**BUSINESS DATA MINING**

**(IDS 572)**

**Solutions to Homework 2**

**Group Members**

* Amey Pophali (apopha2@uic.edu)
* Karthik Varanasi (vvaran3@uic.edu)
* Mrinal Dhawan (mdhawa3@uic.edu)

**Problem 1 -**

* **Load the data and check the attributes of the data. How many variables are in this data set?**

**Ans.** We can get the values by **dim(hw2)** command.

*Attributes = 9,*

*Instances = 768*

Dim command provides us with the aforementioned results.

Note: hw2 is the name of the dataset.

* **Choose the first 80% of the data for training and the remaining 20% data for testing.**

**Ans.** This can be done by using following R commands.

***set.seed(123);***

***hw = sample(2, nrow(hw2), replace = T, prob = c(0.8, 0.2))***

***Trainhw = hw2[hw == 1,]***

***Testhw = hw2[hw == 2,]***

Here, ‘Trainhw’ represents training data and ‘Testhw’ represents test data.

***dim(Testhw)***

[1] 145 9

***dim(Trainhw)***

[1] 623 9

The dim command shows the attributes and columns in the training and testing data. The fist value in the output is the number of instances and second is the number of variables.

* **Use “rpart" function to create a tree using the training data. What is the accuracy of your model based on training data?**

The tree has been created using this command –

**hw2\_Rpart = rpart(Class.variable ~ ., data = Trainhw, parms = list(split = "gini") )**

Printing the plotting the tree using the commands –

**print(hw2\_Rpart)**

Output attached -



Accuracy of the data –

**predict.trainData<-predict(hw2\_Rpart, Trainhw[,1:8], type="class")**

**table(predict.trainData, Trainhw $class)**

The prediction is as follows-

predict.testData 0 1

0 64 12

1 28 41

In the output, the 1st index (i.e. [0,0] index of the matrix signifies the values that are actually ‘0’ in the actual data and have been predicted correctly as ‘0’ by our decision tree. Similarly, [1,1] provides the accuracy of the data point which have been correctly predicted as one. Hence the accuracy of the model is –

Accuracy = [ (340+177)/(340+177+68+38) ] \* 100

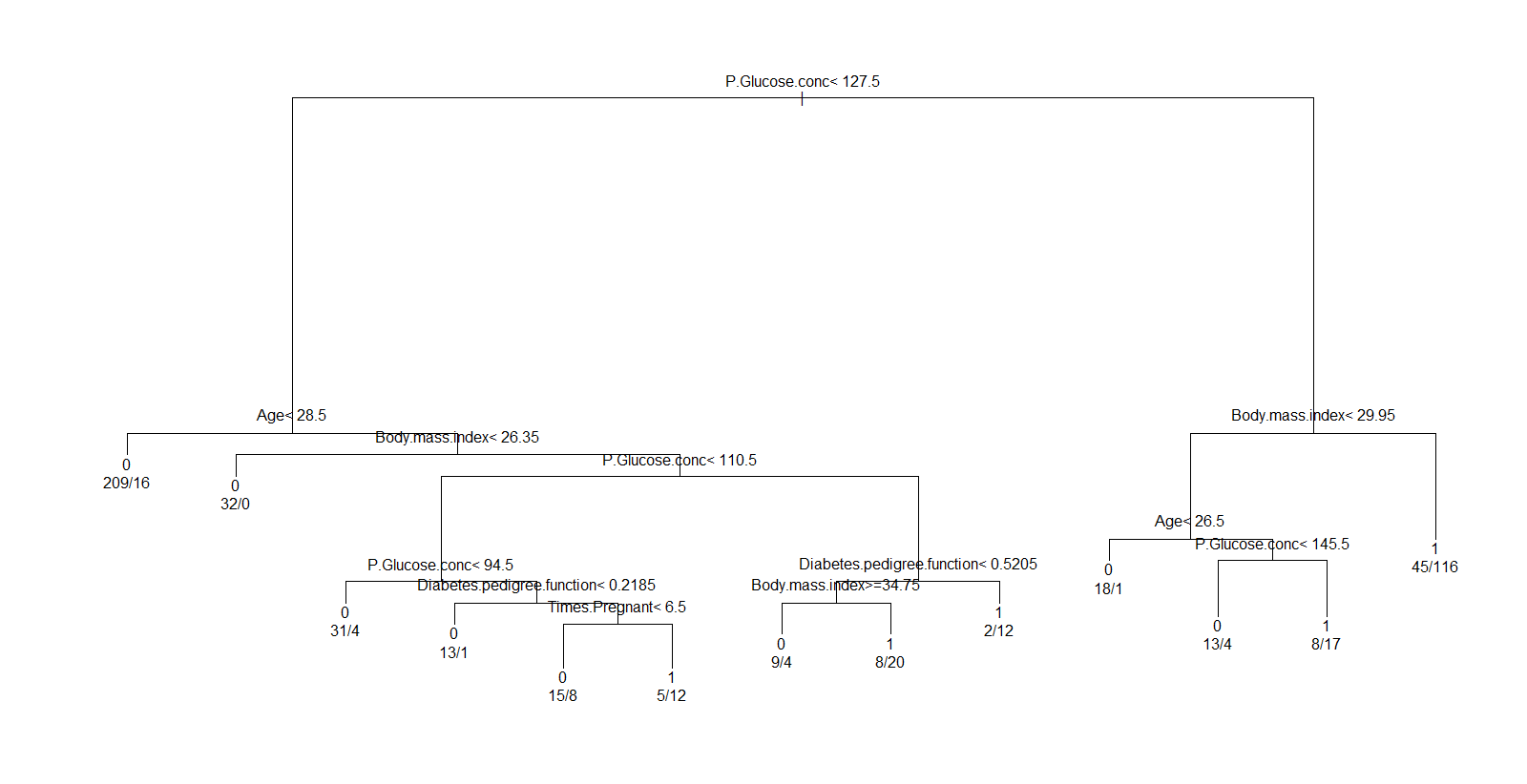
= 517/623 \* 100 = 82%

* **Plot your decision tree. How many leaves are in your tree? Are these leaves pure?**

Plotting the tree with the commands.

**plot(hw2\_Rpart)**

**text(hw2\_Rpart, use.n = T, xpd= T)** #To give label to the tree



The tree has 13 leaves. Only one leaf in the tree is pure.

To select the tree with minimum error we use the code. This code gives us the pruned tree.

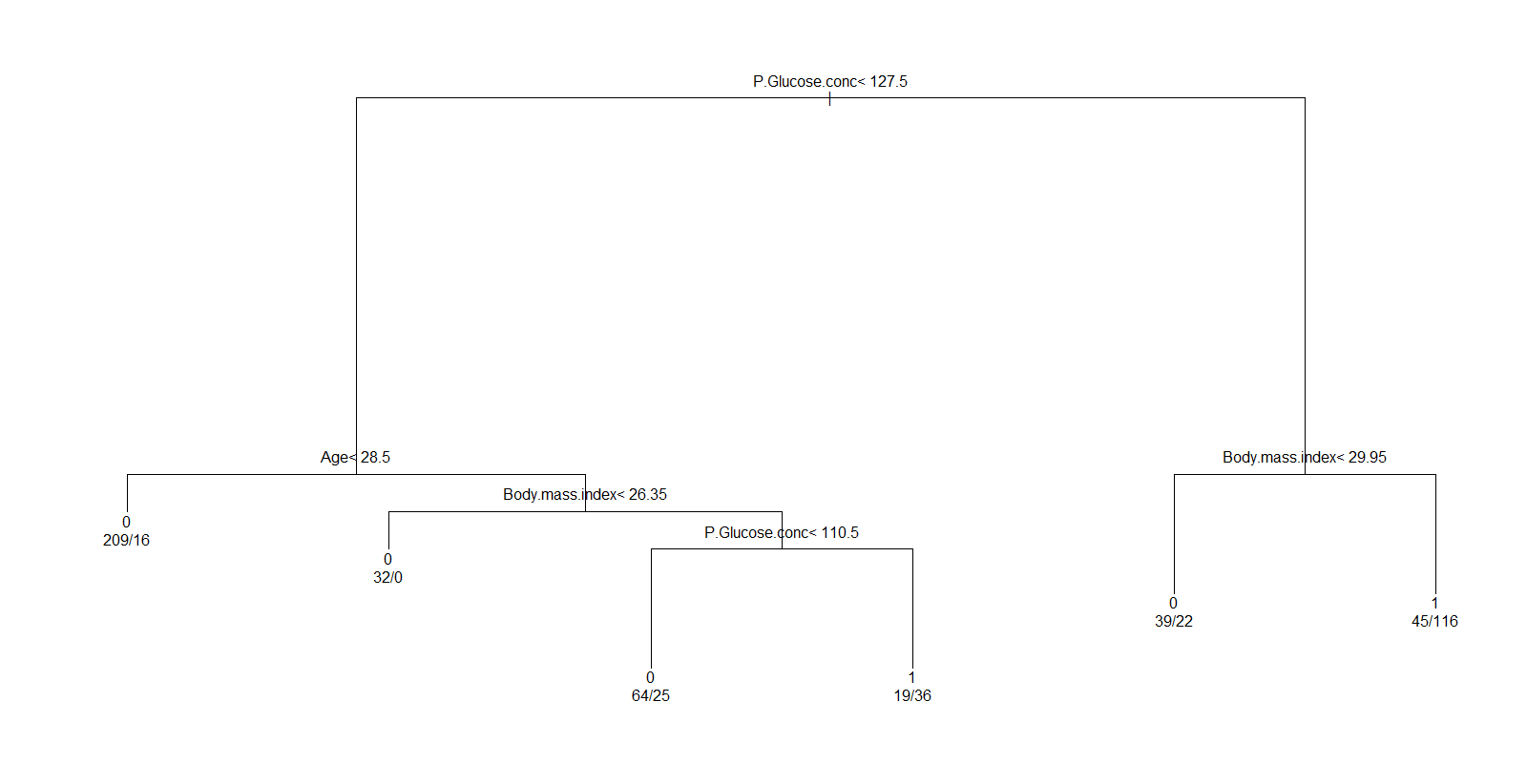
**opt = which.min(hw2\_Rpart$cptable[,"xerror"])**

**cp = hw2\_Rpart$cptable[opt, "CP"]**

**hw2\_pruning = prune(hw2\_Rpart, cp = cp)**

**plot(hw2\_pruning)**

**text(hw2\_pruning, use.n = T, xpd= T)**



* **Provide two strongest If-Then rules from this decision tree. Please explain why these rules are chosen.**

**Two strongest if-else rules for the tree**

1. If ( (Glucose.conc <= 127.5) & (Age > 28.5) &(Body.mass.index<26.35) )

then

Class variable = 0

1. If ( (Glucose.conc <= 127) & (Age <28) &)

Then

Class variable = 0

The if-else rule has been chosen by starting at the root node and going downwards towards the best pure node available in the tree. The second rule has been chosen by moving towards the second best pure node in the tree.

* **Apply the decision tree on test data and report your prediction. What is the accuracy of your model in the test data?**

Applying the tree on the test data using the following commands –

Accuracy of the data –

***predict.test<-predict(hw2\_Rpart, Testhw[,1:8], type="class")***

***P <- table(predict.testData, Testhw$Class.variable)***

The prediction is as follows-

predict.trainData 0 1

0 340 38

1 68 177

In the output, the 1st index (i.e. [0,0] index of the matrix signifies the values that are actually ‘0’ in the actual data and have been predicted correctly as ‘0’ by our decision tree. Similarly [1,1] provides the accuracy of the data point which have been correctly predicted as one. Hence the accuracy of the model is –

accuracy<-sum(diag(P))/sum(P)

accuracy

[1] 0.7241379

Accuracy of the model = 72%

* **Do parts (c), (e), and (f) for a “ctree" function as well. Are there any significant differences between these two decision trees?**

**Ans.**

Tree could be created and plotted using the following commands -

***hw2\_tree = ctree(Class.variable ~ ., data = Trainhw)***

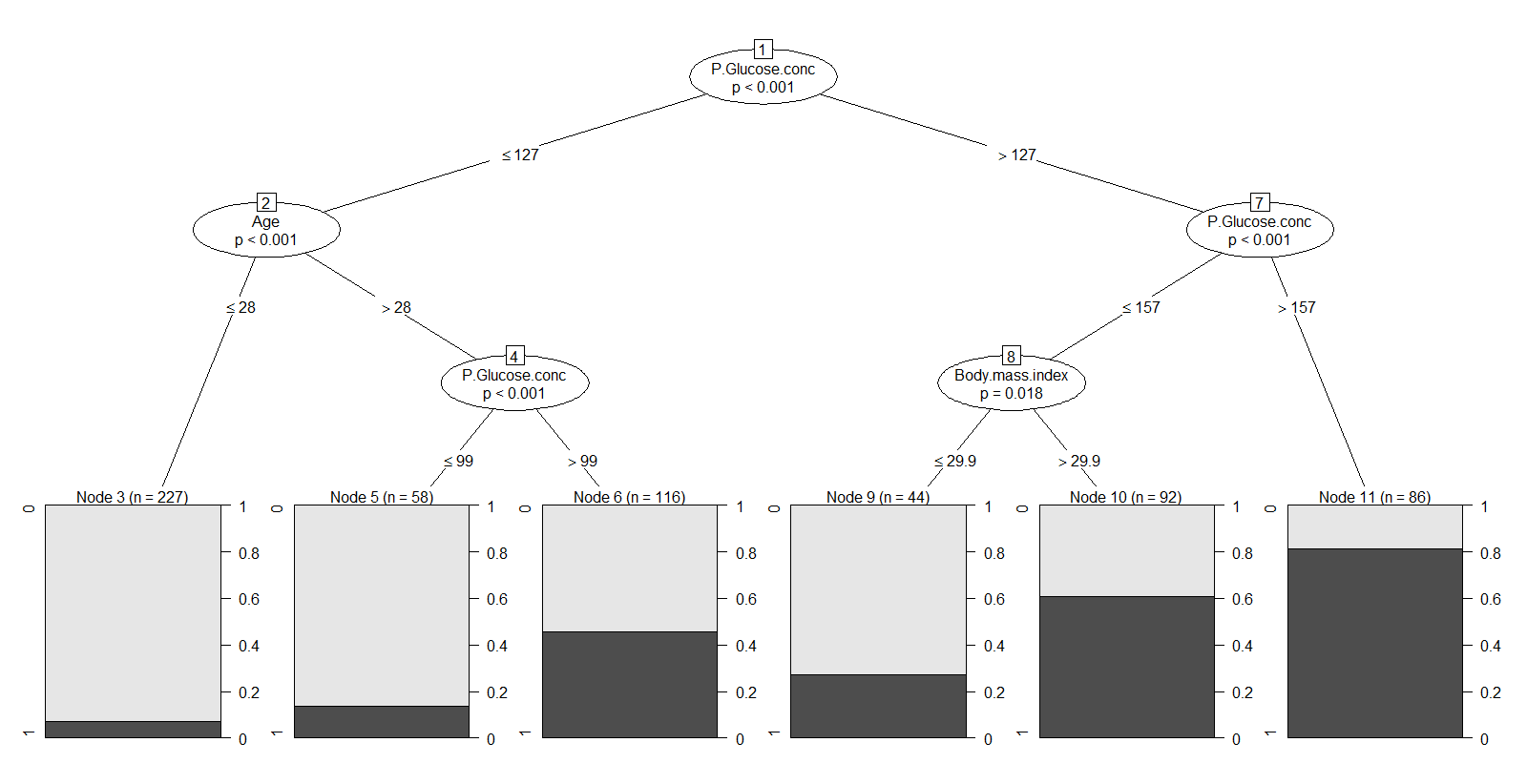
***print(hw2\_tree)***

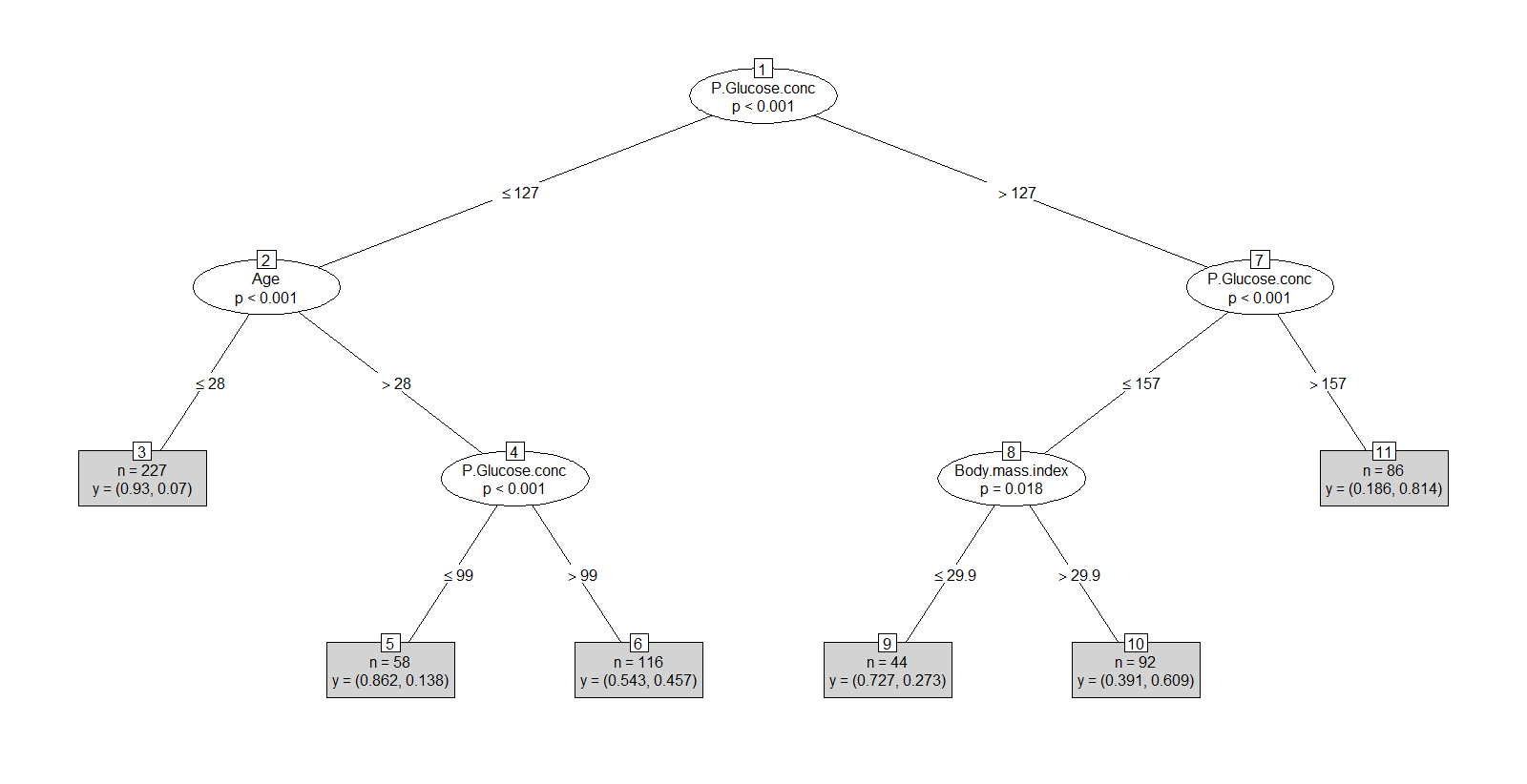
***plot(hw2\_tree)***

Printing the tree. Result attached.



Plotting the Ctree –



Simplified version of the Ctree –

**Number of Leaves in the tree** –

There are 6 leaves in the tree. 1 leaf Node 3 and node 3 is also almost pure.

**Accuracy**

We can get the prediction accuracy of the training and test data using the following command –

***Train\_Accuracy = table(predict(hw2\_tree), Trainhw$Class.variable)***

***Test\_Accuracy = table(predict(hw2\_tree, newdata = Testhw), Testhw$Class.variable)***

Results –

On training data

0 1

0 356 89

1 52 126

Get the accuracy by using the command –

***accuracy<-sum(diag(Train\_Accuracy))/sum(Train\_Accuracy)***

Output - 0.77

Hence accuracy of the model is 77%

On test Data

0 1

0 77 18

1 15 35

***accuracy<-sum(diag(Test\_Accuracy))/sum(Test\_Accuracy)***

Output - 0.77

Hence accuracy of the model is 77%

**Two strongest if-else rules for the tree**

1. If ( (Glucose.conc <= 127) & (Age <=28)) )

then

Class variable = 0

1. If ( (Glucose.conc <= 127) & (Age <=28) & (Glucose.conc <= 99))

Then

Class variable = 0

We start at the root node to initiate the making of the if-else rules. The strongest if-else rule would be the one that moves downwards and ends at a best available pure leaf. Hence the 1st if else rule has been formed by going further downwards toward the best pure leaf of the tree. 2nd rule has been formed by moving towards the second best pure leaf in the tree.

The consolidated R file used for the calculating and plotting the aforementioned matrices and graphs is attached here for your reference. The file also consists of the commands that have been used to filters outliers from the data.



**Problem 2 –**

**Please answer this question without using software. Consider the dataset in the table below for classifying whether a type of food is appealing or not.**

* **What is the information gain associated with choosing Temperature as root? What about Taste and Size? Brief write the steps of your calculations.**
* **Draw a full decision tree for the dataset by choosing appropriate features for splitting. Justify**

**each split.**

****

**Ans.** Total Observations = 10

Information at the beginning of the split –

Information(beginning) = -p1log(p1) – p2log(p2)

= -(1/2) \* log(1/2) – (1/2) \* log(1/2)

= 1

Splitting on Temperature –

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | **Temperature** | | |  |  |
|  |  |  |  |
|  | Hot |  |  |  | Cold |  |
|  |  |  |  |  |  |  |
| **Taste** | **Size** | **Appealing** |  | **Taste** | **Size** | **Appealing** |
| Salty | Small | No |  | Sweet | Large | No |
| Sour | Small | Yes |  | Sweet | Large | No |
| Salty | Large | No |  | Sour | Small | Yes |
| Sour | Large | Yes |  | Sweet | Small | Yes |
| Salty | Large | No |  | Sweet | Small | Yes |

2Y/3N 3Y/2N

Entropy(Hot) = -p1log(p1) – p2log(p2)

= -3/5\*log (3/5) – 2/5 log (3/5)

= 0.95

Entropy(Cold) = -3/5 log2(3/5)-2/5 log2(2/5)

= 0.972

Total Entropy after(Temperature)= (5/10) (0.972) + (5/10) (0.972) = 0.972

Gain(Temp) = 1-0.972 = 0.028

Splitting on Size –

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | **Size** | | |  |  |
|  |  |  |  |
|  | Small |  |  |  | Large |  |
|  |  |  |  |  |  |  |
| **Taste** | **Temperature** | **Appealing** |  | **Taste** | **Temperature** | **Appealing** |
| Salty | Hot | No |  | Sweet | Cold | No |
| Sour | Cold | Yes |  | Sweet | Cold | No |
| Sour | Hot | Yes |  | Salty | Hot | No |
| Sweet | Cold | Yes |  | Sour | Hot | Yes |
| Sweet | Cold | Yes |  | Salty | Hot | No |

4Y/1N 1Y/4N

Entropy (Size) = -5/10 log2(5/10)-5/10 log2(5/10)

= 1

Entropy(Small) = -4/5 log2(4/5)-1/5 log2(1/5)

= 0.72

Entropy(Large) = -4/5 log2(4/5)-1/5 log2(1/5)

= 0.72

Entropyafter(Size) = (5/10) (0.72) + (5/10) (0.72)

= 0.72

Information Gain(Size)=1 - 0.72 = **0.028**

Splitting on Taste –

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | **Taste** | | |  |  |  |
|  | Sour |  |  | Salty |  |  | Sweet |  |
|  |  |  |  |  |  |  |  |  |
| **Size** | **Temp** | **Appealing** | **Size** | **Temp** | **Appealing** | **Size** | **Temp** | **Appealing** |
| Small | Cold | Yes | Small | Hot | No | Large | Cold | No |
| Small | Hot | Yes | Large | Hot | No | Large | Cold | No |
| Large | Hot | Yes | Large | Hot | No | Small | Cold | Yes |
|  |  |  |  |  |  | Small | Cold | Yes |

3Y/0N 0Y/3N 2Y/2N

Entropy (Taste) = -5/10 log2(5/10)-5/10 log2(5/10) = 1

Entropy(Salty) = 0

Entropy(Sour) = 0

Entropy(Sweet) = -2/4 log2(2/4)-2/4 log2(2/4)

= 1

Entropyafter(Taste) = (3/10) (0) + (3/10) (0) + (4/10) (1)

= 0.4

Info Gain(Taste) =1 - 0.4

= 0.6

Since we have highest gain on splitting on taste we take this as the root node and split further on the impure data i.e. Sweet –

|  |  |  |  |
| --- | --- | --- | --- |
| Size | | | |
| Large | | Small | |
| Temp | Appealing | Temp | Appealing |
| Cold | No | Cold | Yes |
| Cold | No | Cold | Yes |

0Y/2N 2Y/0N

Entropy(large)= 0

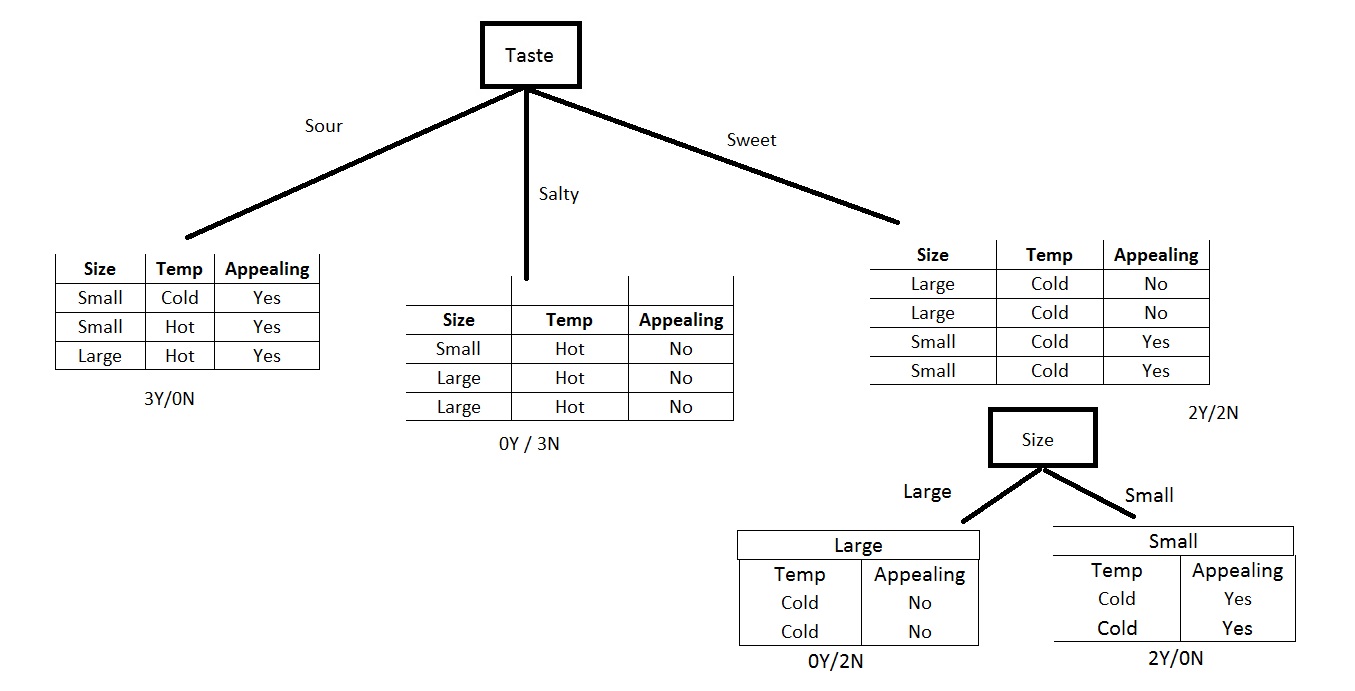
Entropy(small)= 0

Entropyafter(Size) = 2/4(0) +2/4(0)

=0

Gain(Size)=1 – 0 = **1**

**Decision tree –**



**Problem 3 –**

* **Consider two decision trees that always yield the same class label, given the same test sample. Do both the trees need to have the same number of nodes? You can justify your answer with an example.**

If two decision trees that always yield the same class label for the given set of test sample, the trees do not need to have same number of nodes. This is because decision will be the same for the for the given data set but number of attributes of the tree to reach that particular decision might be different.

* **Describe the purpose of separating the data into training and testing data.**

We separate the data into training and testing to avoid under fitting and overfitting of the models. The development of predictive model is completed using the training set of the data. Test data provides us a way to test and cross validate the model. Testing stage of the model is necessary as it cross validates the model by providing the efficiency of the model and its ability to perform on the future data.

* **Which problem do we try to address when using pruning? How do we do that? Please explain.**

Pruning is a technique of reducing the size of a decision tree either by limiting the splits of decision tree or by removing sections of the decision tree that provide little or negligible information.

There are two ways to implement pruning.

1. Pre-Pruning

In pre-pruning, a certain criterion is set to stop the expansion of the decision tree. The criterion is set such that it minimizes the impurity in each node.

1. Post-pruning

In this method, the complete decision tree is plotted and branches of the tree are removed from the bottom up to a certain limit. This technique is time consuming but more efficient than pre-pruning because there is a risk that pre-pruning can stop too early.