

# Decision Trees

---

- Decision trees are a simple hierarchically structured way to guide one's path to a decision.
- Decision tree learning is one of the most widely used techniques for classification.
  - Its classification accuracy is competitive with other methods, and
  - it is very efficient.
- The classification model is a tree, called decision tree.

# Decision Trees Algorithm

---

- Employs the divide and conquer method
  - Recursively divides a training set until each division consists of examples from one class
1. Create a root node and assign all of the training data to it
  2. Select **the best splitting** attribute
  3. Add a branch to the root node for each value of the split. **Split the data into mutually exclusive** subsets along the lines of the specific split
  4. Repeat the steps 2 and 3 for each and every leaf node until the **stopping criteria** is reached

# Decision Trees Algorithm

---

- Decision Tree algorithms mainly differ on
  - Splitting criteria
    - Which variable to split first? – Information Gain
    - What values to use to split?
    - How many splits to form for each node?
  - Stopping criteria
    - When to stop building the tree – Max tolerable error
  - Pruning (generalization method)
    - Pre-pruning versus post-pruning

## Exercise: Decision tree to Predict 'Play'

---

Outlook	Temp	Humidity	Windy	Play
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

Question: How to Select the best splitting attribute

Outlook	Temp	Humidity	Windy	Play
Sunny	Hot	Normal	True	??

# Evaluating the weather attributes

Outlook	Temp	Humidity	Windy	Play
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

<u>Attribute</u>	<u>Rules</u>	<u>Error</u>	<u>Total Error</u>
Outlook	Sunny→No	2/5	

# Evaluating the weather attributes

Outlook	Temp	Humidity	Windy	Play
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

<u>Attribute</u>	<u>Rules</u>	<u>Erro</u> <u>r</u>	<u>Total</u> <u>Error</u>
Outlook	Sunny→No	2/5	
	Overcast →yes	0/4	

# Evaluating the weather attributes

Outlook	Temp	Humidity	Windy	Play
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

<u>Attribute</u>	<u>Rules</u>	<u>Error</u>	<u>Total Error</u>
Outlook	Sunny → No	2/5	4/14
	Overcast → yes	0/4	
	Rainy → yes	2/5	

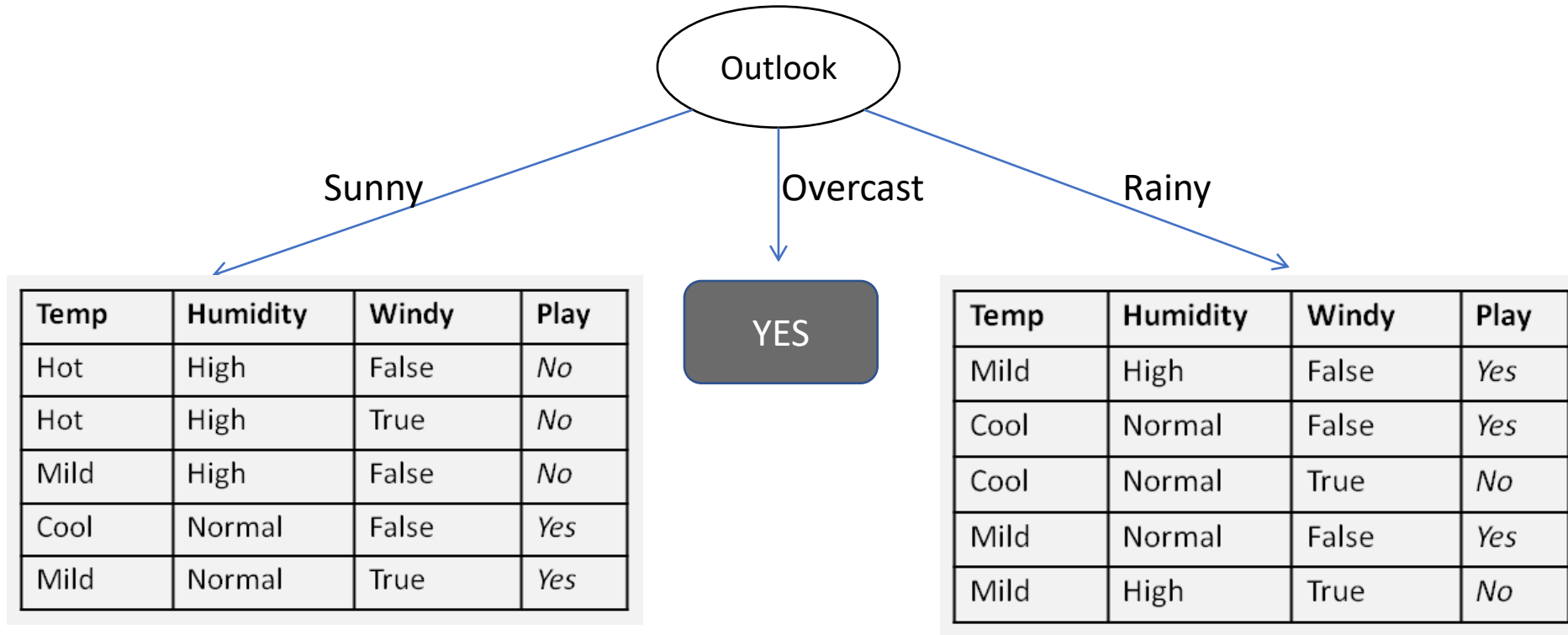
# Evaluating the weather attributes

Outlook	Temp	Humidity	Windy	Play
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

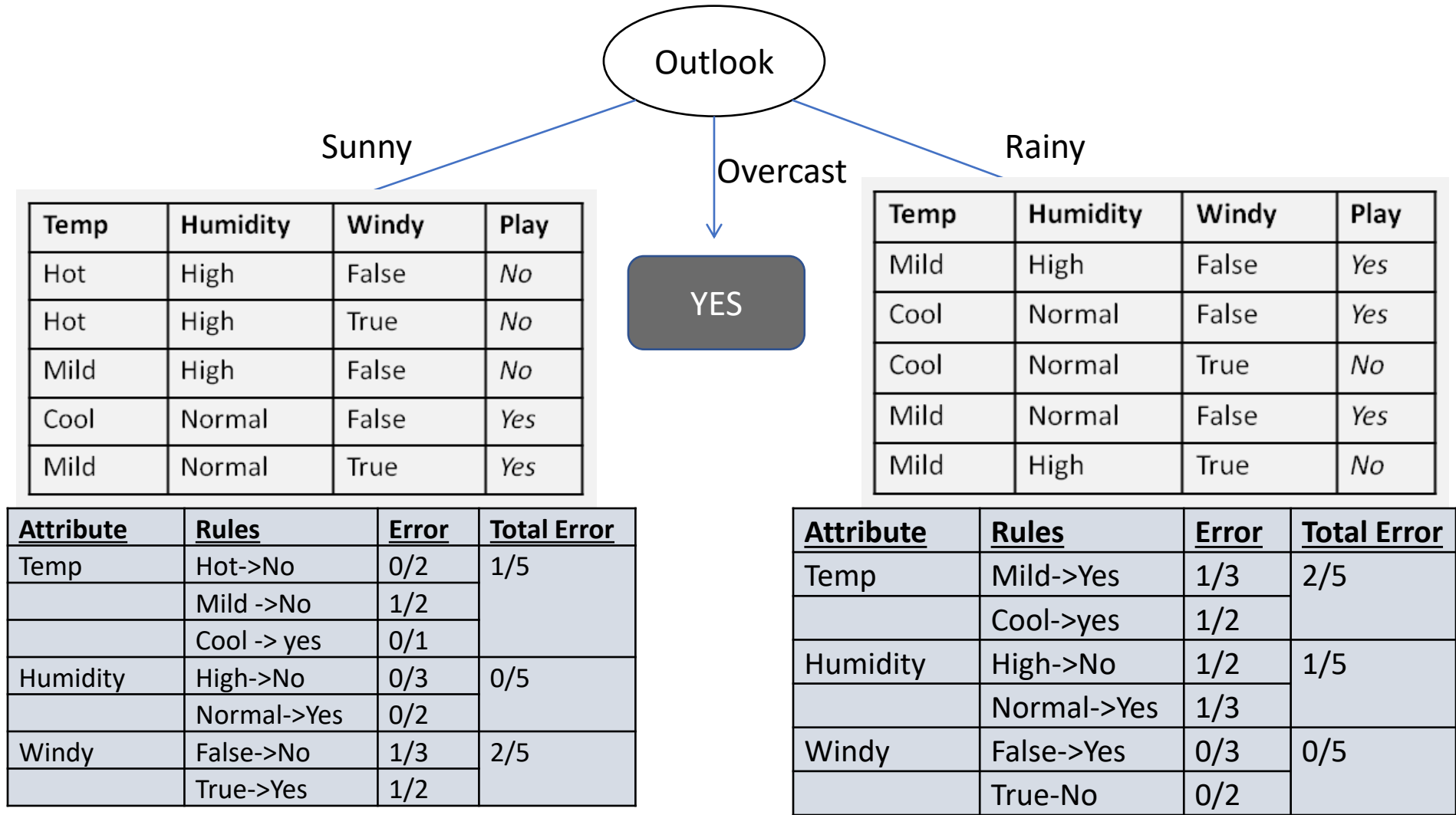
Attribute	Rules	Error	Total Error
Outlook	Sunny → No	2/5	4/14
	Overcast → yes	0/4	
	Rainy → yes	2/5	
Temp	Hot → No	2/4	5/14
	Mild → Yes	2/6	
	Cool → Yes	1/4	
Humidity	High → No	3/7	4/14
	Normal → Yes	1/7	
Windy	False → Yes	2/8	5/14
	True → No	3/6	



# Decision tree after Iteration 1 (for weather/play problem)

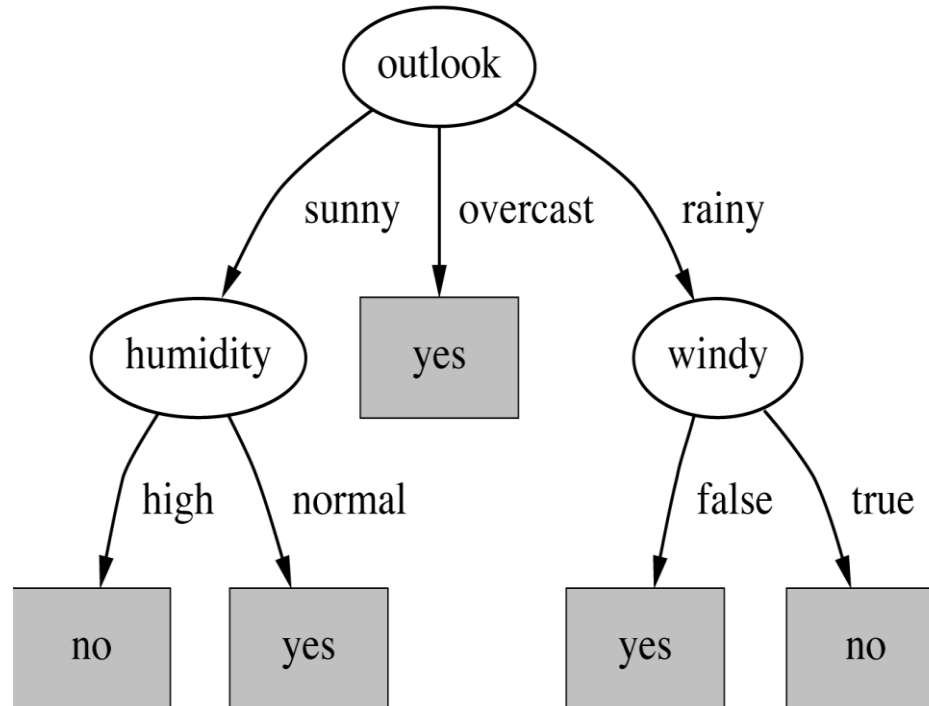


# Decision tree after Iteration 1 (for weather/play problem)



# Decision tree

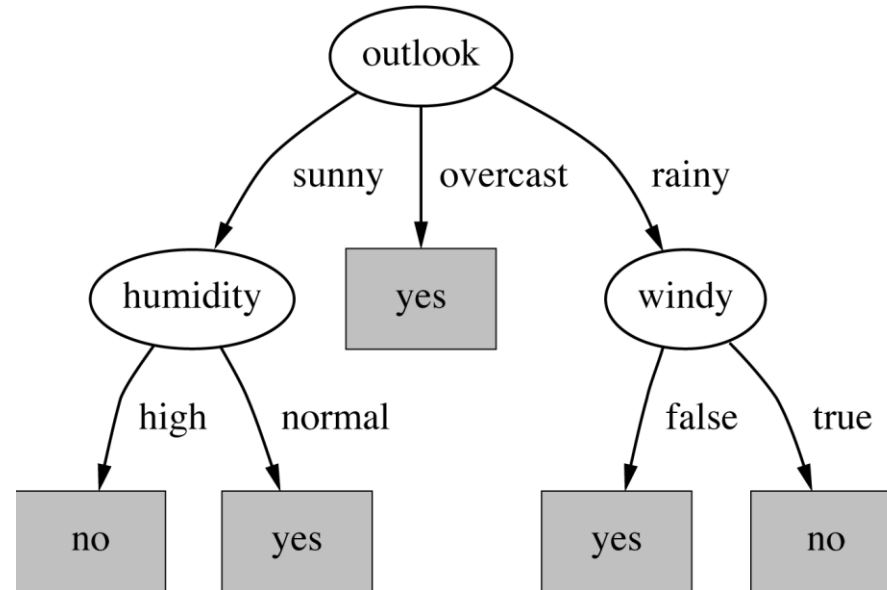
(for weather/play problem)



Outlook	Temp	Humidity	Wind	Play
Sunny	Hot	Normal	True	YES

Predict using the model

# Decision tree (for weather/play problem)



- Not all leaves need to be pure; sometimes identical instances have different class.
- Splitting stops when data can't be split any further

# Decision Tree vs Table Lookup

---

	<b>Decision Tree</b>	<b>Table Lookup</b>
Accuracy	Varied level of accuracy	100% accurate
Generality	General. Applies to all situations	Applies only when a similar case occurred before
Frugality	Only three variables needed	All four variables are needed
Simple	Only one or two questions asked	All four variable values are needed
Easy	Logical, and easy to understand	Can be cumbersome to look up; no understanding of the logic behind the decision

## Decision Trees (Part 2)

---

Now that we have an intuitive ideas how a decision tree is constructed. Let's focus on more precisely how to create a tree.

Remember the important question in tree construction is how to pick which attributes to split the tree on. This brings up the concept of information gain and entropy

<https://towardsdatascience.com/decision-tree-in-python-b433ae57fb93>

<https://towardsdatascience.com/enchanted-random-forest-b08d418cb411#.hh7n1co54>

## Decision Trees (Part 2)

---

### Model Parameters:

- `Max_depth` : maximum depth of the trees
- `Criterion`: default is “gini”, other choice is “entropy”

```
model = DecisionTreeClassifier(max_depth=3, criterion='entropy')  
model = DecisionTreeClassifier(max_depth=3, criterion='gini')
```

# Entropy and Information Gain

---

## High Entropy

- Mixed cases = Heterogenous  
Example: 50% boy + 50% girls

## Low Entropy

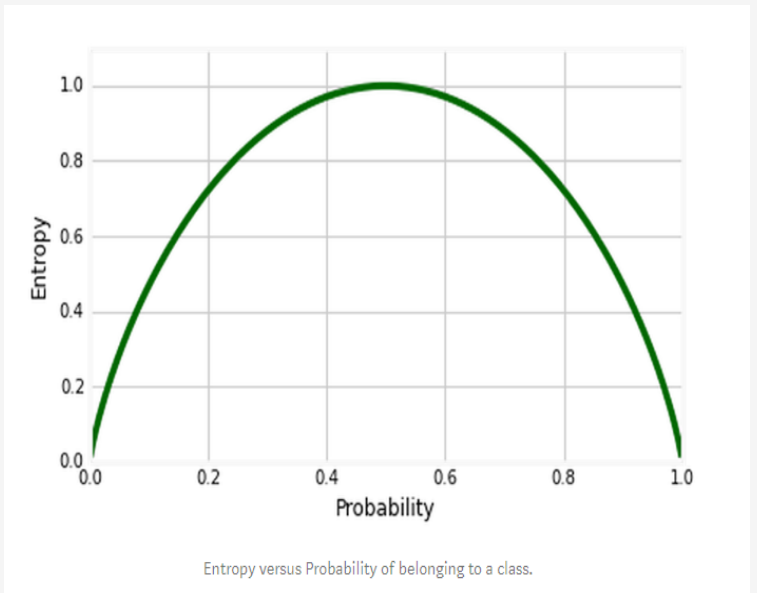
- Pure cases, homogenous
- Example: 90% boy + 10% girls
- or 10% boy + 90% girls

Information Gain from  
splitting a dataset  $S$  into  
different partition  $V$

$$Gain(S, D) = H(S) - \sum_{V \in D} \frac{|V|}{|S|} H(V)$$

Entropy formula:

$$H = - \sum p(x) \log p(x)$$



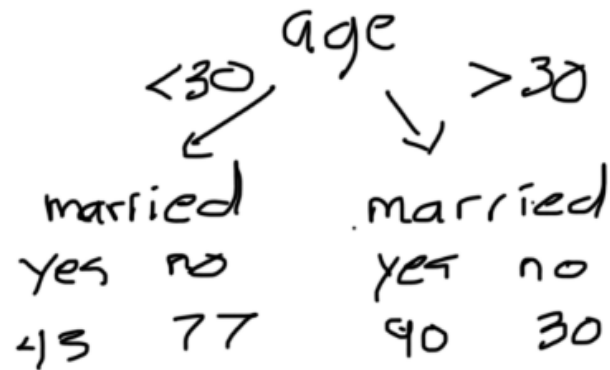


Gini Index:  $G = \sum_{i=1}^C p(i) * (1 - p(i))$

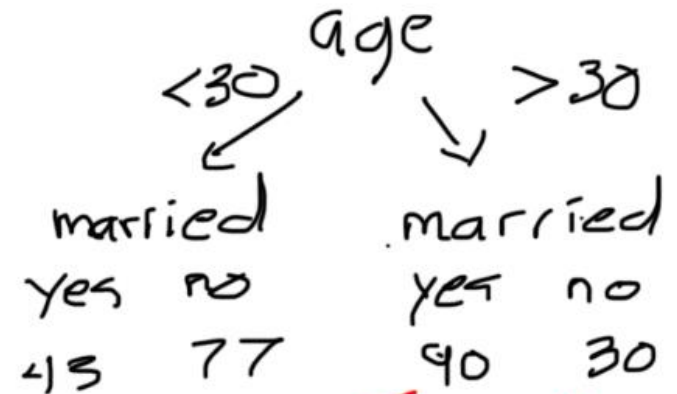
$$G = \sum_{i=1}^C p(i) * (1 - p(i)) = \sum_{i=1}^C p(i) - p^2(i) = 1 - \sum_{i=1}^C p^2(i)$$

For pure class: When  $P(i)$  is 1 or 0,  $G = 0$

For mix class: When  $P(i) = 0.5$ ,  $G = 0.5$



$$\begin{aligned} \text{left Impurity} &= 1 - \left(\frac{43}{43+77}\right)^2 - \left(\frac{77}{43+77}\right)^2 \\ &= 0.460 \end{aligned}$$



$$\begin{aligned} \text{right Impurity} &= 1 - \left(\frac{90}{90+30}\right)^2 - \left(\frac{30}{90+30}\right)^2 \\ &= 0.375 \end{aligned}$$

## Gini Index

Information gain is the difference in impurity before and after the split

Diagram illustrating a decision tree split on the attribute **age**:

```
graph TD
    age -->|<30| node1
    age -->|>30| node2
    node1 -->|married| node1_1
    node1 -->|not married| node1_2
    node2 -->|married| node2_1
    node2 -->|not married| node2_2
    node1_1 --> 45
    node1_2 --> 77
    node2_1 --> 90
    node2_2 --> 30
```

The leaf node values (45, 77, 90, 30) are circled in red.

$$\text{Impurity} = 1 - \left(\frac{120}{240}\right)^2 - \left(\frac{120}{240}\right)^2 = 0.5$$

$$\text{Gain}(S, D) = H(S) - \sum_{V \in D} \frac{|V|}{|S|} H(V)$$

$$\begin{aligned} \text{Information gain} &= 0.5 - \left(\frac{120}{240}\right)(0.375) - \left(\frac{120}{240}\right)(0.460) \\ &= 0.0825 \end{aligned}$$

# Decision Tree Algorithms

---

Decision-Tree	C4.5	CART	CHAID
Full Name	Iterative Dichotomiser (ID3)	Classification and Regression Trees	Chi-square Automatic Interaction Detector
Basic algorithm	Hunt's algorithm	Hunt's algorithm	adjusted significance testing
Developer	Ross Quinlan	Bremman	Gordon Kass
When developed	1986	1984	1980
Types of trees	Classification	Classification & Regression trees	Classification & regression
Serial implementation	Tree-growth & Tree-pruning	Tree-growth & Tree-pruning	Tree-growth & Tree-pruning
Type of data	Discrete & Continuous; Incomplete data	Discrete and Continuous	Non-normal data also accepted
Types of splits	Multi-way splits	Binary splits only; Clever surrogate splits to reduce tree depth	Multi-way splits as default
Splitting criteria	Information gain	Gini's coefficient, and others	<i>Chi</i> -square test
Pruning Criteria	Clever bottom-up technique avoids overfitting		Trees can become very large
Implementation	Publicly available	Publicly available in most packages	Popular in market research, for segmentation

# Random Forests

---

- Repeatedly select data from the data set with replacement and build a separate tree with each new training set. Each of these trees built will be used to make new forecast. The class label that receive the most votes becomes the predicted class for that data point
- Each tree may be a “weak” classifier and is subject to overfitting from the specific training sample dataset. However, by building not just one tree, but multiple trees for different training samples, the hope is that the combined forecast from individual “weak” classifiers may become a “strong” classifier
- This is the basic idea behind the “Ensemble methods”, in which we combine multiple machine learning algorithms to obtain better predictive performance. We’ll run multiple models on the data and use the aggregate predictions, which will be better than a single model alone.

# Decision Trees

---

Learning by doing