**ID3**

**IsPure** <- function(data) {

length(unique(data[,ncol(data)])) == 1 **// purity**

}

**Entropy** <- function( vls ) {

res <- vls/sum(vls) \* log2(vls/sum(vls)) // **entropy**

res[vls == 0] <- 0

-sum(res)

}

**InformationGain** <- function( tble ) {

entropyBefore <- Entropy(colSums(tble)) // **information gain**

s <- rowSums(tble)

entropyAfter <- sum (s / sum(s) \* apply(tble, MARGIN = 1, FUN = Entropy ))

informationGain <- entropyBefore - entropyAfter

return (informationGain)

}

TrainID3 <- function(node, data) {

node$obsCount <- nrow(data)

#if the data-set is pure (e.g. all toxic), then

if (IsPure(data)) {

#construct a leaf having the name of the pure feature (e.g. 'toxic')

child <- node$AddChild(unique(data[,ncol(data)]))

node$feature <- tail(names(data), 1)

child$obsCount <- nrow(data)

child$feature <- ''

} else {

#calculate the information gain

ig <- sapply(colnames(data)[-ncol(data)],

function(x) InformationGain(

table(data[,x], data[,ncol(data)])

)

)

#chose the feature with the highest information gain (e.g. 'color')

#if more than one feature have the same information gain, then take

#the first one

feature <- names(which.max(ig))

node$feature <- feature

#take the subset of the data-set having that feature value

childObs <- split(data[ ,names(data) != feature, drop = FALSE],

data[ ,feature],

drop = TRUE)

for(i in 1:length(childObs)) {

#construct a child having the name of that feature value (e.g. 'red')

child <- node$AddChild(names(childObs)[i])

TrainID3(child, childObs[[i]])

#call the algorithm recursively on the child and the subset } }}

library(data.tree)

data(mushroom)

mushroom

head(mushroom)

tree <- Node$new("mushroom")

TrainID3(tree, mushroom)

print(tree, "feature", "obsCount")

Predict <- function(tree, features) {

if (tree$children[[1]]$isLeaf) return (tree$children[[1]]$name)

child <- tree$children[[features[[tree$feature]]]]

return ( Predict(child, features))

}

Predict(tree, c(color = 'red', size = 'large', points = 'yes'))

plot(tree)