Tree Based methods (22IM10040 , Sunny)

Sunny Kumar

2025-03-19

```r
df <- read.csv("D:\\Sem study materials\\study materials 6th sem\\Study materials by me\\Statistical Lea
```

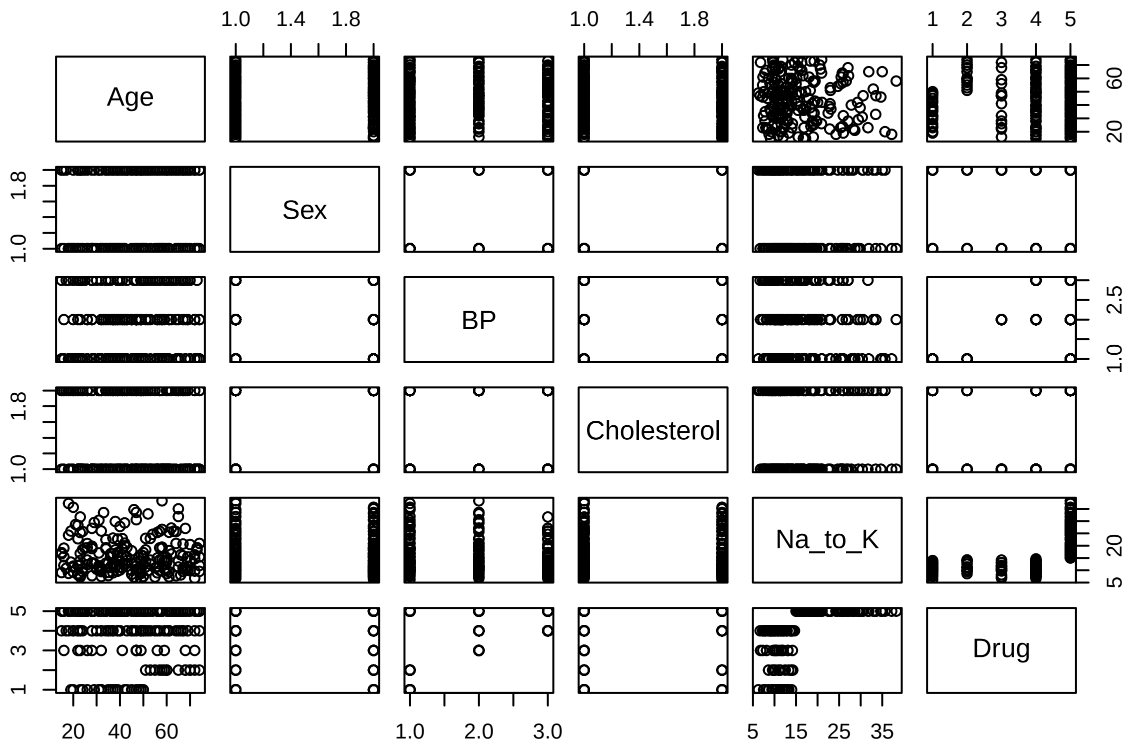
```r
dim(df)
```

```
## [1] 200   6
```

```r
head(df)
```

```
##   Age Sex     BP Cholesterol Na_to_K  Drug
## 1  23   F   HIGH        HIGH  25.355 drugY
## 2  47   M    LOW        HIGH  13.093 drugC
## 3  47   M    LOW        HIGH  10.114 drugC
## 4  28   F NORMAL        HIGH   7.798 drugX
## 5  61   F    LOW        HIGH  18.043 drugY
## 6  22   F NORMAL        HIGH   8.607 drugX
```

```r
plot(df)
```



```r
colSums(is.na(df))
```

```
##         Age         Sex          BP Cholesterol     Na_to_K        Drug
##           0           0           0           0           0           0
```

```r
unique(df$Drug)
```

```
## [1] "drugY" "drugC" "drugX" "drugA" "drugB"
```

From observing the dataset we can say that using the other data we have to predict the type/class of drug

```r
library(rpart)       # For decision tree
```

```
## Warning: package 'rpart' was built under R version 4.4.3
```

```r
library(rpart.plot)  # For visualizing the tree
```

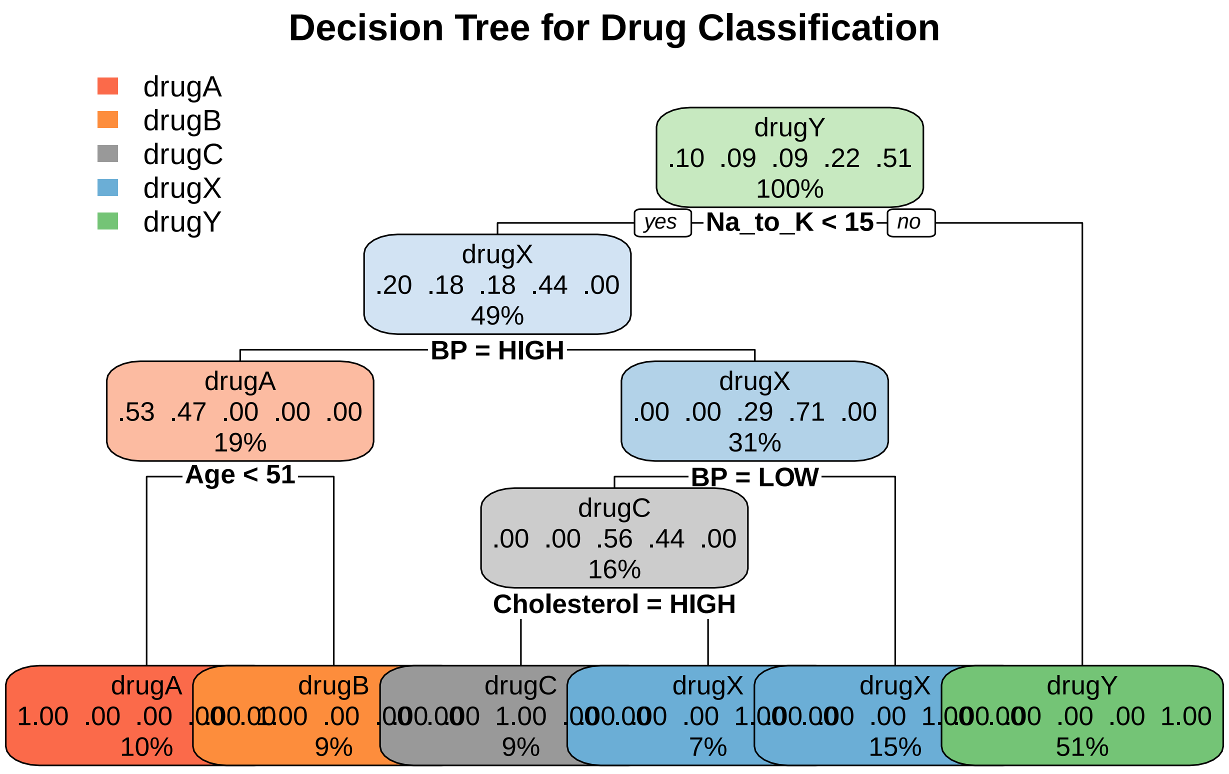
```
## Warning: package 'rpart.plot' was built under R version 4.4.3
```

```r
df$Sex <- as.factor(df$Sex)
df$BP <- as.factor(df$BP)
df$Cholesterol <- as.factor(df$Cholesterol)
df$Drug <- as.factor(df$Drug)

set.seed(123)  # For reproducibility
train_index <- sample(1:nrow(df), 0.8 * nrow(df))  # 80% train, 20% test
df_train <- df[train_index, ]
df_test <- df[-train_index, ]

tree_model <- rpart(Drug ~ ., data = df_train, method = "class")
```

```r
rpart.plot(tree_model, type = 2, extra = 104, cex = 0.8, main = "Decision Tree for Drug Classification")
```



Decision Tree for Drug Classification
drugA
drugB
drugC
drugX
drugY
drugY .10 .09 .09 .22 .51 100%
yes
Na_to_K < 15
no
drugX .20 .18 .18 .44 .00 49%
BP = HIGH
drugA .53 .47 .00 .00 .00 19%
drugX .00 .00 .29 .71 .00 31%
Age < 51
BP = LOW
drugC .00 .00 .56 .44 .00 16%
Cholesterol = HIGH
drugA 1.00 .00 .00 .00 .00 10%
drugB .00 1.00 .00 .00 .00 9%
drugC .00 .00 1.00 .00 .00 9%
drugX .00 .00 .00 1.00 .00 7%
drugX .00 .00 .00 1.00 .00 15%
drugY .00 .00 .00 .00 1.00 51%

```r
printcp(tree_model)  # Displays CP values and tree performance
```

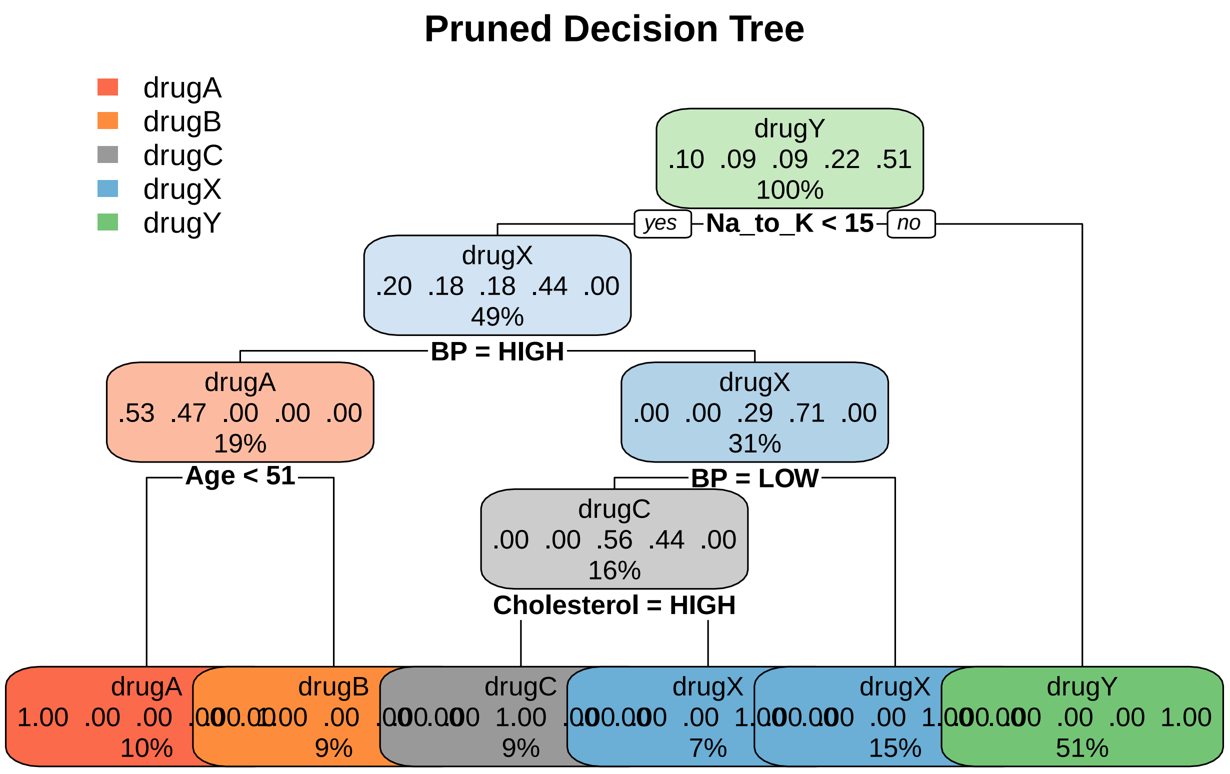
```
##
## Classification tree:
## rpart(formula = Drug ~ ., data = df_train, method = "class")
##
## Variables actually used in tree construction:
## [1] Age         BP          Cholesterol Na_to_K
##
## Root node error: 79/160 = 0.49375
```

```
##
## n= 160
##
##          CP nsplit rel error   xerror     xstd
## 1 0.443038      0   1.00000 1.000000 0.080051
## 2 0.202532      1   0.55696 0.569620 0.071989
## 3 0.177215      2   0.35443 0.417722 0.064785
## 4 0.088608      3   0.17722 0.215190 0.049341
## 5 0.010000      5   0.00000 0.037975 0.021718
```

```
best_cp <- tree_model$cptable[which.min(tree_model$cptable[,"xerror"]), "CP"]
cat("Optimal CP:", best_cp, "\n")
```

```
## Optimal CP: 0.01
```

```
pruned_tree <- prune(tree_model, cp = best_cp)
library(rpart.plot)
rpart.plot(pruned_tree, type = 2, extra = 104, cex = 0.8, main = "Pruned Decision Tree")
```



```
pruned_predictions <- predict(pruned_tree, df_test, type = "class")
pruned_conf_matrix <- table(Predicted = pruned_predictions, Actual = df_test$Drug)

# Print confusion matrix & accuracy
print(pruned_conf_matrix)
```

```
##           Actual
## Predicted drugA drugB drugC drugX drugY
##     drugA     7     0     0     0     0
##     drugB     0     2     0     0     0
##     drugC     0     0     2     0     0
##     drugX     0     0     0    19     0
##     drugY     0     0     0     0    10
```

```
pruned_accuracy <- sum(diag(pruned_conf_matrix)) / sum(pruned_conf_matrix)
cat("Pruned Tree Accuracy:", pruned_accuracy, "\n")
```

```
## Pruned Tree Accuracy: 1
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.4.3
```

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

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

```
set.seed(123)
bagging_model <- randomForest(Drug ~ ., data = df_train, mtry = ncol(df_train) - 1, ntree = 500, importa
bagging_pred <- predict(bagging_model, df_test)
bagging_acc <- mean(bagging_pred == df_test$Drug)
cat("Bagging Model Accuracy:", bagging_acc, "\n")
```

```
## Bagging Model Accuracy: 1
```

```
set.seed(123)
rf_model <- randomForest(Drug ~ ., data = df_train, mtry = sqrt(ncol(df_train) - 1), ntree = 500, impor
rf_pred <- predict(rf_model, df_test)
rf_acc <- mean(rf_pred == df_test$Drug)
cat("Random Forest Accuracy:", rf_acc, "\n")
```

```
## Random Forest Accuracy: 1
```

```
set.seed(123)
rf_mtry3 <- randomForest(Drug ~ ., data = df_train, mtry = 3, ntree = 500)
rf_mtry5 <- randomForest(Drug ~ ., data = df_train, mtry = 5, ntree = 500)

acc_mtry3 <- mean(predict(rf_mtry3, df_test) == df_test$Drug)
acc_mtry5 <- mean(predict(rf_mtry5, df_test) == df_test$Drug)

cat("Accuracy (mtry = 3):", acc_mtry3, "\n")
```

```
## Accuracy (mtry = 3): 1
```

```r
cat("Accuracy (mtry = 5):", acc_mtry5, "\n")
```

```
## Accuracy (mtry = 5): 1
```

```r
library(caret)
```

```
## Loading required package: ggplot2
```

```
##
## Attaching package: 'ggplot2'
```

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

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

```r
set.seed(123)
tune_grid <- expand.grid(mtry = c(2, 3, 4, 5, 6))

control <- trainControl(method = "cv", number = 5)

rf_tuned <- train(Drug ~ ., data = df_train, method = "rf", tuneGrid = tune_grid, trControl = control)

print(rf_tuned)
```

```
## Random Forest
##
## 160 samples
##   5 predictor
##   5 classes: 'drugA', 'drugB', 'drugC', 'drugX', 'drugY'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 128, 127, 128, 128, 129
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##   2     0.9810484  0.9717581
##   3     0.9747984  0.9625095
##   4     0.9747984  0.9625095
##   5     0.9747984  0.9625095
##   6     0.9747984  0.9625095
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

```r
best_mtry <- rf_tuned$bestTune$mtry

cat("Best mtry:", best_mtry, "\n")
```

```
## Best mtry: 2
```

```r
# Train the final best model
final_rf <- randomForest(Drug ~ ., data = df_train, mtry = best_mtry, ntree = 500)
final_pred <- predict(final_rf, df_test)

# Calculate Accuracy
final_acc <- mean(final_pred == df_test$Drug)
cat("Final Tuned Random Forest Accuracy:", final_acc, "\n")
```

```
## Final Tuned Random Forest Accuracy: 1
```

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this: