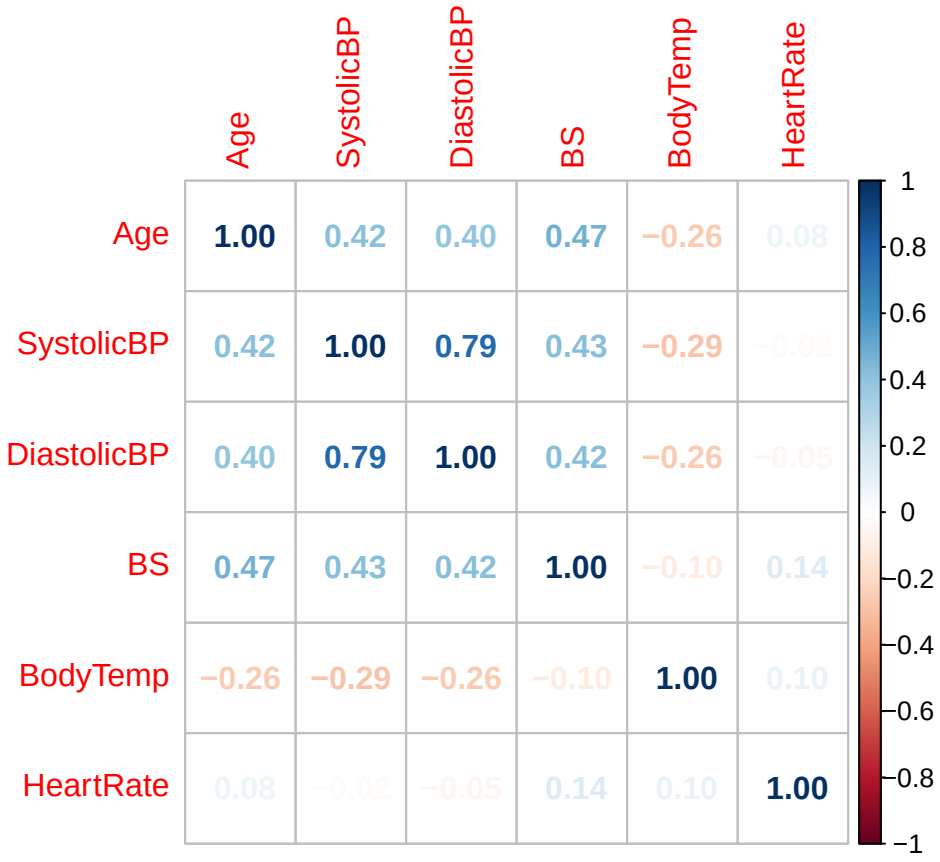

```
# box plot of diastolic bp - risk level
ggplot(df, aes(x = RiskLevel, y = DiastolicBP)) +
  geom_boxplot(aes(fill = RiskLevel), color = "black") +
  scale_y_continuous(breaks = seq(50,100 , by = 5)) +
  ggtitle("Diastolic BP Distribution by Risk Levels") +
  labs(x = "Risk Level",
       y= "Diastolic BP",
       caption = "Prepared by ST",
       fill = "Risk Level") +
  theme_dark()
```




```
# box plot of systolic bp - risk level
ggplot(df, aes(x = RiskLevel, y = SystolicBP)) +
  geom_boxplot(aes(fill = RiskLevel), color = "black") +
  scale_y_continuous(breaks = seq(70,160 , by = 10)) +
  ggtitle("Systolic BP Distribution by Risk Levels") +
  labs(x = "Risk Level",
       y= "Systolic BP",
       caption = "Prepared by ST",
       fill = "Risk Level") +
  theme_dark()
```




```
## correlation
# correlation table
cor.data <- cor(df [, -7])

# correlation graph
corrplot(cor.data, method = "number")
```




```
### OUTLIER DETECTION
## AGE
# outlier detection of the age
boxplot.stats(df$Age)$out # potential outliers of the age
```

```
## [1] 70
```

```
# imputation of age col
uv.1 <- quantile(df$Age, 0.75) + ((1.5) * (quantile(df$Age, 0.75) - quantile(df$Age, 0.25)))
df$Age [df$Age > uv.1] <- uv.1 # all age values greater than uv.1 are replaced with this uv.1

## SYSTOLIC BP
# outlier detection of the systolic bp
boxplot.stats(df$SystolicBP)$out # potential outliers of the systolic bp
```

```
##  [1] 160 160 160 160 160 160 160 160 160 160
```

```
# imputation of the systolic bp
uv.2 <- quantile(df$SystolicBP, 0.75) + ((1.5) * (quantile(df$SystolicBP, 0.75) - quantile(df$SystolicBl
df$SystolicBP [df$SystolicBP > uv.2] <- uv.2 # all systolic bp greater than uv.2 are replaced with this

## BS
# outlier detection of the BS
boxplot.stats(df$BS)$out # potential outliers of the BS
```

```
##   [1] 15 13 11 18 11 15 18 12 16 12 15 15 11 15 18 17 15 15 18 15 11 15 19 18 15
##  [26] 19 11 19 18 15 11 15 19 16 11 12 12 19 18 15 11 15 19 16 15 10 11 17 15 11
##  [51] 15 18 11 12 11 11 13 15 19 18 15 11 11 15 19 16 11 15 18 12 19 18 13 15 16
##  [76] 12 18 11 19 18 15 11 15 19 16 11 16 12 16 15 13 15 13 15 13 10 11 18 11 15
## [101] 11 18 11 15 18 11 17 15 11 15 18 15 11 15 19 18 15 11 15 19 16 11 19 18 15
## [126] 11 15 19 16 15 19 16 15 11 12 17 11 15 13 11 12 10 15 11 15 12 13 11 18 15
## [151] 18 11 12 12 19 11 11 12 16 12 15 16 18 18 11 11 15 16 16 11 11 11 18 11 12
## [176] 12 19 18 15 11 15 19 16 15 10 11 17 15 11 15 18 11 12 11 11 13 15 19 18 15
## [201] 11 11 15 19 16 11 15 18 19 18
```

```
# imputation of the BS
uv.3 <- quantile(df$BS, 0.75) + ((1.5) * (quantile(df$BS, 0.75) - quantile(df$BS, 0.25)))
df$BS [df$BS > uv.3] <- uv.3 # all BS values greater than uv.3 are replaced with this uv.3

## HEARTRATE
# outlier detection of the heart rate
boxplot.stats(df$HeartRate)$out # potential outliers of the heart rate
```

```
## [1] 7 7
```

```
# imputation of the heart rate
uv.4 <- quantile(df$HeartRate, 0.25) - ((1.5) * (quantile(df$HeartRate, 0.75) - quantile(df$HeartRate,
df$HeartRate [df$HeartRate < uv.4] <- uv.4 # all heart rate values lower than uv.4 are replaced with th

### Test & Train Split
# test & train split
split <- sample.split(df$RiskLevel, SplitRatio = 0.7)
test <- subset(df, split == FALSE)
train <- subset(df, split == TRUE)

# dim of the train set
dim(test)
```

```
## [1] 305   7
```

```
# dim of the test set
dim(train)
```

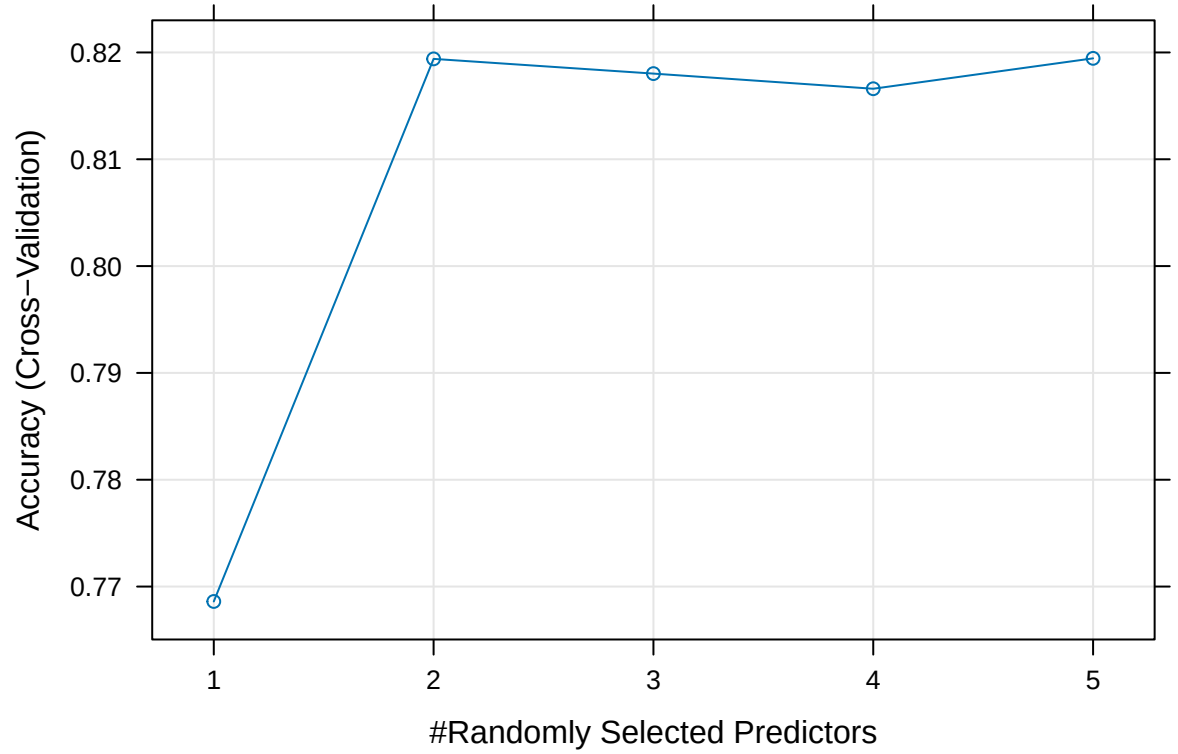
```
## [1] 709   7
```

```
### RANDOM FOREST
# creating parameter grid for random forest model
param.grid.rf <- expand.grid(mtry = seq(1, 5, by = 1))

# setting cross - validation parameters with the control function
ctrl.rf <- trainControl(method = "cv",
                        number = 5)

# select the best parameters for random forest
parameter.search.rf <- train(RiskLevel ~.,
                             data = train,
                             method = "rf",
```

```
                            trControl = ctrl.rf,
                            tuneGrid = param.grid.rf)

# best tune of the random forest model
parameter.search.rf$bestTune$mtry
```

```
## [1] 5
```

```
# plot of cv - predictors
plot(parameter.search.rf)
```

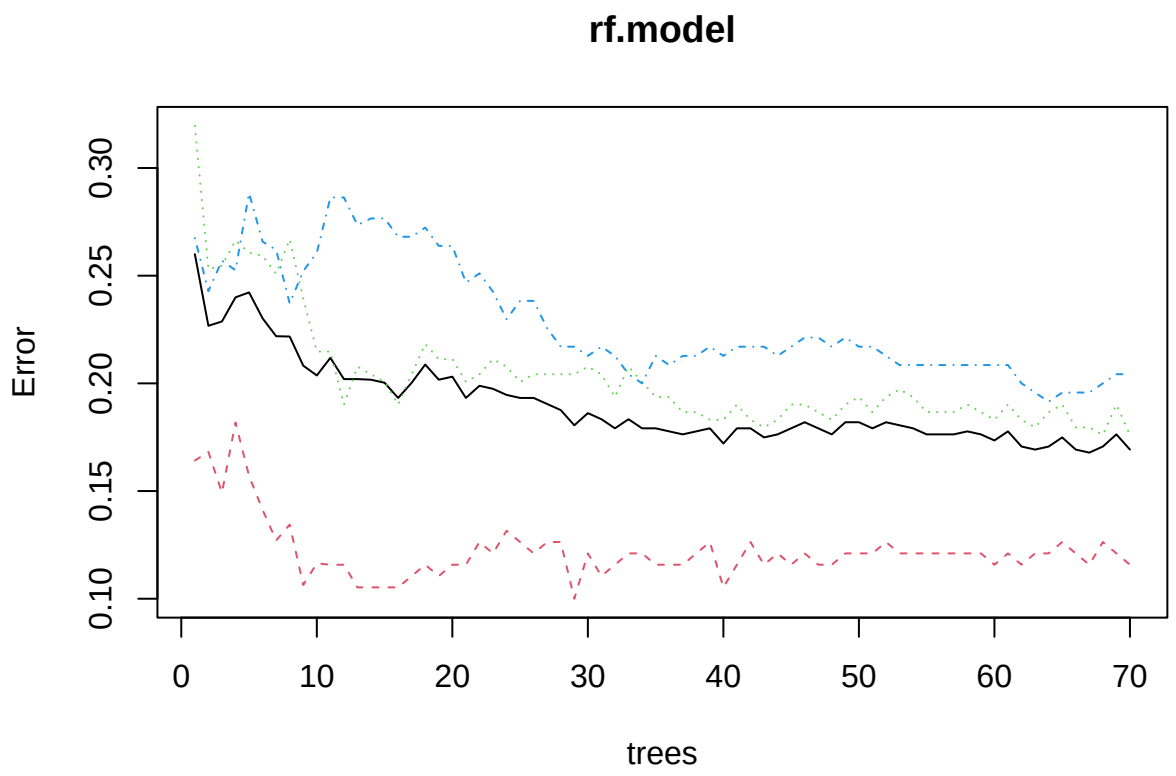



```
# building a rf model with best tune mtry value
set.seed(101)
rf.model <- randomForest(RiskLevel ~.,
                         train,
                         mtry = parameter.search.rf$bestTune$mtry,
                         ntree = 70)

# rf model OOB estimate of error rate
rf.model
```

```
##
## Call:
##  randomForest(formula = RiskLevel ~ ., data = train, mtry = parameter.search.rf$bestTune$mtry,
```

```
##                 Type of random forest: classification
##                       Number of trees: 70
## No. of variables tried at each split: 5
##
##         OOB estimate of  error rate: 16.93%
## Confusion matrix:
##           high risk low risk mid risk class.error
## high risk       168        5       17   0.1157895
## low risk          4      234       46   0.1760563
## mid risk         16       32      187   0.2042553
```

```r
# plot of the trees - error of the random forest model
plot(rf.model)
```

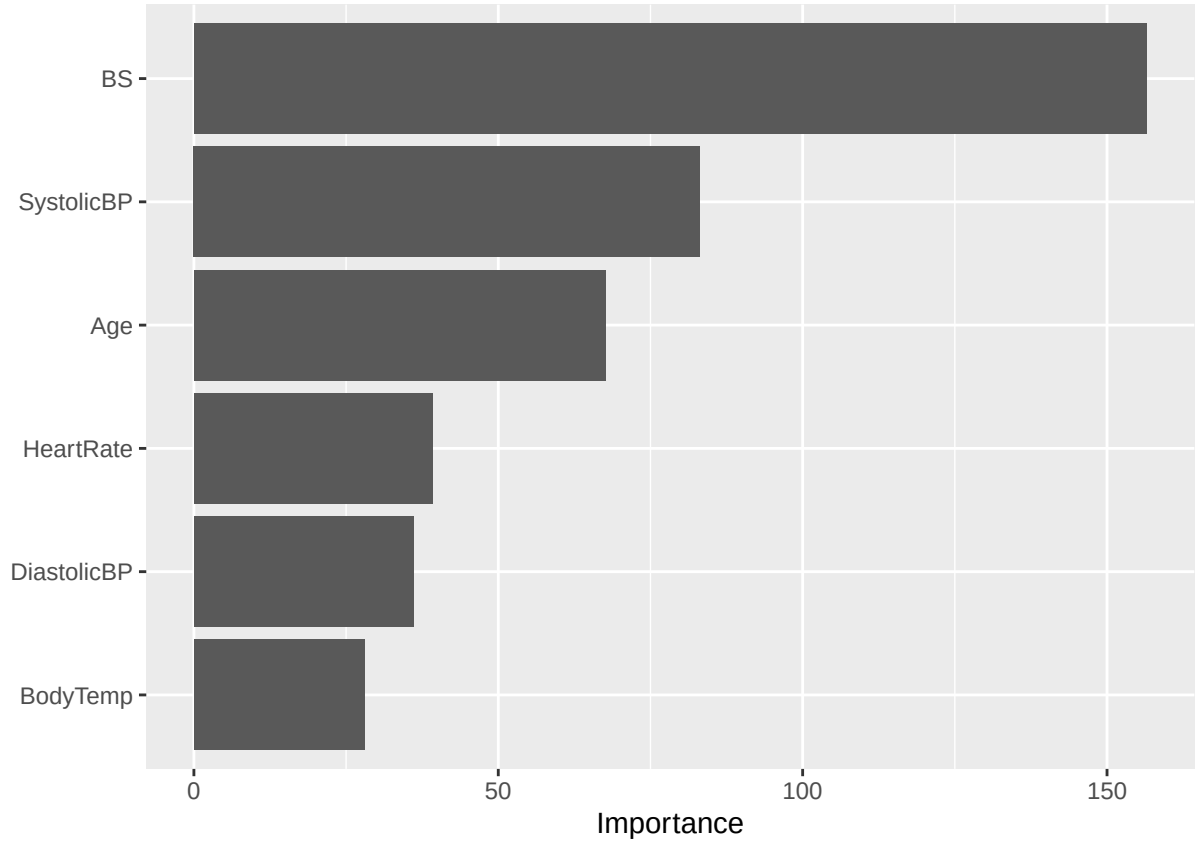



```r
# predictions of the random forest model
rf.preds <- predict(rf.model, test)

# confusion matrix of the random forest model
confusionMatrix(rf.preds, test$RiskLevel)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction  high risk low risk mid risk
##   high risk        64        4        3
```

```
##   low risk          3      101       16
##   mid risk         15       17       82
##
## Overall Statistics
##
##                Accuracy : 0.8098
##                  95% CI : (0.7612, 0.8523)
##     No Information Rate : 0.4
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.7102
##
##  Mcnemar's Test P-Value : 0.04257
##
## Statistics by Class:
##
##                      Class: high risk Class: low risk Class: mid risk
## Sensitivity                    0.7805          0.8279          0.8119
## Specificity                    0.9686          0.8962          0.8431
## Pos Pred Value                 0.9014          0.8417          0.7193
## Neg Pred Value                 0.9231          0.8865          0.9005
## Prevalence                     0.2689          0.4000          0.3311
## Detection Rate                 0.2098          0.3311          0.2689
## Detection Prevalence           0.2328          0.3934          0.3738
## Balanced Accuracy              0.8745          0.8620          0.8275
```

```r
# variable importance plot
vip(rf.model)
```

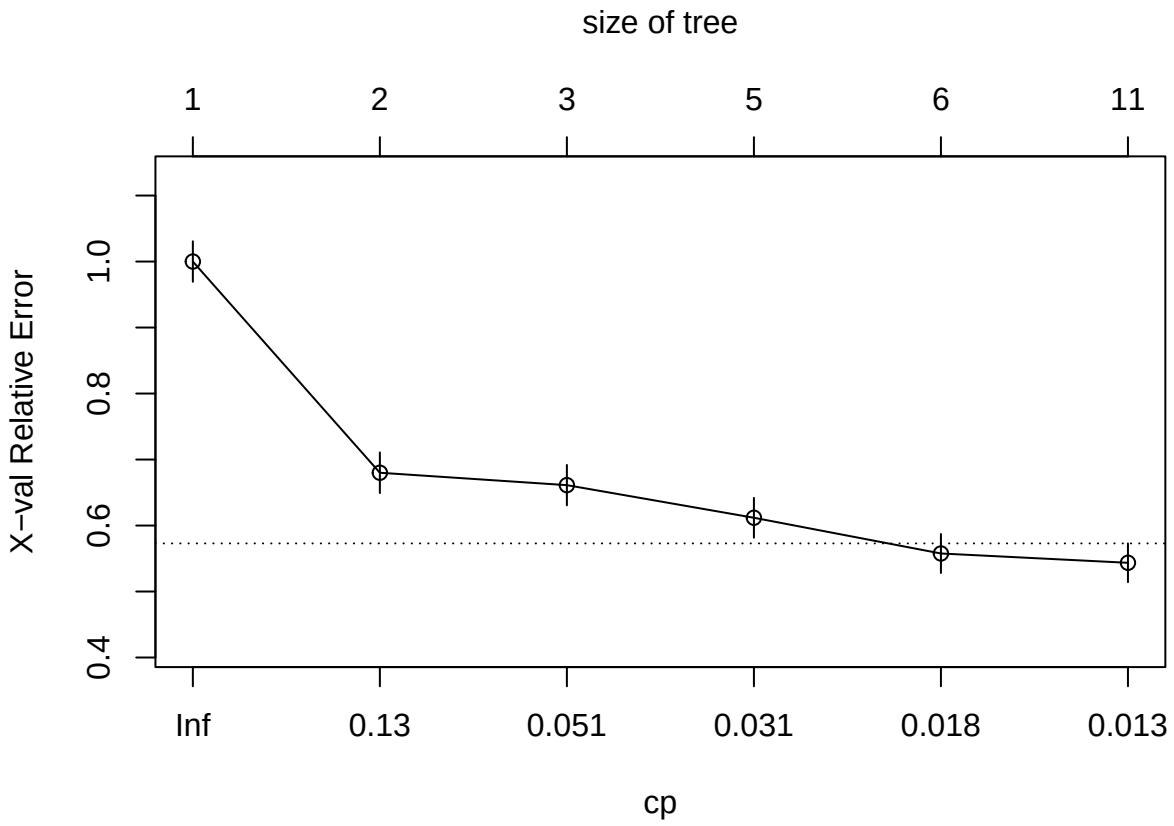



```
### CLASSIFICATION TREE
# define the parameter grid of the ct
param.grid.ct <- expand.grid(cp = seq(0.01, 1, by = 0.01))

# control parameter of the ct
ctrl.ct <- trainControl(method = "cv",
                        number = 5)

# select the best parameters for the ct
parameter.search.ct <- train(RiskLevel ~.,
                             data = train,
                             method = "rpart",
                             trControl = ctrl.ct,
                             tuneGrid = param.grid.ct)

# best tune of the ct
parameter.search.ct$bestTune$cp
```

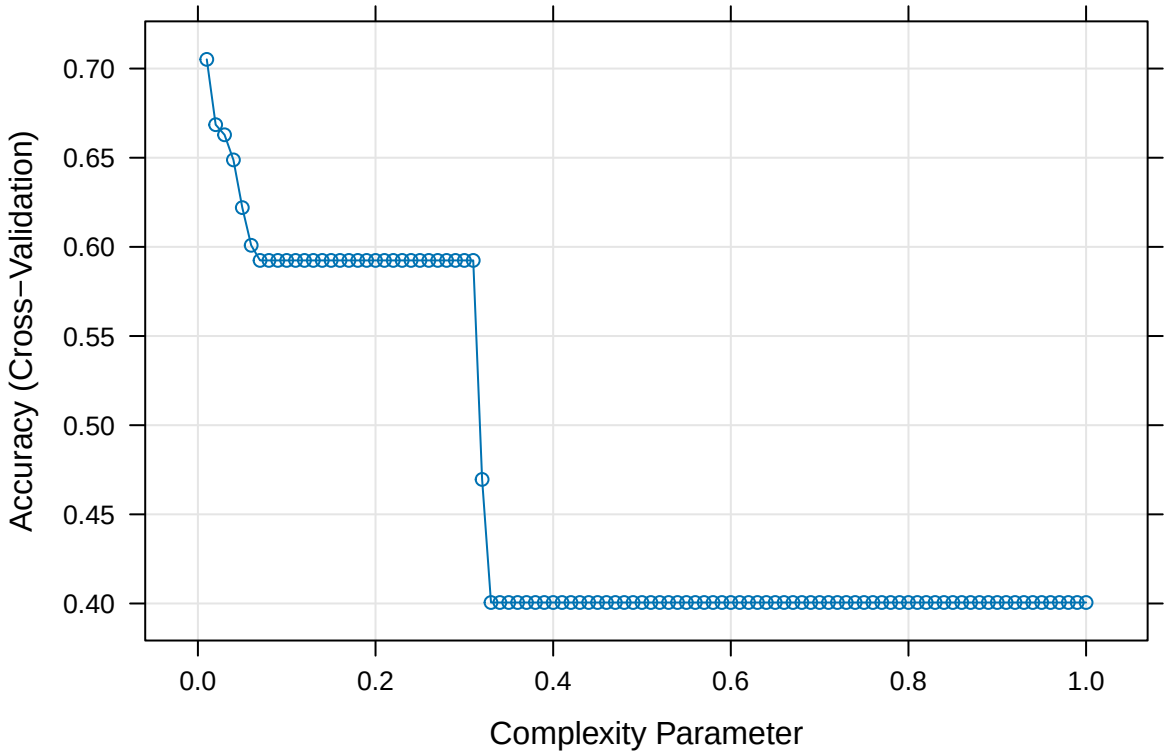
```
## [1] 0.01
```

```
# building the ct model
set.seed(101)
ct.model <- rpart(RiskLevel ~.,
                  data = train,
                  cp = parameter.search.ct$bestTune$cp)
```

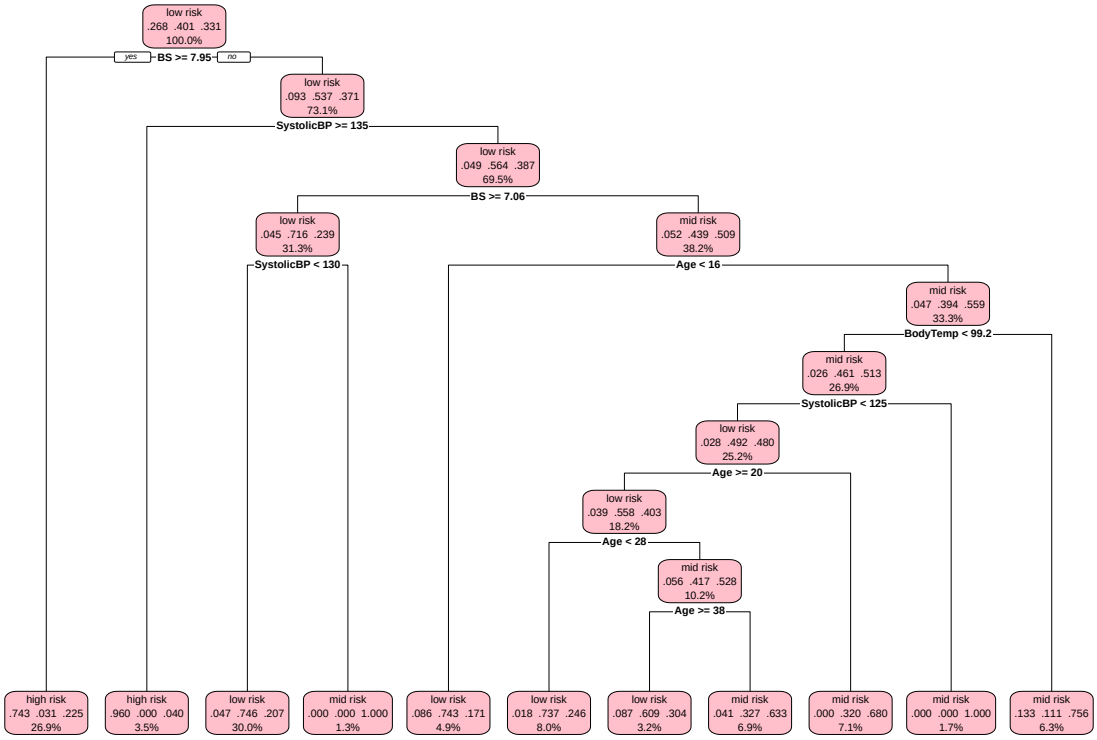
```
# cross - validation error vs. CP
plotcp(ct.model)
```




```
# complexity parameter vs. accuracy (CV) plot
plot(parameter.search.ct)
```




```r
# classification tree plot
rpart.plot(ct.model, digits = 3, box.palette = "pink")
```




```
# predictions of the CT model (test)
ct.preds <- predict(ct.model, test, type = "class")

# confusion matrix of the ct model (test)
confusionMatrix(ct.preds, test$RiskLevel)
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  high risk low risk mid risk
##   high risk        72        2       18
##   low risk          9       99       38
##   mid risk          1       21       45
##
## Overall Statistics
##
##                Accuracy : 0.7082
##                  95% CI : (0.6537, 0.7586)
##     No Information Rate : 0.4
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5543
##
##  Mcnemar's Test P-Value : 1.905e-05
##
```

```
## Statistics by Class:
##
##                      Class: high risk Class: low risk Class: mid risk
## Sensitivity                    0.8780          0.8115          0.4455
## Specificity                    0.9103          0.7432          0.8922
## Pos Pred Value                 0.7826          0.6781          0.6716
## Neg Pred Value                 0.9531          0.8553          0.7647
## Prevalence                     0.2689          0.4000          0.3311
## Detection Rate                 0.2361          0.3246          0.1475
## Detection Prevalence           0.3016          0.4787          0.2197
## Balanced Accuracy              0.8942          0.7773          0.6689
```

```r
# predictions of the CT model (train)
ct.preds.train <- predict(ct.model, train, type = "class")

# confusion matrix of the ct model (train)
confusionMatrix(ct.preds.train, train$RiskLevel)
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  high risk low risk mid risk
##   high risk       166        6       44
##   low risk         16      241       71
##   mid risk          8       37      120
##
## Overall Statistics
##
##                Accuracy : 0.7433
##                  95% CI : (0.7095, 0.7751)
##     No Information Rate : 0.4006
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6086
##
##  Mcnemar's Test P-Value : 9.796e-09
##
## Statistics by Class:
##
##                      Class: high risk Class: low risk Class: mid risk
## Sensitivity                    0.8737          0.8486          0.5106
## Specificity                    0.9037          0.7953          0.9051
## Pos Pred Value                 0.7685          0.7348          0.7273
## Neg Pred Value                 0.9513          0.8871          0.7886
## Prevalence                     0.2680          0.4006          0.3315
## Detection Rate                 0.2341          0.3399          0.1693
## Detection Prevalence           0.3047          0.4626          0.2327
## Balanced Accuracy              0.8887          0.8219          0.7079
```