COMP4388: **MACHINE LEARNING**

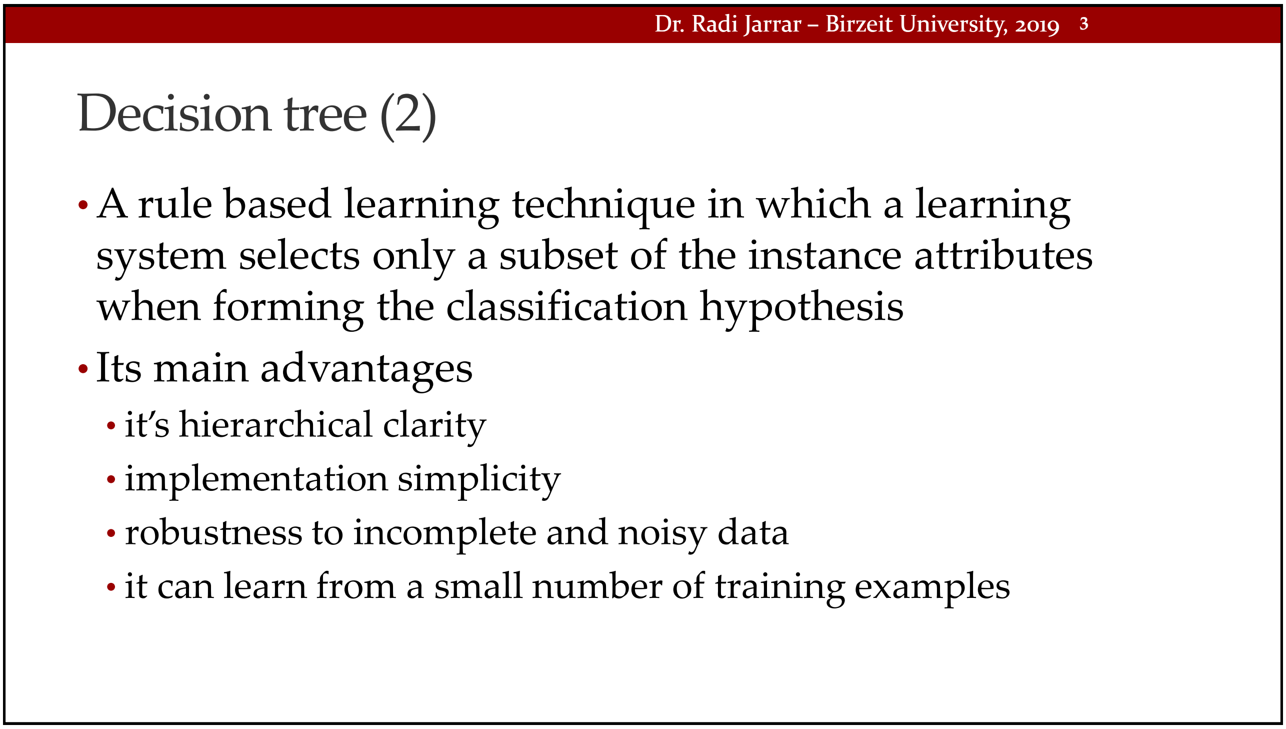
Week 8-Decision Tree Learning

Dr. Radi Jarrar
Department of Computer Science

جامعة بيرزيت
BIRZEIT UNIVERSITY

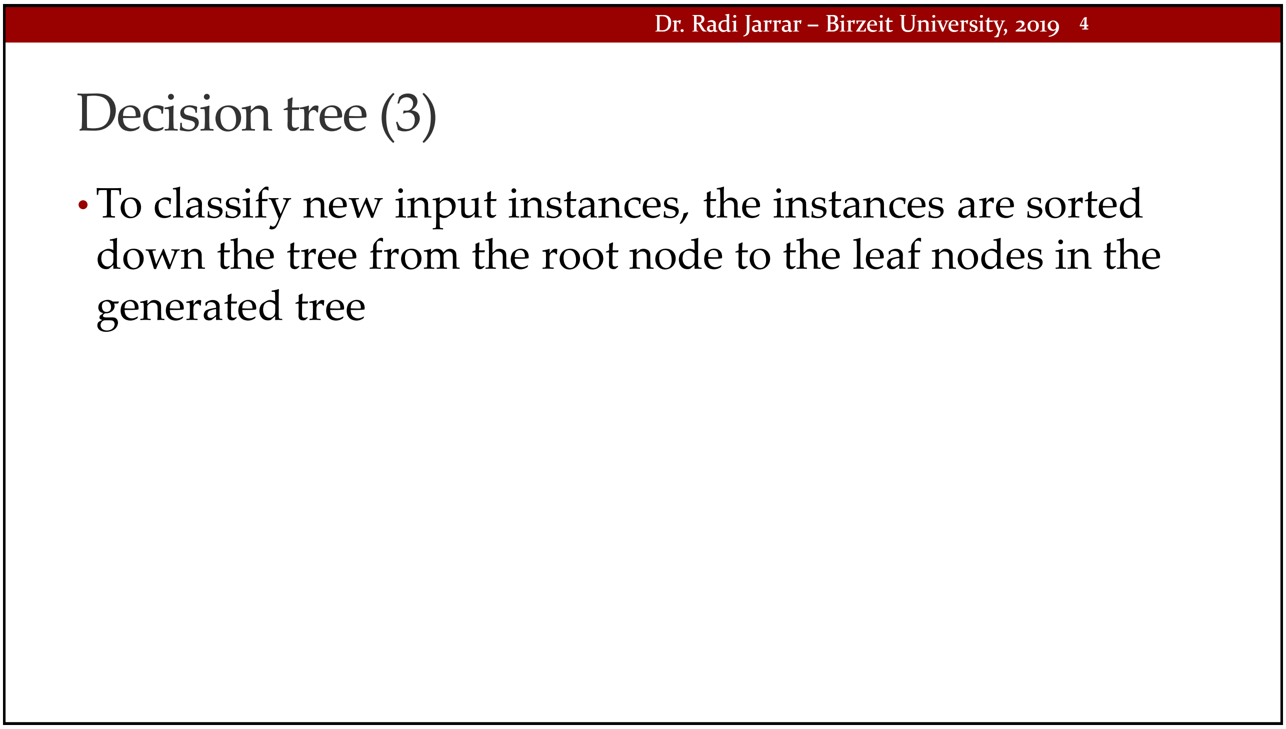
Decision tree

- Decision tree is used for classification and regression problems
- It builds a model in the form of tree structure
- Decision trees use a set of labelled training instances to form the classification hypothesis
- The learned rules are then formed in a tree structure starting with a single root node to some leaf nodes, each of which represents a classification result of an input instance



Decision tree (2)

- A rule based learning technique in which a learning system selects only a subset of the instance attributes when forming the classification hypothesis
- Its main advantages
  - it's hierarchical clarity
  - implementation simplicity
  - robustness to incomplete and noisy data
  - it can learn from a small number of training examples

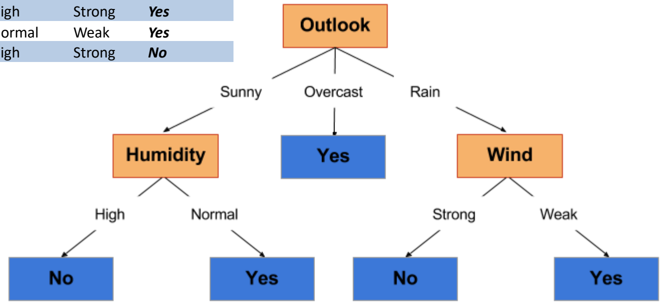


Decision tree (3)

- To classify new input instances, the instances are sorted down the tree from the root node to the leaf nodes in the generated tree

Decision tree (4)

Day	Outlook	Temprature	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***



Decision tree (5)

- The topmost node of the tree is called the **root node**
- The root node is the first test that is executed on the data to split up the data and form up the tree
- It corresponds to **the best predictor** on the data (later in this lecture)
- All internal nodes (except the leaf nodes) are called decision nodes

Decision tree (6)

- Decision nodes make some tests on attributes of the data (i.e., test one attribute X_i)
- The decision node may have multiple branches (depending on the test outcomes)
- Each branch from the internal nodes selects one value of X_i
- The leaf nodes represent the classification or regression (i.e., predict Y (or $P(Y|X \in leaf\_nodes)$)

Which attribute to start with?

Day	**Outlook**	**Temprature**	**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***

Informative features

- One of the important tasks is to select important (informative) feature to start the classification (or regression) with
- What it means to be informative?
  - *Information* is a quantity that reduces the uncertainty about something
- The better the information, the more uncertainty it reduces

Informative features (2)

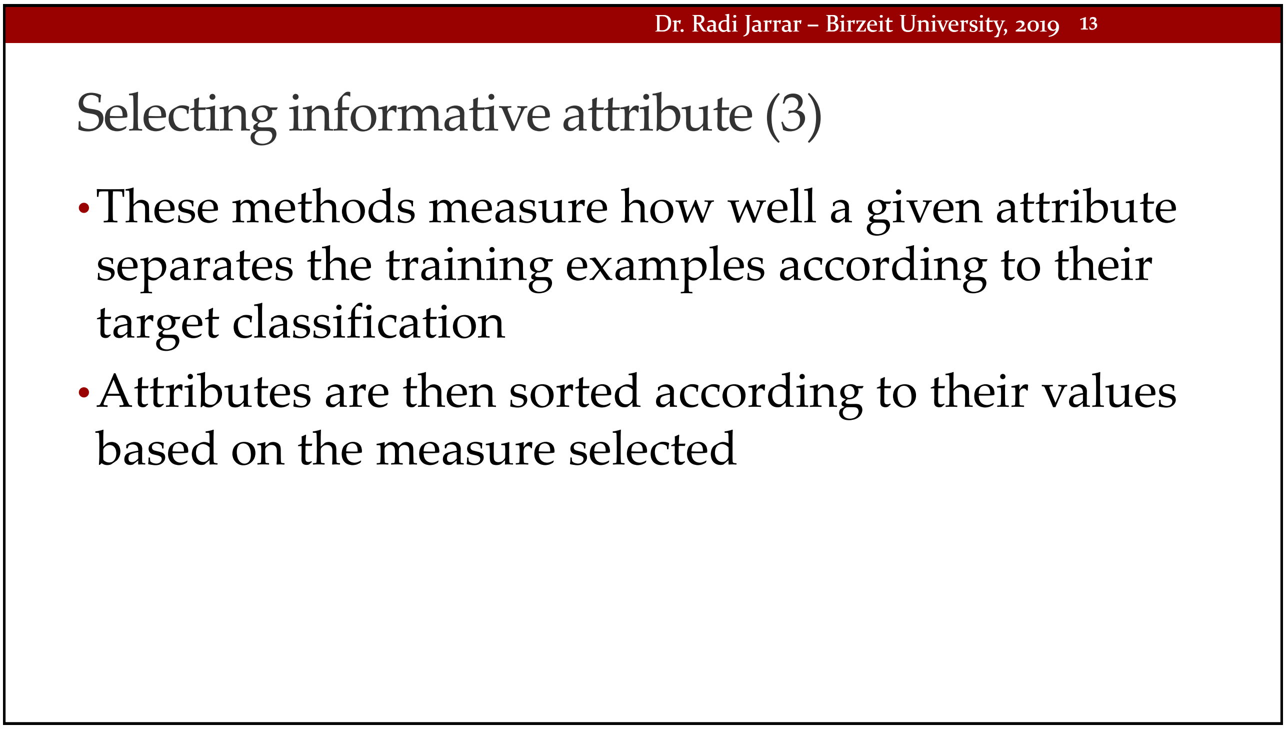
- Task: Is there one (or more) feature that reduces the uncertainty about the value of the target class?
- In other words, which feature correlates better with the target class label? (which, ultimately, reduces the uncertainty in it)
- **Finding informative attribute is the basis for the Decision Tree learning**

Selecting informative attribute

- Given the input data (i.e., the training dataset), how to select an attribute that partitions the data in an informative way?
- In other way, what is the best feature, in the set of input features, to be used first by the decision tree to make decision?

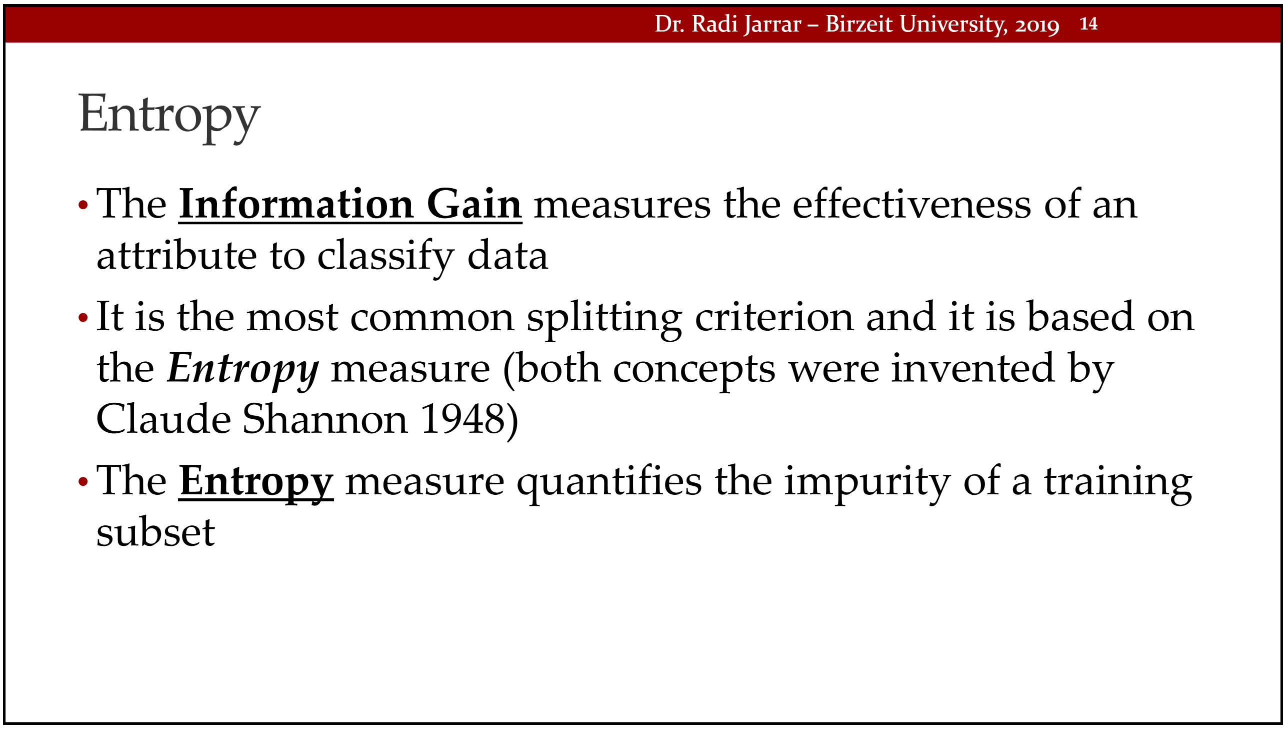
Selecting informative attribute (2)

- The attribute selected first at the root node should be the most significant and discriminant for classifying input instances
- Selecting the most suitable attribute is achieved by statistical methods such as **Information Gain** and **Gain Ratio**



Selecting informative attribute (3)

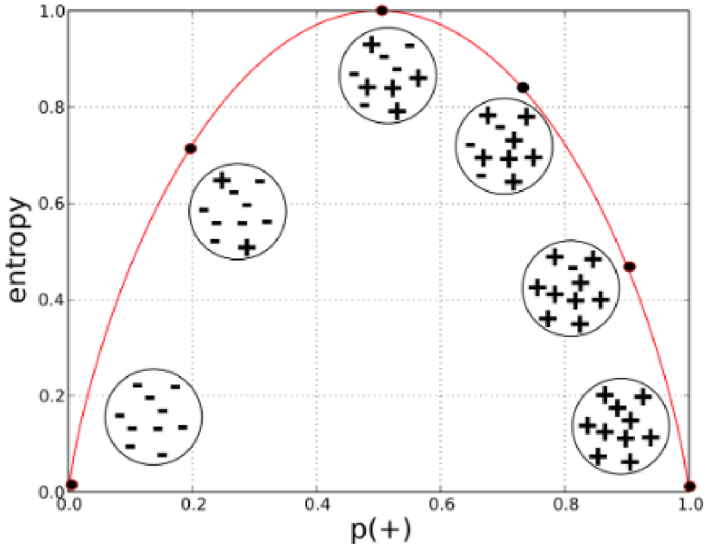
- These methods measure how well a given attribute separates the training examples according to their target classification
- Attributes are then sorted according to their values based on the measure selected



Entropy

- The **Information Gain** measures the effectiveness of an attribute to classify data
- It is the most common splitting criterion and it is based on the ***Entropy*** measure (both concepts were invented by Claude Shannon 1948)
- The **Entropy** measure quantifies the impurity of a training subset

Entropy (2)



Entropy (3)

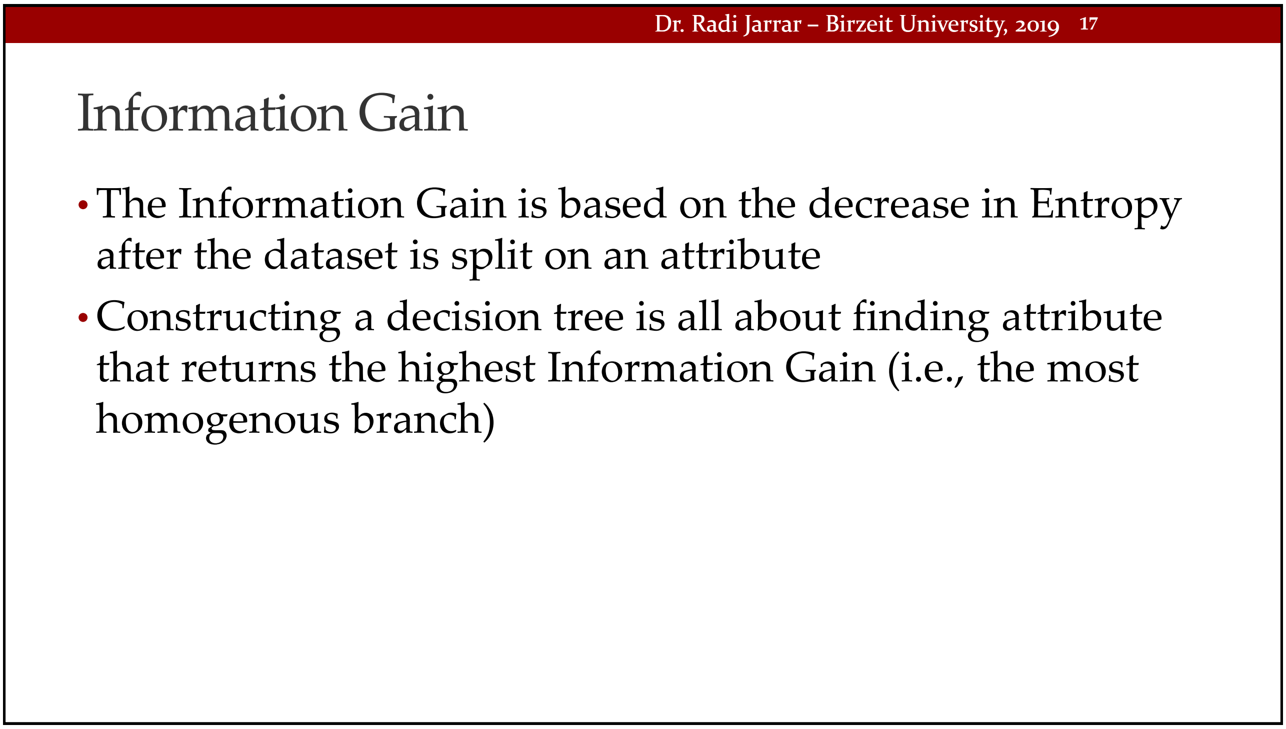
- The Entropy is measured as follows

$$E\left(D_{train}\right) = -\sum_{i=1}^{c} p_i \log_2\left(p_i\right)$$

where p_i is the proportion of D_{train} belonging to class i for all possible classes c

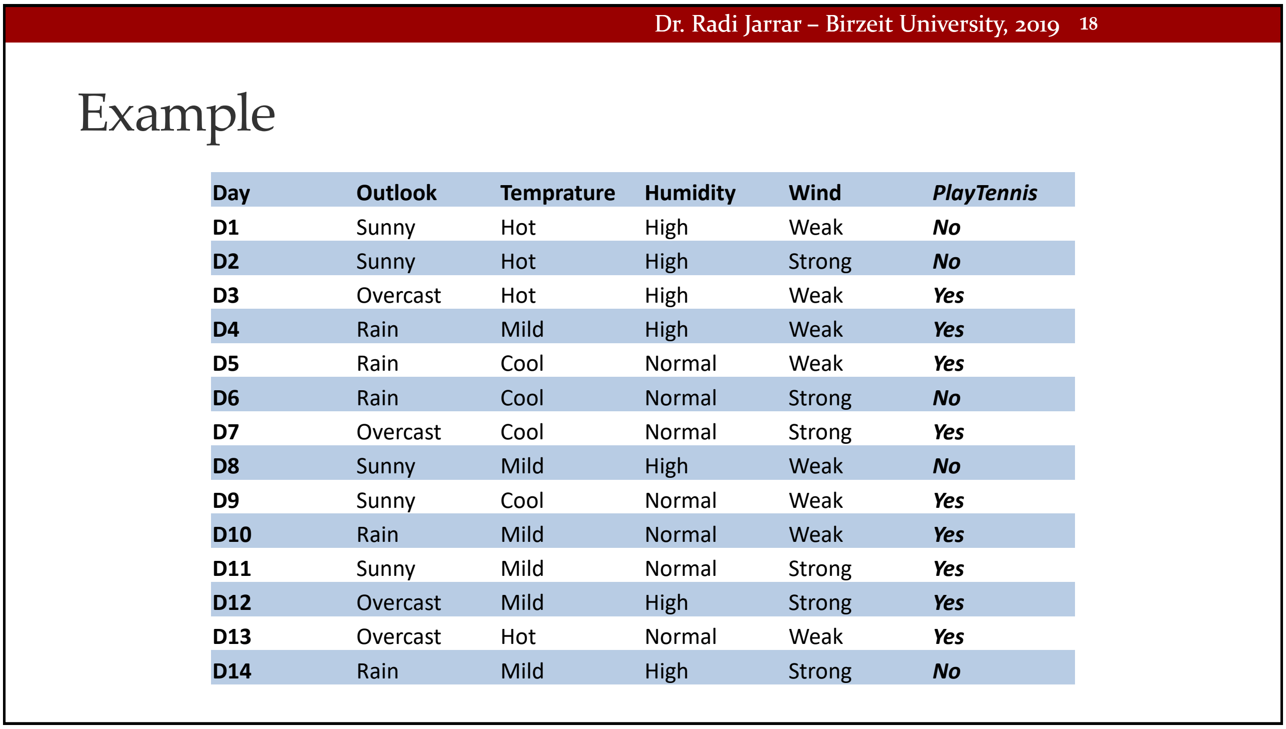
- Then the Information Gain is

$$Gain(S,A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S)$$



Information Gain

- The Information Gain is based on the decrease in Entropy after the dataset is split on an attribute
- Constructing a decision tree is all about finding attribute that returns the highest Information Gain (i.e., the most homogenous branch)



Example

Day	**Outlook**	**Temprature**	**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***

Example (2)

- To build the decision tree, calculate the Entropy using frequency tables

1) Find the Entropy of the target attribute (i.e., PlayTennis)

PlayTennis	
Yes	No
9	5

2) The dataset is split on different attributes. The entropy for each branch is calculated. The resulted entropy is subtracted from the entropy before the split. The result is the information gain (or decrease in entropy)

Example (3)

		PlayTennis		
		Yes	No	Total
Outlook	Sunny	3	2	5
	Overcast	4	0	4
	Rain	2	3	5
				14

		PlayTennis		
		Yes	No	Total
Temprature	Hot	2	2	4
	Mild	4	2	6
	Cool	3	1	4
				14

		PlayTennis		
		Yes	No	Total
Wind	Weak	6	2	8
	Strong	3	3	6
				14

		PlayTennis		
		Yes	No	Total
Humidity	High	3	4	7
	Normal	6	1	7
				14

Example (4)

3) The attribute with the largest Information Gain is selected as the decision node

- In this case, ‘Outlook’ attribute as the GI = 0.38

		PlayTennis		
		Yes	No	Total
Outlook	Sunny	3	2	5
	Overcast	4	0	4
	Rain	2	3	5
Gain Information = 0.38				14

Example (4)

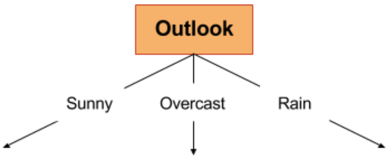
3) The attribute with the largest Information Gain is selected as the decision node

- In this case, ‘Outlook’ attribute as the GI = 0.38

- When the value of Outlook is ‘Overcast’, the entropy is 0

4) When the entropy is zero, a leaf node is created

		PlayTennis		
		Yes	No	Total
Outlook	Sunny	3	2	5
	Overcast	4	0	4
	Rain	2	3	5
Gain Information = 0.38				14



Outlook

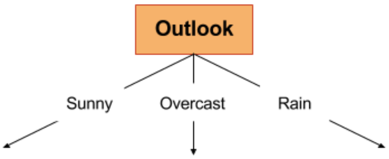
Sunny

Overcast

Rain

Example (5)

5) A branch with entropy more than zero needs further splitting



[D1, D2, D8, D9, D11]
[2+, 3-]

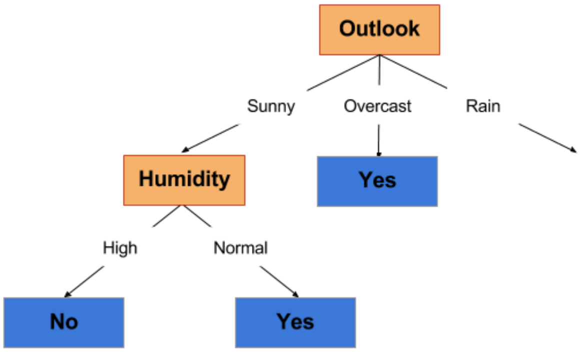
Which attribute shall be used (tested) next?

Subset - Sunny

Day	Outlook	Temprature	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***

Example (6)

After deciding on which attribute to be tested after Sunny



Outlook

Sunny

Overcast

Rain

Humidity

Yes

High

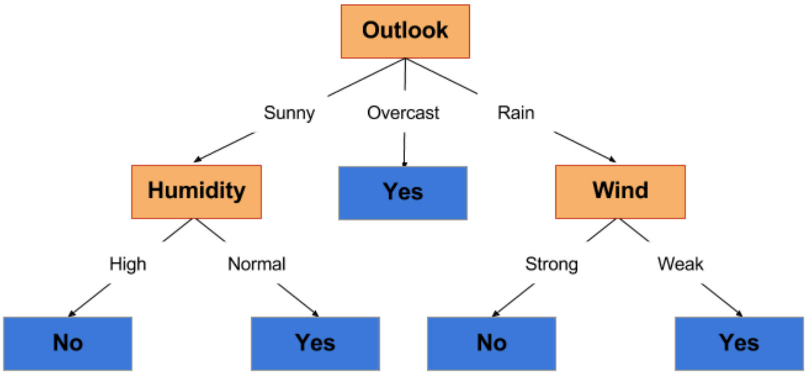
Normal

No

Yes

Example (7)

This algorithm is run recursively on the non-leaf branches until all data is classified



Outlook

Sunny

Overcast

Rain

Humidity

Yes

Wind

High

Normal

Strong

Weak

No

Yes

No

Yes

ID3

- The previous example of the decision tree is called ID3 Algorithm
- Similarly, another well-known top-down induction tree is C4.5 algorithm
- C4.5 (and C5.0) handle continuous values rather than only categorical values as in the ID3 decision tree

The algorithm

- Node = root
- Loop:
  - Decide on N as the best decision attribute for the next node
  - Assign N as decision attribute for node
  - For each value of N, create a new descendant of node
  - Sort the training examples to leaf nodes
  - If the training examples are sorted 'perfectly' then stop, else iterate over the new leaf node

Decision tree to decision rules

- A decision tree can be easily transformed to a set of decision rules by mapping from the root node to the leaf nodes one-by-one

if(outlook = sunny and humidity= normal)
    then PlayTennis = yes
if(outlook = sunny and humidity= high)
    then PlayTennis = no
if(outlook = overcast)
    then PlayTennis = yes
if(outlook = rain and wind = weak)
    then PlayTennis = yes
if(outlook = rain and wind = strong)
    then PlayTennis = no

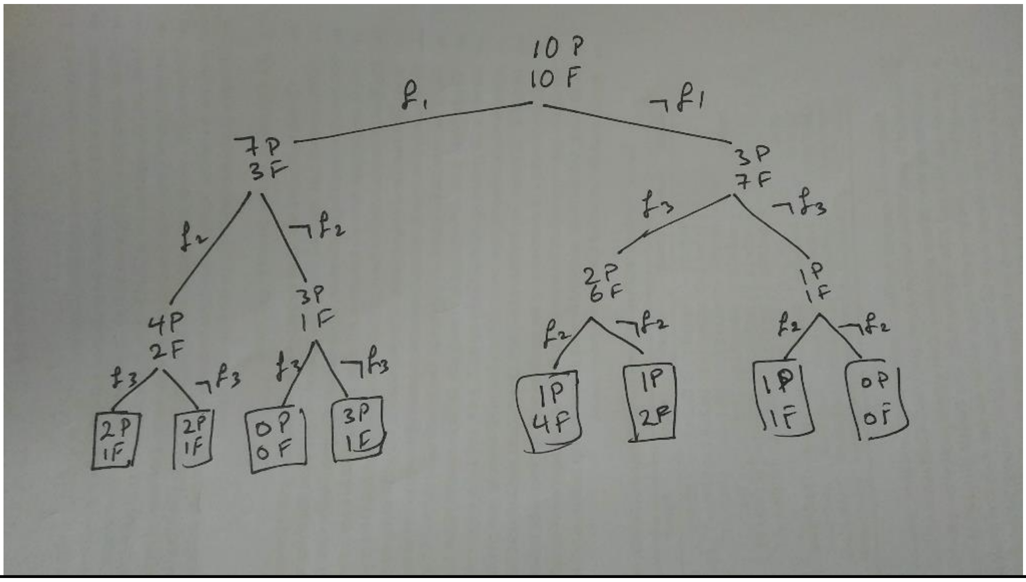
C5.0 Decision Tree algorithm

- Developed by J. Quinlan as an improved version of the previously developed tree, C4.5
- C4.5 was an improvement over the ID3 (Iterative Dichomotoriser 3)
- C5.0 has become an industry standard for producing decision trees

C5.0 Decision Tree algorithm (2)

- C5.0 uses Entropy to measure the purity of data and the Information Gain to decide on which attribute to split
- Accepts both discrete and continuous features
- Handles incomplete features (missing features)
- Solves overfitting by using a 'pruning' technique
- Weights can be applied to different features on the data

Tree pruning



Tree pruning (2)

- Branches reflect anomalies due to noise/outliers (overfitting)
- Pruning: is used to avoid overfitting (which makes the tree to specific to the training data) so it is more generalised to the testing data and more dynamic
- Statistical methods to remove the least-reliable branches
- Pruned trees are less complex, smaller in size, faster, and better at correctly classifying data

Tree pruning (3)

- Two types: pre-pruning and post-pruning
- Pre-pruning: pruning is achieved by halting the build process of the tree early during the construction (no further partitioning)
- When you get to a certain branch, you need to make the decision about whether to branch or not based on how much more accuracy (or predictive power) you will get if you take that route
- Upon halting, the node becomes a leaf and holds the most frequent class among the subset of instances or the probability distribution of those instances
- The decision whether to split or no is made by a specific threshold on the goodness of statistical measure (e.g., Information Gain or Gain Ratio)

Tree pruning (4)

- Post-pruning: pruning is achieved by removing a subtree from a fully grown tree
- A subtree branching at node A is removed and it becomes a node
- The leaf is labelled with the most frequent class in the subtree being replaced
- Considered more expensive as the tree is fully generated first, and then pruned
- More reliable as pre-pruning may remove a node that will have high information gain in later levels

Strengths of Decision Trees

- Does well on most problems
- Uses only the most important features
- Automatic learning process can handle numeric or nominal features and missing data
- The output model (the classifier) can be interpreted without a mathematical background

Weaknesses of Decision Trees

- Easy to overfit or underfit
- Might have troubles in modelling some relationships between attributes due to reliance on axis-parallel splits
- Minor changes in the training data results in big changes in the structure of the tree
- With larger datasets, larger trees are formed which are much harder to interpret