Odd Sem., AY 2024-25

Lecture: 23-24-25

Fond. Of DS (DSPC201)

Course Instructor: Dr Vikram Singh

Supervised vs. Unsupervised Learning

- Clustering: Unsupervised learning
  - The *class labels* of training data is unknown
  - Given a set of measurements, observations, etc. with the aim of establishing the existence of *classes or clusters* in the data
- Classification: Supervised learning
  - Supervision: The training data (observations, measurements, etc.) are accompanied by **labels** indicating the class of the observations
  - New data is classified based on the training set

Classification: Prediction Problems

- Classification
  - predicts categorical class labels (discrete or nominal)
  - classifies data (constructs a model) based on the training set and the values (class labels) in a classifying attribute and uses it in classifying new data

- *Typical applications*
  - Credit/loan approval:
  - Medical diagnosis: if a tumor is cancerous or benign
  - Fraud detection: if a transaction is fraudulent
  - Web page categorization: which category it is

Classification: *A Two-Step Process*

- Model construction: describing a set of predetermined classes
  - Each tuple/sample is assumed to belong to a predefined class, as determined by the class label attribute
  - The set of tuples used for model construction is training set
  - The model is represented as *classification rules, decision trees, or mathematical formulae*
- Model usage: for classifying future or unknown objects
  - Estimate accuracy of the model
    - The known label of test sample is compared with the classified result from the model
    - Accuracy rate is the percentage of test set samples that are correctly classified by the model
    - Test set is independent of training set (otherwise overfitting)
  - If the accuracy is acceptable, use the model to classify new data
- Note: If *the test set* is used to select models, it is called validation (test) set

Accuracy Estimation : *Confusion* metric

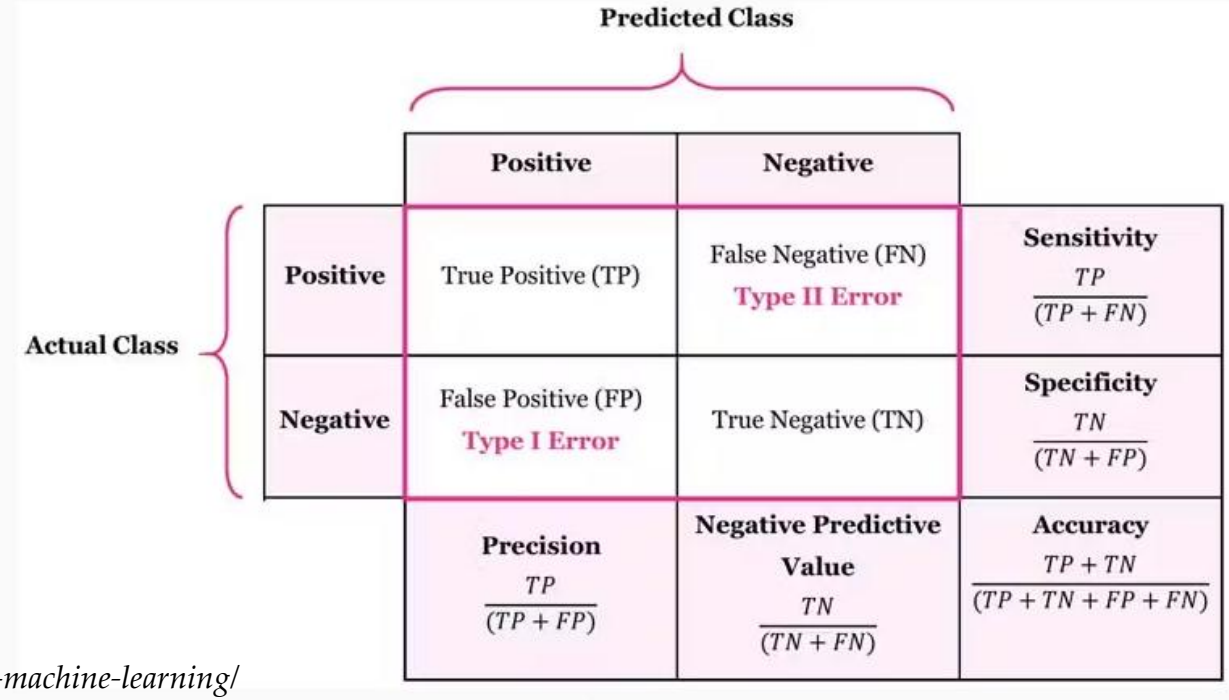
What it does	Compares predicted labels to true labels
What it shows	Number of true positives, true negatives, false positives, and false negatives
How it helps	Analyzes model performance and improves predictive accuracy
When it's used	In supervised learning frameworks to evaluate classification models

Accuracy Estimation : *Confusion* metric

Example: Confusion Matrix for *Dog Image Recognition* with Numbers

Index	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
Actual	Dog	Dog	Dog	Not Dog	Dog	Not Dog	Dog	Dog	Not Dog	Not Dog
Predicted	Dog	Not Dog	Dog	Not Dog	Dog	Dog	Dog	Dog	Not Dog	Not Dog
Result	**TP**	**FN**	**TP**	**TN**	**TP**	**FP**	**TP**	**TP**	**TN**	**TN**

- Actual Dog Counts = 6
- Actual Not Dog Counts = 4
- **True Positive (TP) = 5**
- **False Positive (FP) = 1**
- **True Negative (TN) = 3**
- **False Negative (FN) = 1**



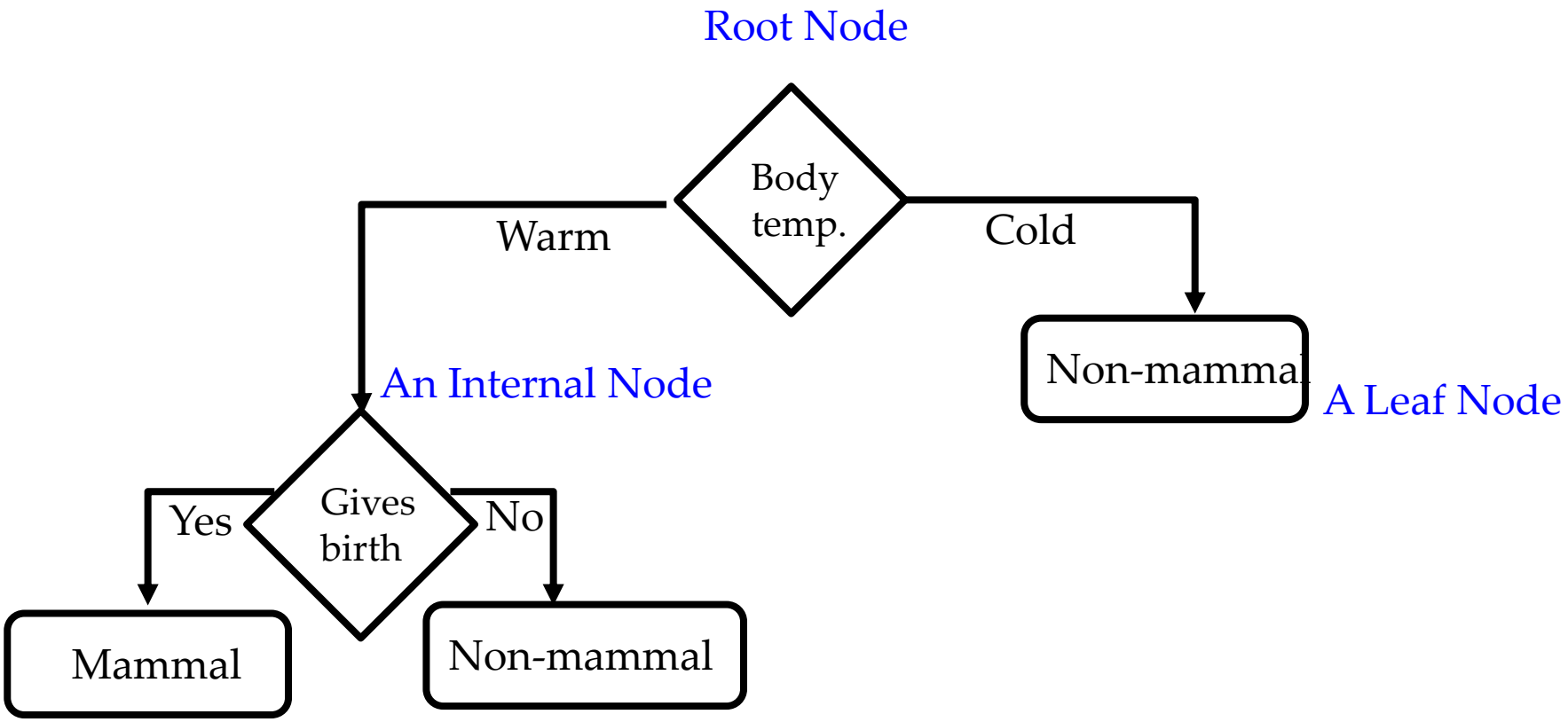
➢*https://www.geeksforgeeks.org/confusion-matrix-machine-learning/*

Issues Affecting Model Selection

- **Accuracy**
  - classifier accuracy: predicting class label
- **Speed**
  - time to construct the model (training time)
  - time to use the model (classification/prediction time)
- **Robustness**: handling noise and missing values
- **Scalability**: efficiency in disk-resident databases
- **Interpretability**
  - understanding and insight provided by the model
- Other measures, e.g., goodness of rules, such as decision tree size or compactness of classification rules

Decision Tree Induction: An Example

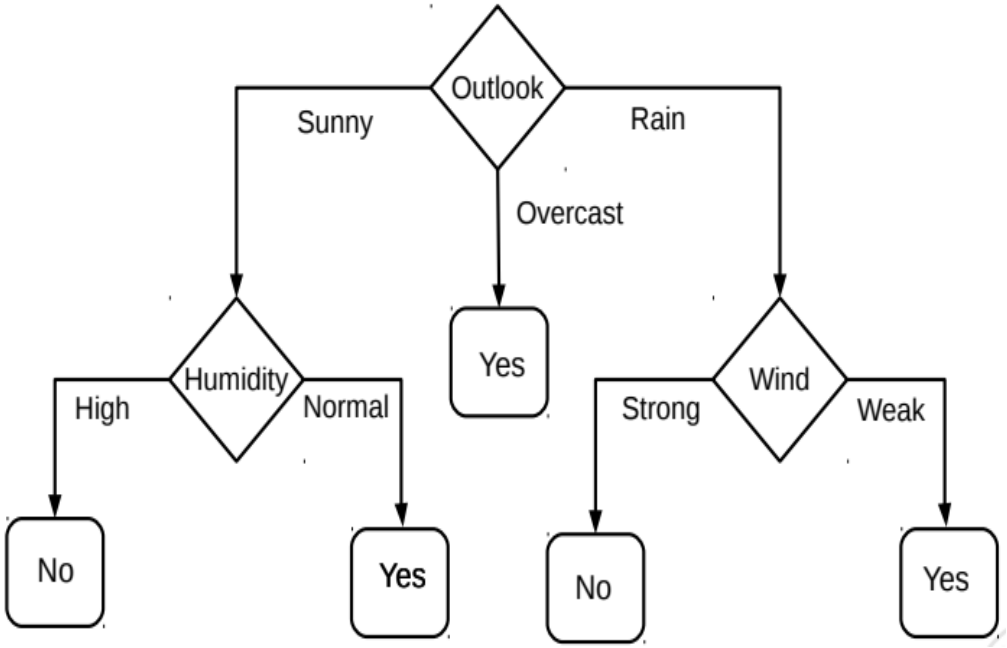
- A Decision Tree (DT) defines a hierarchy of rules to make a prediction.
- *Root* and *internal nodes* test rules. *Leaf nodes make predictions.*
- Decision Tree (DT) learning is about learning such a tree from labeled data.



Decision Trees for Classification: Another Example

- *Deciding whether to play or not to play Tennis on a Saturday*
- Each input (Saturday) has 4 categorical features:
  - Outlook, Temp., Humidity, Wind
- *A binary classification problem: Play vs No-play*
- *Below Left*: Training data, *Below Right*: A decision tree constructed using this data

day	outlook	temperature	humidity	wind	play
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



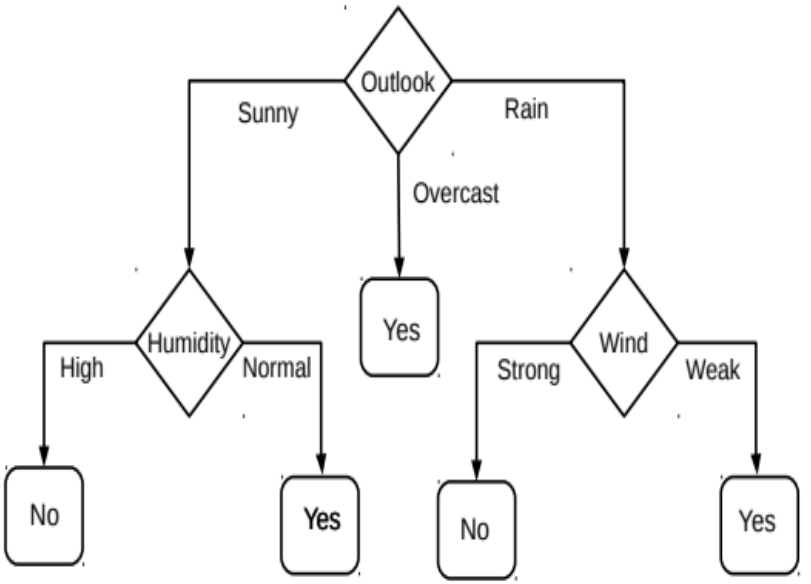
How to Split at Internal Nodes?

- Recall that each internal node receives a subset of all the training inputs
- Regardless of the criterion, the split should result in as "pure" groups as possible
  - A *pure group* means that the majority of the inputs have the same label/output
- For classification problems (discrete outputs), entropy is a measure of purity
  - Low entropy ⇒ high purity (less uniform label distribution)
  - Splits that give the largest reduction (before split vs after split) in entropy are preferred (this reduction is also known as "information gain")

Decision Tree Construction: An Example

- Let's consider the playing Tennis example
- Assume each internal node will test the value of one of the features
- Question: *Why does it make more sense to test the feature "outlook" first?*
- Answer: Of all the 4 features, it's the most informative
  - It has the highest information gain as the root node

day	outlook	temperature	humidity	wind	play
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

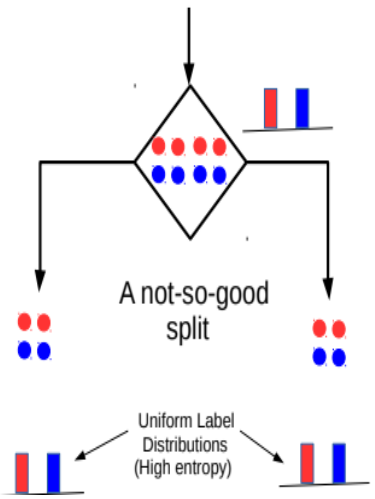


Entropy and Information Gain

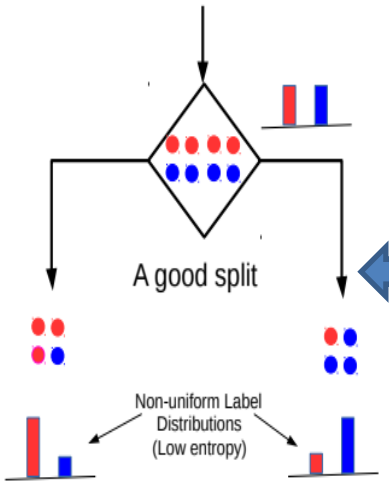
- Assume a set of labelled inputs $\boldsymbol{S}$ from C classes, p_c as fraction of class c inputs
- Entropy of the set $\boldsymbol{S}$ is defined as $\mathrm{H}(\boldsymbol{S}) = -\sum_{c \in C} p_c \log p_c$
- Suppose a rule splits $\boldsymbol{S}$ into two smaller disjoint sets $\boldsymbol{S}_1$ and $\boldsymbol{S}_2$
- Reduction in entropy after the split is called information gain

$$IG = H(S) - \frac{|S_1|}{|S|} H(S_1) - \frac{|S_2|}{|S|} H(S_2)$$

This split has a low IG (in fact zero IG)



VS



This split has higher IG

Entropy and Information Gain

- Let's use IG based criterion to construct a DT for the *Tennis example*
- *At root node, let's compute IG of each of the 4 features*
- Consider feature "wind".
  - Root contains all examples S = [9 +ve, and 5-ve] :

day	outlook	temperature	humidity	wind	play
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

$$H(S) = -(9/14)\log_2(9/14) - (5/14)\log_2(5/14) = 0.94$$

$$S_{weak} = [6+, 2-] \Rightarrow H(S_{weak}) = 0.811$$

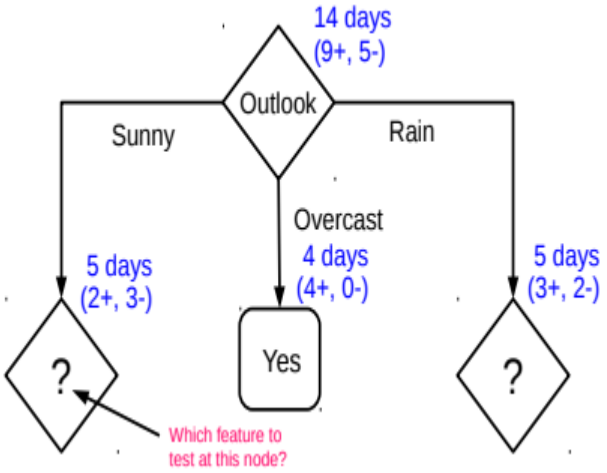
$$S_{strong} = [3+, 3-] \Rightarrow H(S_{strong}) = 1$$

$$IG(S, wind) = H(S) - \frac{|S_{weak}|}{|S|} H(S_{weak}) - \frac{|S_{strong}|}{|S|} H(S_{strong}) = 0.94 - 8/14 * 0.811 - 6/14 * 1 = 0.048$$

- Likewise, at root: **IG(S, outlook) = 0.246, IG(S, humidity) = 0.151, IG(S,temp) = 0.029**
- Thus we choose "outlook" feature to be tested at the root node
- Now how to grow the DT, i.e., what to do at the next level? Which feature to test next?
- Rule: Iterate - for each child node, select the feature with the highest IG

Growing the tree

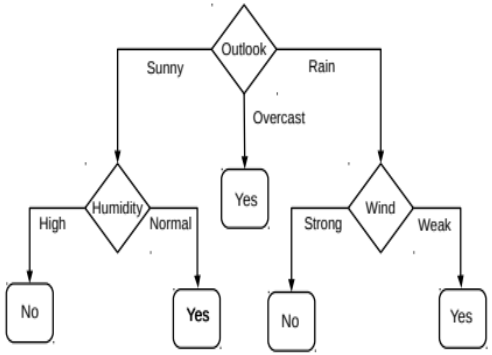
day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	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



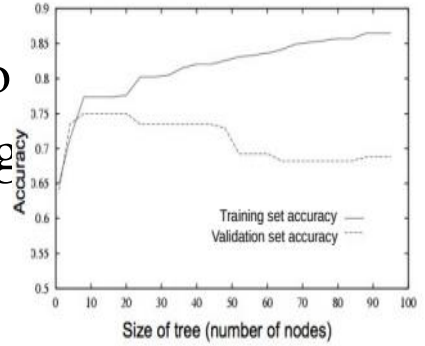
- Proceeding as before, for level 2, left node, we can verify that
  - IG(S,temp) = 0.570, IG(S, humidity) = 0.970, IG(S, wind) = 0.019
- Thus *humidity* chosen as the feature to be tested at level 2, left node
- *No need to expand the middle node (already "pure" - all "yes" training examples)*
- Can also verify that *wind has the largest IG* for the right node
- Note: If a feature has already been tested along a path earlier, we don't consider it again

When to stop growing the tree?

day	outlook	temperature	humidity	wind	play
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



- Stop expanding a node further (i.e., make it a leaf node) when
  - It consist of all training examples having the same label (the node becomes "pure")
  - We run out of features to test along the path to that no
  - The DT starts to overfit (can be checked by monitoring the validation set accuracy)



- **Important:** No need to obsess too much for purity
  - It is okay to have a leaf node that is not fully pure, e.g., this
  - At test inputs that reach an impure leaf, can predict probability of belonging to each class (in above example, p(red) = 3/8, p(green) = 5/8), or simply predict the majority label

Algorithm for Decision Tree Induction

- Basic algorithm (a greedy algorithm)
  - Tree is constructed in a top-down recursive divide-and-conquer manner
  - At start, all the training examples are at the root
  - Attributes are categorical (if continuous-valued, they are discretized in advance)
  - Examples are *partitioned recursively based on selected attributes*
  - Test attributes are selected on the basis of a heuristic or *statistical measure* (e.g., information gain)
- Conditions for stopping partitioning
  - All samples for a given node belong to the same class
  - There are no remaining attributes for further partitioning – majority voting is employed for classifying the leaf
  - There are no samples left