18-661 Introduction to Machine Learning

Decision Trees

Spring 2022

ECE – Carnegie Mellon University

Course Logistics

- Recitation on Friday will review material on SVMs and decision trees (today's lecture) and is designed to help you with HW 4.
  - Kigali recitations will be held on campus, in person in room E305 at 4:10pm local time.
  - Pittsburgh recitations will continue to be recorded and posted to Canvas but will no longer be offered to remote students.
  - SV recitations will be broadcast to the classroom (B23 118) at the usual 1:40pm PT time.
- One HW 4 problem asks about AdaBoost, which will be covered in Monday's lecture. Next week's recitation will discuss boosting in more detail and may be helpful for solving this problem.
- Your final exam has been scheduled for Friday, May 6 at 12-3pm ET. Logistics will be announced later, but please let me know by April 22 if you have any conflicts necessitating a makeup exam (e.g., another final exam scheduled at that time).

Outline

1. Review: Nearest Neighbors
2. Decision Trees: Motivation
3. Learning A Decision Tree
4. Practical Considerations for Decision Trees

Review: Nearest Neighbors

Parametric vs. Nonparametric models

Key difference:

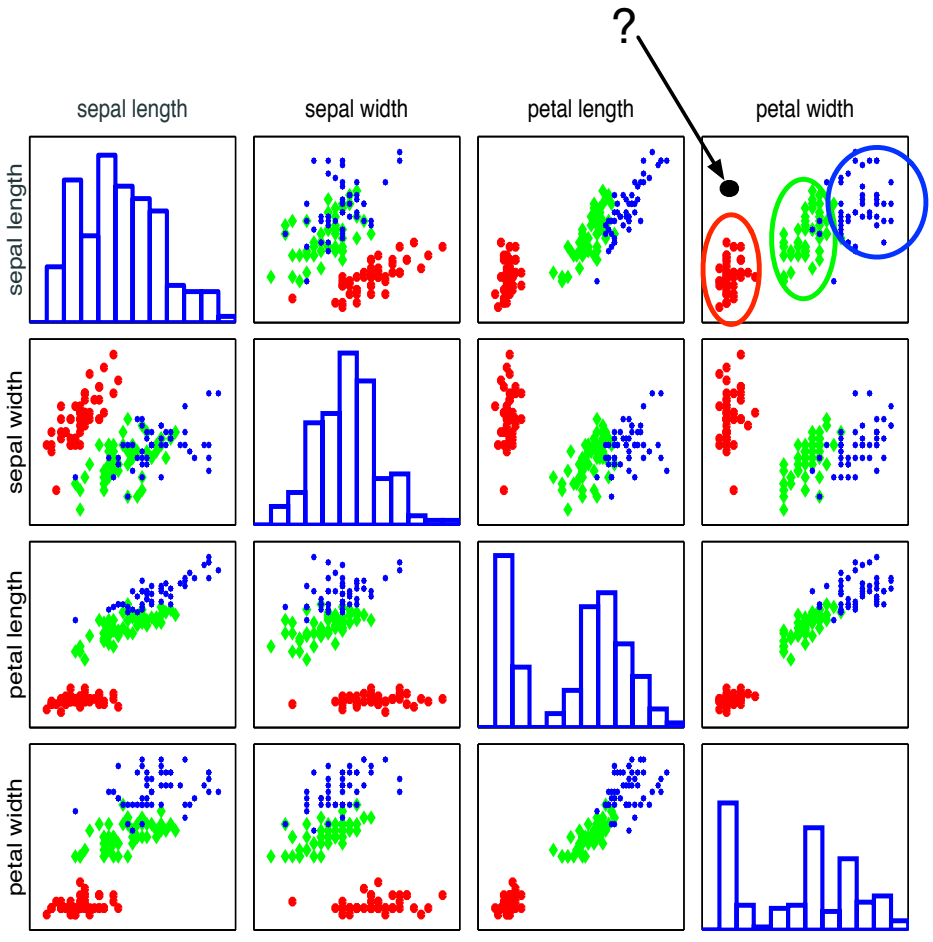
- Parametric models assume that the data can be characterized via some *fixed* set of parameters θ. Given this set of parameters, our future predictions are independent of the data $\mathcal{D}$, i.e., $P(x|\theta, \mathcal{D}) = P(x|\theta)$.
  - Often simpler and faster to learn, but can sometimes be a poor fit
- Nonparametric models instead assume that the model features depend on the data $\mathcal{D}$. The number of features tends to grow with the size of the dataset.
  - More complex and expensive, but can learn more flexible patterns
- Both parametric and nonparametric methods can be used for either regression or classification.

Recognizing Flowers

Types of Iris: setosa, versicolor, and virginica



Labeling an Unknown Flower Type



Closer to red cluster: so labeling it as setosa

Nearest Neighbor Classification (NNC)

Nearest neighbor of a (training or test) data point

$$\boldsymbol{x}(1) = \boldsymbol{x}_{\mathrm{nn}(\boldsymbol{x})}$$

where $\mathrm{nn}(\boldsymbol{x}) \in [\mathrm{N}] = \{1, 2, \cdots, \mathrm{N}\}$, i.e., the index to one of the training instances

$$\mathrm{nn}(\boldsymbol{x}) = \mathrm{argmin}_{n \in [\mathrm{N}]} \|\boldsymbol{x} - \boldsymbol{x}_n\|_2^2 = \mathrm{argmin}_{n \in [\mathrm{N}]} \sum_{d=1}^{\mathrm{D}} (x_d - x_{nd})^2$$

Classification rule

$$y = f(\boldsymbol{x}) = y_{\mathrm{nn}(\boldsymbol{x})}$$

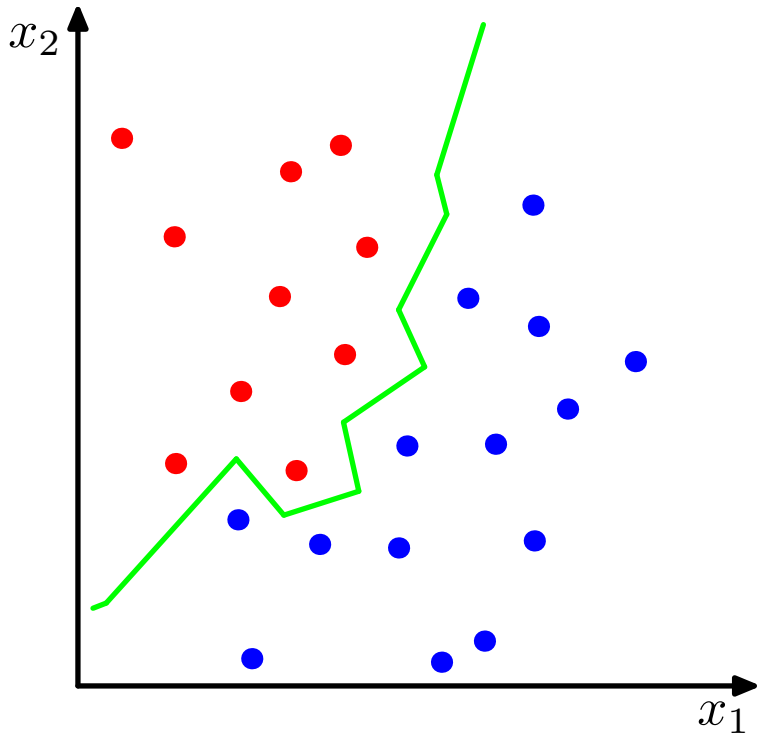
Example: if $\mathrm{nn}(\boldsymbol{x}) = 2$, then

$$y_{\mathrm{nn}(\boldsymbol{x})} = y_2,$$

which is the label of the 2nd data point.

Decision Boundary

For every point in the space, we can determine its label using the NNC rule. This gives rise to a *decision boundary* that partitions the space into different regions.



(b)

Nearest Neighbors for Regression

Recall the nearest neighbor of a (training or test) data point:

$$\boldsymbol{x}(1) = \boldsymbol{x}_{\text{nn}(\boldsymbol{x})}$$

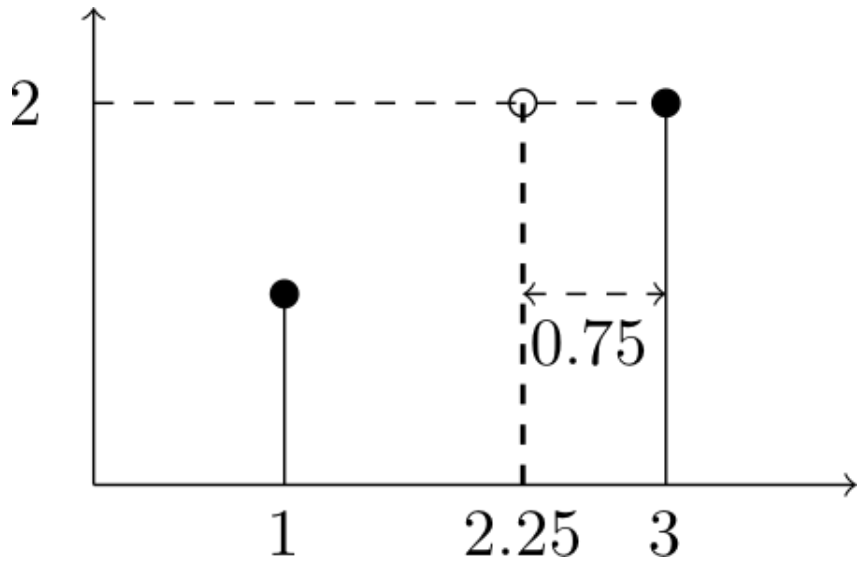
where $\text{nn}(\boldsymbol{x}) \in [\text{N}] = \{1, 2, \cdots, \text{N}\}$ indexes a training instance

$$\text{nn}(\boldsymbol{x}) = \text{argmin}_{n \in [\text{N}]} \|\boldsymbol{x} - \boldsymbol{x}_n\|_2^2 = \text{argmin}_{n \in [\text{N}]} \sum_{d=1}^{\text{D}} (x_d - x_{nd})^2$$

Regression rule

$$y = f(\boldsymbol{x}) = y_{\text{nn}(\boldsymbol{x})}$$

Label **x** with the label of its nearest neighbor!



K-Nearest Neighbor (KNN) Classification

Increase the number of nearest neighbors to use?

- 1st-nearest neighbor: $\text{nn}_1(\boldsymbol{x}) = \text{argmin}_{n \in [\text{N}]} \|\boldsymbol{x} - \boldsymbol{x}_n\|_2^2$
- 2nd-nearest neighbor: $\text{nn}_2(\boldsymbol{x}) = \text{argmin}_{n \in [\text{N}] - \text{nn}_1(\boldsymbol{x})} \|\boldsymbol{x} - \boldsymbol{x}_n\|_2^2$
- 3rd-nearest neighbor: $\text{nn}_3(\boldsymbol{x}) = \text{argmin}_{n \in [\text{N}] - \text{nn}_1(\boldsymbol{x}) - \text{nn}_2(\boldsymbol{x})} \|\boldsymbol{x} - \boldsymbol{x}_n\|_2^2$

The set of K-nearest neighbors

$$\text{knn}(\boldsymbol{x}) = \{\text{nn}_1(\boldsymbol{x}), \text{nn}_2(\boldsymbol{x}), \cdots, \text{nn}_K(\boldsymbol{x})\}$$

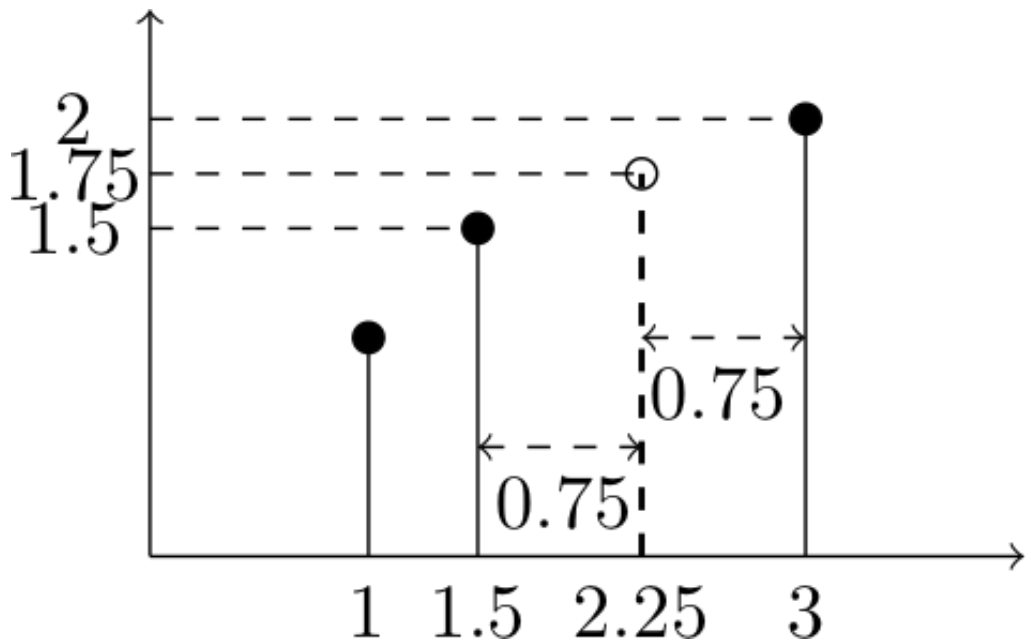
Let $\boldsymbol{x}(k) = \boldsymbol{x}_{\text{nn}_k(\boldsymbol{x})}$, then

$$\|\boldsymbol{x} - \boldsymbol{x}(1)\|_2^2 \leq \|\boldsymbol{x} - \boldsymbol{x}(2)\|_2^2 \cdots \leq \|\boldsymbol{x} - \boldsymbol{x}(K)\|_2^2$$

How to Do Regression with K Neighbors?

- We need a way to aggregate labels from each of the neighbors.
- Average the labels associated with the K points.

$$\hat{y} = \frac{1}{K} \sum_{n \in knn(\mathbf{x})} y_n$$



How to Classify with K Neighbors?

Classification rule

- Every neighbor votes: suppose y_n (the true label) for $\boldsymbol{x}_n$ is c, then
  - vote for c is 1
  - vote for $c' \neq c$ is 0

  We use the *indicator function* $\mathbb{I}(y_n == c)$ to represent the votes.
- Aggregate everyone's vote

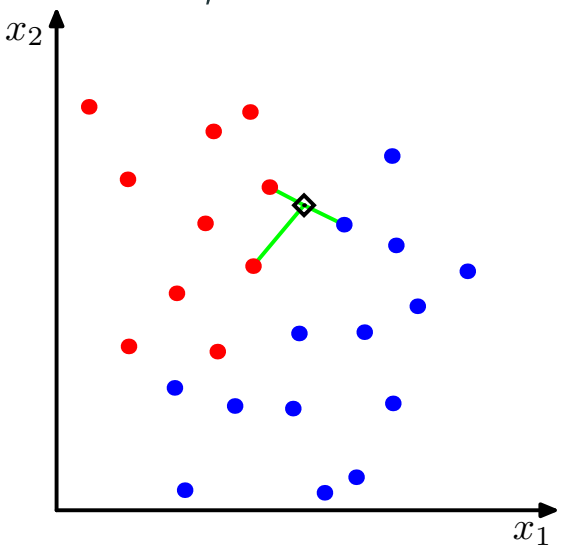
$$v_c = \sum_{n \in \text{knn}(\boldsymbol{x})} \mathbb{I}(y_n == c), \quad \forall \quad c \in [\text{C}]$$

- Label with the majority, breaking ties arbitrarily

$$y = f(\boldsymbol{x}) = \arg\max_{c \in [\text{C}]} v_c$$

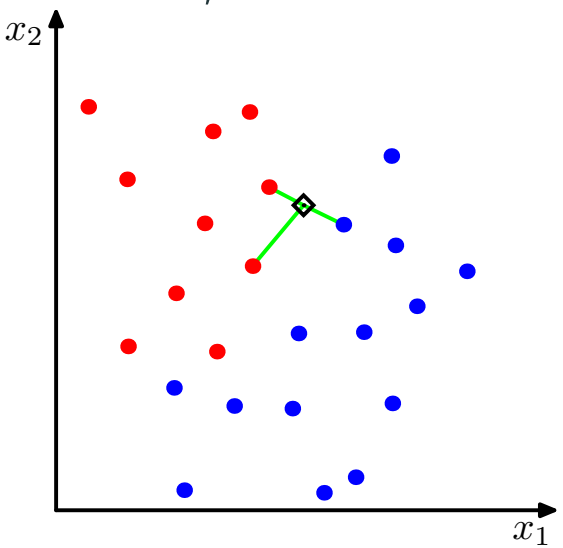
Example

K=1, Label: red



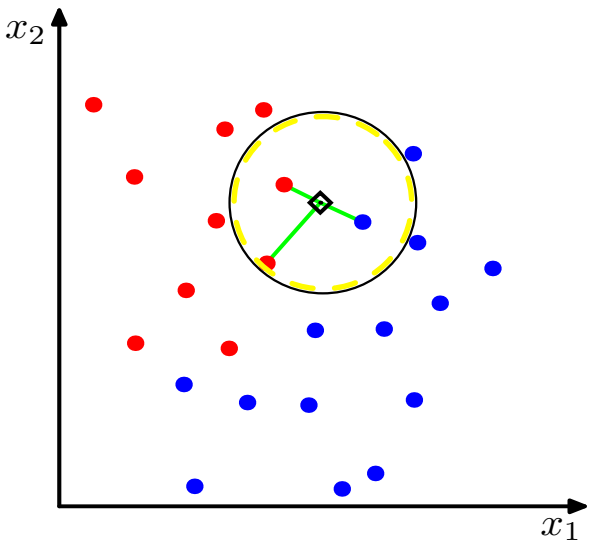
(a)

K=3, Label: red



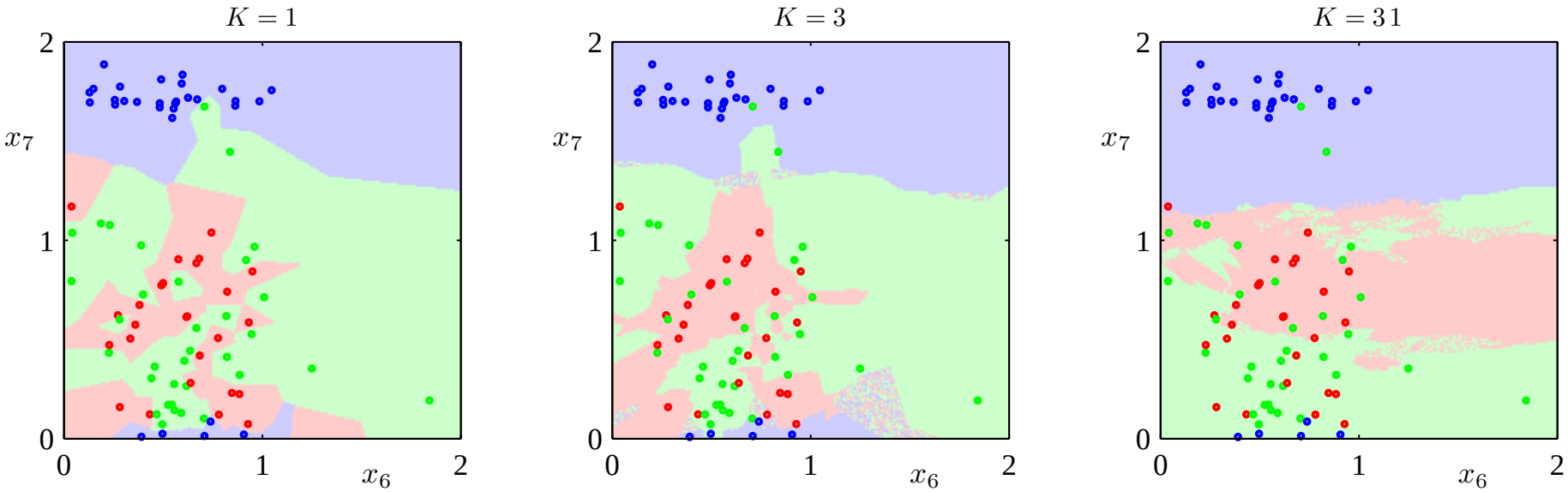
(a)

K=5, Label: blue



(a)

How to Choose an Optimal K?



When K increases, the decision boundary becomes smooth. K is equivalent to a regularizer.

Hyperparameters in NN

Two crucial choices for NN

- Choosing K, i.e., the number of nearest neighbors (default is 1)
- Choosing the right distance measure (default is Euclidean distance), for example, from the following generalized distance measure

$$\|\boldsymbol{x} - \boldsymbol{x}_n\|_p = \left(\sum_d |x_d - x_{nd}|^p \right)^{1/p}$$

for $p \geq 1$.

These are not specified by the algorithm itself — resolving them requires empirical studies and are task/dataset-specific. In practice, we often use validation datasets to choose K.

Preprocess Data

Normalize data to have zero mean and unit standard deviation in each dimension

- Compute the means and standard deviations in each feature

$$\bar{x}_d = \frac{1}{N} \sum_n x_{nd}, \qquad s_d^2 = \frac{1}{N-1} \sum_n (x_{nd} - \bar{x}_d)^2$$

- Scale the feature accordingly

$$x_{nd} \leftarrow \frac{x_{nd} - \bar{x}_d}{s_d}$$

Many other ways of normalizing data — you would need/want to try different ones and pick among them using (cross) validation.

Why Use Nearest Neighbors?

Advantages of NNC

- Computationally, simple and easy to implement – just compute distances, no optimization required
- Can learn complex decision boundaries

Disadvantages of NNC

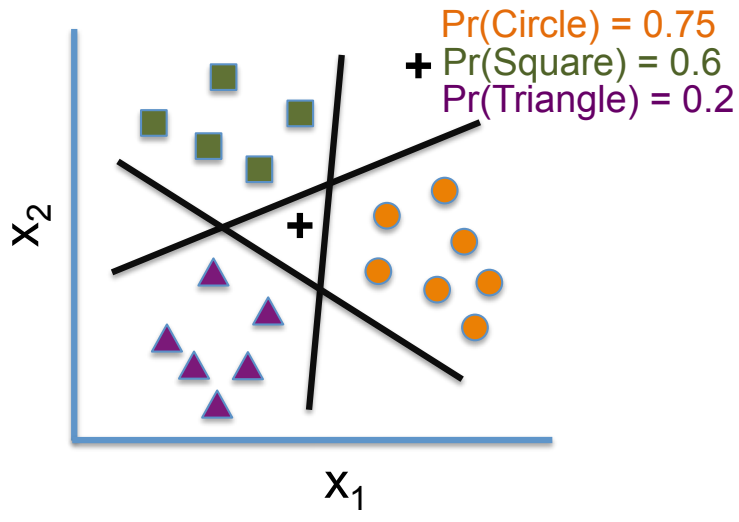
- Computationally intensive for large-scale problems: $O(\mathrm{N}D)$ for labeling a data point.
- We need to "carry" the training data around. Without it, we cannot do classification. This type of method is called *nonparametric*.
- Choosing the right distance measure and K can be difficult.

When Should We Not Use Nearest Neighbors?

- Relies on the existence of training data points "close" to test points. Choose your distance wisely!
- Can break down if the classes are unbalanced.
- Can break down if we have irrelevant features. Solution–feature selection!
    - Essentially, we want to measure the "information" contributed by each feature.
    - More on this in the next lecture on decision trees.

Decision Trees: Motivation

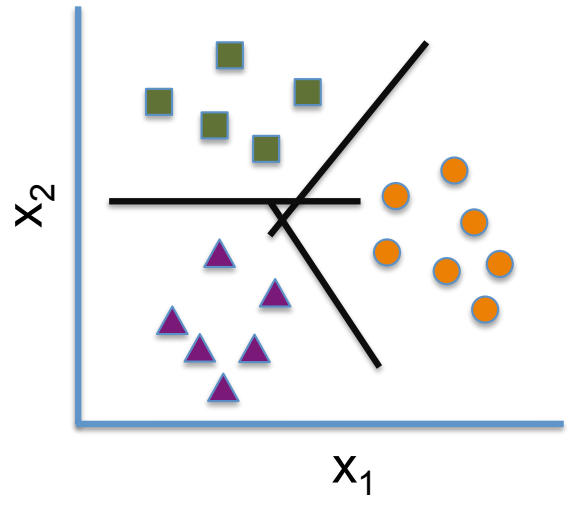
Recall: Multi-class Classification



We combined binary decision boundaries to partition the feature space

- One-versus-all approach

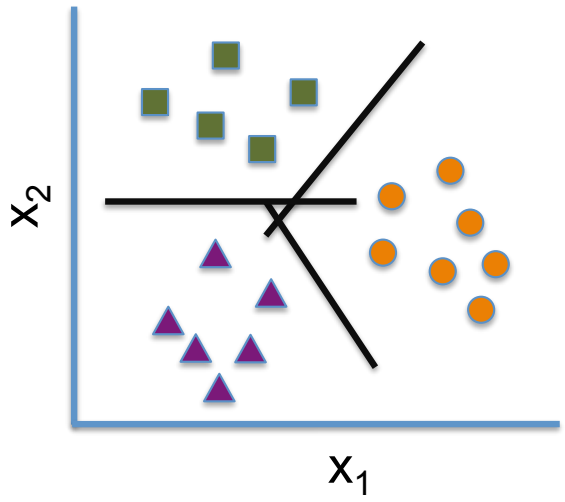
Recall: Multi-class Classification



We combined binary decision boundaries to partition the feature space

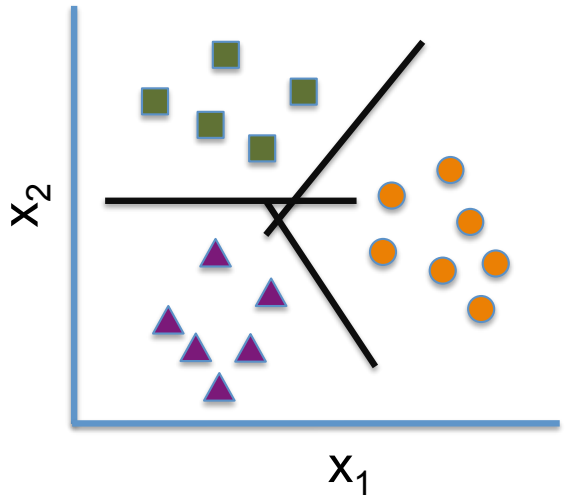
- One-versus-one approach

Recall: Multi-class Classification



- Suppose the 3 classes are 3 possible treatments for an illness and you recommend treatment 1.
- The patient sues you and your lawyer needs to explain the reasoning behind the decision in court. What would she say?
  - "$\mathbf{w}_{(1)}^\top \mathbf{x} > 0$ and $\mathbf{w}_{(2)}^\top \mathbf{x} < 0$"? This might not convince the judge.
  - "Treatment 1 worked for similar patients"? This ignores the structure of your data.

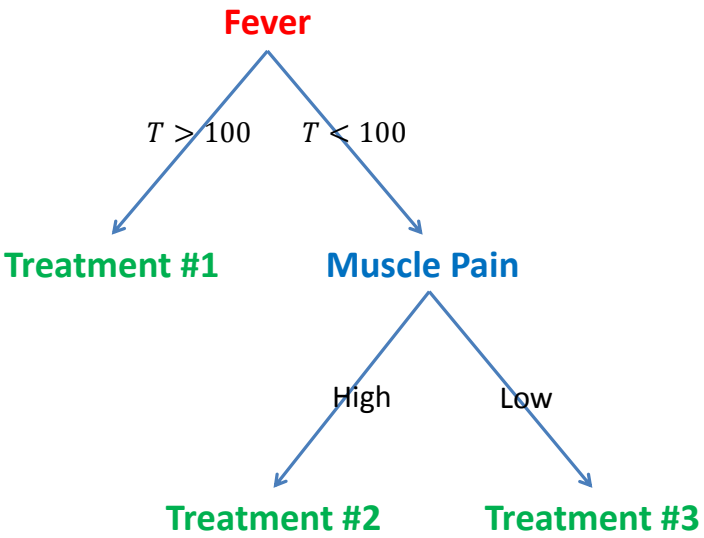
Need Interpretable Decision Boundaries



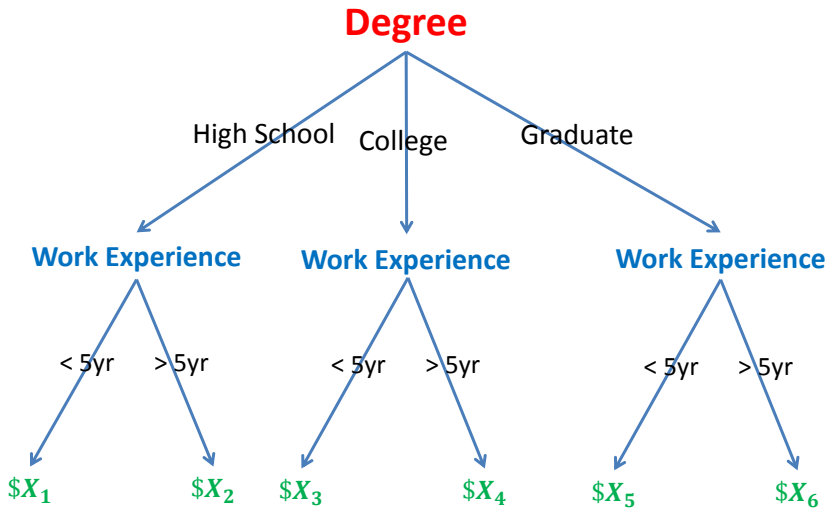
- Should be able to explain the reasoning in clear terms, e.g., "I always recommend treatment 1 when a patient has fever $\geq 100F$"
- The rules that you use to make decisions can be easily used by a lay-person without performing complex computations
- Decision trees can provide such simple decision rules

Many Decisions are Tree-Structured

Medical treatment

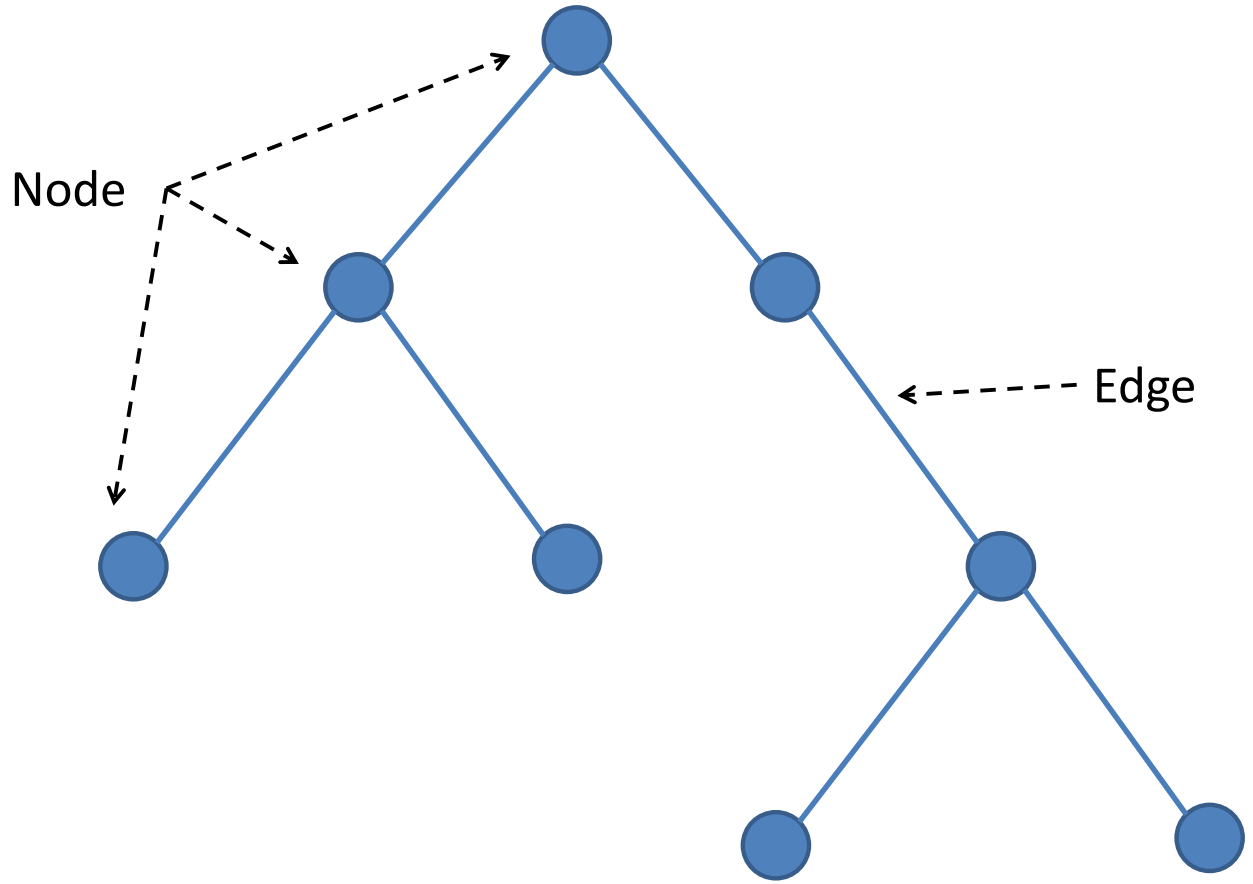


Salary in a company

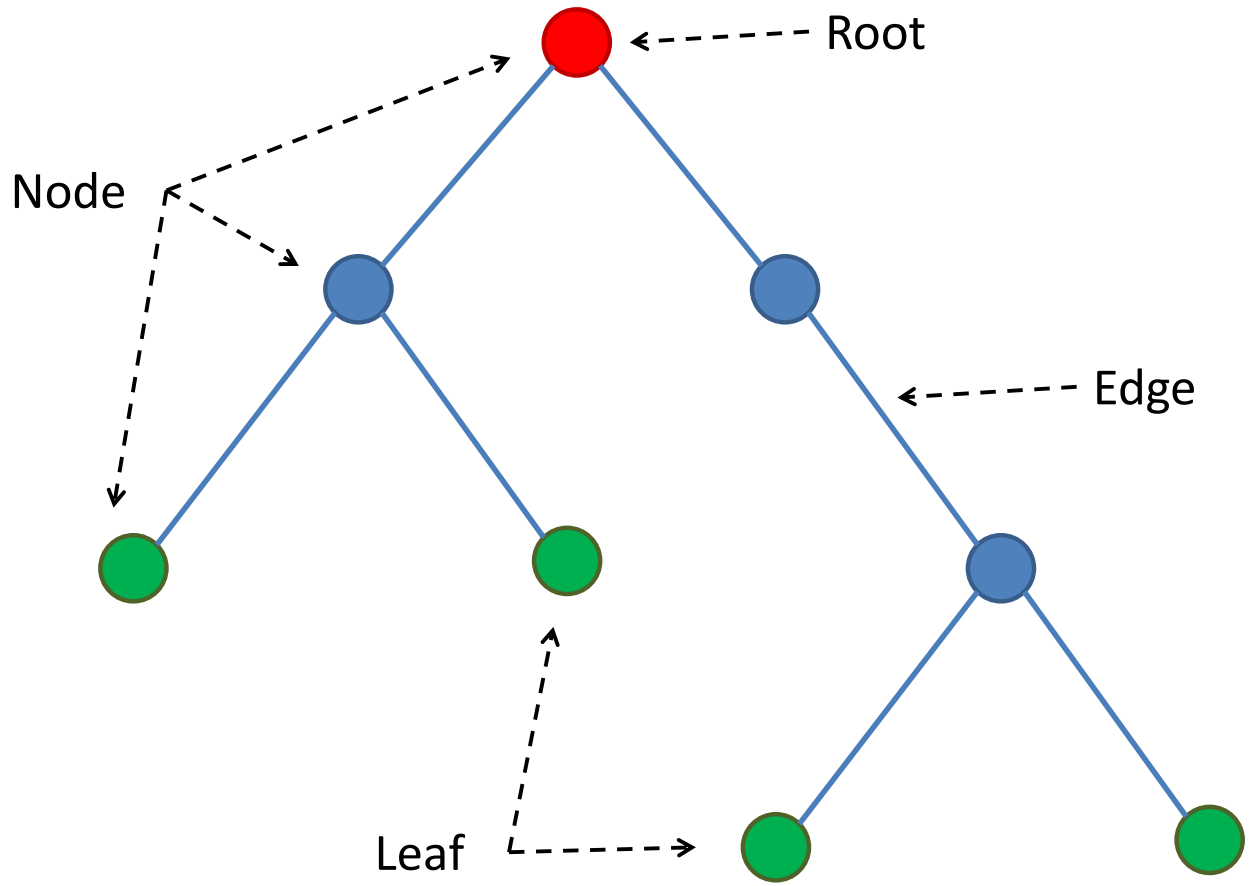


Other examples: fault detection in manufacturing systems, student admissions decisions, jail/parole decisions

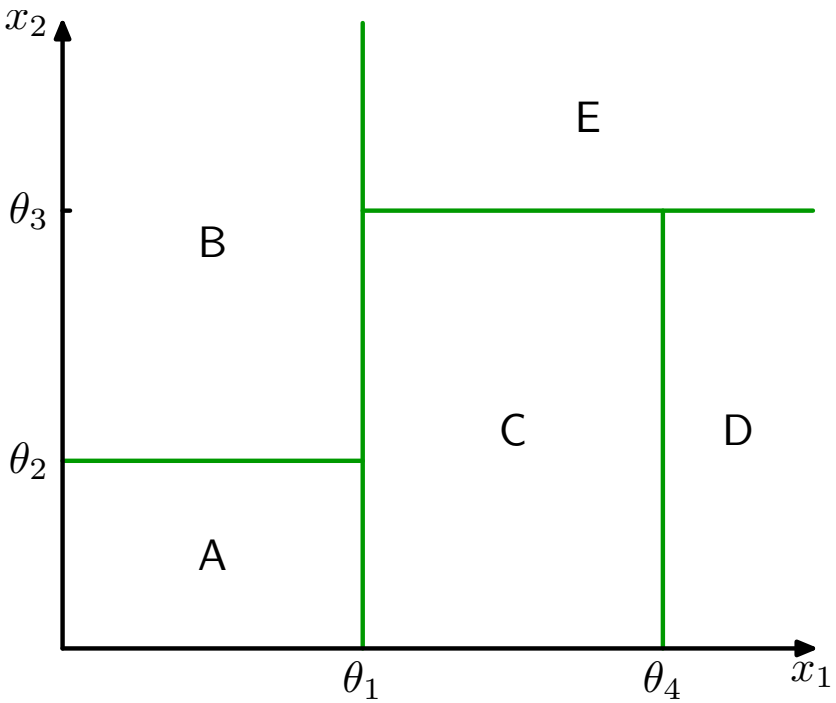
What Is a Tree?

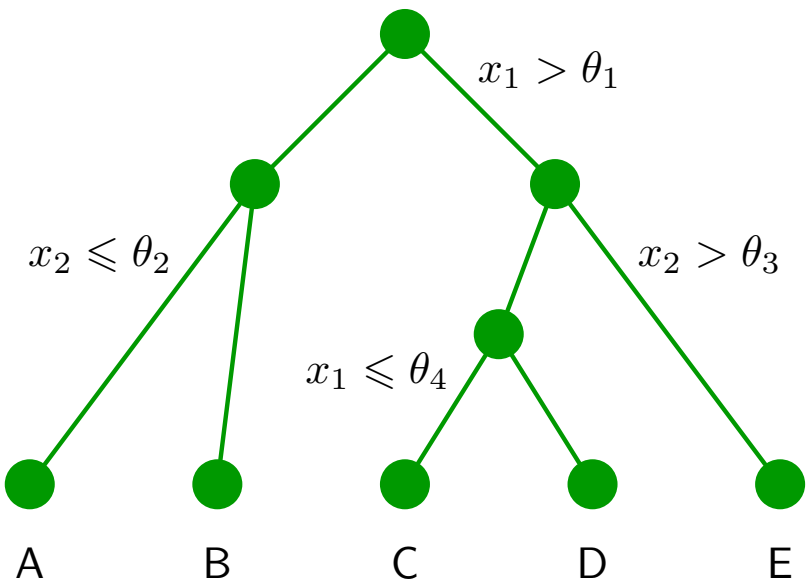


Special Names for Nodes in a Tree



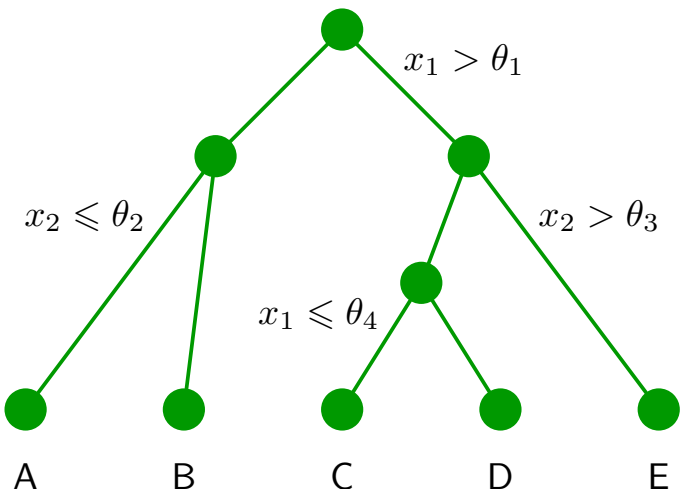
A Tree Partitions the Feature Space





Learning A Decision Tree

Learning a Tree Model



Three things to learn:

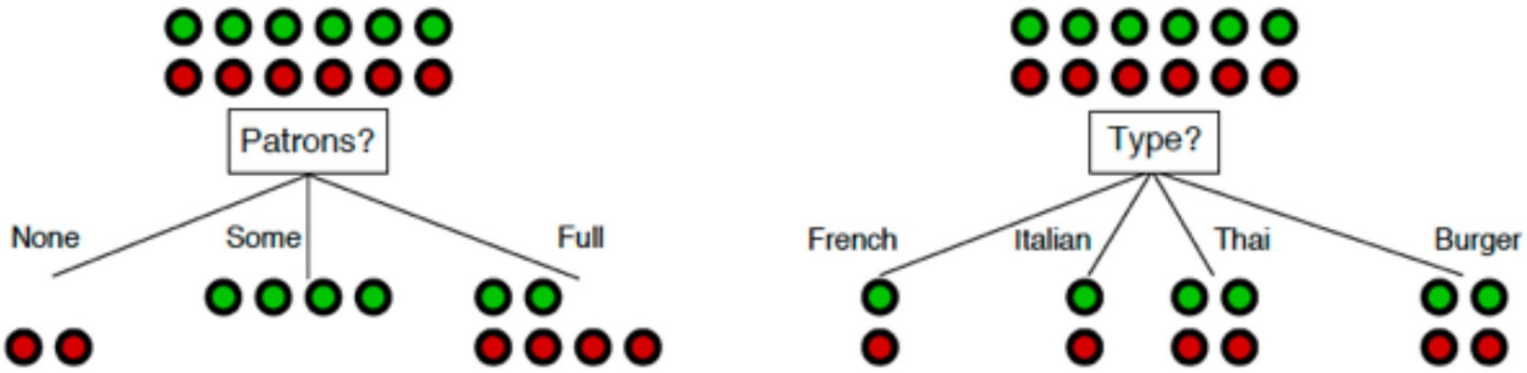
1. The structure of the tree.
2. The threshold values (θ_i).
3. The values for the leaves ($A, B, \ldots$).

Example: Choosing Whether to Wait at a Restaurant

Attributes										Target
Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
T	*F*	*F*	*T*	*Some*	*$$$*	*F*	*T*	*French*	*0–10*	*T*
T	*F*	*F*	*T*	*Full*	*$*	*F*	*F*	*Thai*	*30–60*	*F*
F	*T*	*F*	*F*	*Some*	*$*	*F*	*F*	*Burger*	*0–10*	*T*
T	*F*	*T*	*T*	*Full*	*$*	*F*	*F*	*Thai*	*10–30*	*T*
T	*F*	*T*	*F*	*Full*	*$$$*	*F*	*T*	*French*	*>60*	*F*
F	*T*	*F*	*T*	*Some*	*$$*	*T*	*T*	*Italian*	*0–10*	*T*
F	*T*	*F*	*F*	*None*	*$*	*T*	*F*	*Burger*	*0–10*	*F*
F	*F*	*F*	*T*	*Some*	*$$*	*T*	*T*	*Thai*	*0–10*	*T*
F	*T*	*T*	*F*	*Full*	*$*	*T*	*F*	*Burger*	*>60*	*F*
T	*T*	*T*	*T*	*Full*	*$$$*	*F*	*T*	*Italian*	*10–30*	*F*
F	*F*	*F*	*F*	*None*	*$*	*F*	*F*	*Thai*	*0–10*	*F*
T	*T*	*T*	*T*	*Full*	*$*	*F*	*F*	*Burger*	*30–60*	*T*

Use the attributes to decide whether to wait (T) or not wait (F)

Which Attribute to Split First?



- Patron is a better choice – gives more information to help distinguish between the labels
- Intuition: Like playing 20 questions and choosing carefully which question to ask first
- More formally: use information gain to choose which attribute to split

How to Measure Information Gain $I(X; Y)$?

Gaining information is equivalent to reducing our uncertainty.

- Use entropy $H(Y)$ to measure uncertainty in Y.
- We define $H(Y)$ and $H(Y|X)$ next

Definition (Entropy)
If a random variable Y takes K different values, a_1, $a_2 ... a_K$, then its entropy is

$$H[Y] = -\sum_{i=1}^{K} \Pr(Y = a_i) \log \Pr(Y = a_i)$$

Convention: $0 \log 0$ is considered as 0

Example: Entropy of a Bernoulli Random Variable

What is the entropy $H(Y)$ of Y, which is 1 with probability p and 0 otherwise?

Find the entropy $H(Y)$ for $p = 0.5$, $p = 0.25$, $p = 0$.

- For $p = 0.5$

  $$H(Y) = -(0.5 \log 0.5 + 0.5 \log 0.5) = \log 2 = 1 \text{ bit (log is base 2)}$$

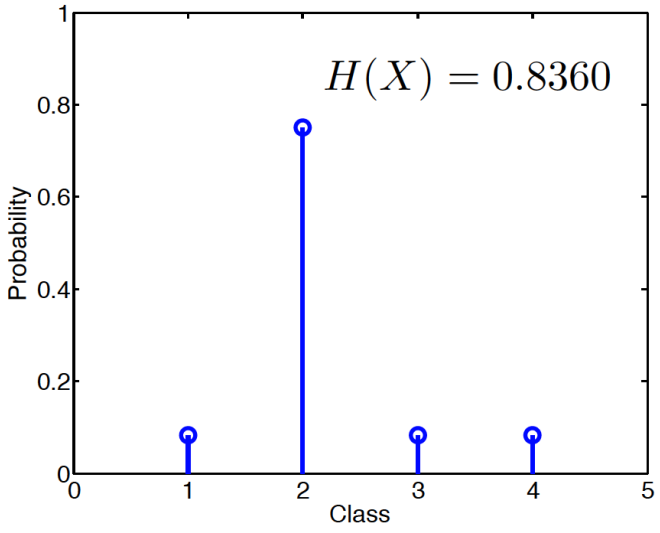
- For $p = 0.25$

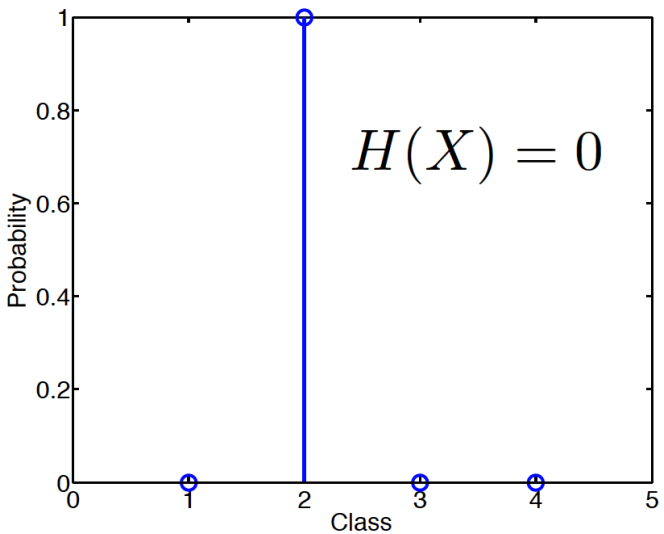
  $$H(Y) = -(0.25 \log 0.25 + 0.75 \log 0.75) = 2 \log 2 - 0.75 \log 3 = 0.81 \text{ bits}$$

- For $p = 0$, $H(Y) = 0$

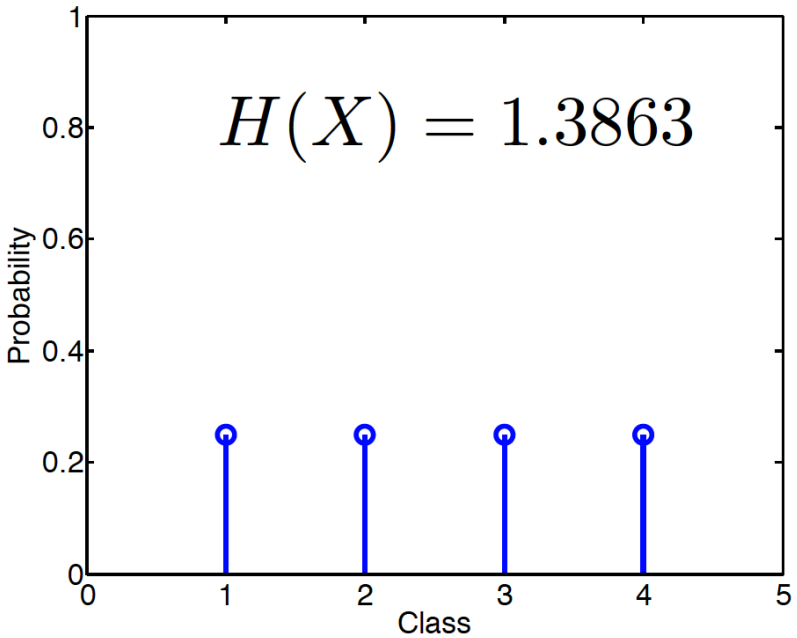
With more uncertainty ($p = 0.5$), we have a larger entropy.

Illustrating Entropy





Given a range of possible values, entropy is maximized with a uniform distribution.



Conditional Entropy

Definition (Conditional Entropy)
Given two random variables X and Y

$$H[Y|X] = \sum_k P(X = a_k) H[Y|X = a_k]$$

In our restaurant example:

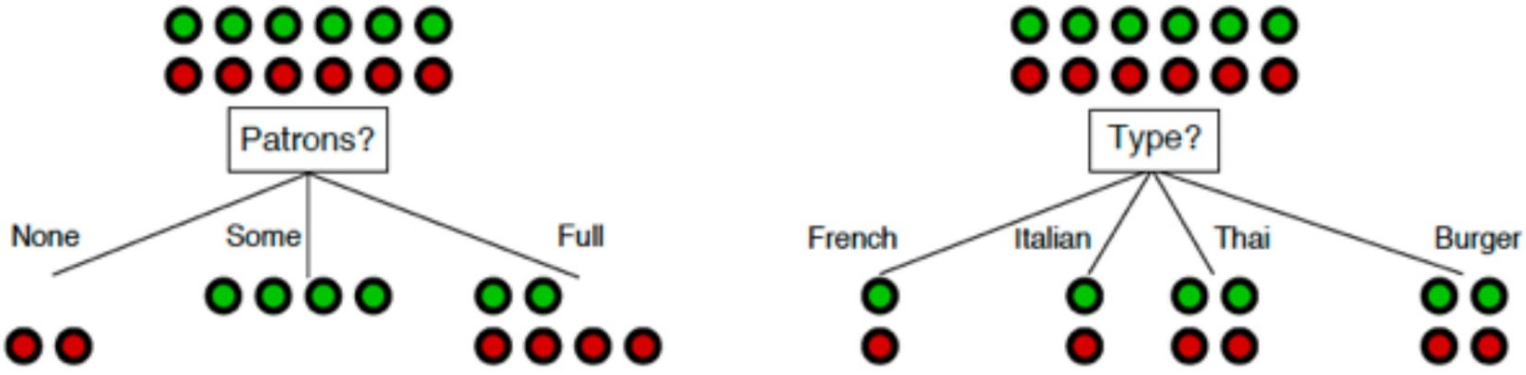
- X: the attribute to be split
- Y: wait or not (the labels)

Definition (Information Gain)

$$I(X; Y) = H[Y] - H[Y|X]$$

Measures the reduction in entropy (i.e., the reduction of uncertainty in Y) when we also consider X.

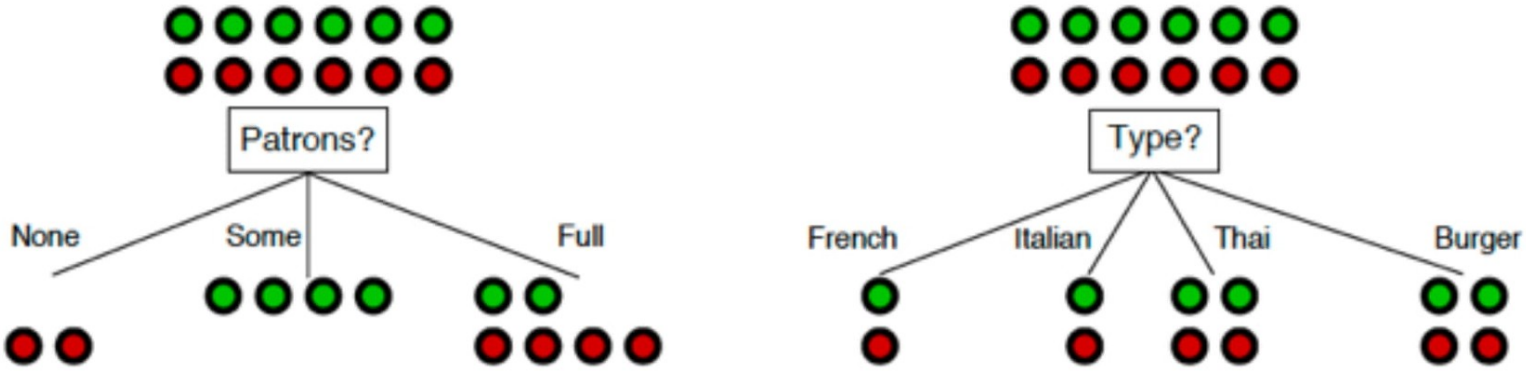
Which Attribute to Split?



Patron vs. Type?

- Let us compute the information gain $I(X; Y) = H[Y] - H[Y|X]$ for Patron and Type
- When $H[Y]$ is fixed, we need only to compare conditional entropies

Information Gain if We Split "Patron"

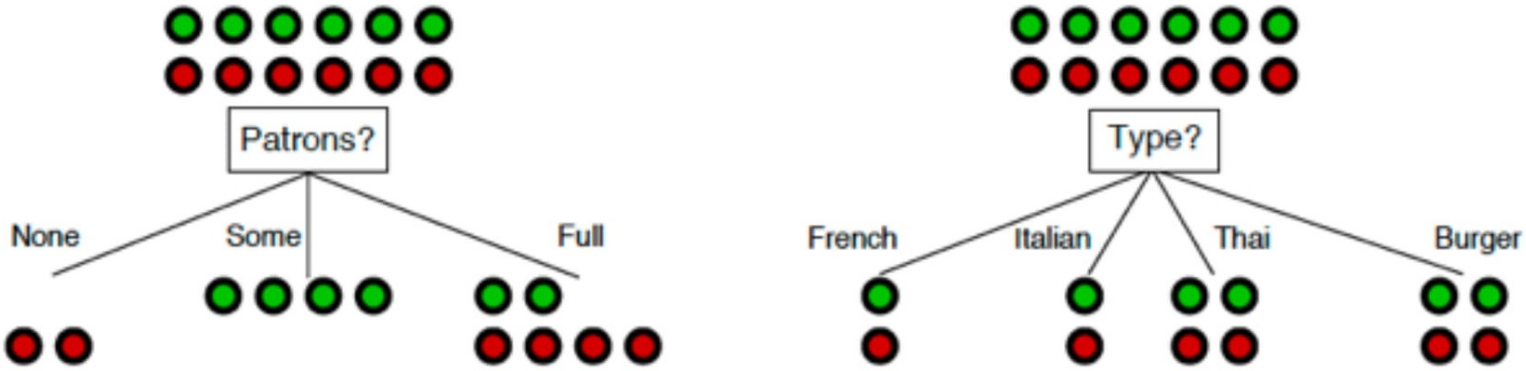


- $H(Y) = -\frac{6}{12}\log\frac{6}{12} - \frac{6}{12}\log\frac{6}{12} = 1$ bit
- $H(Y|X = none) = 0$
- $H(Y|X = some) = 0$
- $H(Y|X = full) = -\left(\frac{2}{2+4}\log\frac{2}{2+4} + \frac{4}{2+4}\log\frac{4}{2+4}\right) \approx 0.9$ bits
- Thus the conditional entropy is

$$H(Y|X) = \left(\frac{2}{12} \times 0 + \frac{4}{12} \times 0 + \frac{6}{12} \times 0.9\right) = 0.45 \text{ bits}$$

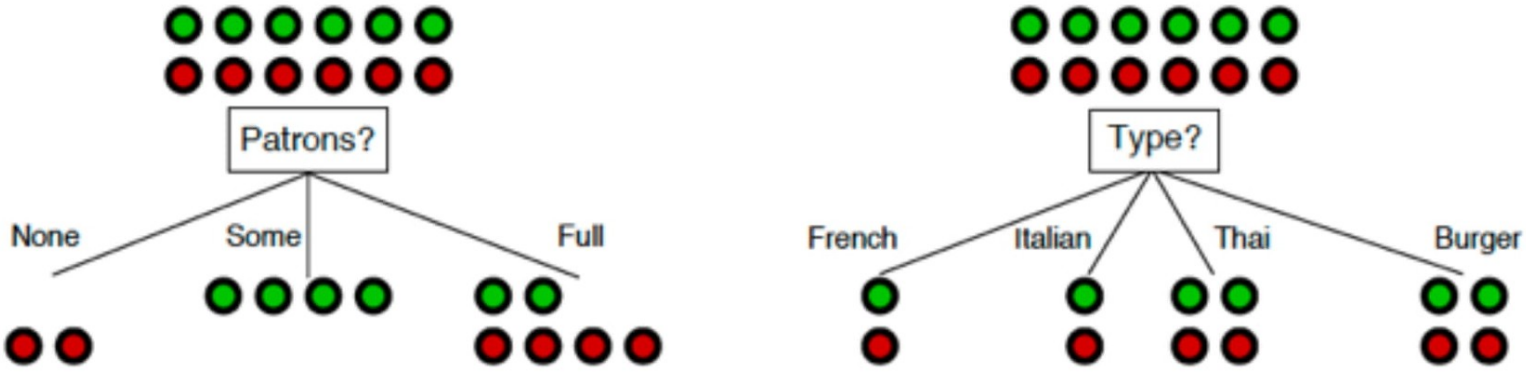
- Information Gain $I(X; Y) = 1 - 0.45 = 0.55$ bits

Information Gain if We Split "Type"



- $H(Y) = -\frac{6}{12}\log\frac{6}{12} - \frac{6}{12}\log\frac{6}{12} = 1$ bit
- $H(Y|X = \textit{french}) = \log 2 = 1$ bit
- $H(Y|X = \textit{italian}) = \log 2 = 1$ bit
- $H(Y|X = \textit{thai}) = \log 2 = 1$ bit
- $H(Y|X = \textit{burger}) = \log 2 = 1$ bit
- Thus the conditional entropy is
  $H(Y|X) = \frac{2}{12} \times 1 + \frac{2}{12} \times 1 + \frac{4}{12} \times 1 + \frac{4}{12} \times 1 = 1$ bit
- Information Gain $I(X; Y) = 1 - 1 = 0$ bits

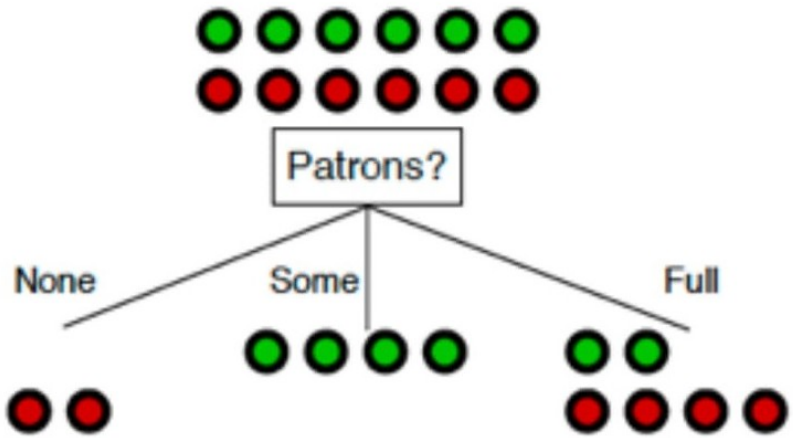
Splitting on "Patron" or "Type"?



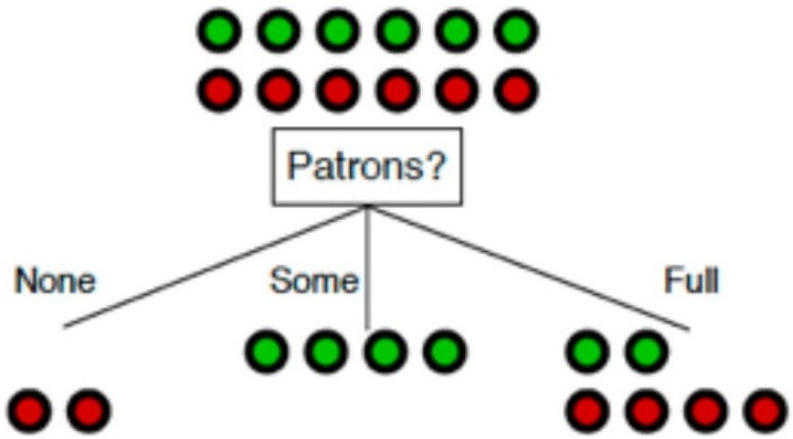
- Information gain from "Patron" is 0.55 bits.
- Information gain from "Type" is 0 bits.

Thus, we should split on "Patron" and not "Type" (higher information gain). This is consistent with our intuition.

Next Split?



Do We Split on "None" or "Some"?



- No, we do not
- The decision is deterministic, as seen from the training data.

Next Split

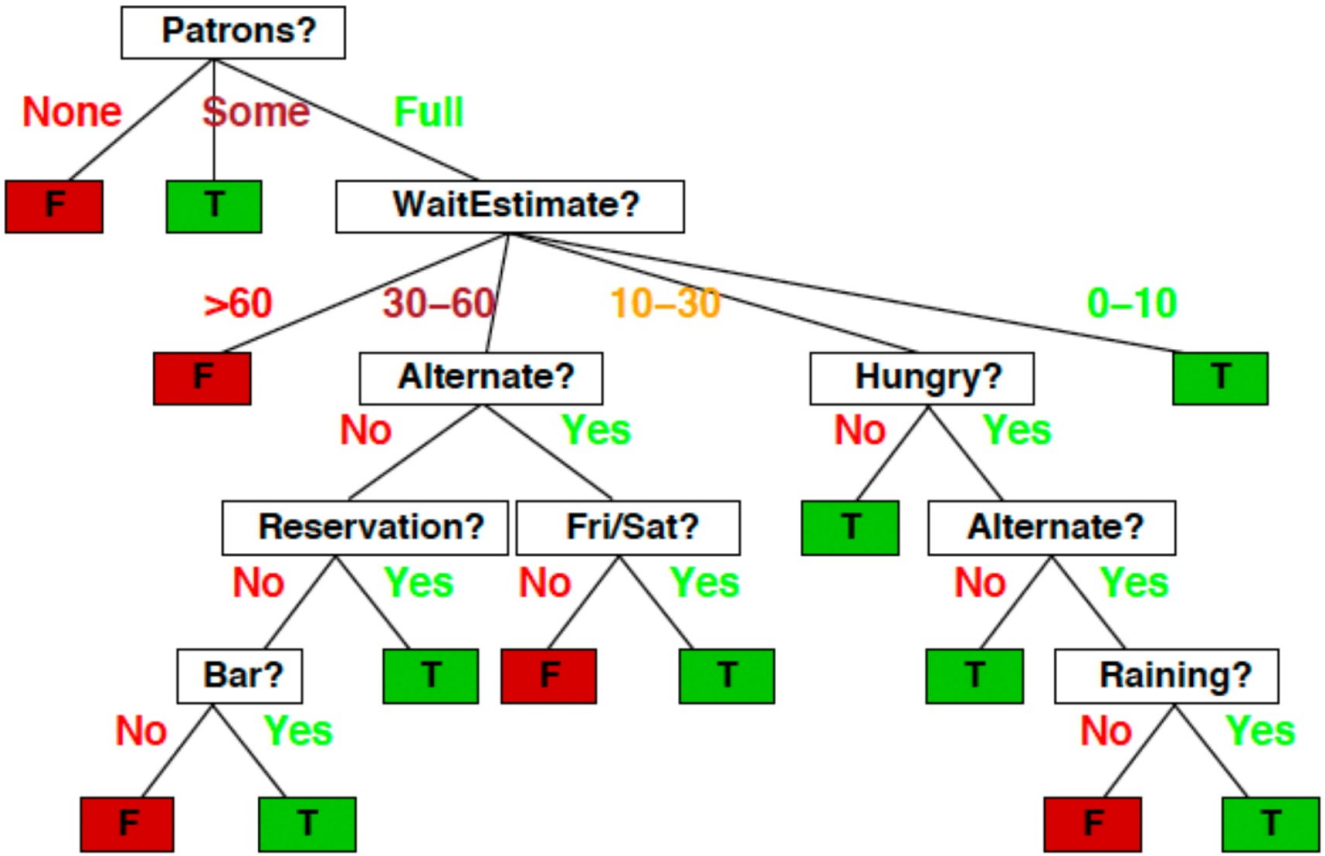
Next split?

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T

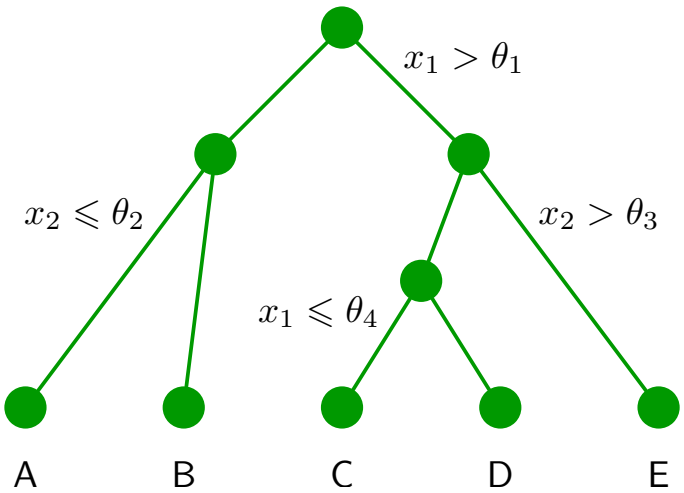
Classification of examples is positive (T) or negative (F)

We will look only at the 6 instances with Patrons == Full

Greedily We Build the Tree and Get...



Learning a Tree Model



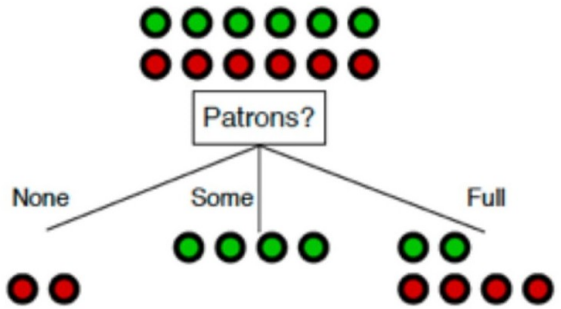
Three things to learn:

1. The structure of the tree.
   - Split based on information gain.
2. The threshold values (θ_i).
   - One outcome per category.
   - What if features are continuous?
3. The values for the leaves $(A, B, \ldots)$.
   - Obvious if only one value exists in the leaf.
   - Otherwise?

Practical Considerations for Decision Trees

Often, We Use Binary Decision Trees

In our example, we split each node among all categories.

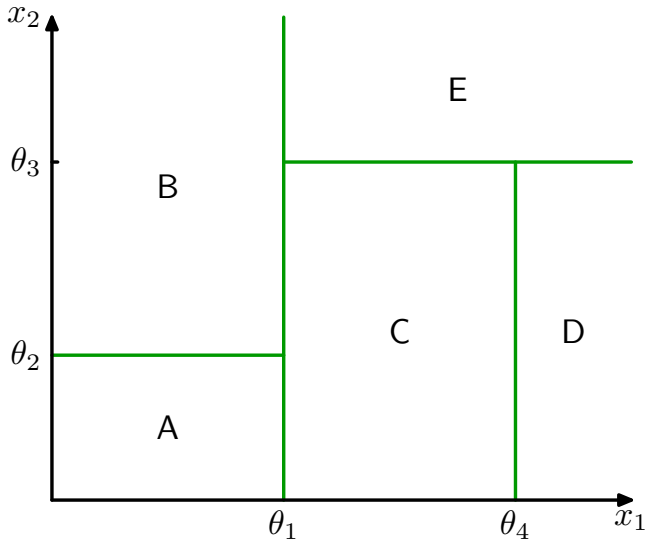


This may overfit with a large number of categories!

- Many implementations only allow two outcomes for each split.
- Binary decision trees can capture multiple splits by adding an additional level of splits. (e.g., first split on "none" and "some + full", then split the "some + full" node on "some" and "full').

Computational Considerations: Numerical Features

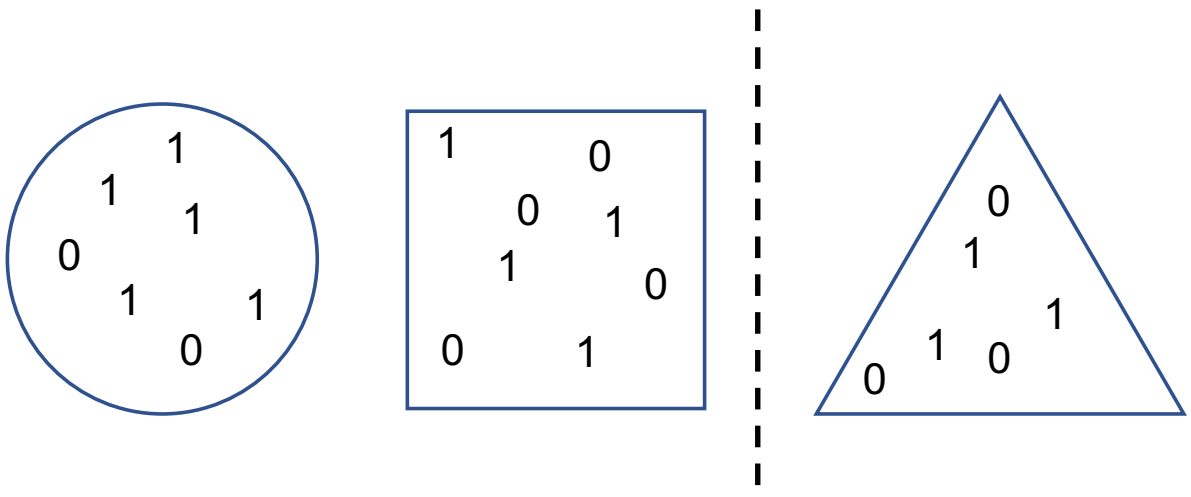
- How should we decide the threshold to use in splitting the feature?
- Can we do this efficiently?
  - Yes – for a given feature we only need to consider the n values in the training data!
  - If we sort each feature by these n values, we can quickly compute and maximize the information gain along each possible threshold.
  - This takes $O(dn \log n)$ time, where d is the number of features



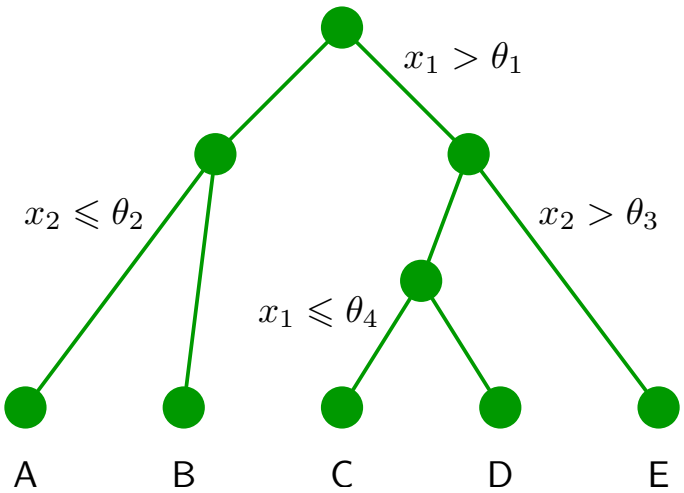
$$x_1^{(1)} \quad x_1^{(2)} \quad x_1^{(3)} \;\Big|^{\theta_5}\; x_1^{(4)} \quad \cdots \quad x_1^{(n)}$$

Computational Considerations: Categorical Features

- Assuming q distinct categories, there are $\frac{1}{2}\left(2^q - 2\right) = 2^{q-1} - 1$ possible partitions!
- Things simplify in the case of binary classification or regression
  - Can sort the categories by the fraction of labels falling in class 1
  - Suffices to consider only $q - 1$ possible splits (see Section 9.2.4 in ESL)
- Example: Suppose we have two labels (0 or 1) and the feature is "shape," which has three categories (circle, square, or triangle).



Learning a Tree Model



Three things to learn:

1. The structure of the tree.
   - Split based on information gain.
2. The threshold values (θ_i).
   - (Often) binary decision trees.
   - Split based on information gain.
3. The values for the leaves $(A, B, \ldots)$.
   - Obvious if only one value exists in the leaf.
   - Otherwise?

What Is the Optimal Tree Depth?

- What happens if we pick the wrong depth?
  - If the tree is too deep, we can overfit
  - If the tree is too shallow, we underfit
- Max depth is a hyperparameter that should be tuned by the data
- Alternative strategy is to create a very deep tree, and then to prune it (see Section 9.2.2 in ESL for details)

Cost Complexity Pruning

Pruning means collapsing non-terminal nodes to eliminate a split.

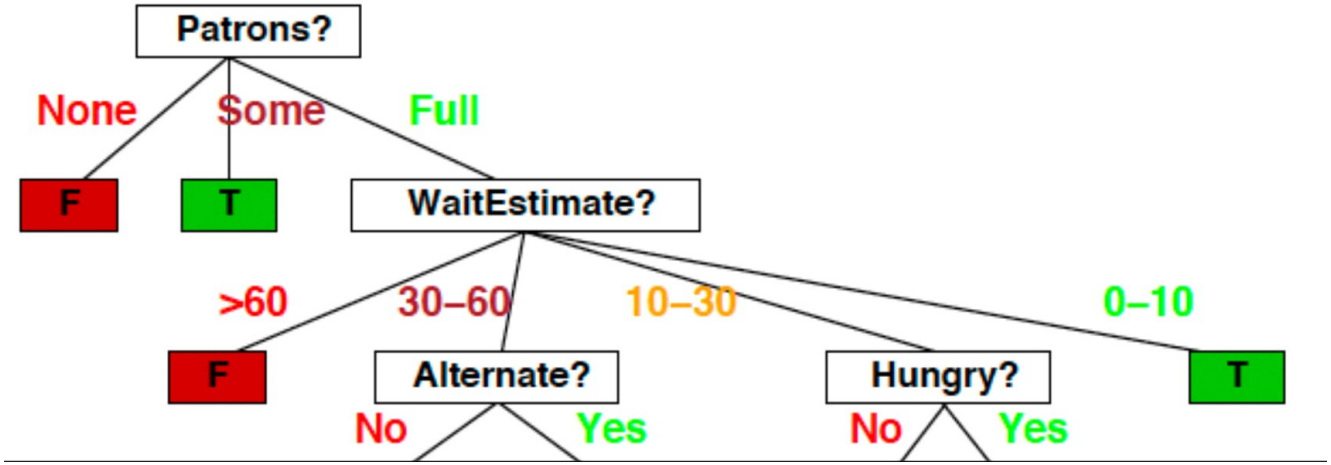
Cost complexity criterion

$$C_\alpha(T) = \sum_{m=1}^{|T|} \text{error}_m(T) + \alpha|T|$$

- Find the tree T that minimizes the cost $C_\alpha(T)$, where $m = 1, 2, \ldots, |T|$ indexes the leaf nodes.
- Measure error of the training data at each leaf node as before (misclassification rate, residual sum of squares,...).
- Choose α as a hyperparameter (similar to regularization).

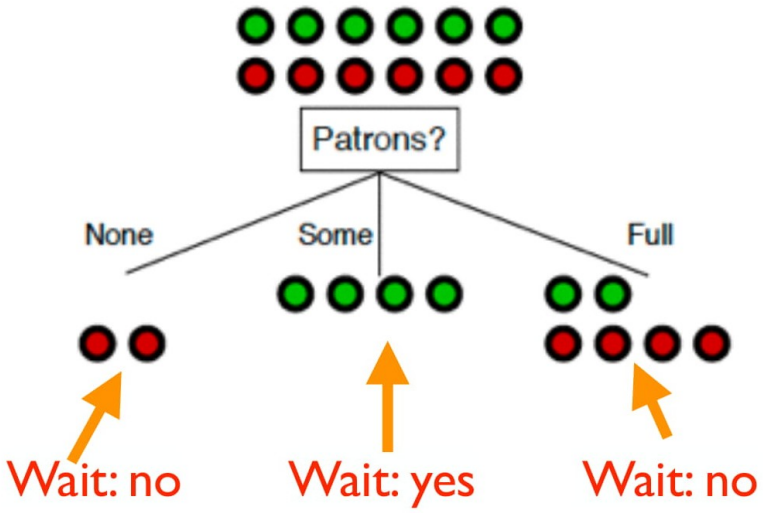
To find the tree that minimizes C_α, greedily collapse the node in the full tree that increases the error rate the least.

How to Classify with a Pruned Decision Tree?



- If we stop here, not all training samples would be classified correctly
- More importantly, how do we classify a new instance?
- We label the leaves of this smaller tree with the majority of training sample's labels.

Example



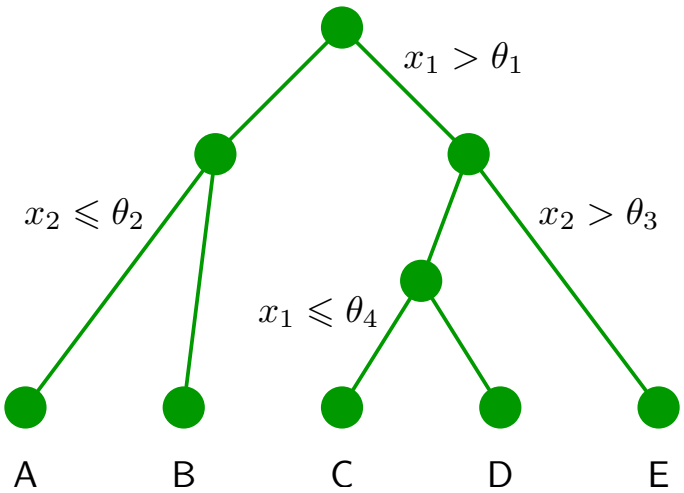
Overfitting in Decision Trees

- Including irrelevant attributes can result in overfitting the training example data.
- If we have too little training data, even a reasonable hypothesis space will overfit.

Strategies to avoid overfitting

- Stop growing when data split is not statistically significant.
- Acquire more training data.
- Remove irrelevant attributes (manual process — not always possible).
- Grow full tree, then post-prune (e.g., cost complexity)

Learning a Tree Model

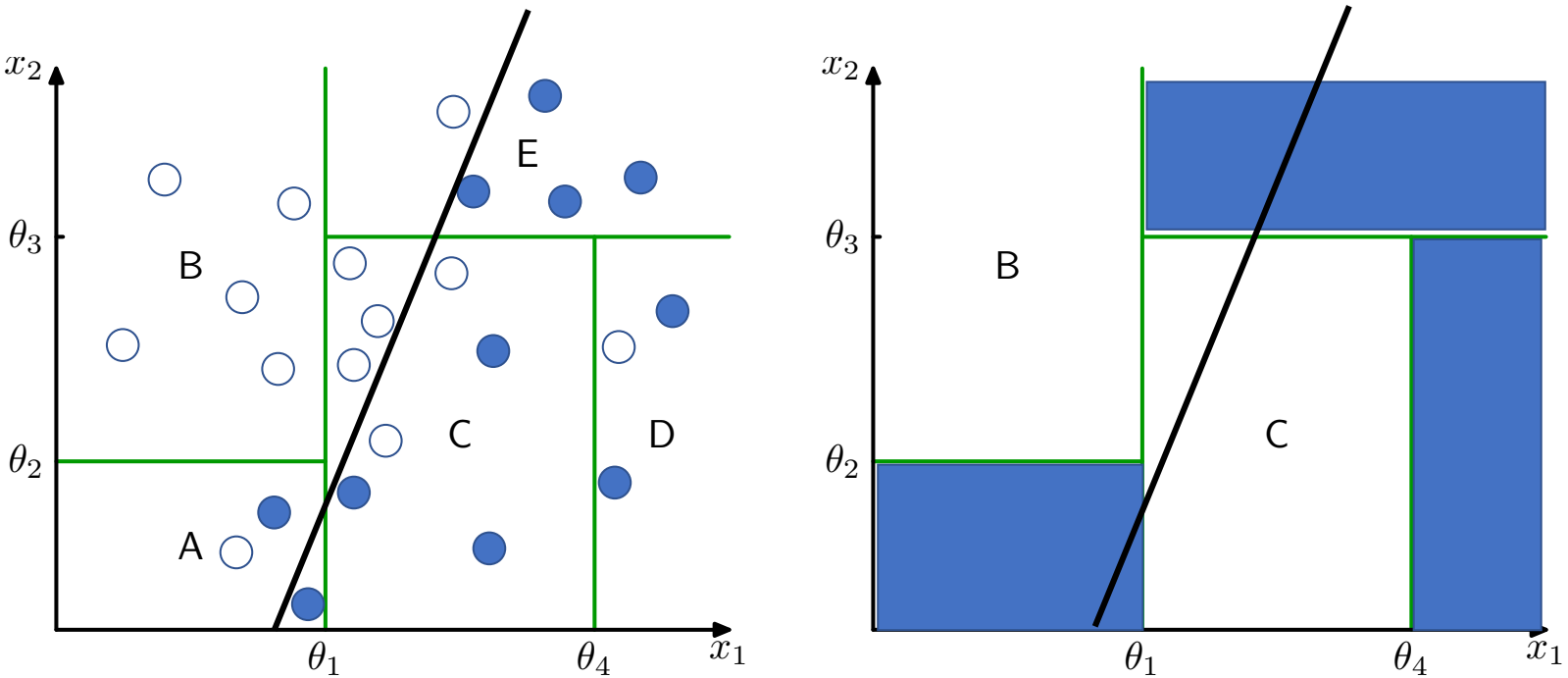


Three things to learn:

1. The structure of the tree.
   - Split based on information gain.
2. The threshold values (θ_i).
   - (Usually) binary decision trees.
   - Split based on information gain.
3. The values for the leaves ($A, B, \ldots$).
   - Majority vote.

Disadvantages of Decision Trees

- Binary decision trees find it hard to learn linear boundaries.
- Decision trees can have high variance due to dependence on the training data.
- We use heuristic training techniques: finding the optimal partition is NP-hard.



Advantages of Decision Trees

- Can be interpreted by humans (as long as the tree is not too big)
- Computationally efficient (for shallow trees)
- Handles both numerical and categorical data
- Can be used for both classification and regression
- Compact representation: unlike Nearest Neighbors we don't need training data at test time
- But, like NN, decision trees are nonparametric because the number of parameters depends on the data

Summary of Decision Trees

You should know:

- Motivation for considering decision trees
- How to construct a decision tree
- Techniques for ensuring the tree does not overfit
- Disadvantages of decision tree methods

Decision trees are a common building block for various ensemble methods (more on this next lecture).