NAYANA DAVIS

CLASSIFICATICATION AND REGRESSION TREES (CARTS)

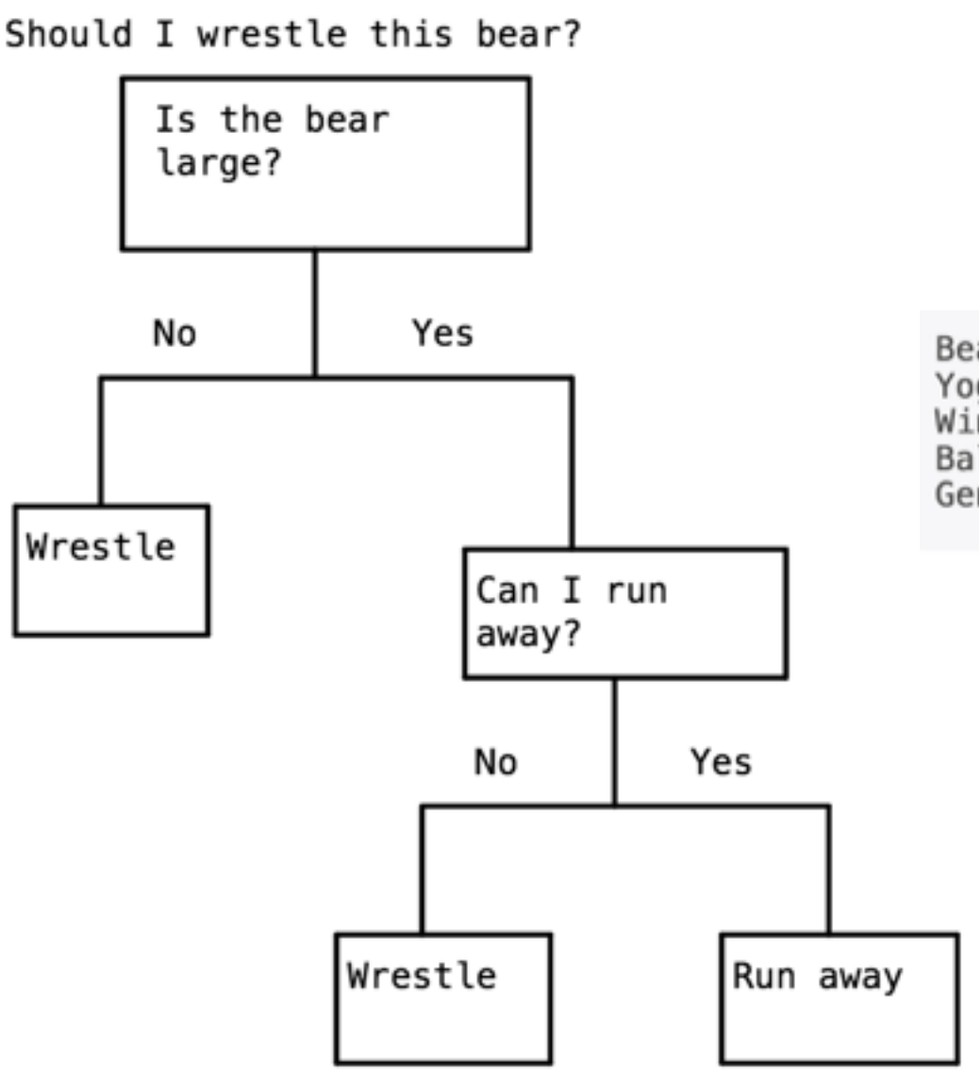
LEARNING OBJECTIVES

- Describe what a decision tree is
- Explain how a classification tree works
- Explain how a regression tree works

WHAT IS A DECISION TREE?

- Powerful machine learning technique that tells us what outcomes we should predict in certain situations
  - Supervised learning algorithm—we first construct the tree with historical data, and then use it to predict an outcome
- Can be used for classification or regression problems

SHOULD I WRESTLE THIS BEAR?



Bear name	Size	Escape possible?	Action
Yogi	Small	No	Wrestle
Winnie	Small	Yes	Wrestle
Baloo	Large	Yes	Run away
Gentle Ben	Large	No	Wrestle

WHAT IS A DECISION TREE?

Decision trees are a non-parametric hierarchical classification technique.

- Non-parametric methods stand in contrast to models like logistic regression or ordinary least squares regression. There are no underlying assumptions about the distribution of the data or the errors. Non-parametric models essentially start with no assumed parameters about the data and construct them based on the observed data.
- Hierarchical means that the model is defined by a sequence of questions which yield a class label or value when applied to any observation. Once trained, the model behaves like a recipe, a series of "if this then that" conditions that yields a specific result for our input data.

DECISION TREE STRUCTURE

- Decision trees constructed through a series of nodes and edges
- Nodes are where we split the data based on a variable (question)
- An edge is one side of the split. A possible answer to a given question
- Decision trees work by splitting data based on variables

YOU DO: WORK IN PAIRS, DRAW OUT DECISION TREE

- 5 minutes

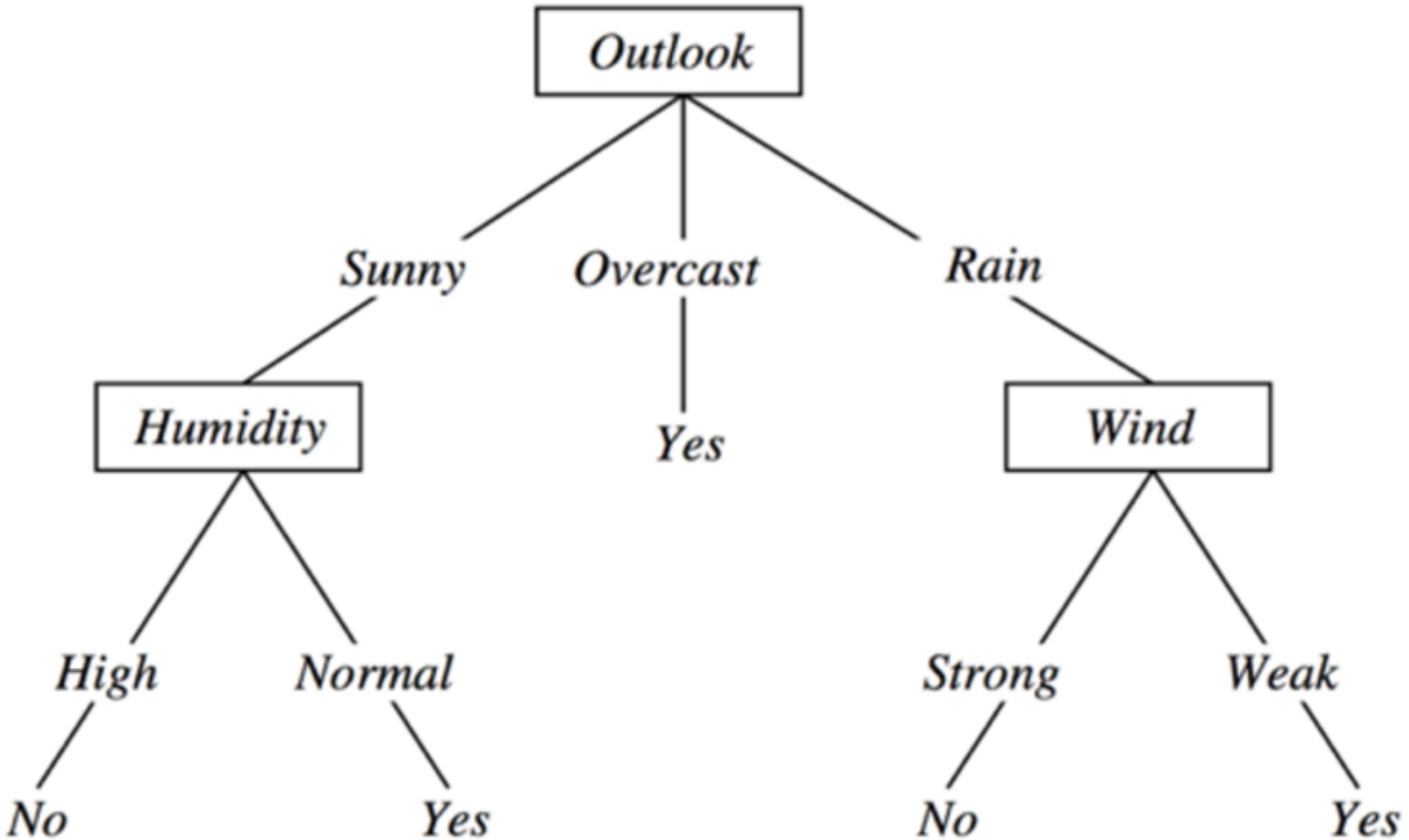
DIRECTED ACYCLIC GRAPHS (DAG)

CART models are a case of what's known as a Directed Acyclic Graph. DAGs have nodes and edges.

The Acyclic part means the edges don't cycle back onto themselves

- The top node is called the root node
- Internal nodes test a condition on a specific feature
- A leaf node contains a class label (or regression value)

SHOULD I PLAY GOLF?



BUILDING A DECISION TREE

Building decision trees requires algorithms capable of determining an optimal choice at each node.

One such algorithm is Hunt's algorithm. This is a greedy, recursive algorithm that leads to a local optimum.

HUNT'S ALGORITHM

- Greedy: the algorithm makes the most optimal decision it can at each step.
- Recursive: the algorithm splits task into subtasks and solves each the same way.
- Local optimum: the algorithm finds a solution just for the given neighborhood of points.

HUNT'S ALGORITHM

The algorithm works by recursively partitioning records into smaller and smaller subsets. The partitioning decision is made at each node according to a metric called purity. A node is said to be 100% pure when all of its records belong to a single class (or have the same value).

HUNT'S ALGORITHM

Let Dt be the set of training records that reach a node t. The general recursive procedure is defined as below:

- If Dt contains records that belong the same class yt, then t is a leaf node labeled as yt
- If Dt is an empty set, then t is a leaf node labeled by the default class, yd
- If Dt contains records that belong to more than one class, use an attribute test to split the data into smaller subsets.

PSEUDOCODE CLASSIFICATION DECISION TREE ALGORITHM

Given a set of records Dt at node t:

If all records in Dt belong to class A:

t is a leaf node corresponding to class (Base case)

Else if Dt contains records from both A and B:

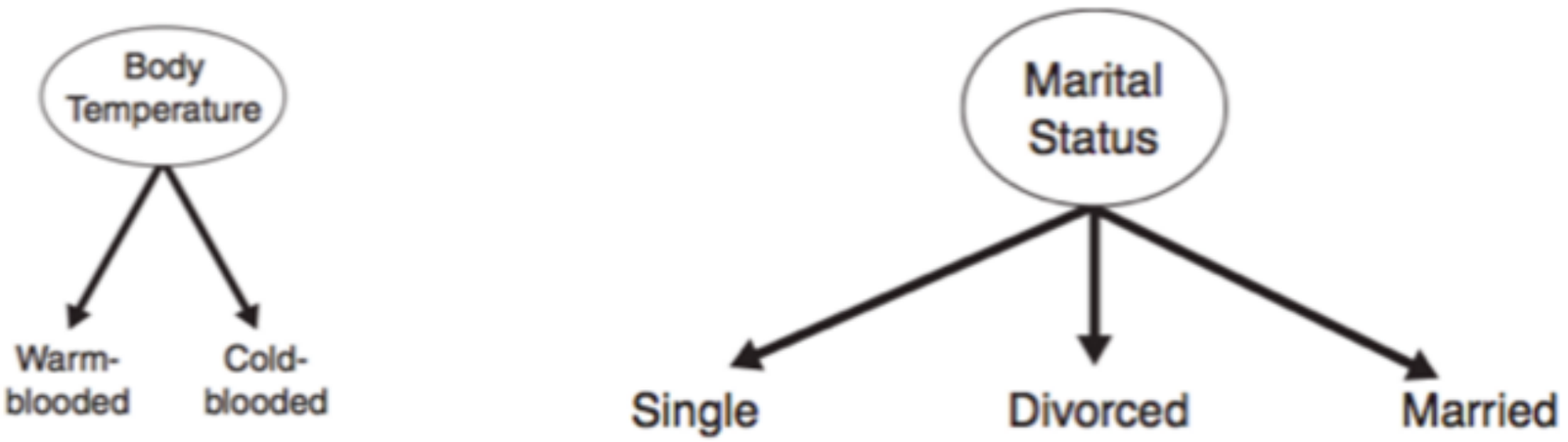
Create test condition to partition the observations

Define t as an internal node, with outgoing edges to child nodes

partition records in Dt with conditional test logic to child nodes

Recursively apply steps at each child node.

SPLITS CAN BE BINARY WAY OR MULTI-WAY



(a) Multiway split

Figure 4.8. Test condition for binary attributes.

FEATURES CAN BE CATEGORICAL OR CONTINUOUS

Continuous measure decisions (regression trees)

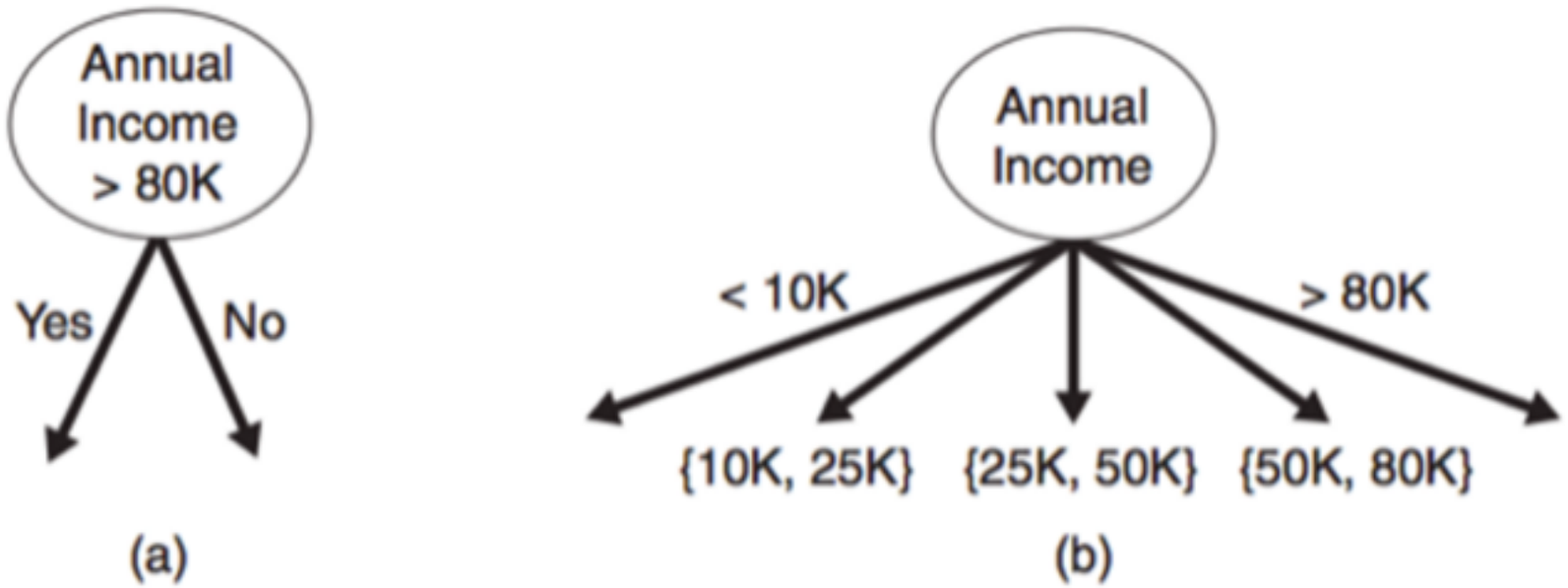


Figure 4.11. Test condition for continuous attributes.

OPTIMIZATION AND "PURITY"

Recall from the algorithm we iteratively create test conditions to split the data.

- A maximum impurity partition is given by the distribution (classification), where both classes are presented equally

$$p(0|t) = p(1|t) = 0.5|$$

OPTIMIZATION AND "PURITY"

- Maximum purity is obtained when only one class is present, i.e:

$$p(0|t) = 1 - p(1|t) = 1|$$

PURITY OBJECTIVE FUNCTION

To achieve maximum purity we need an objective function to optimize.

We want our objective function to measure the gain in purity from a particular split. Therefore it depends on the class distribution over the nodes (before and after the split).

For example, let

$p(i|t)$

be the probability of class i at node t (e.g., the fraction of records labeled i at node t)

We then define an impurity function that will smoothly vary between the two extreme cases of minimum impurity (one class or the other only) and the maximum impurity case as an equal mix.

COMMON PURITY FUNCTIONS

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

$$\text{Gini}(t) = 1 - \sum_{i=0}^{c-1} [p(i|t)]^2$$

$$\text{Classification error}(t) = 1 - \max_i [p(i|t)]$$

GAIN

Impurity measures on their own they are not enough to tell us how a split will do. We need to look at impurity before & after the split. We can make this comparison using what is called the gain. Where I is the impurity measure, Nj denotes the number of records at child node j, and N denotes the number of records at the parent node. When I is the entropy function, this quantity is called the information gain.

$$\Delta = I(\text{parent}) - \sum_{\text{children}} \frac{N_j}{N} I(\text{child}_j)$$

OVERFITTING

Overfitting occurs when a statistical model describes random error or noise instead of the underlying relationship. Overfitting generally occurs when a model is excessively complex, such as having too many parameters relative to the number of observations. A model that has been overfit will generally have poor predictive performance, as it can exaggerate minor fluctuations in the data.

OVERFITTING

- Subtree replacement: One possible way to prevent overfitting is pre-pruning, which involves setting a minimum threshold on the gain, and stopping when no split achieves a gain above this threshold. It is difficult to calibrate in practice.

- Subtree raising: Alternatively we can build the full tree and then perform pruning as a post-processing step. To prune a tree, the nodes are examined from the bottom-up and pieces of the tree are simplified according to some criteria. Complicated subtrees can be replaced either with a single node or with a simpler (child) subtree.

CART ADVANTAGES

-Simple to understand and interpret. People are able to understand decision tree models after a brief explanation.

—Useful to work with non technical departments (marketing/sales).

-Requires little data preparation.

—Other techniques often require data normalization, dummy variables need to be created and blank values to be removed.

-Able to handle both numerical and categorical data.

—Other techniques are usually specialized in analyzing datasets that have only one type of variable.

-Uses a white box model.

—If a given situation is observable in a model the explanation for the condition is easily explained by boolean logic.

—By contrast, in a black box model, the explanation for the results is typically difficult to understand.

-Possible to validate a model using statistical tests. That makes it possible to account for the reliability of the model.

-Robust. Performs well even if its assumptions are somewhat violated by the true model from which the data were generated.

-Performs well with large datasets. Large amounts of data can be analyzed using standard computing resources in reasonable time.

-Once trained can be implemented on hardware and has extremely fast execution.

-Real-time applications like trading, for example.

TEXT

CART DISADVANTAGES

-Locally-optimal.

—Practical decision-tree learning algorithms are based on heuristics such as the greedy algorithm where locally-optimal decisions are made at each node.

—Such algorithms cannot guarantee to return the globally-optimal decision tree.

-Overfitting.

—Decision-tree learners can create over-complex trees that do not generalize well from the training data.

-There are concepts that are hard to learn because decision trees do not express them easily. In such cases, the decision tree becomes prohibitively large.

-Decision tree learners create biased trees if some classes dominate. It is therefore recommended to balance the dataset prior to fitting with the decision tree.