Data Mining & Knowledge Discovery

Classification

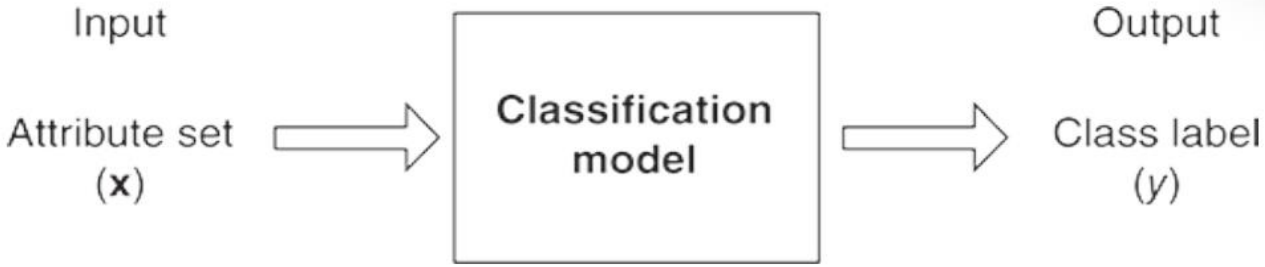
Md. Biplob Hosen

Lecturer, IIT-JU

Email: biplob.hosen@juniv.edu

Classification: Definition

- Classification is the process of identifying and and grouping objects or ideas into predetermined categories.
- Given a collection of records (training set). Each record is characterized by a tuple (x,y), where x is the attribute set and y is the class label.
  - x: attribute, predictor, independent variable, input
  - y: class, response, dependent variable, output
- We can express it mathematically as a target function f that takes as input the attribute set x and produces an output corresponding to the predicted class label. The model is said to classify an instance (x , y) correctly if f (x) = y.
- Classification Task: Learn a model that maps each attribute set x into one of the predefined class labels y.



A schematic illustration of a classification task

Examples of Classification Task

- Spam filtering and tumor identification are examples of binary classification problems, in which each data instance can be categorized into one of two classes.
- If the number of classes is larger than 2, as in the galaxy classification example, then it is called a multiclass classification problem.

Task	Attribute set, x	Class label, y
Categorizing email messages	Features extracted from email message header and content	spam or non-spam
Identifying tumor cells	Features extracted from x-rays or MRI scans	malignant or benign cells
Cataloging galaxies	Features extracted from telescope images	

Example: Vertebrate Classification

- Following dataset can be used for a binary classification task such as mammal classification.
- The attribute set includes characteristics of the vertebrate.

	Name	Warm-blooded	Gives Birth	Aquatic Creature	Aerial Creature	Has Legs	Hibernates	Class
0	human	1	1	0	0	1	0	mammals
1	python	0	0	0	0	0	1	non-mammals
2	salmon	0	0	1	0	0	0	non-mammals
3	whale	1	1	1	0	0	0	mammals
4	frog	0	0	1	0	1	1	non-mammals
5	komodo	0	0	0	0	1	0	non-mammals
6	bat	1	1	0	1	1	1	mammals
7	pigeon	1	0	0	1	1	0	non-mammals
8	cat	1	1	0	0	1	0	mammals
9	leopard shark	0	1	1	0	0	0	non-mammals
10	turtle	0	0	1	0	1	0	non-mammals
11	penguin	1	0	1	0	1	0	non-mammals
12	porcupine	1	1	0	0	1	1	mammals
13	eel	0	0	1	0	0	0	non-mammals
14	salamander	0	0	1	0	1	1	non-mammals

Example: Loan Borrower Classification

- Consider the problem of predicting whether a loan borrower will repay the loan or default on the loan payments.
- The data set used to build the classification model.
- The attribute set includes personal information of the borrower such as marital status and annual income, while the class label indicates whether the borrower had defaulted on the loan payments.

ID	Home Owner	Marital Status	Annual Income	Defaulted?
1	Yes	Single	125000	No
2	No	Married	100000	No
3	No	Single	70000	No
4	Yes	Married	120000	No
5	No	Divorced	95000	Yes
6	No	Single	60000	No
7	Yes	Divorced	220000	No
8	No	Single	85000	Yes
9	No	Married	75000	No
10	No	Single	90000	Yes

Classification Tasks

- A classification model serves two important roles in data mining.
- First, it is used as a **predictive model** to classify previously unlabeled instances.
- A good classification model must provide accurate predictions with a fast response time.
- Second, it serves as a **descriptive model** to identify the characteristics that distinguish instances from different classes.
- This is particularly useful for critical applications, such as medical diagnosis, where it is insufficient to have a model that makes a prediction without justifying how it reaches such a decision.

Vertebrate Name	Body Temperature	Skin Cover	Gives Birth	Aquatic Creature	Aerial Creature	Has Legs	Hibernates	Class Label
gila monster	cold-blooded	scales	no	no	no	yes	yes	?

- In this example, it can be used as a descriptive model to help to determine characteristics that define a vertebrate as a mammal, a reptile, a bird, a fish, or an amphibian.
- For example, the model may identify mammals as warmblooded vertebrates that give birth to their young.

General Framework for Classification

- A classifier model is created using a given set of instances, known as the training set, which contains attribute values as well as class labels for each instance.
- The systematic approach for learning a classification model given a training set is known as a learning algorithm.
- The process of using a learning algorithm to build a classification model from the training data is known as **induction &** the process of applying a classification model on unseen test instances to predict their class labels is known as **deduction**.
- Thus, the process of classification involves two steps: applying a learning algorithm to training data to learn a model, and then applying the model to assign labels to unlabeled instances.

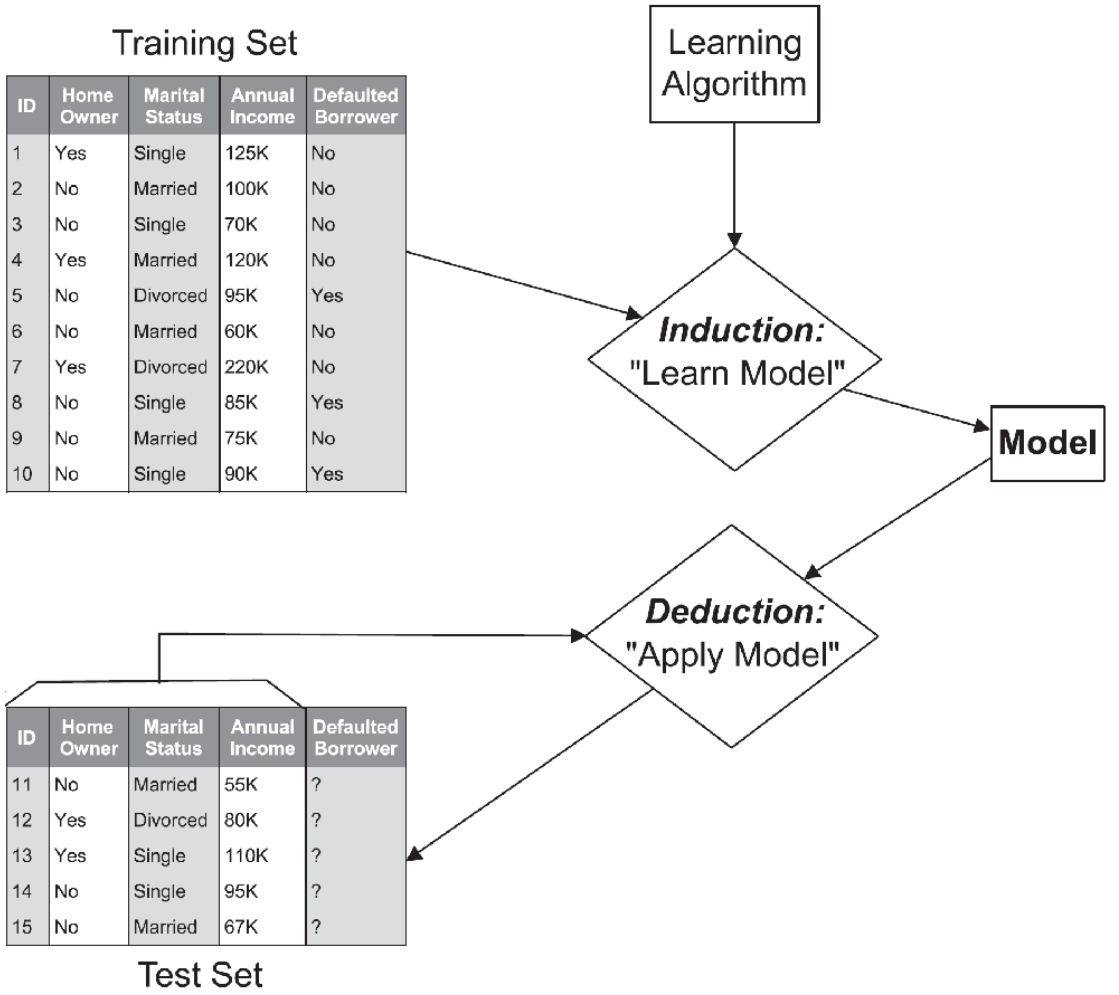


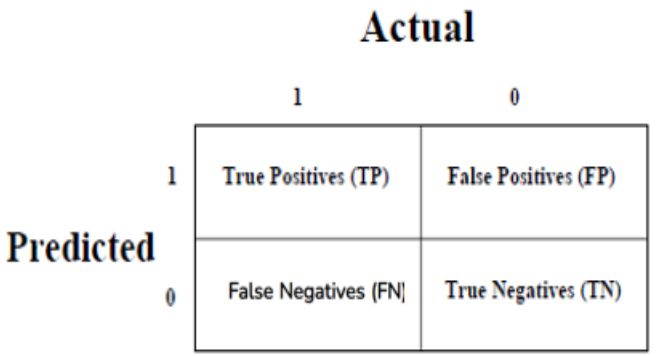
Figure . General framework for building a classification model.

Classification Techniques

- A classification technique refers to a general approach to classification.
- Base Classifiers:
  - Decision Tree based Methods
  - Rule-based Methods
  - Nearest-neighbor
  - Naïve Bayes and Bayesian Belief Networks
  - Support Vector Machines
  - Neural Networks, Deep Neural Networks
- Ensemble Classifiers:
  - Boosting, Bagging, Random Forests

Confusion Matrix

- The performance of a model (classifier) can be evaluated by comparing the predicted labels against the true labels of instances.
- This information can be summarized in a table called a confusion matrix.
- A confusion matrix is nothing but a table with two dimensions.
- `“Actual” and “Predicted” and furthermore, both the dimensions have “True Positives (TP)”, “True Negatives (TN)”, “False Positives (FP)”, “False Negatives (FN)” as shown below –
- TP – It is the case when both actual class & predicted class of data point is 1.
- TN – It is the case when both actual class & predicted class of data point is 0.
- FP – It is the case when actual class of data point is 0 & predicted class of data point is 1.
- FN – It is the case when actual class of data point is 1 & predicted class of data point is 0.



Confusion Matrix

Accuracy:

It may be defined as the number of correct predictions made by our ML model. We can easily calculate it by confusion matrix with the help of following formula-

accuracy =Number of correct predictions/Total number of predictions

=(f10+f01)/(f11+f10+f01+f00)

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

Error Rate:

- Error rate is another related metric, which is defined as follows for binary classification problems:

  Error_rate=Number of wrong predictions/Total number of predictions

  =(f10+f01)/(f11+f10+f01+f00)

Confusion Matrix

Precision:

- Precision, used in document retrievals, may be defined as the number of correct documents returned by our ML model. We can easily calculate it by confusion matrix with the help of following formula −

$$Precision = \frac{TP}{TP + FP}$$

Recall or Sensitivity:

- Recall may be defined as the number of positives returned by our ML model. We can easily calculate it by confusion matrix with the help of following formula −

$$Recall = \frac{TP}{TP + FN}$$

F1 Score:

- The F1 Score is the weighted average of Precision and Recall, this score takes both false positives and false negatives into account.
- Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if we have an uneven class distribution.
- This F1 score is better to use in the natural distribution as it gives a more accurate performance of the result.

$$F1\ Score = 2 * \frac{Precision\ * Recall}{Precision + Recall}$$

Confusion Matrix – Example - 01

- Consider the following confusion matrix-

```
[[ 73  7]
 [ 4 144]]
```

- Here, TP + TN = 73+144 = 217 and TP+FP+FN+TN = 73+7+4+144=228.
  - Accuracy = 217/228 = 0.952.
  - Error_rate = 11/228 = 0.048
  - Precision = 73/80 = 0.915
  - Recall = 73/77 = 0.948

Confusion Matrix – Example - 02

- Let's say you want to predict how many people are infected with a contagious virus in times before they show the symptoms, and isolate them from the healthy population. The two values for our target variable would be Sick and Not Sick.
- Now, you must be wondering why we need a confusion matrix when we have accuracy.
- Let's, there are 947 data points for the negative class and 53 data points for the positive class.
- Assume, The total outcome values by a classification mode are: TP = 30, TN = 930, FP = 30, FN = 10
- So, the accuracy of our model is: 960/1000=0.96
- 96%! Not bad!
- But it gives the wrong idea about the result.

ID	Actual Sick?	Predicted Sick?	Outcome
1	1	1	TP
2	0	0	TN
3	0	0	TN
4	1	1	TP
5	0	0	TN
6	0	0	TN
7	1	0	FN
8	0	1	FP
9	0	0	TN
10	1	0	FN
:	:	:	:
1000	0	0	TN

Confusion Matrix – Example - 02

- Our model is saying, “I can predict sick people 96% of the time”.
- However, it is doing the opposite. It predicts the people who will not get sick with 96% accuracy while the sick are spreading the virus!
- Do you think this is a correct metric for our model, given the seriousness of the issue?
- Shouldn’t we be measuring how many positive cases we can predict correctly to arrest the spread of the contagious virus?
- Or maybe, out of the correct predictions, how many are positive cases to check the reliability of our model?
- This is where we come across the dual concept of Precision and Recall.

$$Precision = \frac{30}{30+30} = 0.5 \qquad Recall = \frac{30}{30+10} = 0.75$$

- 50% percent of the correctly predicted cases turned out to be positive cases.
- Whereas 75% of the positives were successfully predicted by our model.

Confusion Matrix – Example - 03

- Suppose there is a set, S contains information about patient. If the value is 1 the patient has cancer. If the value is 0 then the patient is not diagnosed with cancer.
  - S = {0,0,0,1,0,1,0,1,0,1,0,0,0,1,1} (Actual)
  - S’ = {0,0,1,1,0,1,0,1,0,0,1,1,0,1,1} (Predicted)
- Our Main Task is to detect the cancer patient.
  - Detecting cancer patient correctly (1-1) = True Positive
  - Detecting non-Cancer patient correctly (0-0) = True Negative
  - Cancer Patient Detected as non-Cancer Patient (1-0) = False Negative
  - Non-Cancer Patient Detected as Cancer Patient (0-1) = False Positive

Confusion Matrix – Example - 03

CONFUSION MATRIX

		ACTUAL DATA (15)	
		CANCER (6)	NON-CANCER (9)
PREDICTED DATA (15)	CANCER (8)	5 (TP)	3 (FP)
	NON-CANCER (7)	1 (FN)	6 (TN)

RESULT

$$Accuracy = \frac{5+6}{5+3+1+6} = \frac{11}{15} = 0.7333 = 73.33\%$$

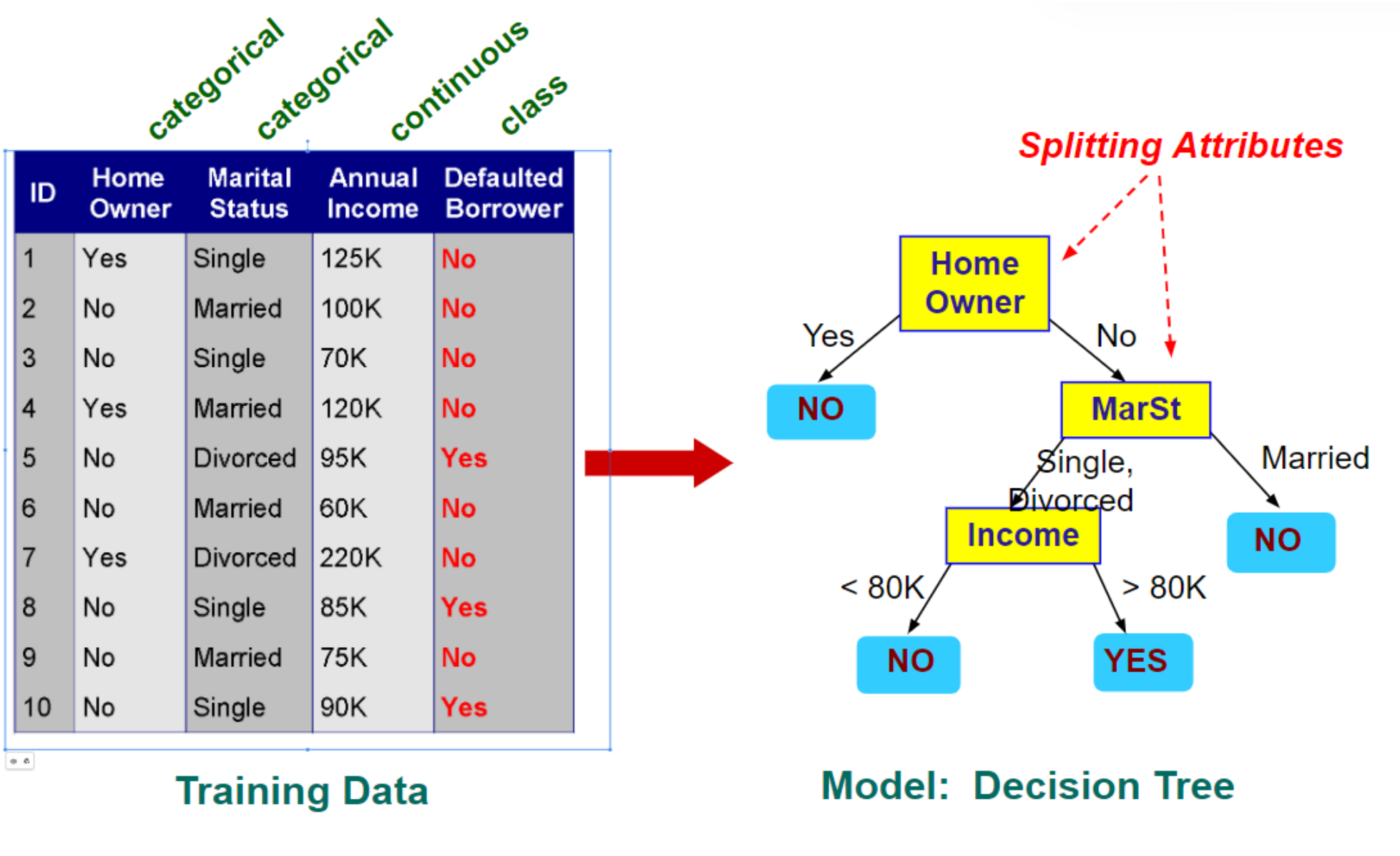
$$Precision = \frac{5}{5+3} = \frac{5}{8} = 0.625 = 62.5\%$$

$$Recall = \frac{5}{5+1} = \frac{5}{6} = 0.8333 = 83.33\%$$

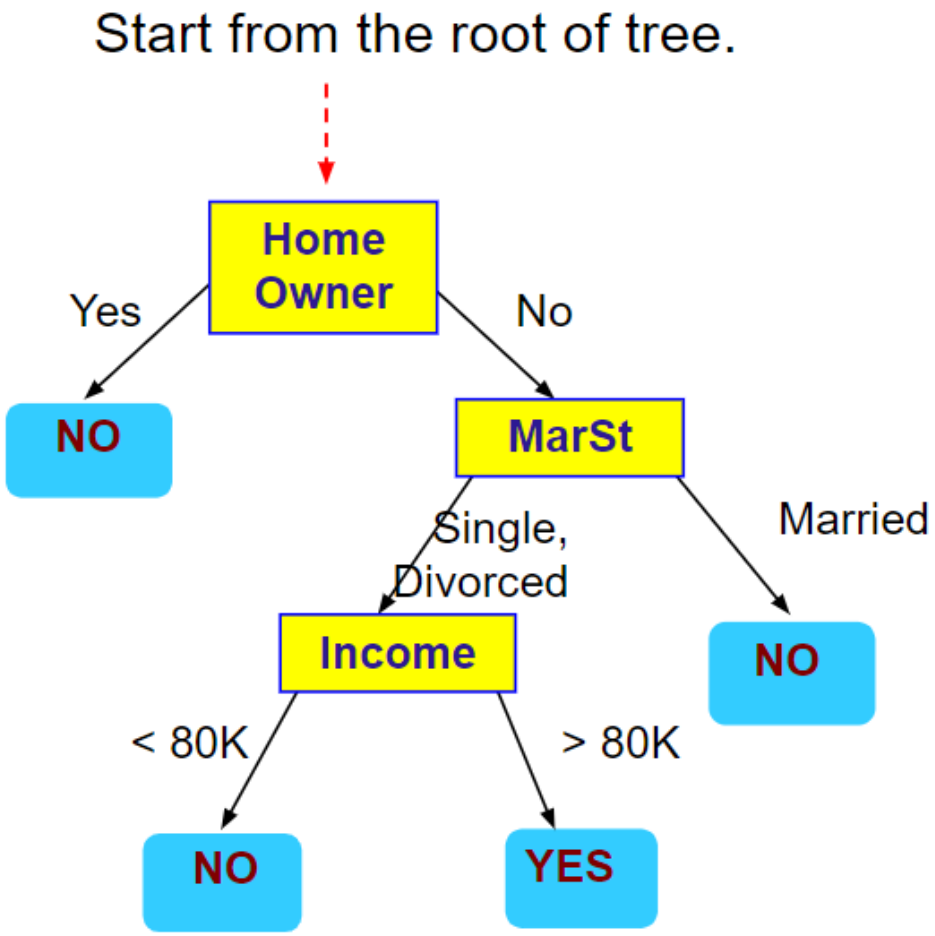
$$F1\ Score = 2 * \frac{0.625 * 0.833}{0.625+0.833} = 0.7142 = 71.42\%$$

Actual Data (S)	Predicted Data (S')	Remark
0	0	TN
0	0	TN
0	1	FP
1	1	TP
0	0	TN
1	1	TP
0	0	TN
1	1	TP
0	0	TN
1	0	FN
0	1	FP
0	1	FP
0	0	TN
1	1	TP
1	1	TP

Example of a Decision Tree



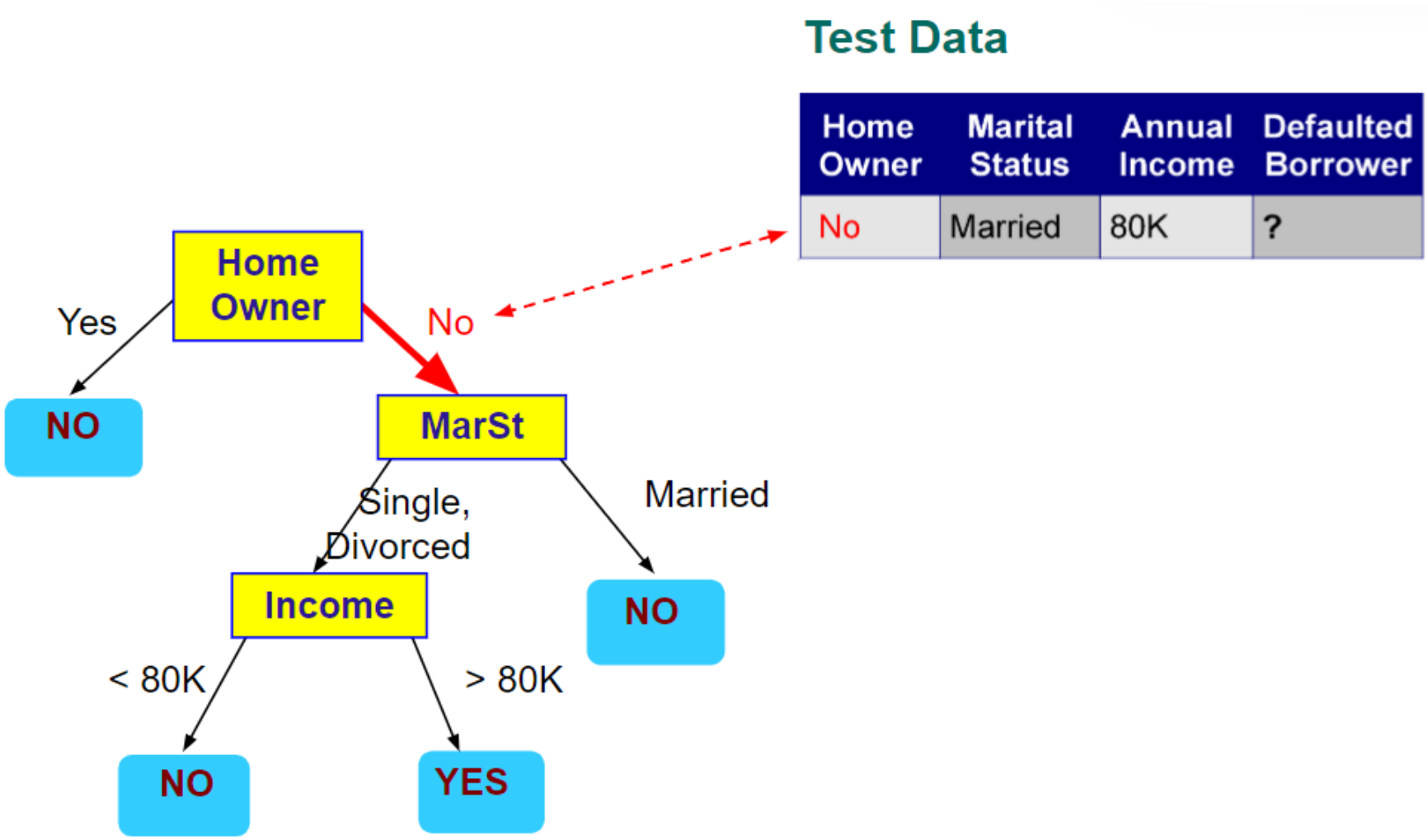
Apply Model to Test Data



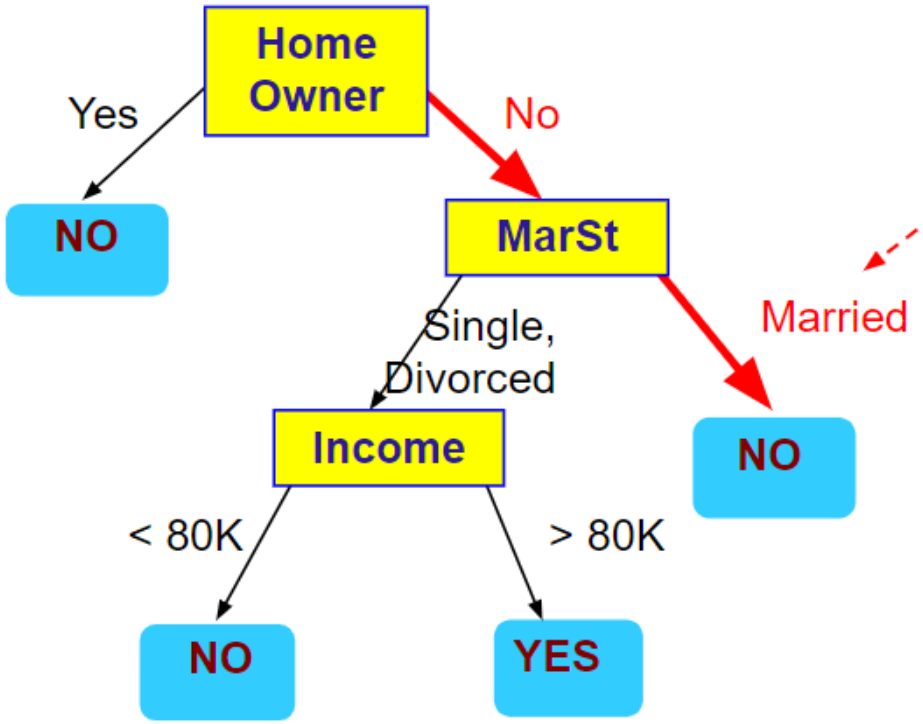
Test Data

Home Owner	Marital Status	Annual Income	Defaulted Borrower
No	Married	80K	?

Apply Model to Test Data



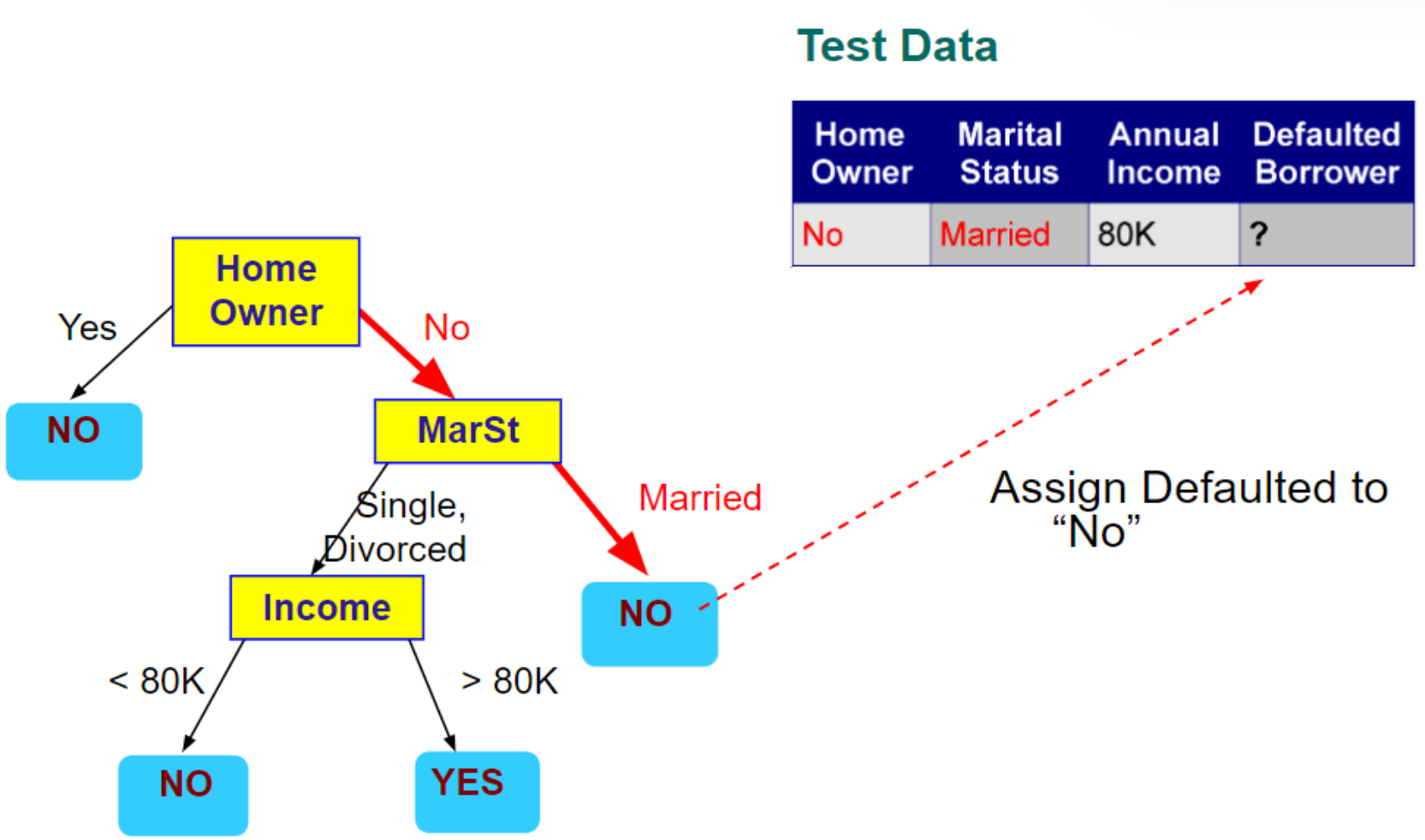
Apply Model to Test Data



Test Data

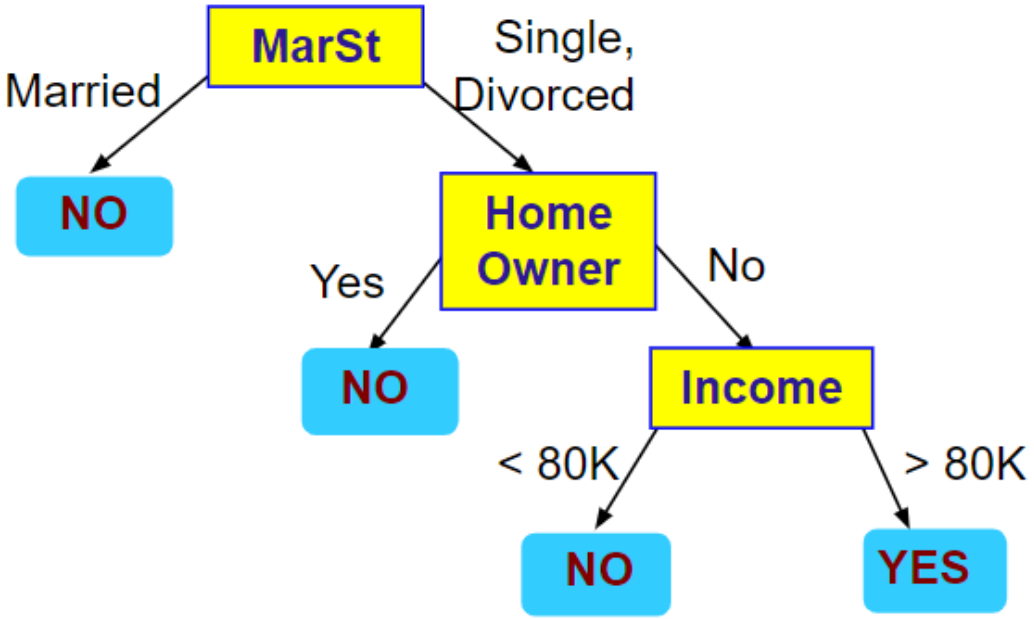
Home Owner	Marital Status	Annual Income	Defaulted Borrower
No	Married	80K	?

Apply Model to Test Data



Another Way for the Decision Tree

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
		categorical	categorical	continuous	class
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



There could be more than one tree that fits the same data!

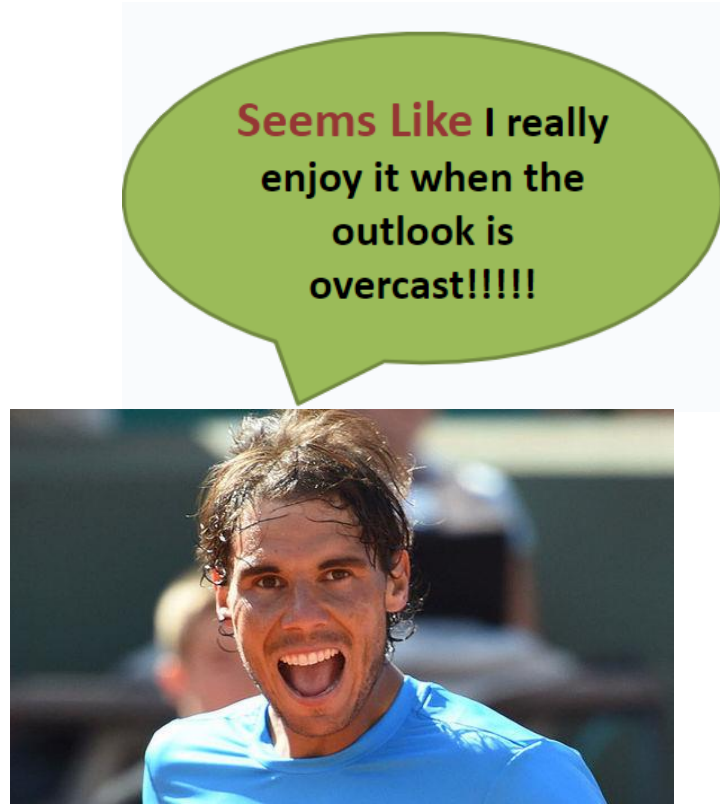
Another Example of Decision Tree

Continue…

Day	Outlook	Temperature	Humidity	Wind	PlayTenni
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

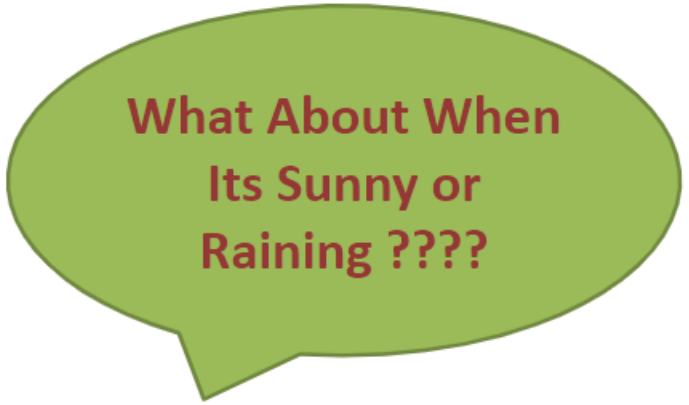
Continue…

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No



Continue…

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No



Continue…

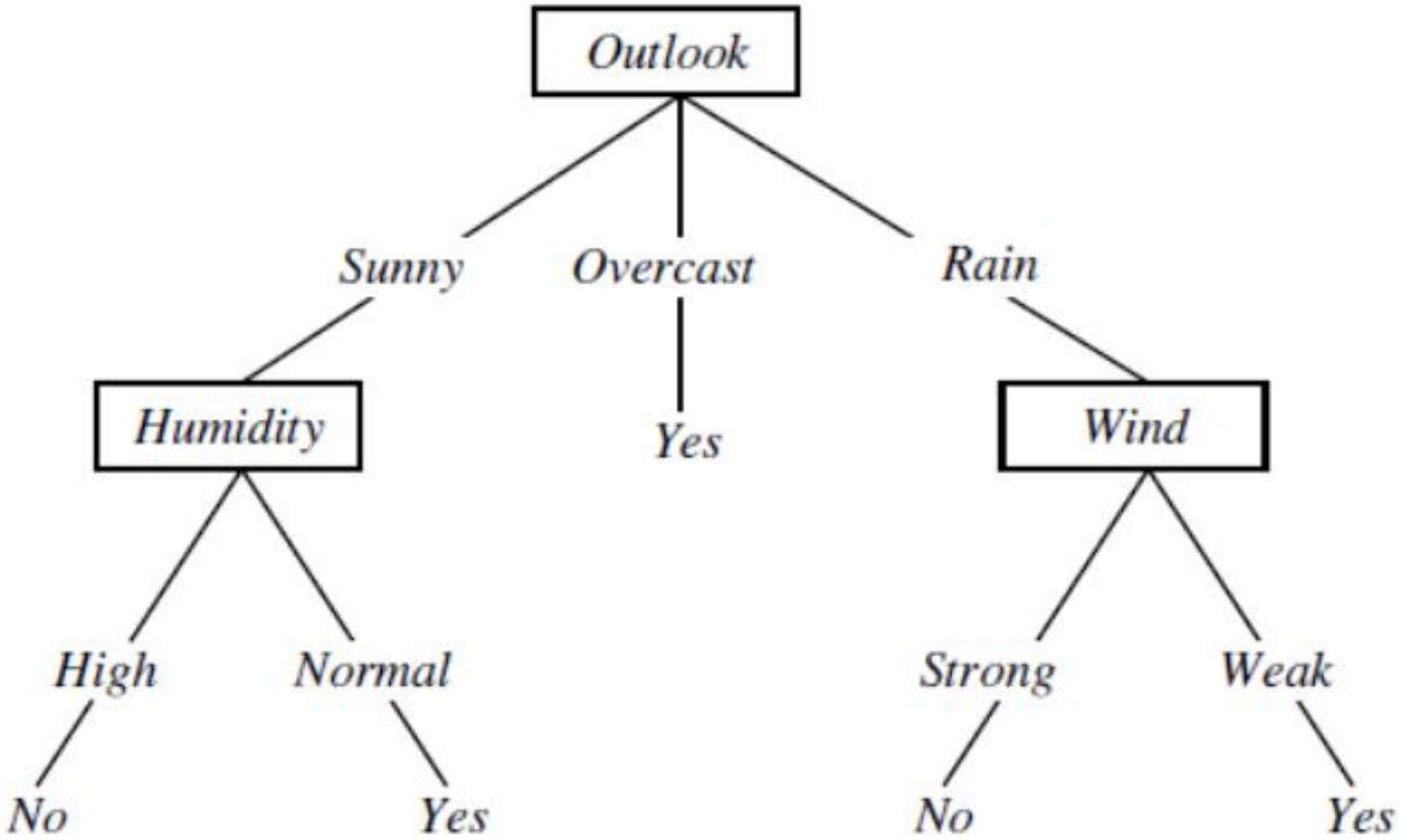
Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes

So, Its Humidity !!!!!!!!

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D14	Rain	Mild	High	Strong	No

Its Wind !

Continue…


Outlook
Sunny
Overcast
Rain
Humidity
Yes
Wind
High
Normal
Strong
Weak
No
Yes
No
Yes

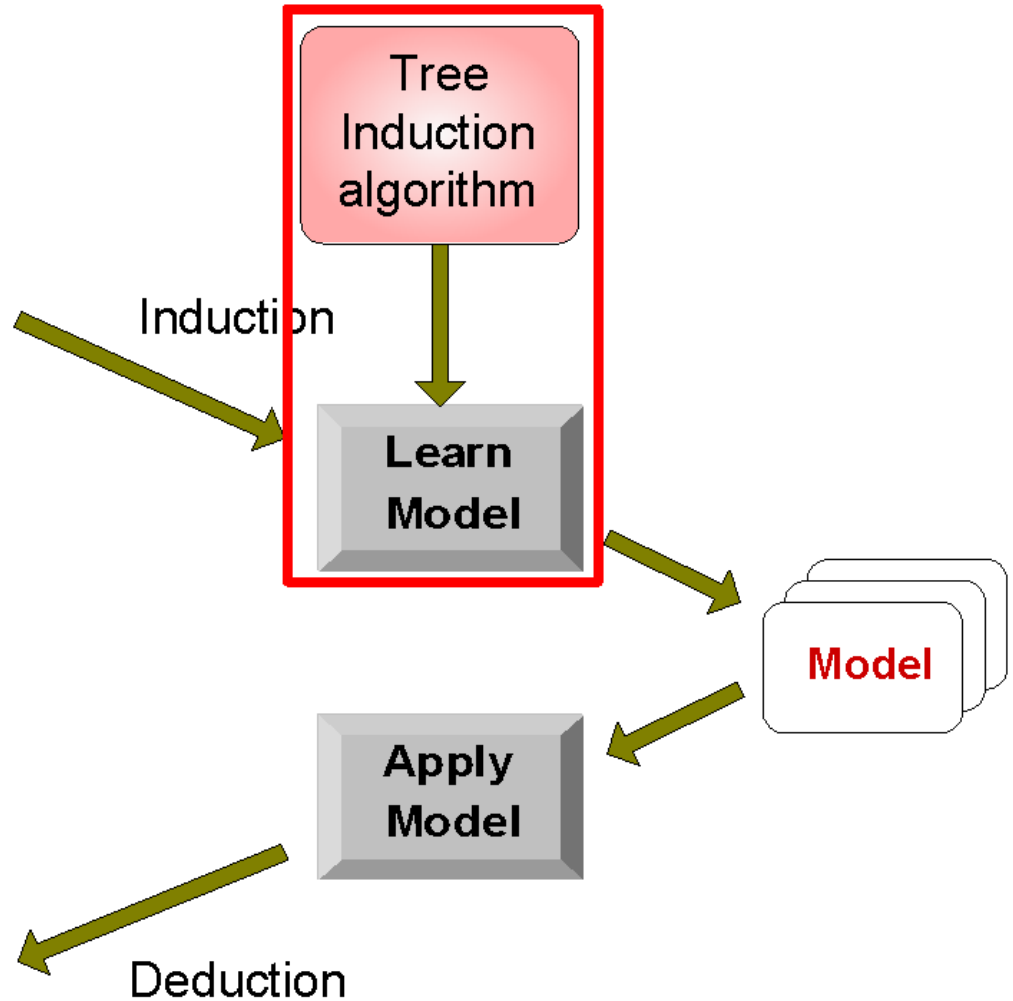
Decision Tree Classification Task

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



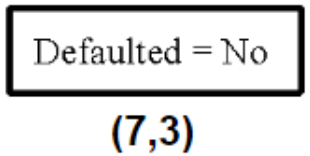
Decision Tree Induction

- Many Algorithms:
  - Hunt's Algorithm (one of the earliest)
  - CART
  - ID3, C4.5
  - SLIQ,SPRINT

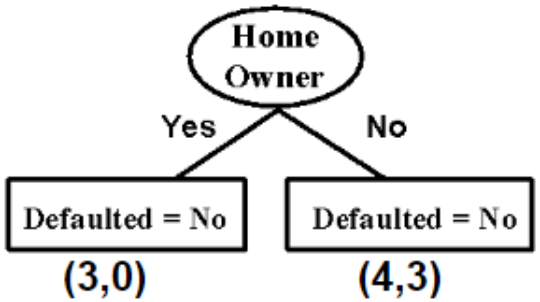
General Structure of Hunt's Algorithm

- Let D_t be the set of training records that reach a node t.
- General Procedure:
  - If D_t contains records that belong the same class y_t, then t is a leaf node labeled as y_t.
  - If D_t contains records that belong to more than one class, use an attribute test to split the data into smaller subsets. Recursively apply the procedure to each subset.

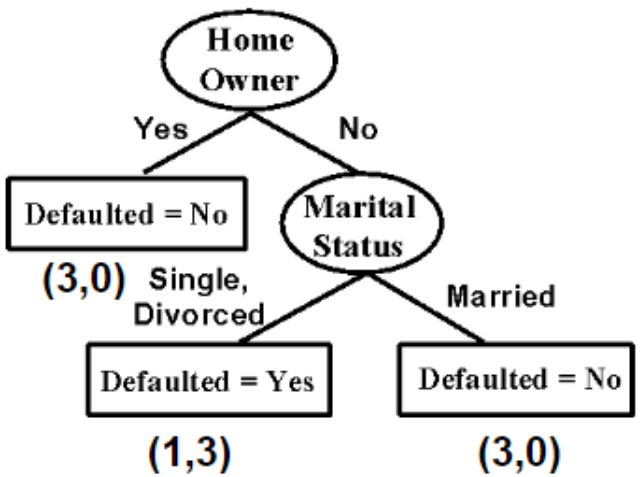
Hunt's Algorithm



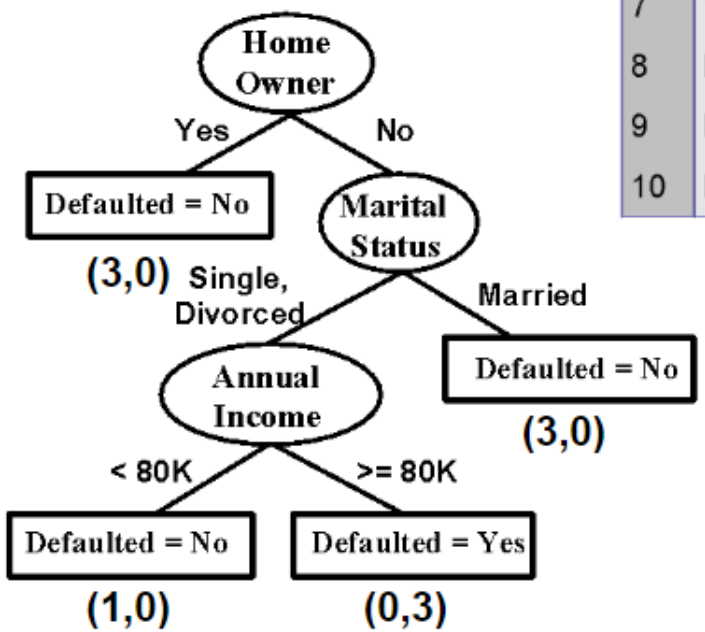
(a)



(b)



(c)



(d)

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Design Issues of Decision Tree Induction

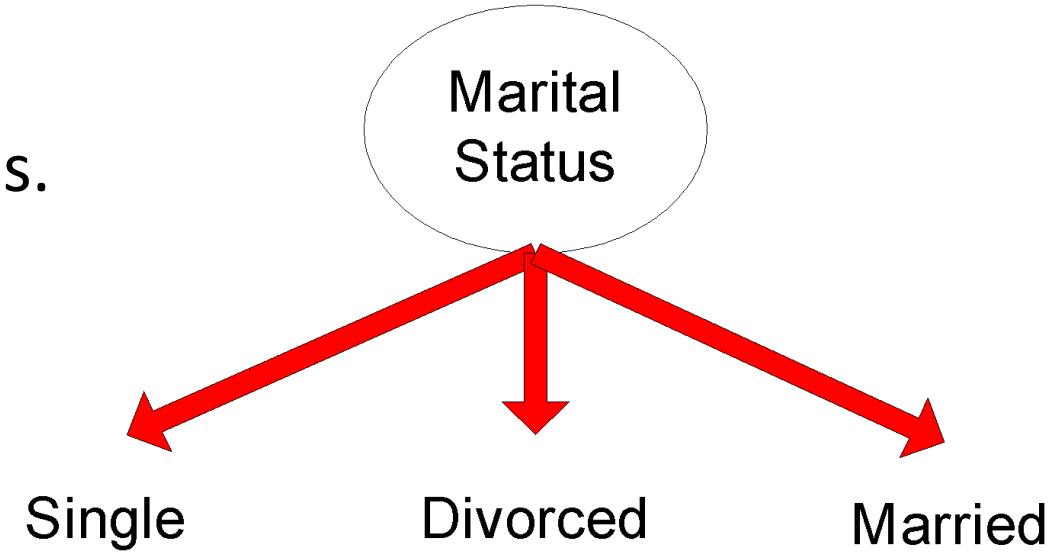
- How should training records be split?
  - Method for expressing test condition
    - depending on attribute types
  - Measure for evaluating the goodness of a test condition

- How should the splitting procedure stop?
  - Stop splitting if all the records belong to the same class or have identical attribute values
  - Early termination

Test Condition for Nominal Attributes

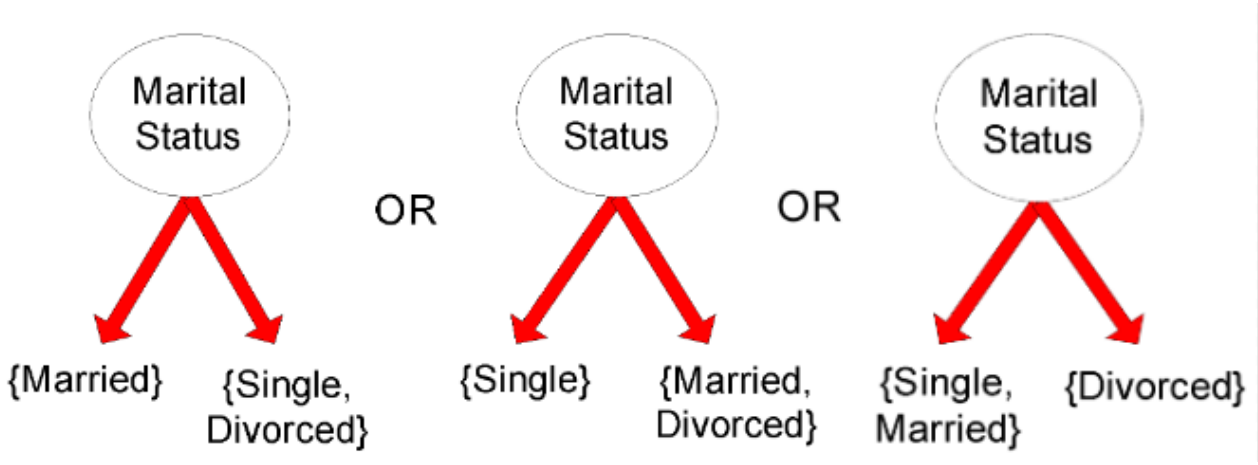
Multi-way split:

- Use as many partitions as distinct values.



Binary split:

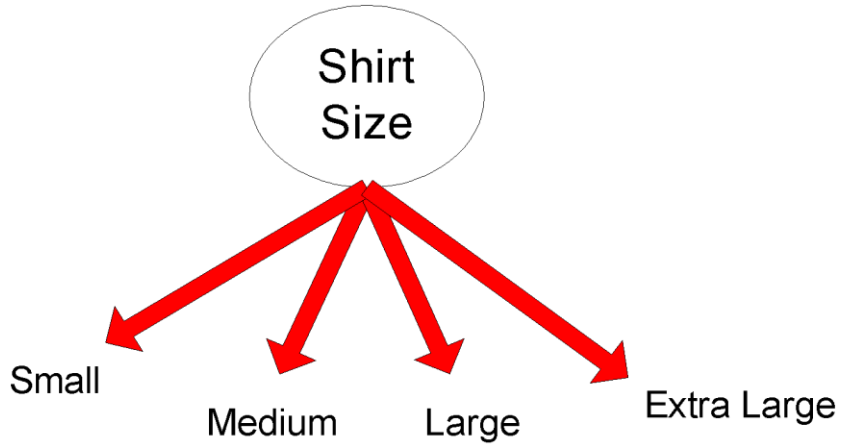
- Divides values into two subsets



Test Condition for Ordinal Attributes

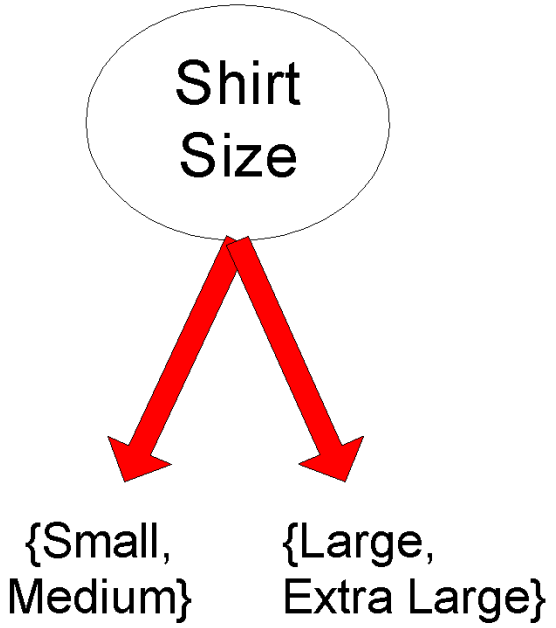
Multi-way split:

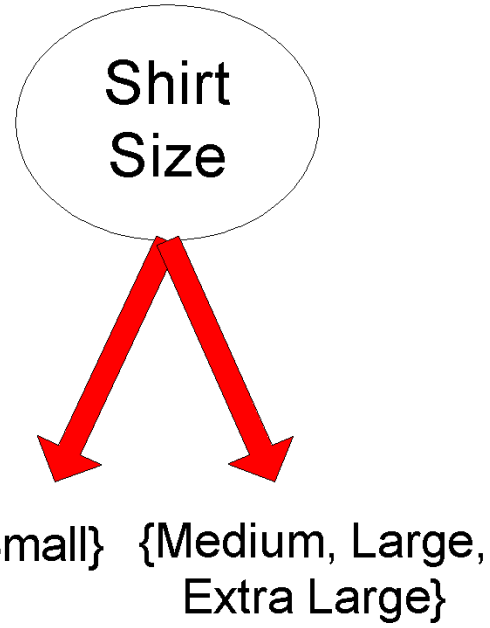
- Use as many partitions as distinct values

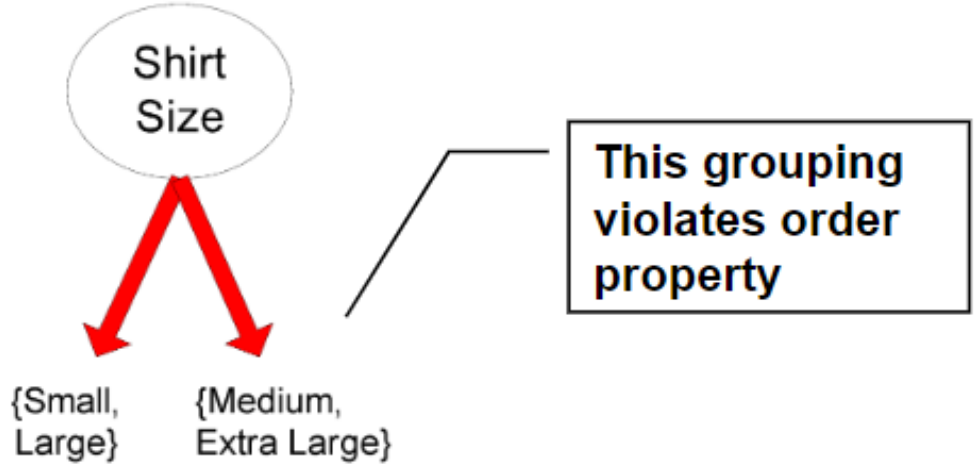


Binary split:

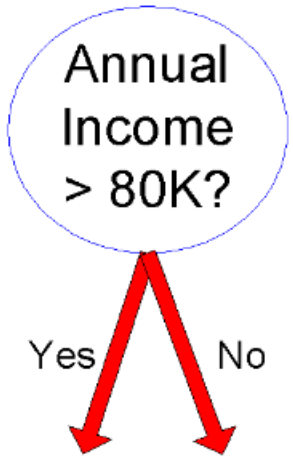
- Divides values into two subsets
- Preserve order property among attribute values



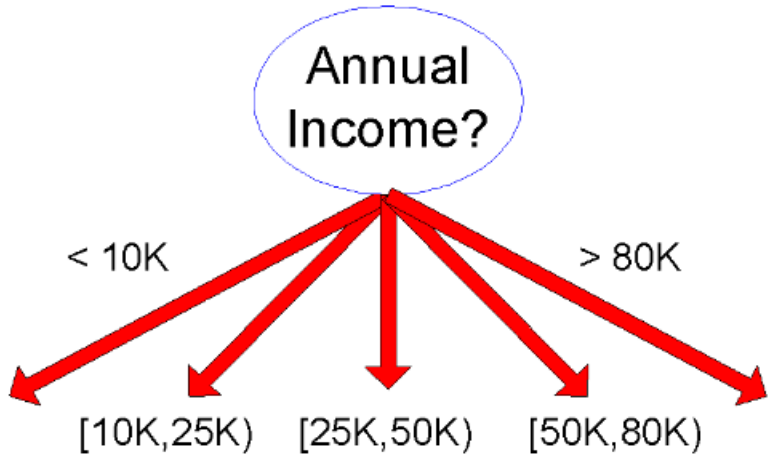




Test Condition for Continuous Attributes



(i) Binary split



(ii) Multi-way split

Test Condition for Continuous Attributes

Different ways of handling:

- Discretization to form an ordinal categorical attribute
  - Ranges can be found by equal interval bucketing, equal frequency bucketing (percentiles), or clustering.
  - Static – discretize once at the beginning
  - Dynamic – repeat at each node

- Binary Decision: $(A < v)$ or $(A \geq v)$
  - consider all possible splits and finds the best cut
  - can be more compute intensive

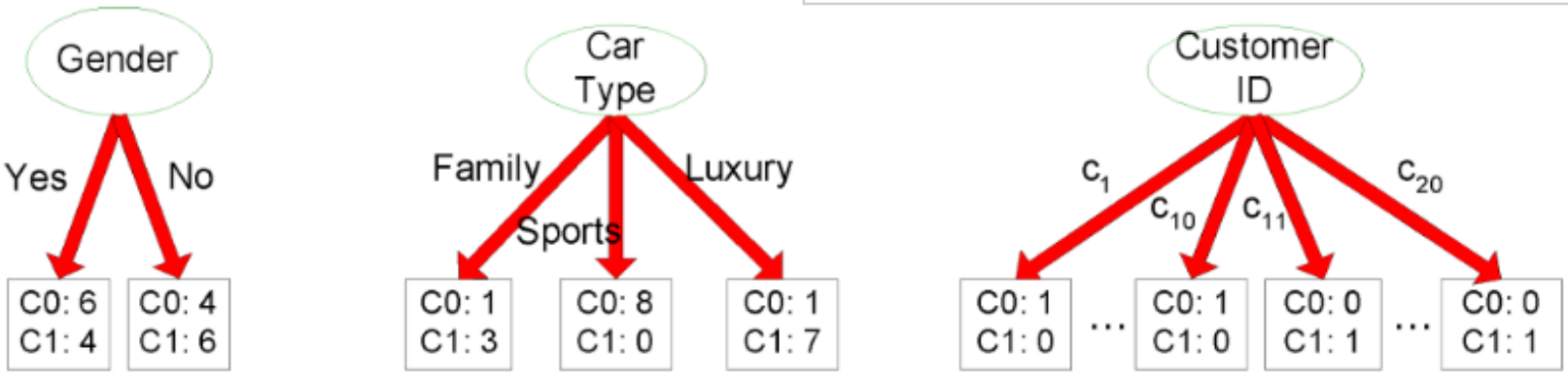
Measures for Selecting an Attribute Test Condition

- There are many measures that can be used to determine the goodness of an attribute test condition.
- These measures try to give preference to attribute test conditions that partition the training instances into purer subsets in the child nodes, which mostly have the same class labels.
- Having purer nodes is useful since a node that has all of its training instances from the same class does not need to be expanded further.
- In contrast, an impure node containing training instances from multiple classes is likely to require several levels of node expansions, thereby increasing the depth of the tree considerably.
- Larger trees are less desirable as they are more susceptible to model overfitting, a condition that may degrade the classification performance on unseen instances.
- They are also difficult to interpret and incur more training and test time as compared to smaller trees.
- In the following, we present different ways of measuring the impurity of a node and the collective impurity of its child nodes, both of which will be used to identify the best attribute test condition for a node.

How to determine the Best Split

Customer Id	Gender	Car Type	Shirt Size	Class
1	M	Family	Small	C0
2	M	Sports	Medium	C0
3	M	Sports	Medium	C0
4	M	Sports	Large	C0
5	M	Sports	Extra Large	C0
6	M	Sports	Extra Large	C0
7	F	Sports	Small	C0
8	F	Sports	Small	C0
9	F	Sports	Medium	C0
10	F	Luxury	Large	C0
11	M	Family	Large	C1
12	M	Family	Extra Large	C1
13	M	Family	Medium	C1
14	M	Luxury	Extra Large	C1
15	F	Luxury	Small	C1
16	F	Luxury	Small	C1
17	F	Luxury	Medium	C1
18	F	Luxury	Medium	C1
19	F	Luxury	Medium	C1
20	F	Luxury	Large	C1

Before Splitting: 10 records of class 0, 10 records of class 1



Which test condition is the best?

Measures of Node Impurity

- Gini Index

$$Gini\ Index = 1 - \sum_{i=0}^{c-1} p_i(t)^2$$

Where $\boldsymbol{p_i(t)}$ is the frequency of class $\boldsymbol{i}$ at node $\mathbf{t}$, and $\boldsymbol{c}$ is the total number of classes

- Entropy

$$Entropy = -\sum_{i=0}^{c-1} p_i(t) log_2 p_i(t)$$

- Misclassification error

$$Classification\ error = 1 - \max[p_i(t)]$$

Entropy – Binary Class

- S is a sample of training examples
- $p_{\oplus}$ is the proportion of positive examples in S
- $p_{\ominus}$ is the proportion of negative examples in S
- Entropy measures the impurity of S

$$Entropy(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

- Entropy Is **Zero (Minimum)**, When all the examples belong to **same class**!!!
- Entropy Is **1(Maximum)**, When there are **equal** number of positive and negative examples!

Entropy – Binary Class

To illustrate, suppose S is a collection of 14 examples of some boolean concept, including 9 positive and 5 negative examples (we adopt the notation [9+, 5−] to summarize such a sample of data). Then the entropy of S relative to this boolean classification is

$$\begin{aligned} Entropy([9+, 5-]) &= -(9/14)\log_2(9/14) - (5/14)\log_2(5/14) \\ &= 0.940 \end{aligned}$$

$$Entropy(S) \equiv \sum_{i=1}^{c} -p_i \log_2 p_i$$

Decision Tree Induction – ID3

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

Complete entropy of dataset is:

```
H(S) = - p(yes) * log2(p(yes)) - p(no) * log2(p(no))
     = - (9/14) * log2(9/14) - (5/14) * log2(5/14)
     = - (-0.41) - (-0.53)
     = 0.94
```

Continue…

First Attribute - Outlook

```
Categorical values - sunny, overcast and rain
H(Outlook=sunny) = -(2/5)*log(2/5)-(3/5)*log(3/5) =0.971
H(Outlook=rain) = -(3/5)*log(3/5)-(2/5)*log(2/5) =0.971
H(Outlook=overcast) = -(4/4)*log(4/4)-0 = 0

Average Entropy Information for Outlook -
I(Outlook) = p(sunny) * H(Outlook=sunny) + p(rain) * H(Outlook=rain) + p(overcast) * H(Outlook=overcast)
= (5/14)*0.971 + (5/14)*0.971 + (4/14)*0
= 0.693

Information Gain = H(S) - I(Outlook)
                 = 0.94 - 0.693
                 = 0.247
```

Continue…

Second Attribute - Temperature

```
Categorical values - hot, mild, cool
H(Temperature=hot) = -(2/4)*log(2/4)-(2/4)*log(2/4) = 1
H(Temperature=cool) = -(3/4)*log(3/4)-(1/4)*log(1/4) = 0.811
H(Temperature=mild) = -(4/6)*log(4/6)-(2/6)*log(2/6) = 0.9179
Average Entropy Information for Temperature -
I(Temperature) = p(hot)*H(Temperature=hot) + p(mild)*H(Temperature=mild) + p(cool)*H(Temperature=cool)
= (4/14)*1 + (6/14)*0.9179 + (4/14)*0.811
= 0.9108

Information Gain = H(S) - I(Temperature)
                 = 0.94 - 0.9108
                 = 0.0292
```

Continue…

Third Attribute - Humidity

```
Categorical values - high, normal
H(Humidity=high) = -(3/7)*log(3/7)-(4/7)*log(4/7) = 0.983
H(Humidity=normal) = -(6/7)*log(6/7)-(1/7)*log(1/7) = 0.591

Average Entropy Information for Humidity -
I(Humidity) = p(high)*H(Humidity=high) + p(normal)*H(Humidity=normal)
= (7/14)*0.983 + (7/14)*0.591
= 0.787

Information Gain = H(S) - I(Humidity)
                 = 0.94 - 0.787
                 = 0.153
```

Continue…

Fourth Attribute - Wind

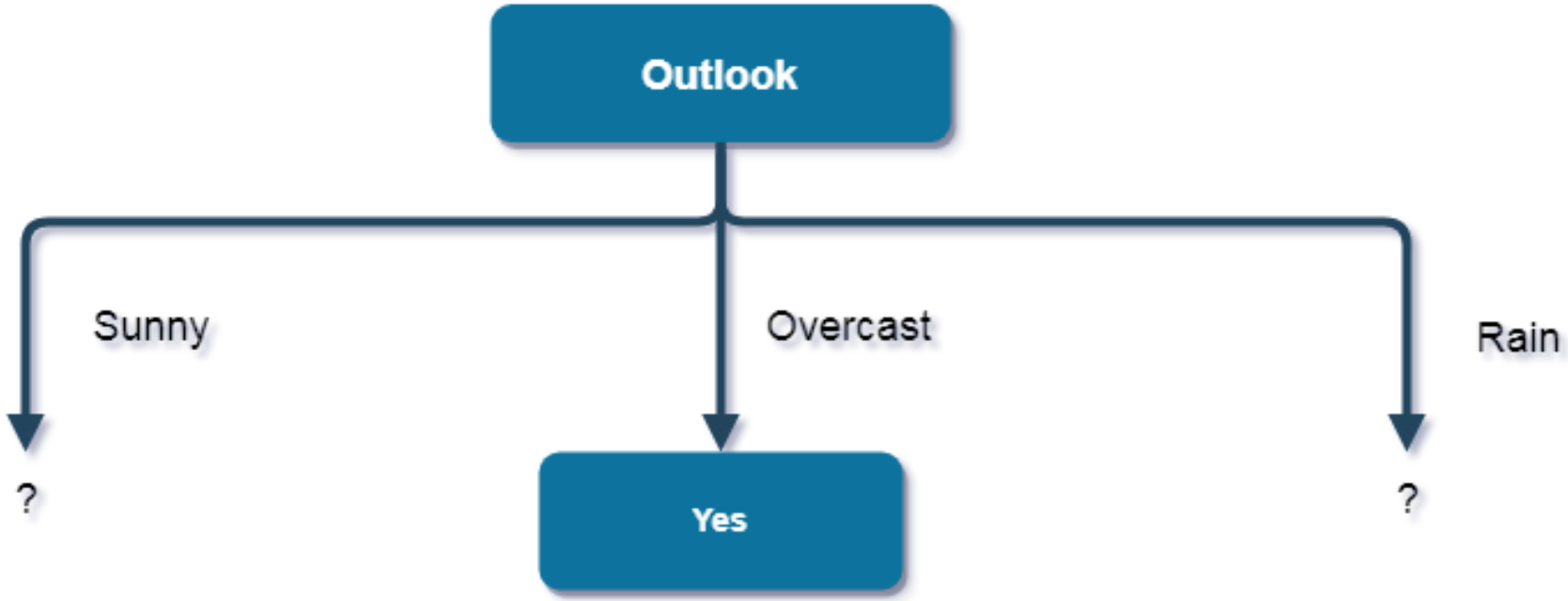
```
Categorical values - weak, strong
H(Wind=weak) = -(6/8)*log(6/8)-(2/8)*log(2/8) = 0.811
H(Wind=strong) = -(3/6)*log(3/6)-(3/6)*log(3/6) = 1

Average Entropy Information for Wind -
I(Wind) = p(weak)*H(Wind=weak) + p(strong)*H(Wind=strong)
= (8/14)*0.811 + (6/14)*1
= 0.892

Information Gain = H(S) - I(Wind)
                 = 0.94 - 0.892
                 = 0.048
```

Continue…

- Here, the attribute with maximum information gain is Outlook. So, the decision tree built so far -



Continue…

- Now, finding the best attribute for splitting the data with Outlook=Sunny values{ Dataset rows = [1, 2, 8, 9, 11]}.

```
Complete entropy of Sunny is -
H(S) = - p(yes) * log2(p(yes)) - p(no) * log2(p(no))
     = - (2/5) * log2(2/5) - (3/5) * log2(3/5)
     = 0.971
```

First Attribute - Temperature

```
Categorical values - hot, mild, cool
H(Sunny, Temperature=hot) = -0-(2/2)*log(2/2) = 0
H(Sunny, Temperature=cool) = -(1)*log(1)- 0 = 0
H(Sunny, Temperature=mild) = -(1/2)*log(1/2)-(1/2)*log(1/2) = 1
Average Entropy Information for Temperature -
= (2/5)*0 + (1/5)*0 + (2/5)*1
= 0.4

Information Gain = H(Sunny) - I(Sunny, Temperature)
                 = 0.971 - 0.4
                 = 0.571
```

Continue…

Second Attribute - Humidity

```
Categorical values - high, normal
H(Sunny, Humidity=high) = - 0 - (3/3)*log(3/3) = 0
H(Sunny, Humidity=normal) = -(2/2)*log(2/2)-0 = 0

Average Entropy Information for Humidity -
I(Sunny, Humidity) = p(Sunny, high)*H(Sunny, Humidity=high) + p(Sunny, normal)*H(Sunny, Humidity=normal)
= (3/5)*0 + (2/5)*0
= 0

Information Gain = H(Sunny) - I(Sunny, Humidity)
                 = 0.971 - 0
                 = 0.971
```

Third Attribute - Wind

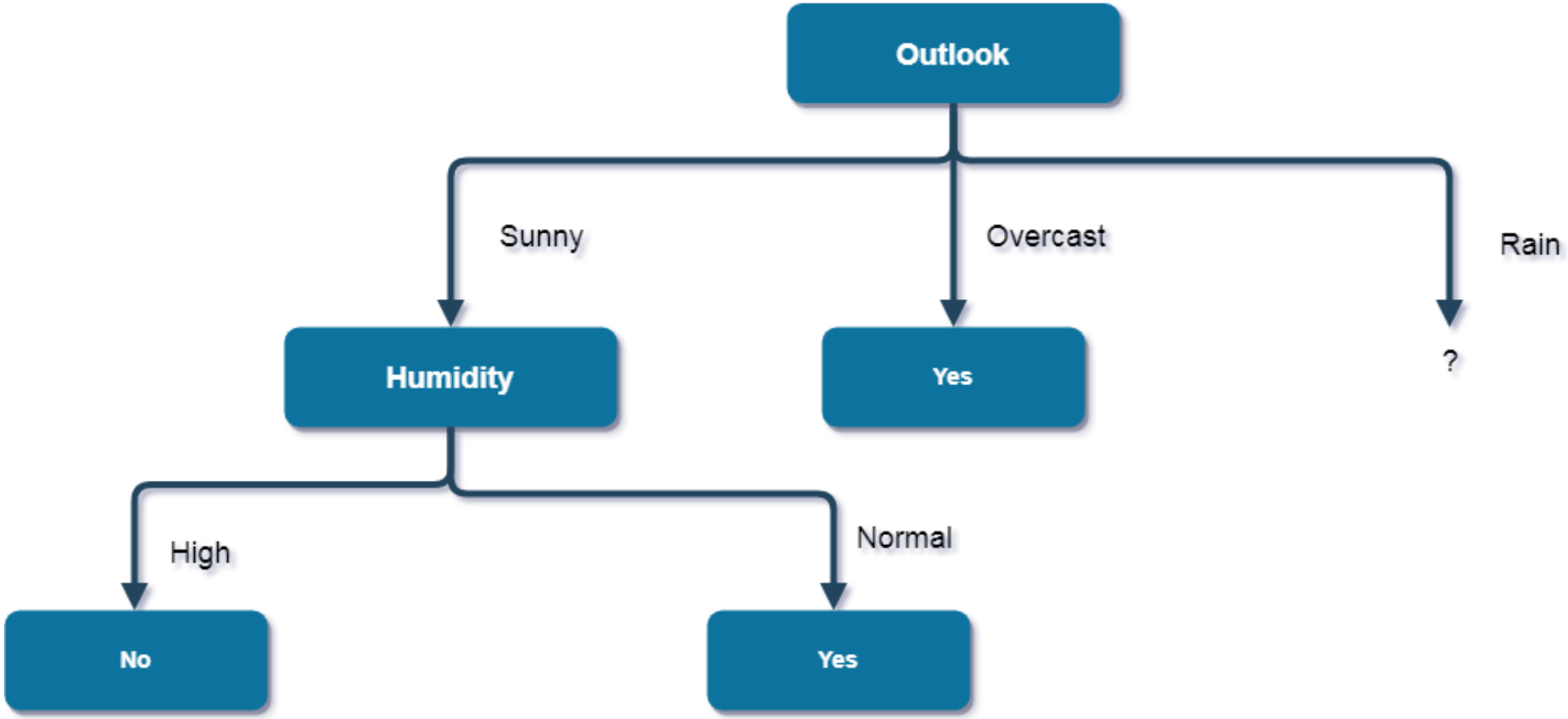
```
Categorical values - weak, strong
H(Sunny, Wind=weak) = -(1/3)*log(1/3)-(2/3)*log(2/3) = 0.918
H(Sunny, Wind=strong) = -(1/2)*log(1/2)-(1/2)*log(1/2) = 1

Average Entropy Information for Wind -
I(Sunny, Wind) = p(Sunny, weak)*H(Sunny, Wind=weak) + p(Sunny, strong)*H(Sunny, Wind=strong)
= (3/5)*0.918 + (2/5)*1
= 0.9508

Information Gain = H(Sunny) - I(Sunny, Wind)
                 = 0.971 - 0.9508
                 = 0.0202
```

Continue…

- Here, the attribute with maximum information gain is Humidity. So, the decision tree built so far -



Continue…

- Now, finding the best attribute for splitting the data with Outlook=Rain values{ Dataset rows = [4, 5, 6, 10, 14]}.

```
Complete entropy of Rain is -
H(S) = - p(yes) * log2(p(yes)) - p(no) * log2(p(no))
     = - (3/5) * log(3/5) - (2/5) * log(2/5)
     = 0.971
```

First Attribute - Temperature

```
Categorical values - mild, cool
H(Rain, Temperature=cool) = -(1/2)*log(1/2)- (1/2)*log(1/2) = 1
H(Rain, Temperature=mild) = -(2/3)*log(2/3)-(1/3)*log(1/3) = 0.918
Average Entropy Information for Temperature -
I(Rain, Temperature) = p(Rain, mild)*H(Rain, Temperature=mild) + p(Rain, cool)*H(Rain, Temperature=cool)
= (2/5)*1 + (3/5)*0.918
= 0.9508

Information Gain = H(Rain) - I(Rain, Temperature)
                 = 0.971 - 0.9508
                 = 0.0202
```

Continue…

Second Attribute - Humidity

Categorical values - high, normal

Information Gain $0.97 - \frac{2}{5}\,1.0 - \frac{3}{5}0.918 = 0.0192$

Third Attribute - Wind

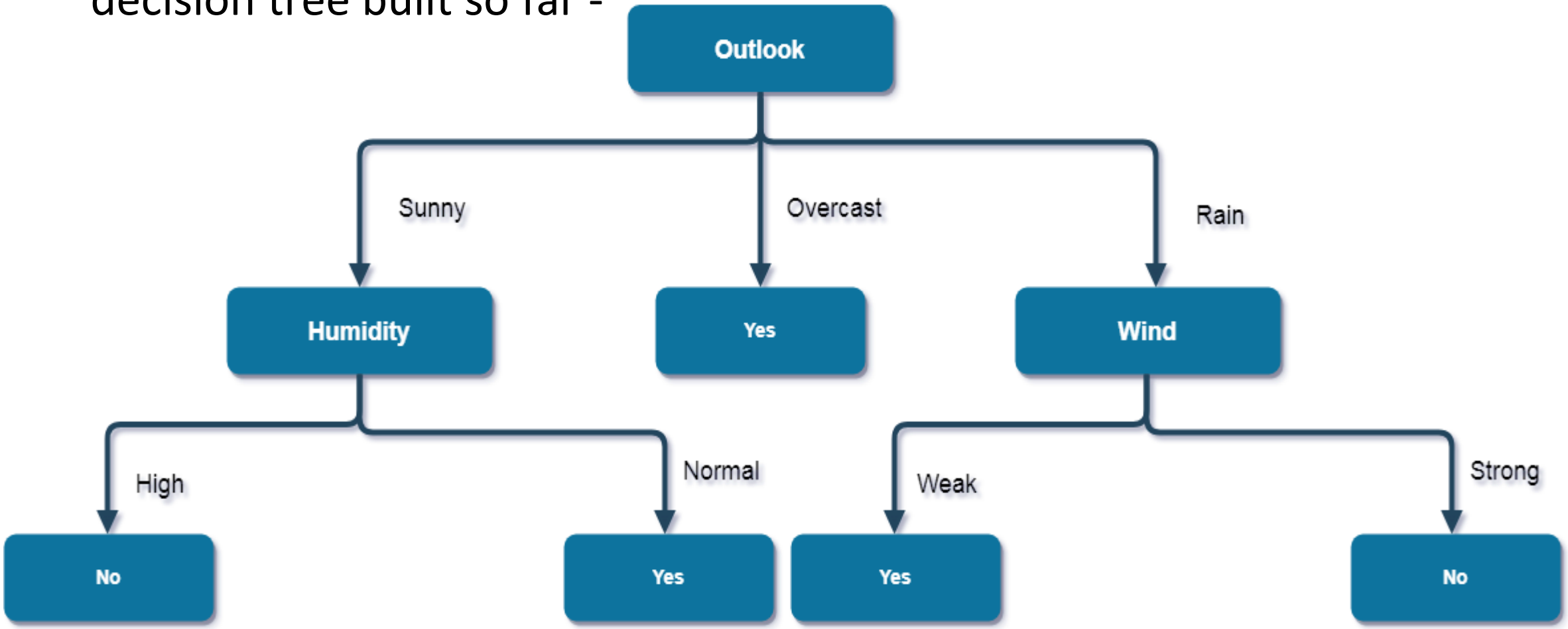
```
Categorical values - weak, strong
H(Wind=weak) = -(3/3)*log(3/3)-0 = 0
H(Wind=strong) = 0-(2/2)*log(2/2) = 0

Average Entropy Information for Wind -
I(Wind) = p(Rain, weak)*H(Rain, Wind=weak) + p(Rain, strong)*H(Rain, Wind=strong)
= (3/5)*0 + (2/5)*0
= 0

Information Gain = H(Rain) - I(Rain, Wind)
                 = 0.971 - 0
                 = 0.971
```

Continue…

- Here, the attribute with maximum information gain is Wind. So, the decision tree built so far -



Reference Link

- https://www.youtube.com/watch?v=coOTEc-0OGw&ab_channel=MaheshHuddar
- https://iq.opengenus.org/id3-algorithm/

Thank You!