8 . DECISION TREE

23CSEG28

```
# Loading the library
library(dplyr)

library(lattice)
library(ggplot2)
library(caret)
library(rpart)
library(rpart.plot)
library(entropy)

#Loading the dataset
drug200 =read.csv("C:/Users/kavin/Downloads/drug200.csv")

#Summary structure of the data
summary(drug200)
```

```
##       Age            Sex                 BP            Cholesterol
##  Min.   :15.00   Length:200         Length:200         Length:200
##  1st Qu.:31.00   Class :character   Class :character   Class :character
##  Median :45.00   Mode  :character   Mode  :character   Mode  :character
##  Mean   :44.31
##  3rd Qu.:58.00
##  Max.   :74.00
##     Na_to_K           Drug
##  Min.   : 6.269   Length:200
##  1st Qu.:10.445   Class :character
##  Median :13.937   Mode  :character
##  Mean   :16.084
##  3rd Qu.:19.380
##  Max.   :38.247
```

```
str(drug200)
```

```
## 'data.frame':    200 obs. of  6 variables:
##  $ Age        : int  23 47 47 28 61 22 49 41 60 43 ...
##  $ Sex        : chr  "F" "M" "M" "F" ...
##  $ BP         : chr  "HIGH" "LOW" "LOW" "NORMAL" ...
##  $ Cholesterol: chr  "HIGH" "HIGH" "HIGH" "HIGH" ...
##  $ Na_to_K    : num  25.4 13.1 10.1 7.8 18 ...
##  $ Drug       : chr  "drugY" "drugC" "drugC" "drugX" ...
```

```
#Checking missing values
sum(is.na(drug200))
```

```
## [1] 0
```

```
#Frequency of Age
histogram(~Age,data = drug200,col='Maroon',main="#1A Frequency of Age")
```

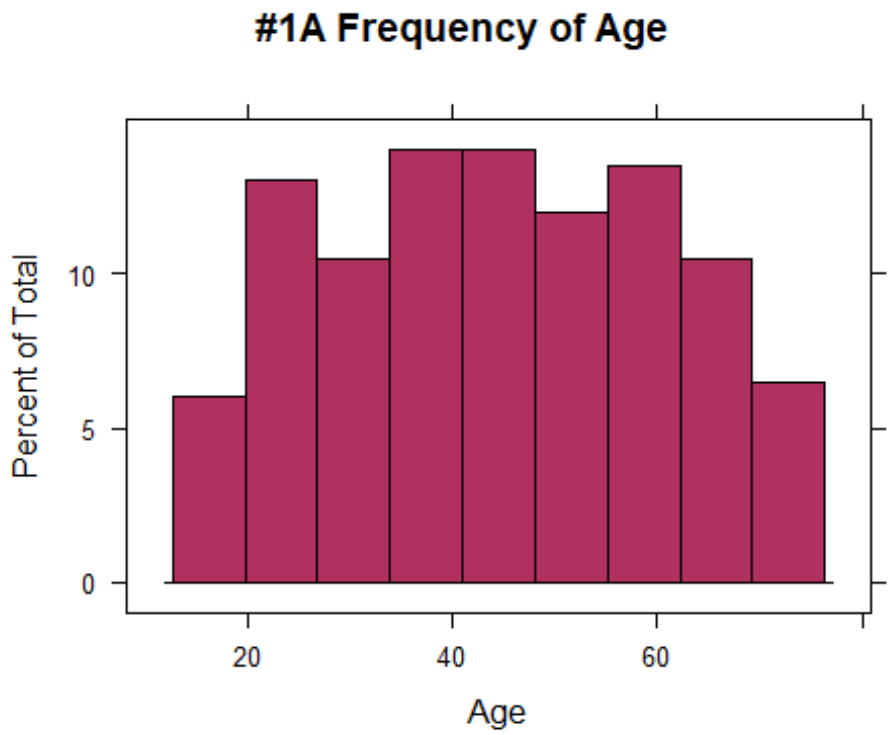


Fig 8.1

```
#Frequency of Age wrt Sex
histogram(~Age|Sex,data=drug200,breaks = 20,col='darkgreen',main="#1B Freq
uency of Age wrt Sex")
```

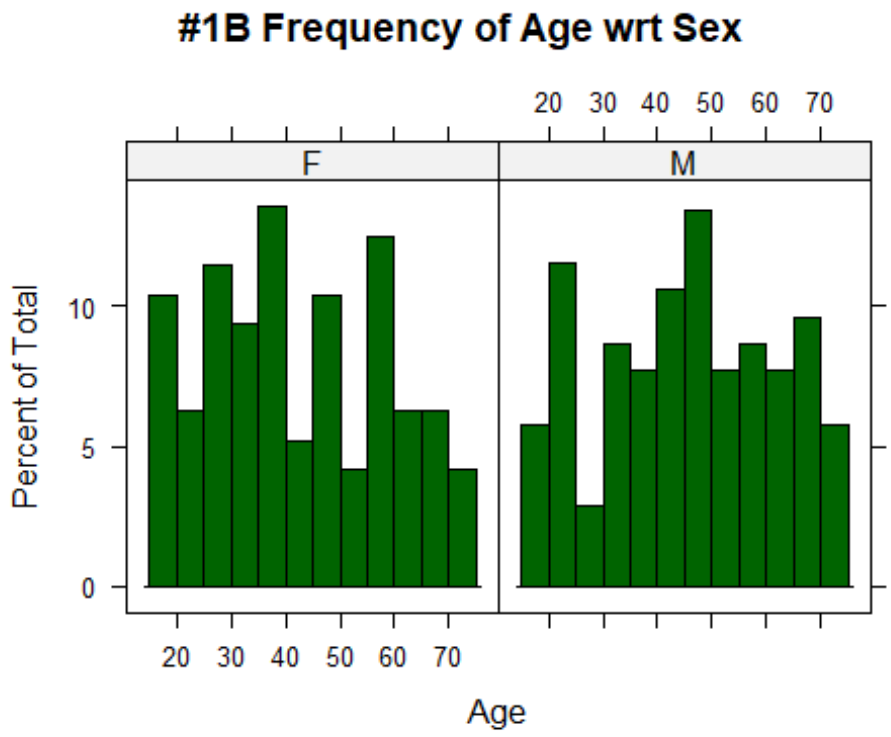


Fig 8.2

```
#Frequency of Age wrt BP
histogram(~Age|BP,data=drug200,breaks = 20,col='red',main="#1C Frequency o
f Age wrt BP")
```

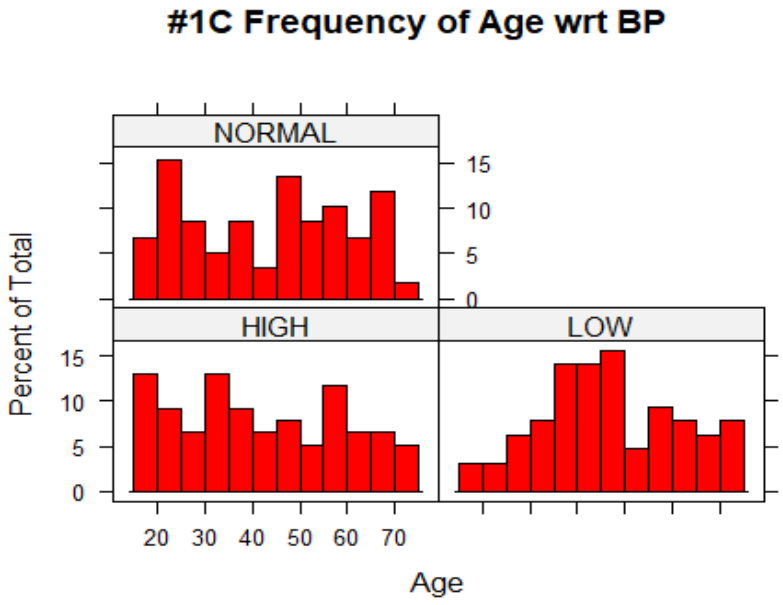


Fig 8.3

```
#Frequency of Age wrt Cholesterol
histogram(~Age|Cholesterol,data=drug200,breaks = 20,col='skyblue',main="#1
D Frequency of Age wrt Cholesterol")
```

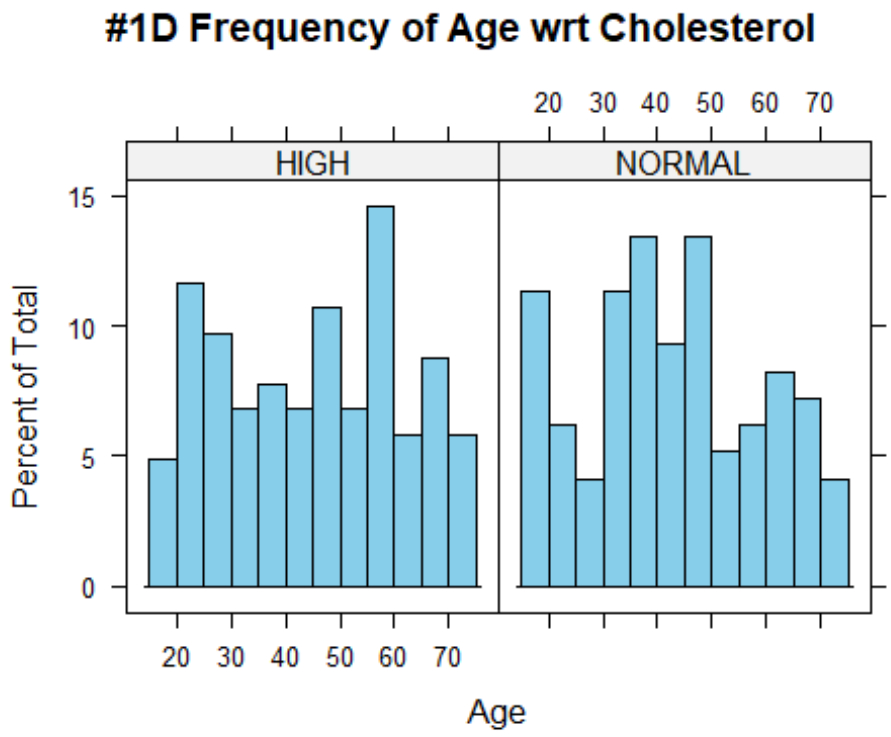


Fig 8.4

```
#Frequency of Age wrt Drug
histogram(~Age|Drug,data=drug200,breaks = 10,col='yellow',main="#1E Freque
ncy of Age wrt Drug")
```

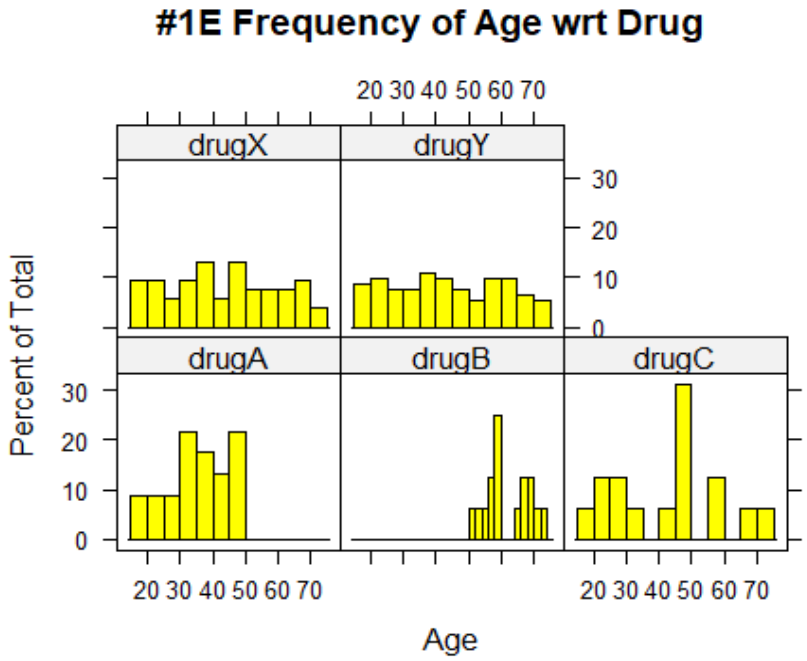


Fig 8.5

```
# Frequency of NA to K
histogram(~Na_to_K,data = drug200,breaks=20,main="#1F Frequency of NA to K
")
```

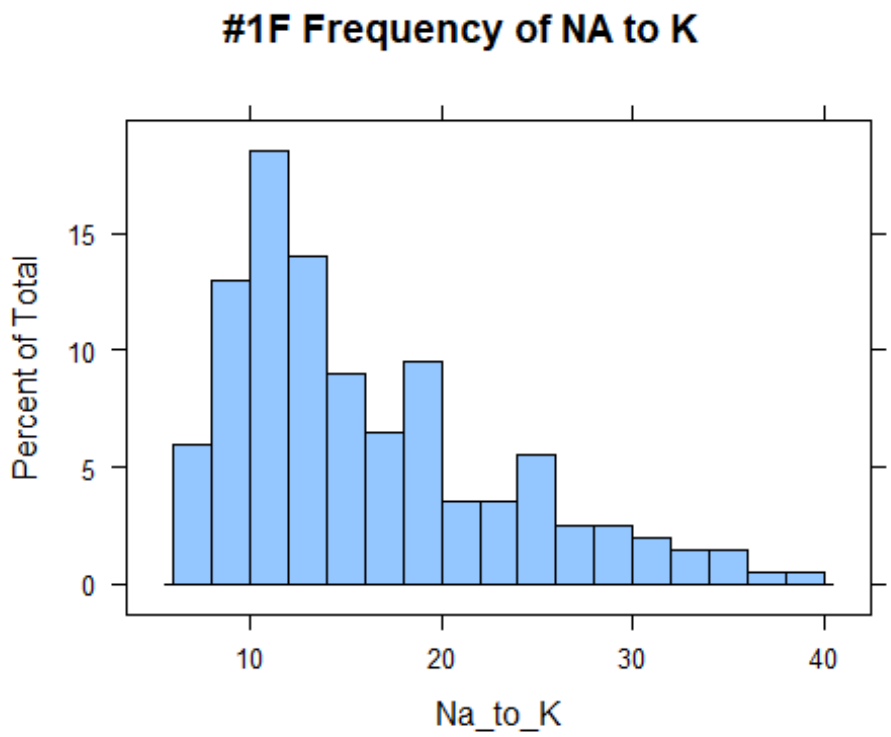


Fig 8.6

```
#Categorize the drug according to Age

bwplot(Age~Drug|Sex,data =drug200,col='green',main='#2A Categorize the dru
g according to Age' ,xlab= 'Drug  category')
```

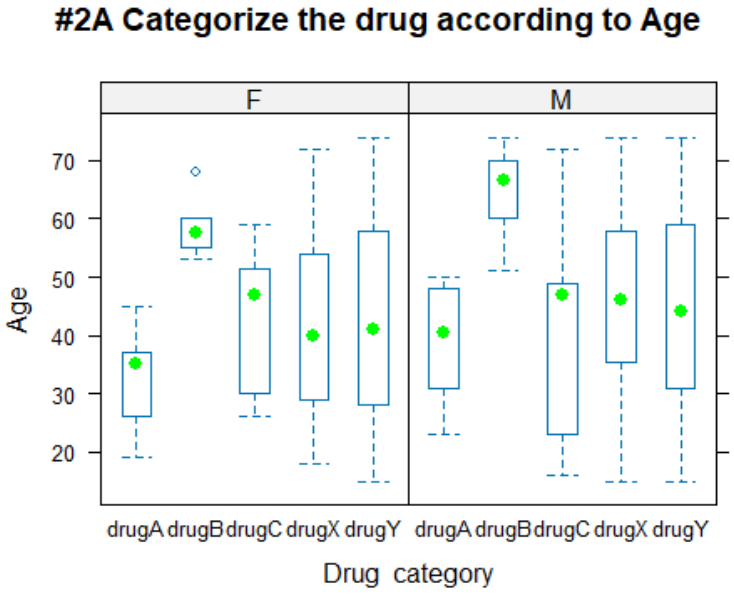


Fig 8.7

```
#Categorize the Cholesterol according to Age
bwplot(Age~Cholesterol|Sex,data =drug200,col='red',main='#2B Categorize th
e Cholesterol according to Age ',xlab = 'Cholesterol Category')
```

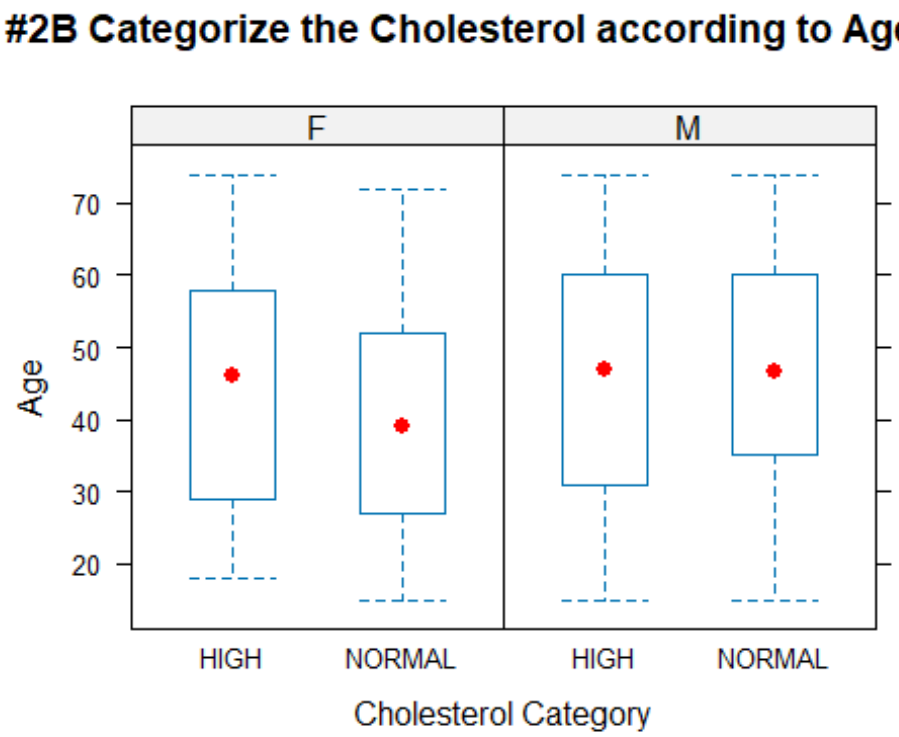


Fig 8.8

```
#Categorize the BP according to Age
bwplot(Age~BP|Sex,data =drug200,col='red',main='#2C Categorize the BP acco
rding to Age',xlab='Age Category')
```

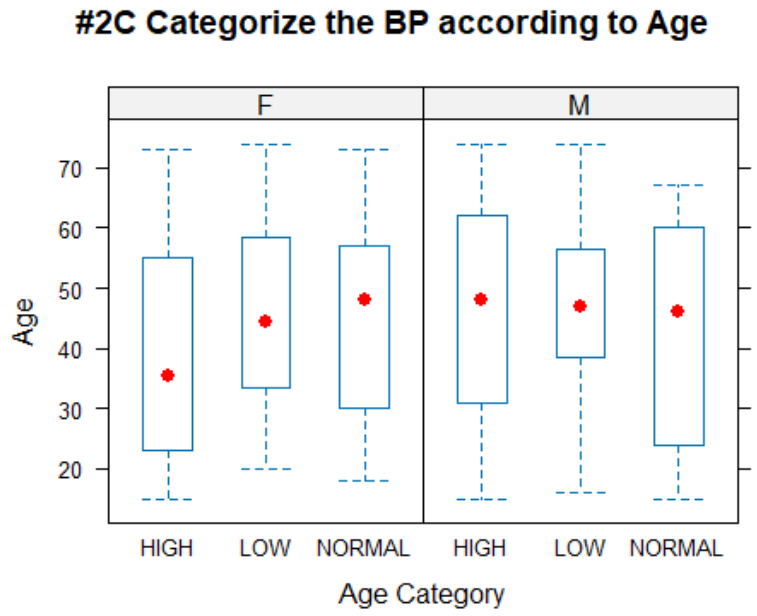


Fig 8.9

```
#Categorize  Na according to Sex
bwplot(Sex~Na_to_K,data =drug200,col='red',main='#2D Categorize  Na accord
ing to Sex',xlab = 'Na_to_K')
```

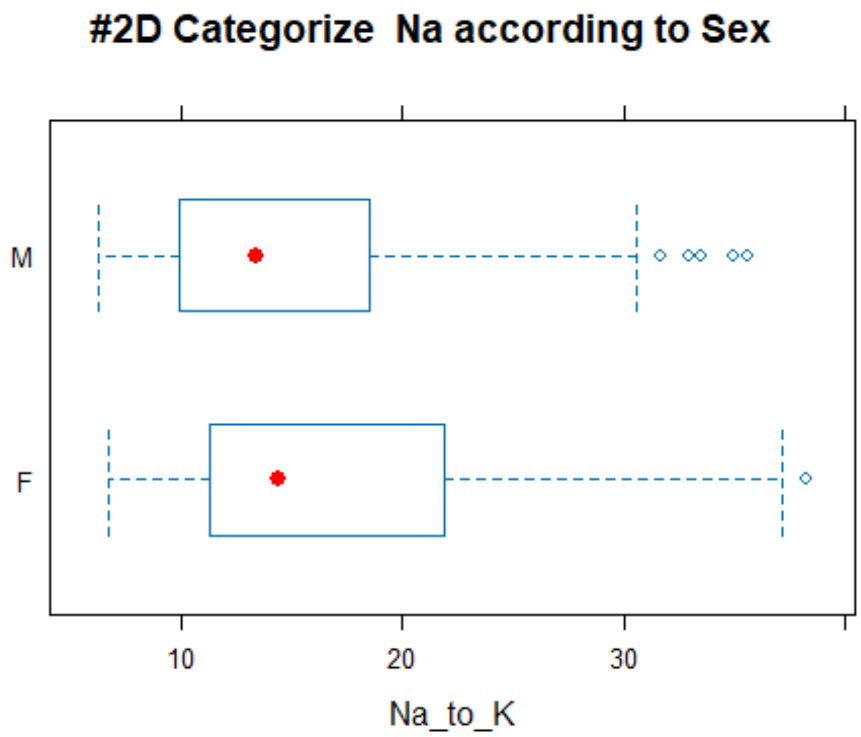


Fig 8.10

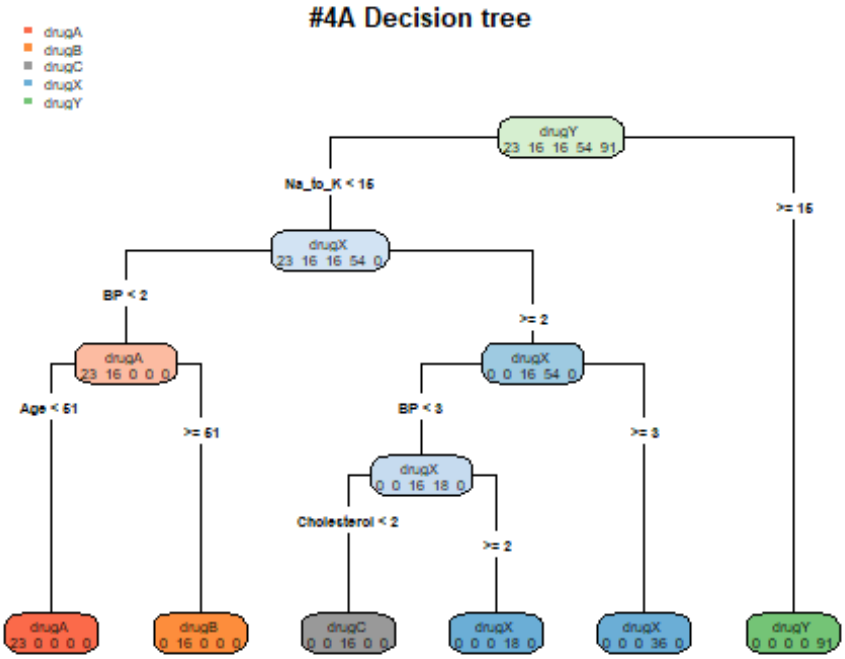
```
#parameteric test anova test
anova_result <- aov(Na_to_K~Drug, data = drug200)
# Extract p-value
p_value <- summary(anova_result)[[1]]$"Pr(>F)"[1]
# Check the p-value
if (p_value < 0.05) {
  print("Assumption1 : Na to K  and Drug are related to each other(p-value
< 0.05)" )
} else {
  print("Assumption1 :Na to k is not related to Drug (p-value >= 0.05)")
}

## [1] "Assumption1 : Na to K  and Drug are related to each other(p-value
< 0.05)"

# Assuming 'drug200' is your dataframe with categorical variables
# Convert categorical variables to factors

num <- drug200[, -ncol(drug200)]
Drug <- drug200[, ncol(drug200)]
num$Sex <- as.integer(factor(num$Sex))
num$BP <- as.integer(factor(num$BP))
num$Cholesterol <- as.integer(factor(num$Cholesterol))
num=cbind(num,Drug)
head(num)
```

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

```
#Model Building
X_train_test <- createDataPartition(Drug, p = 0.7, list = FALSE)

# Split input into training and testing sets
X_train <- num[X_train_test, ]
X_test <- num[-X_train_test, ]

# Split target into training and testing sets

y_train <- Drug[X_train_test]
y_test <- Drug[-X_train_test]
# Train decision tree model
clf <- rpart(y_train ~ ., data = X_train,method="class")

# Make predictions on test set
predictions <- predict(clf, X_test, type = "class")
# Calculate accuracy
accuracy =mean(predictions == y_test)
accuracy
```

```
## [1] 0.9824561
```

```
fit<-rpart(Drug~.,method="class",data = num,control=rpart.control(minsplit
= 1),parms=list(split='information'))
rpart.plot(fit,type=4,extra=1,main="#4A Decision tree")
```




```
drug_df <- data.frame(num$Age,num$Na_to_K,num$Drug,num$BP,num$Cholesterol)
entropy_drug <- entropy::entropy(table(drug_df$Drug))
print("Entropy for Drug variable:")
```

```
## [1] "Entropy for Drug variable:"
```

```
print(entropy_drug)
```

```
## [1] 0
```

```
predictions <- predict(fit, newdata = num, type = "class")
confusion_matrix <- table(predictions, num$Drug)
print("Confusion Matrix:")
```

```
## [1] "Confusion Matrix:"
```

```
print(confusion_matrix)
## predictions drugA drugB drugC drugX drugY
##       drugA    23     0     0     0     0
##       drugB     0    16     0     0     0
##       drugC     0     0    16     0     0
##       drugX     0     0     0    54     0
##       drugY     0     0     0     0    91
```

```
precision <- diag(confusion_matrix) / rowSums(confusion_matrix)
print("Precision:")
```

```
## [1] "Precision:"
```

```
print(precision)
```

```
## drugA drugB drugC drugX drugY
##     1     1     1     1     1
```

```r
# Calculate Recall (Sensitivity)
recall <- diag(confusion_matrix) / colSums(confusion_matrix)
print("Recall:")
```

```
## [1] "Recall:"
```

```r
print(recall)
```

```
## drugA drugB drugC drugX drugY
##     1     1     1     1     1
```

```r
accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
print("Accuracy:")
```

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

```r
print(accuracy)
```

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