Birla Institute of Technology and Science

Foundations of Data Science (CS F320)

SECOND SEMESTER 2021-2022

Lab Sheet #7: Decision Tree

Decision tree is a supervised learning algorithm which is used for both classification and regression. As the name goes, it uses a tree-like model of decisions, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label.

It consists of a set of rules for dividing large heterogeneous population into smaller groups with respect to the target label, the algorithm used for this tree construction is recursive partitioning. It follows top- down approach.

Decision tree Terminology:

Root node: Represents entire population (entire training data - impure)

Splitting: Process of dividing the population into child nodes (2 or more). Purpose of splitting is to reduce impurity in the child nodes and to eventually get pure leaf nodes.

Leaf Nodes: Pure node containing data from only 1 class Intermediate Nodes: Nodes that are neither root nor leaf nodes (any node between root & leaf nodes)

Pruning: Reducing the size of the tree (post/pre pruning)

The following are some of the popular algorithms to build a decision tree:

1. CART :- uses Gini Index (Classification) as metric.

2. ID3:- uses Entropy function and Information gain as metrics.

3. C4.5

4. Hunt's algorithm

In order to grow decision tree please load the Carseats dataset using the rpart package in R.

install the ISLR and rpart packages in your R Studio environment first.

```
rm(list=ls())
library(ISLR)
library(tree)
```

```
library(ISLR)
library(rpart)
```

#Let's first load the Carseats dataframe

```
attach(Carseats)
head(Carseats)
```

Run the View() command on the Carseats data to see what the data set looks like.

```
View(Carseats)
```

 # Our goal will be to classify whether Carseat sales in a store are high or not.

Let's take a look at the histogram of car sales:

```
hist(Carseats$Sales)

# Observe that Sales is a quantitative variable.
```

You want to demonstrate it using trees with a binary response. To do so, you turn Sales into a binary variable, which will be called High. If the sales is less than 8, it will be not high. Otherwise, it will be high. Then you can put that new variable High back into the dataframe

```
#creat a categorical variable based on sales

High=ifelse(Sales>=8,"Yes","No")
Carseats=data.frame(Carseats, High)
head(Carseats)
```

To understand the proportion of stores in High variable use the prop command and view it as table

```
prop.table(table(Carseats$High))
##
##   No  Yes
## 0.59 0.41
We see that 41% of locations had high carseat sales.
```

Use the tree command to fit a decision tree to every other variable in the Carseats data other than Sales.

```
Now let's fill a model using decision trees. Of course, you can't have the Sales variable
here because your response variable High was created from Sales. Thus, let's exclude it
and fit the tree.

#remove first varilabe
Carseats=Carseats[,-1]
names(Carseats)

#split data into training and test set
set.seed(2)
train=sample(1:nrow(Carseats),nrow(Carseats)/2)
test=-train
test

training_data=Carseats[train,]
testing_data=Carseats[test, ]
testing_High=High[test]

#fit the tree model using training data
tree_model=rpart(High~.,training_data)
tree_model
```

#Run the plot and text commands on your tree object.

```
plot(tree_model)
text(tree_model, pretty = 0)
```

#Run summary on your tree object.

```
summary(tree_model)
```

```
##check how the model is doing using the test data

tree_pred=predict(tree_model, testing_data, type="class")
mean(tree_pred!=testing_High)
```

To make it more visual, let's plot the tree as well, then annotate it using the handy text function:

```
# Examine the complexity plot
printcp(tree_model)
plotcp(tree_model)
```

There are so many variables, making it very complicated to look at the tree. At least, you can see that at each of the terminal nodes, they're labeled Yes or No. At each splitting node, the variables and the value of the splitting choice are shown (for example, Price < 92.5 or Advertising < 13.5). For a detailed summary of the tree, simply print it. It'll be handy if you want to extract details from the tree for other purposes:

```
tree_model
```

Overfitting happens when the learning algorithm continues to develop hypotheses that reduce training set error at the cost of an increased test set error. There are several approaches to avoiding overfitting in building decision trees.

There are two types of pruning: pre-pruning and post-pruning.

1. Pre-pruning

Prepruning is also known as early stopping criteria. As the name suggests, the criteria are set as parameter values while building the rpart model. Below are some of the pre-pruning criteria that can be used. The tree stops growing when it meets any of these pre-pruning criteria, or it discovers the pure classes.

- maxdepth: This parameter is used to set the maximum depth of a tree. Depth is thelength of the longest path from a Root node to a Leaf node. Setting this parameterwill stop growing the tree when the depth is equal the value set for maxdepth.

- minsplit: It is the minimum number of records that must exist in a node for a split to happen or be attempted. For example, we set minimum records in a split to be 5; then, a node can be further split for achieving purity when the number of records in each split node is more than 5.

```
Preprune_tree_model = rpart(High~., data=training_data, method="class", control =
rpart.control(cp = 0, maxdepth = 8,minsplit = 100))
```

2. Post-pruning

The idea here is to allow the decision tree to grow fully and observe the CP value. Next,

we prune/cut the tree with the optimal CP value as the parameter as shown in below code:

```
Postprune_ tree_model <- prune(tree_model, cp = 0.0026 )
```

Exercises

- Compare the accuracy of decision tree by randomly selecting 70% data for testing and 30% for training using holdout method, random subsampling and 5- fold cross validation techniques.
- How do you deal with continuous attribute in terms of splitting?
- What are the reasons for overfitting in decision tree?