Exp 3 - ID3 Algorithm

May 21, 2022

ID3 or **Iterative Dichotomiser 3** is a method to separate features into two or more groups iteratively (repeatedly) at each stage. **It is a top-down greedy way of building a decision tree**. In simple terms, the top-down strategy indicates that we build the tree from top to down, whereas the greedy approach means that we choose the best available feature at a time to generate a node at each iteration. **ID3 is often only used for classification tasks involving nominal features.**

1 Experimental Description

1.1 Objective

To create a decision tree based on the ID3 algorithm using an appropriate dataset.

1.2 Algorithm

1. At each phase, the ID3 algorithm splits features into two or more groups iteratively.
2. It chooses the best feature with the highest Information Gain to generate a node at each iteration. Information gain can be calculated using Entropy.
3. **Entropy** is calculated as, $Entropy(S) = -\sum_{1}^{n} p_i * log_2(p_i)$, where S denotes the dataset in use, n denotes the total number of classes in the target column and p_i is the probability of the occurrence of class 'i' of the target column.
4. **Information Gain** is calculated as, $IG(S, A) = Entropy(S) - \sum((|S_v|/|S|) * Entropy(S_v))$, where S_v denotes the set of rows in S for which the feature column A has value v and $|S|$ denotes the number of rows of S.

1.3 Procedure

- Import the dataset into a variable.
- Calculate each feature's Information Gain.
- Divide the dataset into subsets using the feature with the highest Information Gain, given that not all rows belong to the same class.
- Make a decision tree node depending on the feature that provides the most information.
- Make the current node a leaf node if all rows belong to a single class.
- Repeat for the rest of the features until the decision tree is devoid of leaf nodes or we've exhausted all of them.

1.4 System Requirements

Windows/Linux OS/Mac OS with R. Required package is **data.tree**.

1.5 Dataset Summary

For this project, we used a dataset of **tennis playing provisions** based on different weather conditions. The features of this dataset are different weather conditions.

2 Code and Output

```
rm(list = ls())
version$version.string
```

```
## [1] "R version 4.1.2 (2021-11-01)"
```

```
# Install the "data.tree" package by uncommenting and running the following command
#install.packages("data.tree")

library(data.tree)
```

```
## Warning: package 'data.tree' was built under R version 4.1.3
```

```
# Function for checking for more than one unique decisions
PurityCheck <- function(data)
{
  length(unique(data[,ncol(data)])) == 1
}
```

```
# Function for calculating the entropy
calculate_entropy <- function( v )
{
  out <- v/sum(v) * log2(v/sum(v)) # Calculating entropy values for each vector
  out[v == 0] <- 0 # Assigning zero to the entropy vectors having -Inf values
  -sum(out)
}
```

```
# Function for calculating Information Gain (IG)
calculate_ig <- function( table ) {
  table <- as.data.frame.matrix(table)
  ent_before <- calculate_entropy(colSums(table)) # Calculating Entropy before IG
  s <- rowSums(table)
  ent_after <- sum (s / sum(s) * apply(table, MARGIN = 1, FUN = calculate_entropy )) # Calculating Entr
  info_gain <- ent_before - ent_after  # Calculating IG

  return (info_gain)
}
```

```
# Function for creating the decision tree
tree_id3 <- function(node, data) {

  if (PurityCheck(data)) { # Creating tree with one unique decision
    child <- node$AddChild(unique(data[,ncol(data)])) # Adding the only decision as a child
    node$feature <- tail(names(data), 1) # Adding the decision parameter as node feature
    child$obs_Count <- nrow(data)
    child$feature <- ''
  } else { # Creating tree with two or more decisions

    # Calculating IG for all the columns
    info_gain <- sapply(colnames(data)[-ncol(data)],
```

```
                function(x) calculate_ig(
                  table(data[,x], data[,ncol(data)])
                )
    )
    # Storing the column name with max IG as feature
    feature <- names(info_gain)[info_gain == max(info_gain)][1]
    node$feature <- feature

    # Setting the other nodes as child nodes
    children_obs <- split(data[,!(names(data) %in% feature)], data[,feature], drop = TRUE)

    # Adopting a recursive approach for the entire tree
    for(i in 1:length(children_obs)) {
      child <- node$AddChild(names(children_obs)[i])
      tree_id3(child, children_obs[[i]])
    }
  }
}
```

```
# Importing data
PlayTennis <- read.csv("PlayTennis.csv")

# Creating the first Node
tree <- Node$new("PlayTennis")

# Creating the tree
tree_id3(tree,PlayTennis)
print(tree)
```

```
##          levelName
## 1  PlayTennis
## 2   ¦--overcast
## 3   ¦   °--yes
## 4   ¦--rainy
## 5   ¦   °--no
## 6   °--sunny
## 7       ¦--high
## 8       ¦   °--no
## 9       °--normal
## 10          °--yes
```

From the output, we can interpret the following:

- This algorithm constructed a decision tree using Information Gain and Entropy calculations. Since the feature **outlook** has the highest Information Gain, it was used to create the root node. The root node contained three branches **overcast**, **rainy** and **sunny**.
- Based on the generated decision tree, predictions can be made regarding a person's willingness to play tennis depending upon the given weather conditions.

3 Conclusion

ID3 Algorithm got implemented successfully over the given dataset.