Decision Trees:

Hunt's Algorithm (greedy) - recursive algorithm used to build decision trees for classification problems

1.) At each step:
   - → splits the dataset based on the value of an attribute (feature)
   - → recursively repeats this process on each subset (branch of decision tree) until it reaches a base case

2.) Base cases:
   1.) If after a split, all the data in that branch are of the same class
      - → we've found a data, where the outcome is always the same → predict that outcome/class
   2.) If after a split, there are no examples left
      - → predict a reasonable default
         - → maybe most common class at the parent node
         - → some prior probability

Many ways to split a given attribute:
- → Binary split
- → Multi-way split

Continous Variables - How do we handle continous variables using continous trees?

1.) Using binning before running the decision tree
   - → binning is converting continous values into discrete intervals
   - → If age is continous, bin it into categories like "18-25," "26-35," etc.
   - Instead of manually defining the bin ranges, we can use a clustering algorithm like K-means to find natural groupings in the data

2.) Compute a threshold while building the tree:
   - → Instead of binning ahead of time, we can determine the optimal threshold during training
   - → For example, for a variable A, the tree may decide that splitting on $A > t$ vs. $A < t$

We want to split the data at a node that results in "purer" nodes → nodes that mostly contain data from a single class.

→ **GINI Index:** measures how pure a node is

gini index at a node t:

$$Gini(t) = 1 - \sum_j p(j \mid t)^2$$

← GINI of a node t

↳ relative frequency of class j at node t

A lower gini value means that the node is more pure → we choose the lower gini value split

each question we ask to split the data creates new nodes (data partitions)

$1 - (1/8)^2 - (1/7)^2 = 0.219$

$1 - (1/5)^2 - (4/5)^2 = 0.32$

$1 - (3/3)^2 - (0^2)$
$1 - 1 - 0 = 0$

$1 - (2/3)^2 - (1/3)^2 - \cos^2$
$= 0.44$

- → the gini index will always be 0 when the node is, that is, when all data points in the node belong to the same class
   - → when the node contains data from only one class.
   - → all zeros and 1 contribution

GINI of the split:

$$GINI_{split} = \sum_{t=1}^{k} \frac{n_t}{n} GINI(t)$$

- n_t: number of data points at node t (in next child node)
- $GINI(t)$: gini of node t
- n: number of data points before the split (parent node)

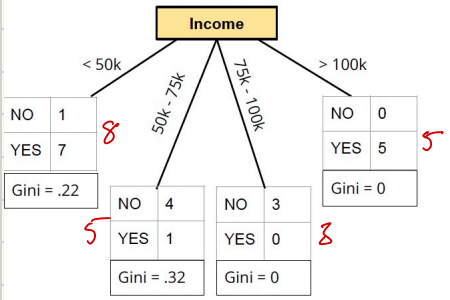
GINI(t) measures how pure a single node is

$GINI_{split}$ evaluates the split with multiple child nodes

weighted as not all child nodes are equal!

GINI gain = $gini_{\text{before splitting}} - gini_{split}$

GINI of the split



$$GINI_{split} = \sum_{t=1}^{k} \frac{n_t}{n}(GINI(t))$$

n = 21

$$= \sum_{t=1}^{k} \left(\frac{n_t}{n}\left(1 - \sum_{j} p(j|t)^2\right)\right)$$

$GINI_{split}$ = .22 * 8/21
+ .32 * 5/21
+ 0 * 3/21
+ 0 * 5/21
= .16

Limitations: overfitting, where we reach a point where all of the leaf nodes classify only one data point with gini = 0 (perfectly pure)
→ but doesn't generalize well, because the tree memorized the training data

1.) Early Termination
- → Stop tree growth before it becomes too specific
  - → stop at a fixed depth
  - → stop if a node size is too small
  - → stop if gini doesn't improve significantly
- → at this point, use majority vote → assign the most common class in that group (at the leaf node)

2.) Pruning – we can let the tree grow fully and then cut / prune the branches