AI/Ml

### Decision Tree

Step 1: Set worrking enviroment and import data

# setwd("H:/My Drive/R Project/SleepHealthandLifestyle")  
data <- read.csv("Sleep\_health\_and\_lifestyle\_dataset.csv")

data <- data[, c(2:3, 5:13)]

Step 2: Check the data class and transform data

data$Blood.Pressure <- sub("/", " ", data$Blood.Pressure)  
data <- separate(data, Blood.Pressure, into = c("Systolic", "Diastolic"), sep = " ")  
data$Systolic <- as.numeric(data$Systolic)  
data$Diastolic <- as.numeric(data$Diastolic)  
  
data$BMI.Category<- gsub("Normal Weight", "Normal", data$BMI.Category)  
  
data$BMI.Category<- gsub("Obese", "Obesity", data$BMI.Category)  
  
data$Sleep.Disorder[data$Sleep.Disorder == "Sleep Apnea"] = "Sleep.Apnea"  
data <- data %>% transform(Sleep.Disorder = as.factor(Sleep.Disorder))  
  
data$Gender <- as.factor(data$Gender)  
  
# data$Occupation <- as.factor(data$Occupation)  
  
data$BMI.Category <- as.factor(data$BMI.Category)

Step 3: Divide data into training and testing grou[ and run the demo test

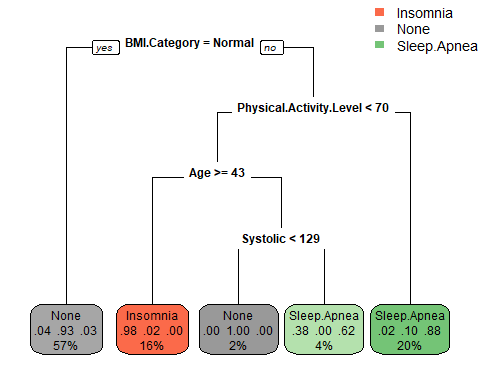
set.seed(111)  
ind <- sample(2, nrow(data), replace = TRUE, prob = c(0.8, 0.2))  
train\_data <- data[ind==1,]  
test\_data <- data[ind==2,]  
  
addmargins(prop.table(table(ind)))

## ind  
## 1 2 Sum   
## 0.7941176 0.2058824 1.0000000

decision\_tree<- rpart(Sleep.Disorder ~ .,  
 data = train\_data,  
 method = 'class',  
 xval = 10, # k-fold với k = 10  
 )  
  
printcp(decision\_tree)

##   
## Classification tree:  
## rpart(formula = Sleep.Disorder ~ ., data = train\_data, method = "class",   
## xval = 10)  
##   
## Variables actually used in tree construction:  
## [1] Age BMI.Category Physical.Activity.Level  
## [4] Systolic   
##   
## Root node error: 125/297 = 0.42088  
##   
## n= 297   
##   
## CP nsplit rel error xerror xstd  
## 1 0.376 0 1.000 1.000 0.068066  
## 2 0.344 1 0.624 0.800 0.065155  
## 3 0.040 2 0.280 0.288 0.044997  
## 4 0.010 4 0.200 0.320 0.047066

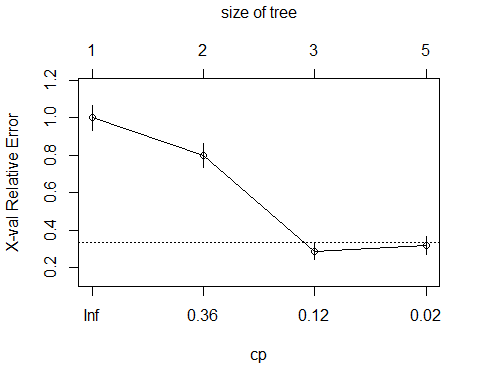
rpart.plot(decision\_tree,type=0, yesno = TRUE)



rpart.rules(decision\_tree)

## Sleep.Disorder Inso None Slee   
## Insomnia [ .98 .02 .00] when BMI.Category is Obesity or Overweight & Physical.Activity.Level < 70 & Age >= 43   
## None [ .04 .93 .03] when BMI.Category is Normal   
## None [ .00 1.00 .00] when BMI.Category is Obesity or Overweight & Physical.Activity.Level < 70 & Age < 43 & Systolic < 129  
## Sleep.Apnea [ .38 .00 .62] when BMI.Category is Obesity or Overweight & Physical.Activity.Level < 70 & Age < 43 & Systolic >= 129  
## Sleep.Apnea [ .02 .10 .88] when BMI.Category is Obesity or Overweight & Physical.Activity.Level >= 70

plotcp(decision\_tree)

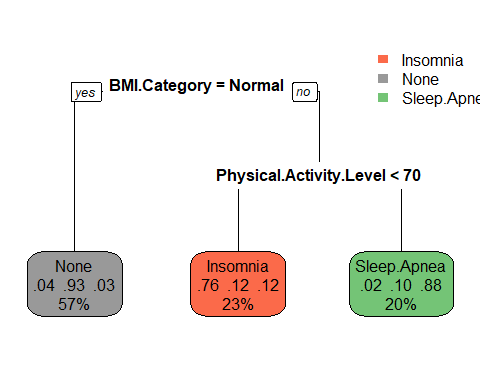


Step 4: Prune the tree to optimal size

library(caret)  
decision\_tree\_prune <- prune(decision\_tree,   
 cp = decision\_tree$cptable[which.min(decision\_tree$cptable[, "xerror"]), "CP"])  
  
printcp(decision\_tree\_prune)

##   
## Classification tree:  
## rpart(formula = Sleep.Disorder ~ ., data = train\_data, method = "class",   
## xval = 10)  
##   
## Variables actually used in tree construction:  
## [1] BMI.Category Physical.Activity.Level  
##   
## Root node error: 125/297 = 0.42088  
##   
## n= 297   
##   
## CP nsplit rel error xerror xstd  
## 1 0.376 0 1.000 1.000 0.068066  
## 2 0.344 1 0.624 0.800 0.065155  
## 3 0.040 2 0.280 0.288 0.044997

rpart.plot(decision\_tree\_prune, type=0, yesno = TRUE)



(rpart.rules(decision\_tree\_prune))

## Sleep.Disorder Ins Non Sle   
## Insomnia [.76 .12 .12] when BMI.Category is Obesity or Overweight & Physical.Activity.Level < 70  
## None [.04 .93 .03] when BMI.Category is Normal   
## Sleep.Apnea [.02 .10 .88] when BMI.Category is Obesity or Overweight & Physical.Activity.Level >= 70

Step 5: Save Decision tree Model

save(decision\_tree\_prune, file = "DecisionTree.rda")

## Random Forest

Step 1: set working environment and import data.

data <- read.csv("Sleep\_health\_and\_lifestyle\_dataset.csv")  
data <- data[, c(2:3, 5:13)]

Step 2: Check data and Transform data.

summary(data)

## Gender Age Sleep.Duration Quality.of.Sleep  
## Length:374 Min. :27.00 Min. :5.800 Min. :4.000   
## Class :character 1st Qu.:35.25 1st Qu.:6.400 1st Qu.:6.000   
## Mode :character Median :43.00 Median :7.200 Median :7.000   
## Mean :42.18 Mean :7.132 Mean :7.313   
## 3rd Qu.:50.00 3rd Qu.:7.800 3rd Qu.:8.000   
## Max. :59.00 Max. :8.500 Max. :9.000   
## Physical.Activity.Level Stress.Level BMI.Category Blood.Pressure   
## Min. :30.00 Min. :3.000 Length:374 Length:374   
## 1st Qu.:45.00 1st Qu.:4.000 Class :character Class :character   
## Median :60.00 Median :5.000 Mode :character Mode :character   
## Mean :59.17 Mean :5.385   
## 3rd Qu.:75.00 3rd Qu.:7.000   
## Max. :90.00 Max. :8.000   
## Heart.Rate Daily.Steps Sleep.Disorder   
## Min. :65.00 Min. : 3000 Length:374   
## 1st Qu.:68.00 1st Qu.: 5600 Class :character   
## Median :70.00 Median : 7000 Mode :character   
## Mean :70.17 Mean : 6817   
## 3rd Qu.:72.00 3rd Qu.: 8000   
## Max. :86.00 Max. :10000

data$Blood.Pressure <- sub("/", " ", data$Blood.Pressure)  
data <- separate(data, Blood.Pressure, into = c("Systolic", "Diastolic"), sep = " ")  
data$Systolic <- as.numeric(data$Systolic)  
data$Diastolic <- as.numeric(data$Diastolic)  
  
data$BMI.Category<- gsub("Normal Weight", "Normal", data$BMI.Category)  
  
data$BMI.Category<- gsub("Obese", "Obesity", data$BMI.Category)  
  
data$Sleep.Disorder[data$Sleep.Disorder == "Sleep Apnea"] = "Sleep.Apnea"  
data <- data %>% transform(Sleep.Disorder = as.factor(Sleep.Disorder))  
  
data$Gender <- as.factor(data$Gender)  
  
# data$Occupation <- as.factor(data$Occupation)  
  
data$BMI.Category <- as.factor(data$BMI.Category)

Step 3: Divide data into training and testing group and run the demo model

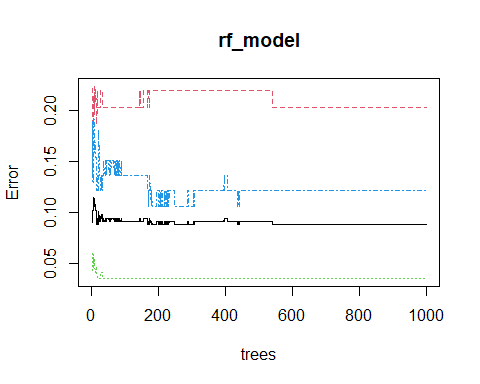
set.seed(111)  
ind <- sample(2, nrow(data), replace = TRUE, prob = c(0.8, 0.2))  
train\_data <- data[ind==1,]  
test\_data <- data[ind==2,]  
  
addmargins(prop.table(table(ind)))

## ind  
## 1 2 Sum   
## 0.7941176 0.2058824 1.0000000

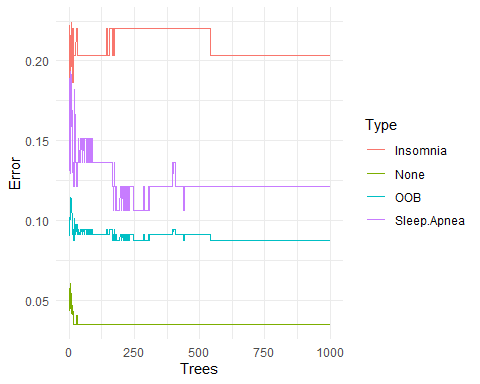
rf\_model <- randomForest(Sleep.Disorder~., data = train\_data, ntree = 1000, important = TRUE)  
rf\_model

##   
## Call:  
## randomForest(formula = Sleep.Disorder ~ ., data = train\_data, ntree = 1000, important = TRUE)   
## Type of random forest: classification  
## Number of trees: 1000  
## No. of variables tried at each split: 3  
##   
## OOB estimate of error rate: 8.75%  
## Confusion matrix:  
## Insomnia None Sleep.Apnea class.error  
## Insomnia 47 5 7 0.20338983  
## None 1 166 5 0.03488372  
## Sleep.Apnea 5 3 58 0.12121212

plot(rf\_model)



library(randomForest)  
oob\_err\_data <- data.frame(  
 Trees = rep(1:nrow(rf\_model$err.rate), 4),   
 Type = rep(c("OOB", "Insomnia", "None", "Sleep.Apnea"), each = nrow(rf\_model$err.rate)),  
 Error = c(rf\_model$err.rate[,"OOB"], rf\_model$err.rate[,"Insomnia"], rf\_model$err.rate[,"None"], rf\_model$err.rate[, "Sleep.Apnea"]))  
  
ggplot(data = oob\_err\_data, aes(x = Trees, y= Error)) + geom\_line(aes(color = Type))+ theme(text = element\_text(size = 20)) + theme\_minimal()



Step 4: Find the optimal mtry value

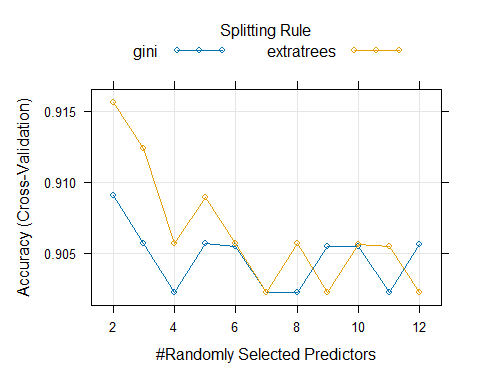
rf\_fit <- train(Sleep.Disorder~.,  
 data = train\_data,  
 method = "ranger",  
 tuneLength = 12,  
 metric = "ROC",  
 trControl = trainControl(  
 method = "cv", # k-fold cross validation  
 number = 10, # 10 folds  
 savePredictions = "final",   
 classProbs = TRUE,   
 summaryFunction = defaultSummary   
 )  
 )

## note: only 11 unique complexity parameters in default grid. Truncating the grid to 11 .

rf\_fit

## Random Forest   
##   
## 297 samples  
## 11 predictor  
## 3 classes: 'Insomnia', 'None', 'Sleep.Apnea'   
##   
## No pre-processing  
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 268, 266, 268, 267, 266, 268, ...   
## Resampling results across tuning parameters:  
##   
## mtry splitrule Accuracy Kappa   
## 2 gini 0.9090434 0.8413850  
## 2 extratrees 0.9156025 0.8528083  
## 3 gini 0.9057100 0.8356049  
## 3 extratrees 0.9123767 0.8471099  
## 4 gini 0.9022618 0.8296133  
## 4 extratrees 0.9057100 0.8356049  
## 5 gini 0.9057100 0.8356049  
## 5 extratrees 0.9089359 0.8413033  
## 6 gini 0.9054876 0.8353118  
## 6 extratrees 0.9057100 0.8356049  
## 7 gini 0.9022618 0.8296133  
## 7 extratrees 0.9022618 0.8296133  
## 8 gini 0.9022618 0.8296133  
## 8 extratrees 0.9057100 0.8356049  
## 9 gini 0.9054876 0.8353118  
## 9 extratrees 0.9022618 0.8296133  
## 10 gini 0.9054876 0.8353118  
## 10 extratrees 0.9055951 0.8353934  
## 11 gini 0.9022618 0.8296133  
## 11 extratrees 0.9054876 0.8353118  
## 12 gini 0.9055951 0.8353934  
## 12 extratrees 0.9022618 0.8296133  
##   
## Tuning parameter 'min.node.size' was held constant at a value of 1  
## Accuracy was used to select the optimal model using the largest value.  
## The final values used for the model were mtry = 2, splitrule = extratrees  
## and min.node.size = 1.

plot(rf\_fit)



Step 5: Build the model with the best mtry

set.seed(111)  
rf\_model\_2 <- randomForest(Sleep.Disorder~., data = train\_data, mtry = rf\_fit$bestTune$mtry, splitrule = rf\_fit$bestTune$splitrule, importance = TRUE, ntree = 1000)  
rf\_model\_2

##   
## Call:  
## randomForest(formula = Sleep.Disorder ~ ., data = train\_data, mtry = rf\_fit$bestTune$mtry, splitrule = rf\_fit$bestTune$splitrule, importance = TRUE, ntree = 1000)   
## Type of random forest: classification  
## Number of trees: 1000  
## No. of variables tried at each split: 2  
##   
## OOB estimate of error rate: 8.42%  
## Confusion matrix:  
## Insomnia None Sleep.Apnea class.error  
## Insomnia 48 5 6 0.18644068  
## None 1 166 5 0.03488372  
## Sleep.Apnea 5 3 58 0.12121212

Step 6: Save model

save(rf\_model\_2, file = "RFModel.rda")

# Validation

Test validation of the Decision Tree

predict\_decision\_tree <- predict(decision\_tree, newdata = test\_data, type="class")  
confusionMatrix(test\_data$Sleep.Disorder, predict\_decision\_tree)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Insomnia None Sleep.Apnea  
## Insomnia 16 2 0  
## None 4 43 0  
## Sleep.Apnea 1 2 9  
##   
## Overall Statistics  
##   
## Accuracy : 0.8831   
## 95% CI : (0.7897, 0.9451)  
## No Information Rate : 0.6104   
## P-Value [Acc > NIR] : 1.098e-07   
##   
## Kappa : 0.7857   
##   
## Mcnemar's Test P-Value : 0.2998   
##   
## Statistics by Class:  
##   
## Class: Insomnia Class: None Class: Sleep.Apnea  
## Sensitivity 0.7619 0.9149 1.0000  
## Specificity 0.9643 0.8667 0.9559  
## Pos Pred Value 0.8889 0.9149 0.7500  
## Neg Pred Value 0.9153 0.8667 1.0000  
## Prevalence 0.2727 0.6104 0.1169  
## Detection Rate 0.2078 0.5584 0.1169  
## Detection Prevalence 0.2338 0.6104 0.1558  
## Balanced Accuracy 0.8631 0.8908 0.9779

Test validation of the Random Forest

predict\_rf <- predict(rf\_model\_2, newdata = test\_data, type = "class")  
confusionMatrix(test\_data$Sleep.Disorder,predict\_rf)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Insomnia None Sleep.Apnea  
## Insomnia 16 2 0  
## None 3 44 0  
## Sleep.Apnea 1 2 9  
##   
## Overall Statistics  
##   
## Accuracy : 0.8961   
## 95% CI : (0.8055, 0.9541)  
## No Information Rate : 0.6234   
## P-Value [Acc > NIR] : 7.181e-08   
##   
## Kappa : 0.8078   
##   
## Mcnemar's Test P-Value : 0.3618   
##   
## Statistics by Class:  
##   
## Class: Insomnia Class: None Class: Sleep.Apnea  
## Sensitivity 0.8000 0.9167 1.0000  
## Specificity 0.9649 0.8966 0.9559  
## Pos Pred Value 0.8889 0.9362 0.7500  
## Neg Pred Value 0.9322 0.8667 1.0000  
## Prevalence 0.2597 0.6234 0.1169  
## Detection Rate 0.2078 0.5714 0.1169  
## Detection Prevalence 0.2338 0.6104 0.1558  
## Balanced Accuracy 0.8825 0.9066 0.9779