Decision Trees

Mehul Motani
Electrical & Computer Engineering
National University of Singapore
Email: motani@nus.edu.sg
Office: E4-05-18
Tel: 6516 6918

Motivation for Learning

- Modern systems are complex and may have many parameters.
- It is impractical and often impossible to encode all the knowledge a system needs.
- Different types of data may require very different parameters.
- Instead of trying to hard code all the knowledge, it makes sense to learn what we need from the data itself.
- Three broad approaches to learning
  - Supervised learning
  - Unsupervised learning
  - Reinforcement learning

Learning from Observations

- Supervised Learning – learn a function from a set of training examples which are preclassified feature vectors.

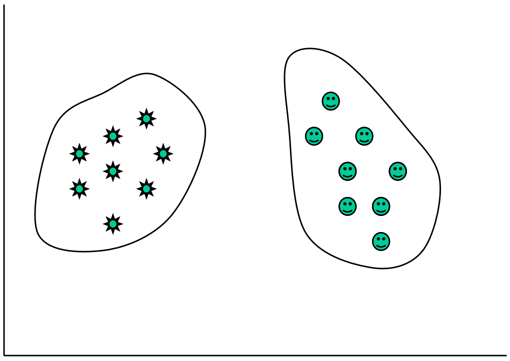
Feature vector	Class
(square, red)	1
(square, blue)	1
(circle, red)	2
(circle blue)	2
(triangle, red)	1
(triangle, green)	1
(ellipse, blue)	2
(ellipse, red)	2

Given a previously unseen feature vector, what is the rule that tells us if it is in class 1 or class 2?

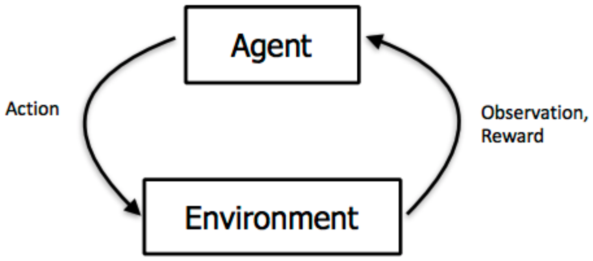
(circle, green) ?
(triangle, blue) ?

Learning from Observations

- Unsupervised Learning – No classes are given. The idea is to find patterns in the data. This generally involves clustering.



- Reinforcement Learning – learn from feedback after a decision is made.

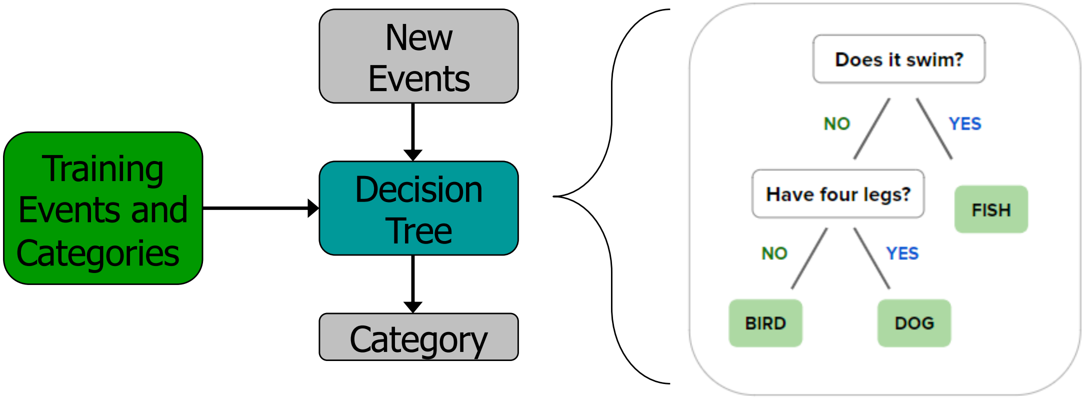


Decision Trees

- The theory behind decision trees is well-understood.
- Decision trees have the nice property that you can easily understand the decision rule that was learned.
- Easy to explain and interpret (rule-based)
- There exist fast deterministic algorithms for computing decision trees.
- Can handle discrete and continuous parameters
- Cautionary notes
  - Complexity can grow large.
  - Prone to overfitting

Decision Trees

- Use training data to build the decision tree.
- Use a decision tree to predict categories for new events.



New Events

Training Events and Categories

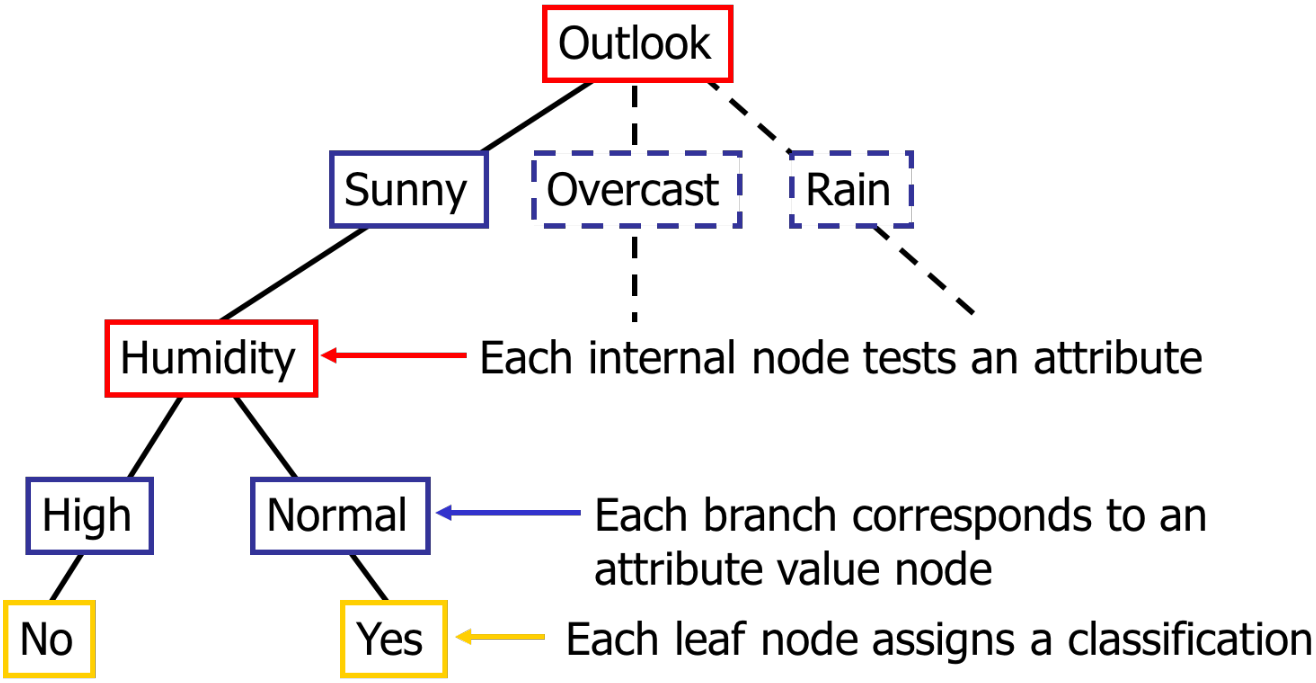
Decision Tree

Category

Example – Should I play tennis?

Day	Outlook	Temp	Humidity	Wind	Tennis?
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*

Tennis Example – Decision Tree

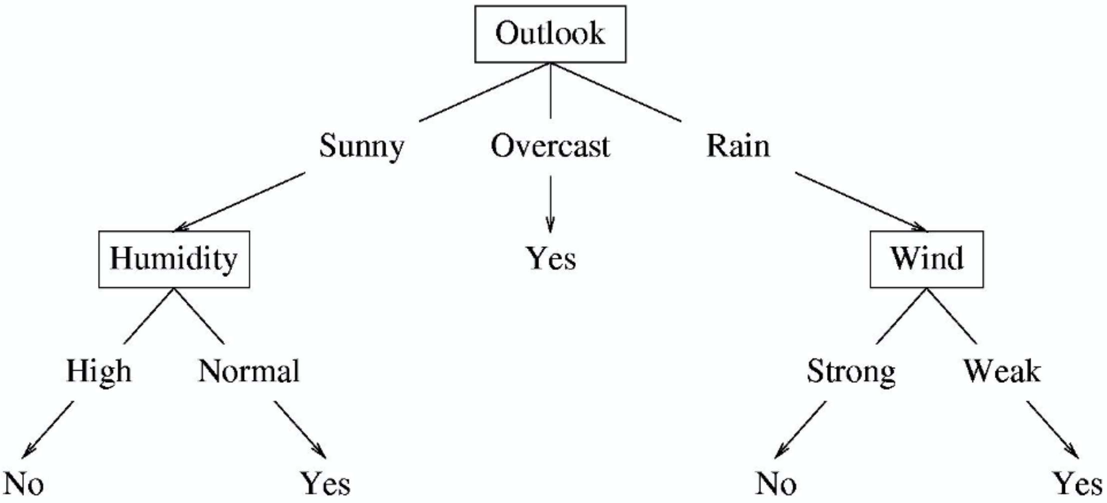


Each internal node tests an attribute

Each branch corresponds to an attribute value node

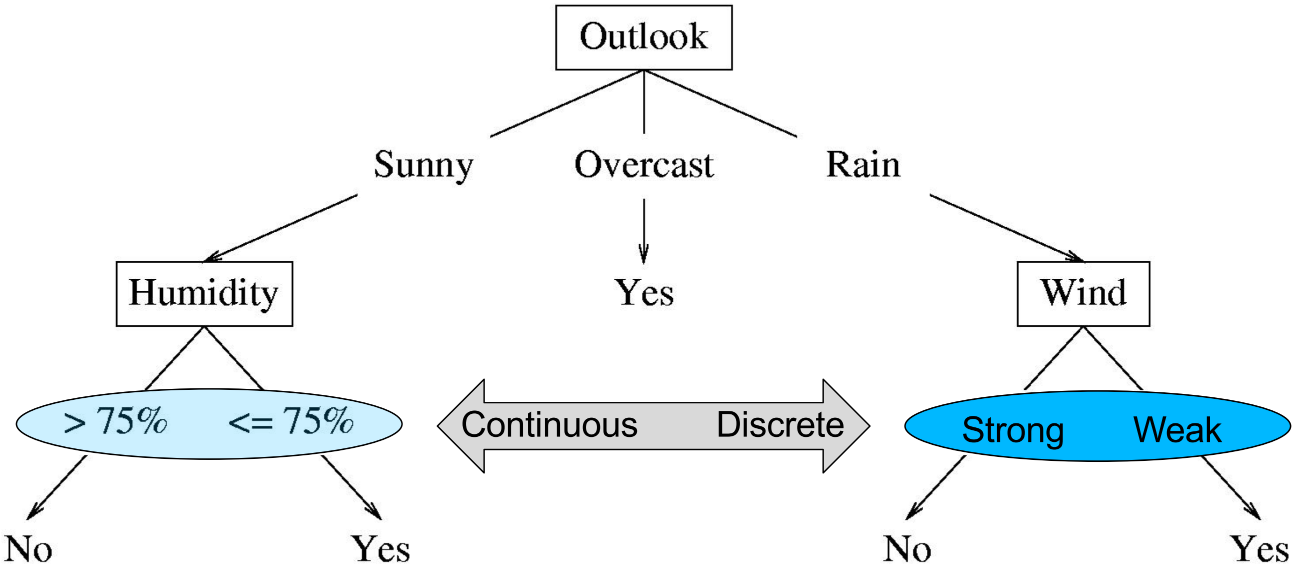
Each leaf node assigns a classification

Decision Tree for Tennis Example



- This is the full decision tree for the Tennis example.
- Compare this to the dataset and notice that the decision tree captures the dataset precisely.
- Note that the Temperature feature is not relevant.

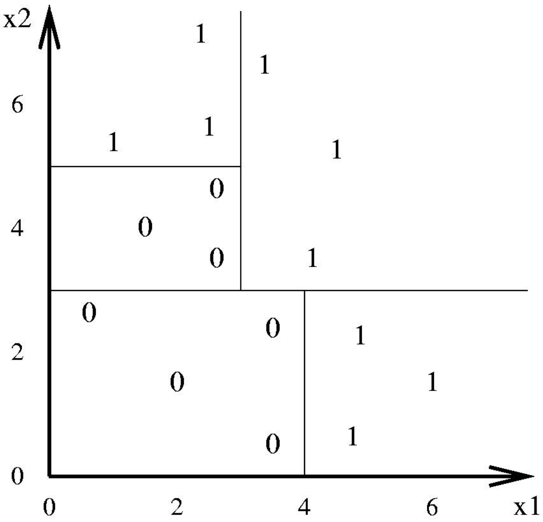
Continuous vs. Discrete Feature Spaces

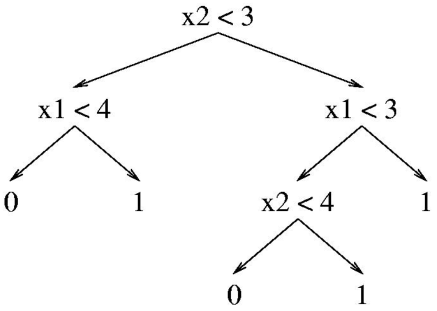


- Decision trees can handle both continuous and discrete feature spaces

Decision Tree Decision Boundaries

Decision trees divide the feature space into axis-parallel rectangles, and label each rectangle with one of the K classes.





Example - Word Sense Disambiguation

- Given an occurrence of a word, decide which sense, or meaning, was intended.
- Example: "run"
  - run1: move swiftly (I ran to the store.)
  - run2: operate (I run a store.)
  - run3: flow (Water runs from the spring.)
  - run4: length of torn stitches (Her stockings had a run.)

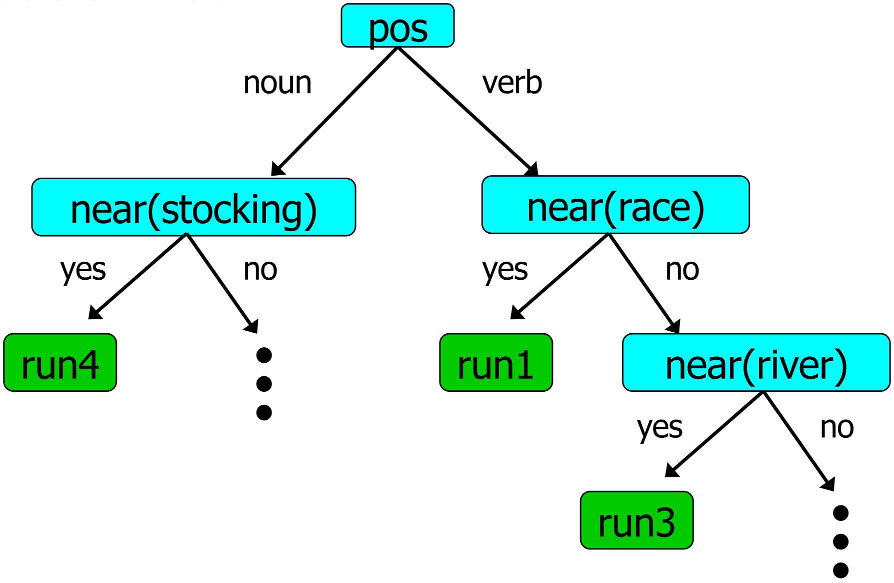
Features				Word Sense
pos	near(race)	near(river)	near(stockings)	
noun	no	no	no	run4
verb	no	no	no	run1
verb	no	yes	no	run3
noun	yes	yes	yes	run4
verb	no	no	yes	run1
verb	yes	yes	no	run2
verb	no	yes	yes	run3

Example - Word Sense Disambiguation

- Categories
  - Use word sense labels (run1, run2, etc.) to name the possible categories.
- Features
  - Features describe the *context* of the word we want to disambiguate.
  - Possible features include:
    - near(w): is the given word near an occurrence of word w?
    - pos: the word's part of speech
    - left(w): is the word immediately preceded by the word w?
    - etc.

Example - Word Sense Disambiguation

- Example decision tree:

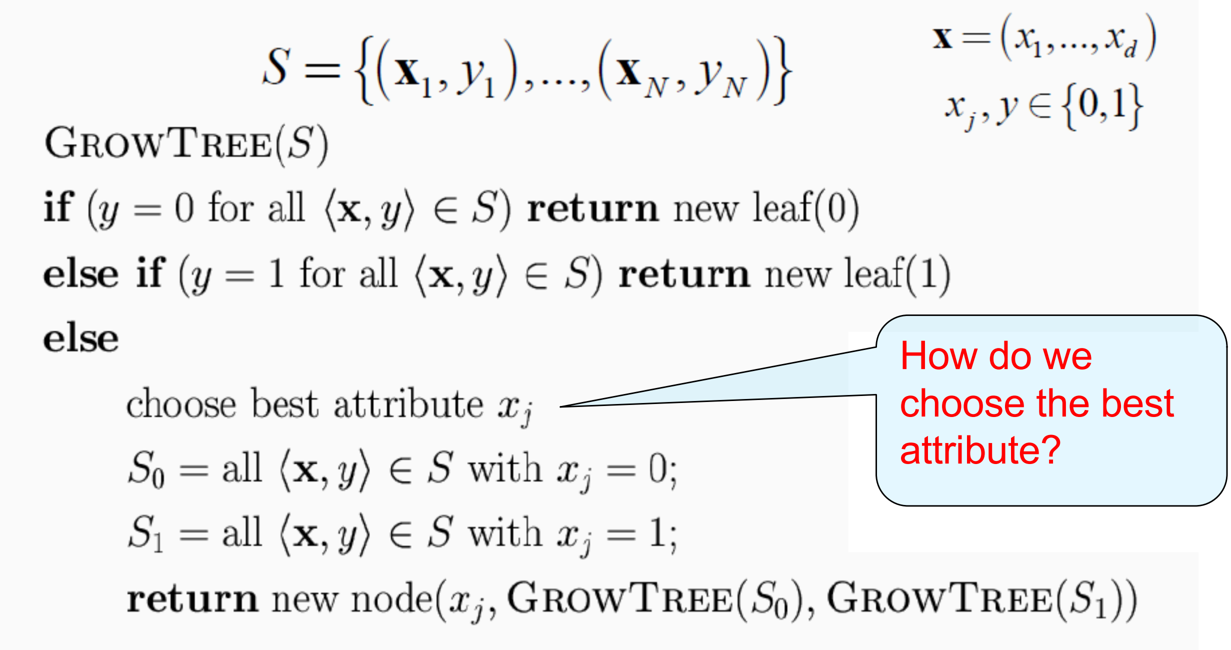


(Note: Decision trees for WSD tend to be quite large)

Learning Algorithm for Decision Trees

- The decision tree encodes the optimal sequence of questions to ask to make the classification decision.
- Finding the optimal decision tree is NP-hard.
  - The number of decision trees is huge!
  - With 6 binary attributes, there are 18,446,744,073,709,551,616 possible trees!
- The decision tree algorithm used in practice is a fast, greedy but suboptimal algorithm.
- In practice, the algorithm has good competitive performance.

Learning Algorithm for Decision Trees



$$S = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$$

$\mathbf{x} = (x_1, \dots, x_d)$

$x_j, y \in \{0,1\}$

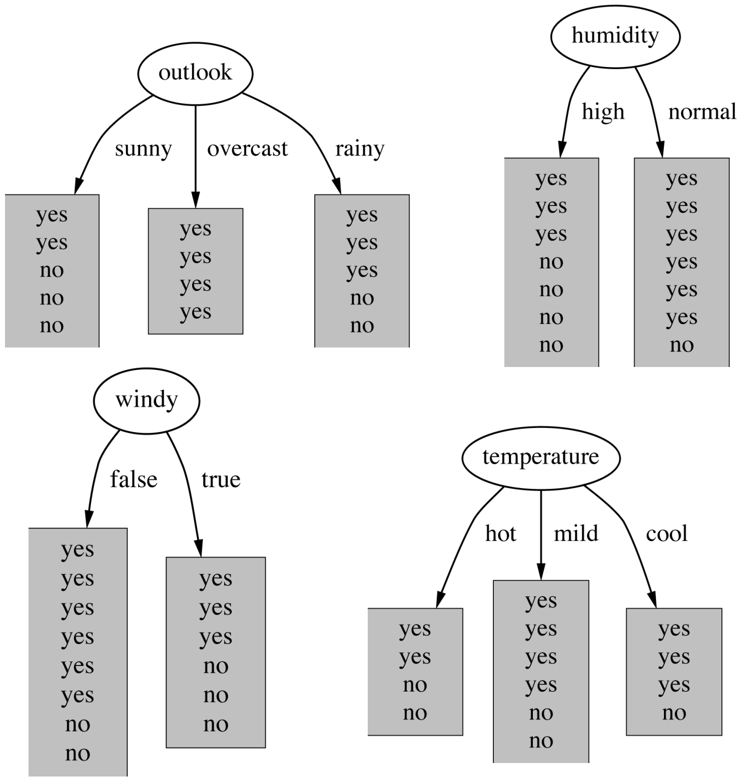
```
GROWTREE($S$)
if ($y = 0$ for all $\langle \mathbf{x}, y \rangle \in S$) return new leaf(0)
else if ($y = 1$ for all $\langle \mathbf{x}, y \rangle \in S$) return new leaf(1)
else
    choose best attribute $x_j$
    $S_0 =$ all $\langle \mathbf{x}, y \rangle \in S$ with $x_j = 0$;
    $S_1 =$ all $\langle \mathbf{x}, y \rangle \in S$ with $x_j = 1$;
    return new node($x_j$, GROWTREE($S_0$), GROWTREE($S_1$))
```

How do we choose the best attribute?

Example – Should I play tennis?

Day	Outlook	Temp	Humidity	Wind	Tennis?
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*

Which attribute to select?



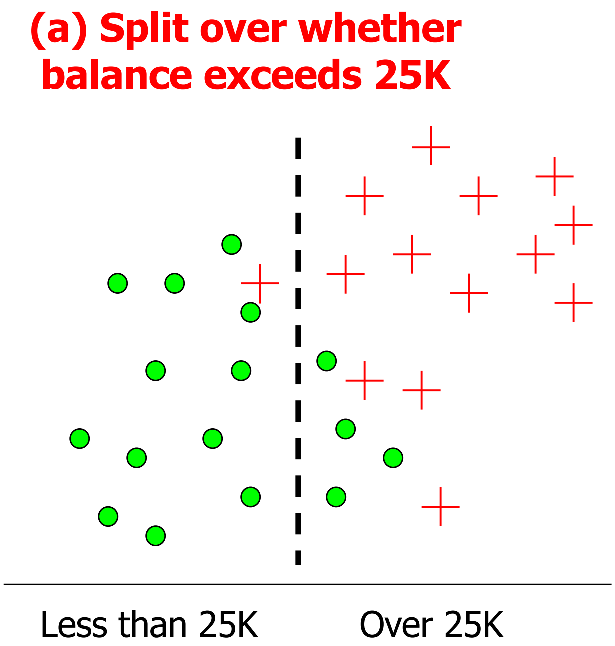
Which is the best attribute?

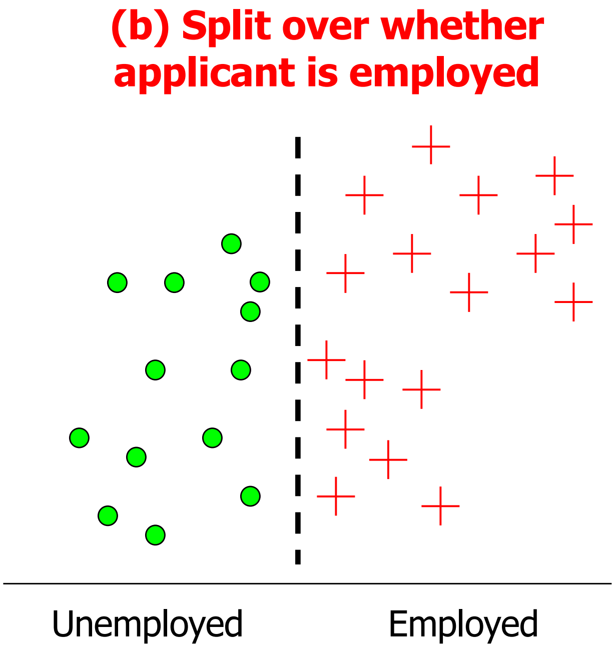
- The one which will result in the smallest tree
- Heuristic: choose the attribute that produces the "purest" nodes

Need a good measure of purity!

Example – Which split is more informative?

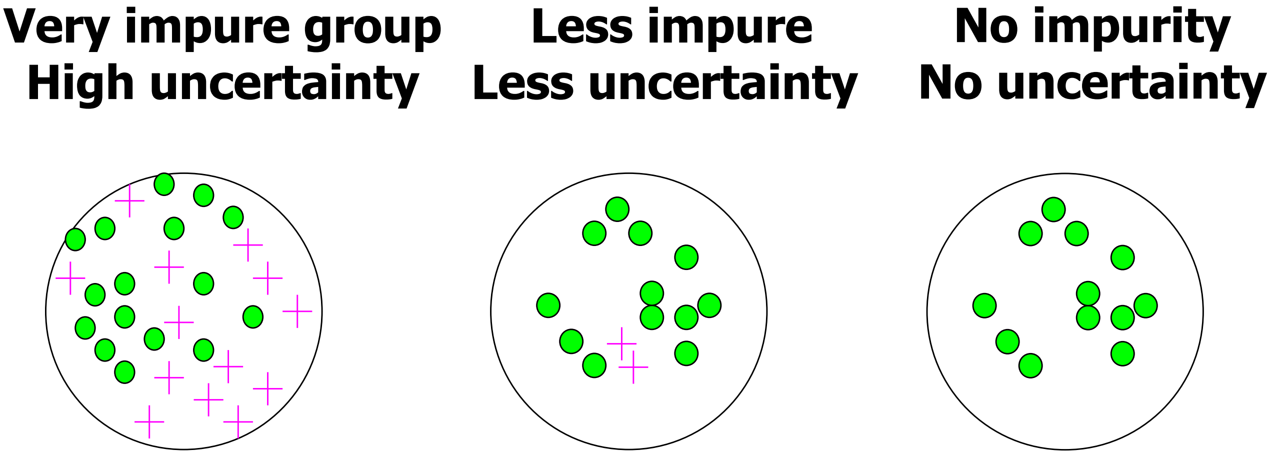
Which attribute results in a more impure split?





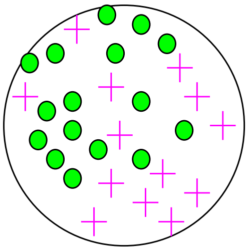
Impurity is Uncertainty

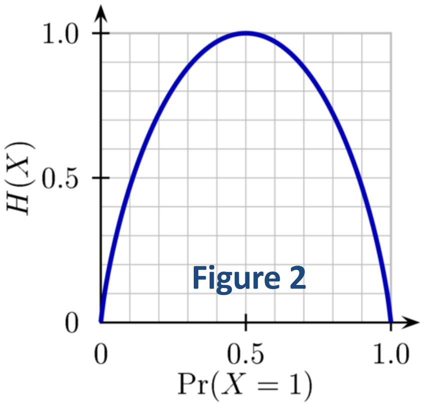
- The key idea is to think of Impurity as Uncertainty
- We will use the counts at the leaves to define probability distributions and use them to measure uncertainty



Entropy: a common way to measure uncertainty

- Entropy = *H(X)* = $\sum_i - p_i \log_2 p_i$ **(1)**
- p_i is the probability of class *i*
- p_i is the fraction of class *i* in the set
- Entropy comes from information theory
  - Cover and Thomas, *Elements of information Theory,* Wiley & Sons, 2012.
- Entropy = Uncertainty = Impurity
  - Higher entropy → More uncertainty
  - Higher entropy → More impure
  - Lower entropy → Less uncertainty
  - Lower entropy → Less impure





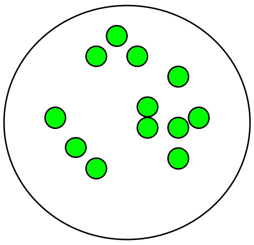
Binary entropy function

What does this mean for learning from data?

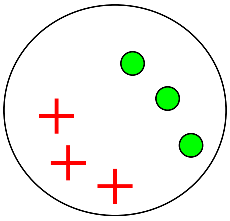
Example: Two Classes

- What is the entropy of a group in which all examples belong to the same class?
  - Entropy = *H(X)* = - 1 $\log_2 1$ = 0
  - Not a good training set for learning
- What is the entropy of a group with 50% in either class?
  - *H(X)* = -0.5 $\log_2 0.5$ – 0.5 $\log_2 0.5$ =1
  - Good training set for learning

Minimum impurity



Maximum impurity

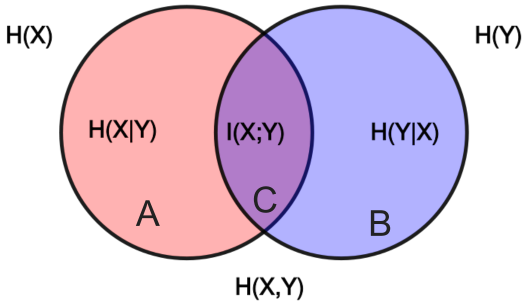


Information Gain

- We want to determine which attribute in the training set is most useful for discriminating between the classes to be learned.
- Information gain tells us how much information a given attribute tells us about the class label.
  - Information Gain: $I(X;Y) = H(Y) - H(Y|X)$

    Conditional Entropy
  - Information Gain is the decrease in entropy after splitting
- We will use information gain to decide the ordering of attributes in the nodes of the decision tree.
  - The higher the information gain, the more information that attribute contains about the label
  - Attributes with higher information gain are selected before attributes with lower information gain.
- Information gain is also known as the mutual information between the features and the label.

Entropy and Mutual Information



- The area contained by both circles (A+B+C) is the joint entropy H(*X*,*Y*).
- The circle on the left (A+C) is the individual entropy H(*X*)
- The part labeled A is the conditional entropy H(*X*|*Y*).
- The circle on the right (B+C) is H(*Y*).
- The part labeled B is the conditional entropy H(*Y*|*X*).
- The part labeled C is the mutual information *I*(*X*;*Y*).

$$H(X) = -\sum_{x\in X} p(x)\log_2 p(x)$$

$$\mathrm{H}(Y|X) \equiv \sum_{x\in\mathcal{X}} p(x)\,\mathrm{H}(Y|X=x)$$

$$\begin{aligned}\mathrm{I}(X;Y) &\equiv \mathrm{H}(X) - \mathrm{H}(X|Y)\\ &\equiv \mathrm{H}(Y) - \mathrm{H}(Y|X)\end{aligned}$$

(Method 1)

$$\mathrm{I}(X;Y) = \sum_{y\in\mathcal{Y}}\sum_{x\in\mathcal{X}} p_{(X,Y)}(x,y)\log\left(\frac{p_{(X,Y)}(x,y)}{p_X(x)\,p_Y(y)}\right)$$

(Method 2)

Calculating Information Gain – Method 1

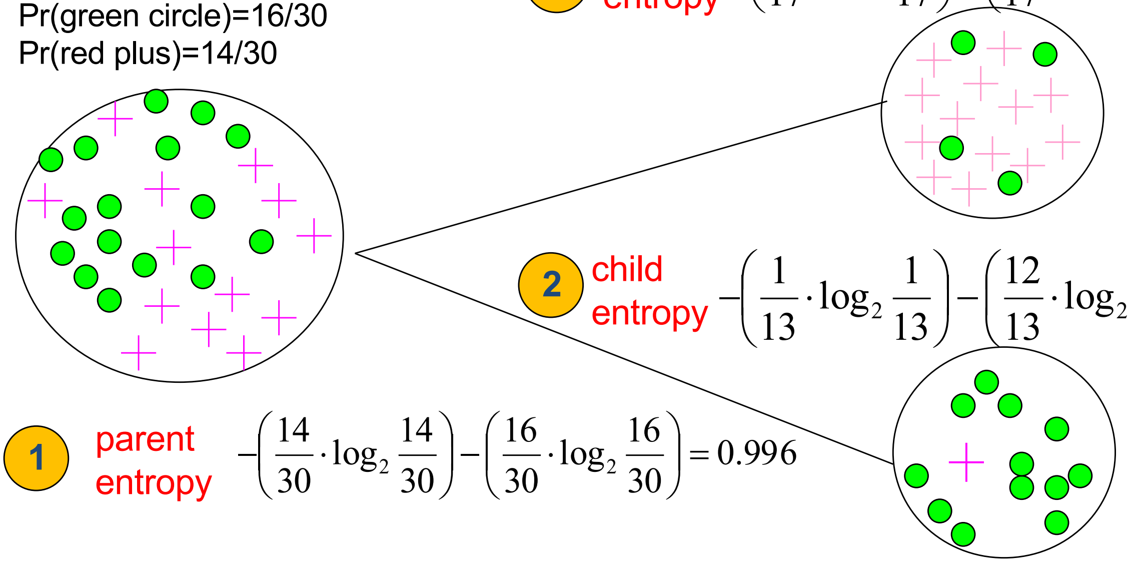
Information Gain = Entropy(parent) – [Average Entropy(children)]

Entire population (30 instances)

Pr(green circle)=16/30

Pr(red plus)=14/30

2 child entropy $-\left(\frac{13}{17}\cdot\log_2\frac{13}{17}\right)-\left(\frac{4}{17}\cdot\log_2\frac{4}{17}\right)=0.787$



2 child entropy $-\left(\frac{1}{13}\cdot\log_2\frac{1}{13}\right)-\left(\frac{12}{13}\cdot\log_2\frac{12}{13}\right)=0.391$

1 parent entropy $-\left(\frac{14}{30}\cdot\log_2\frac{14}{30}\right)-\left(\frac{16}{30}\cdot\log_2\frac{16}{30}\right)=0.996$

3 Average Entropy of Children = $\left(\frac{17}{30}\cdot 0.787\right)+\left(\frac{13}{30}\cdot 0.391\right)=0.615$

4 **Information Gain = 0.996 - 0.615 = 0.38**

Calculating Information Gain – Method 2

- The information gain *I(C;F)* of the class variable *C* with possible values $\{c_1, c_2, \ldots c_m\}$ with respect to the feature variable *F* with possible values $\{f_1, f_2, \ldots, f_d\}$ is defined by:

$$I(C;F) = \sum_{i=1}^{m}\sum_{j=1}^{d} P(C=c_i, F=f_j) log_2 \frac{P(C=c_i, F=f_j)}{P(C=c_i)P(F=f_j)} \qquad \textbf{(1)}$$

- $P(C = c_i)$ is the probability of class *C* having value c_i.
- $P(F=f_j)$ is the probability of feature F having value f_j.
- $P(C=c_i, F=f_j)$ is the joint probability of class $C = c_i$ and variable $F = f_j$.

These are estimated from frequencies in the training data.

Calculating Information Gain – Method 2

X	Y	Z	C
1	1	1	I
1	1	0	I
0	0	1	II
1	0	0	II

How would you distinguish class I from class II?

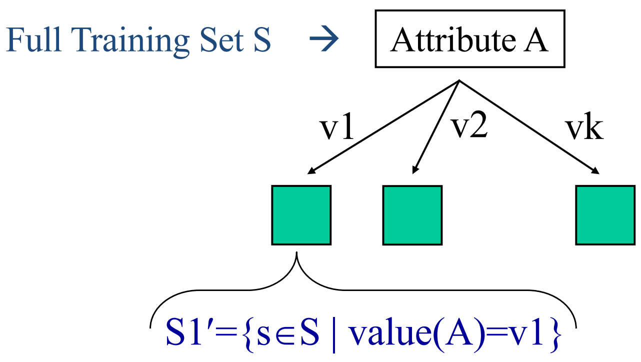
$$
\begin{aligned}
I(C,X) &= P(C=I,X=1)log_2\frac{P(C=I,X=1)}{P(C=I)P(X=1)} \\
&+ P(C=I,X=0)log_2\frac{P(C=I,X=0)}{P(C=I)P(X=0)} \\
&+ P(C=II,X=1)log_2\frac{P(C=II,X=1)}{P(C=II)P(X=1)} \\
&+ P(C=II,X=0)log_2\frac{P(C=II,X=0)}{P(C=II)P(X=0)} \\
&= .5log_2\frac{.5}{.5\times .75} + 0 + .25log_2\frac{.25}{.5\times .25} + .25log_2\frac{.25}{.5\times .75} \\
&= 0.311
\end{aligned} \tag{1}
$$

$$
\begin{aligned}
I(C,Y) &= .5log_2\frac{.5}{.5\times .5} + 0 + .5log_2\frac{.5}{.5\times .5} + 0 \\
&= 1.0
\end{aligned} \tag{2}
$$

$$
\begin{aligned}
I(C,Z) &= .25log_2\frac{.25}{.5\times .5} + .25log_2\frac{.25}{.5\times .5} + .25log_2\frac{.25}{.5\times .5} + .25log_2\frac{.25}{.5\times .5} \\
&= 0.0
\end{aligned} \tag{3}
$$

Which attribute is best? Which is worst? Does it make sense?

Using Information Gain to Construct a Decision Tree



1. Start with the root of the decision tree and the whole training set.
2. Choose the attribute A with highest information gain for the full training set at the root of the tree.
3. Construct child nodes for each value of A. Each has an associated subset of vectors in which A has a particular value.
4. Repeat recursively.

- Quinlan suggested Information Gain in his ID3 system and later the Gain Ratio, both based on Entropy.
- Information Gain has the disadvantage that it prefers attributes with large number of values that split the data into small, pure subsets.
- Quinlan suggested the Gain Ratio to improve this by normalization.
- Reference: Quinlan, J. R. (1986). Induction of decision trees. Machine Learning, 1(1):81-106

Common Measures of impurity

- Entropy – measures uncertainty

$$Entropy = -\sum_j p_j \log_2 p_j \qquad (1)$$

- Gini Index – minimizes the probability of misclassification

$$Gini = 1 - \sum_j p_j^2 \qquad (2)$$

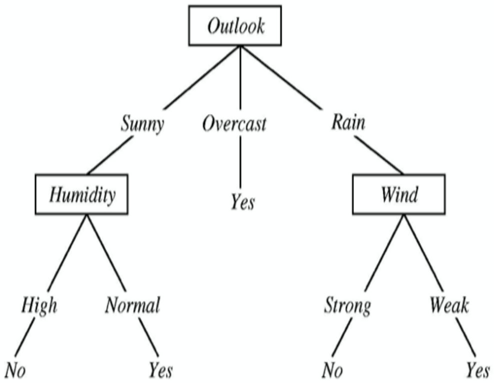
- Classification Error

$$ClassificationError = 1 - \max p_j \qquad (3)$$

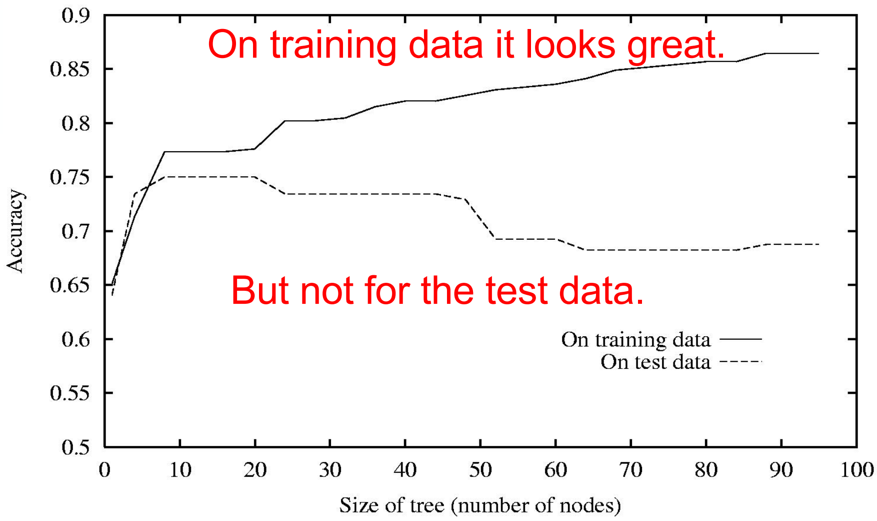
Dealing with non-binary features in Decision Trees

- Features with multiple discrete values
  - Construct a multiway split
  - Test for one value versus all of the others
  - Group the values into two disjoint subsets
- Real-valued features
  - Consider a threshold split using each observed valued of the feature
- Whichever method is used, the information gain can be computed to choose the best split.

Overfitting in Decision Trees



Look at the performance of the Decision Tree on the Training Data and Test Data versus the size of the tree.



On training data it looks great.

But not for the test data.

Consider adding a noisy training sample:

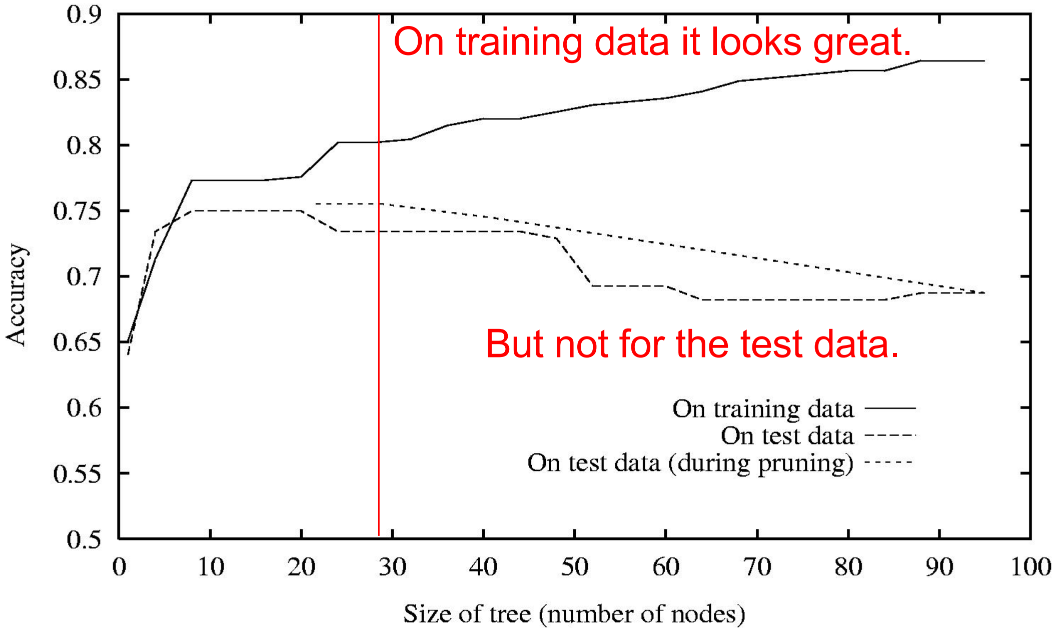
Sunny, Hot, Normal, Strong, PlayTennis=No

What is the effect on the decision tree?

Avoid Overfitting

- Occam's Razor
  - ”If two theories explain the facts equally well, then the simpler theory is to be preferred”
  - Fewer short hypotheses than long hypotheses
  - A short hypothesis that fits the data is unlikely to be a coincidence
  - A long hypothesis that fits the data might be a coincidence
- Stop growing when split not statistically significant
- Grow full tree, then post-prune
  - Prune tree to reduce errors or improve accuracy

Effects of Reduced-error Pruning



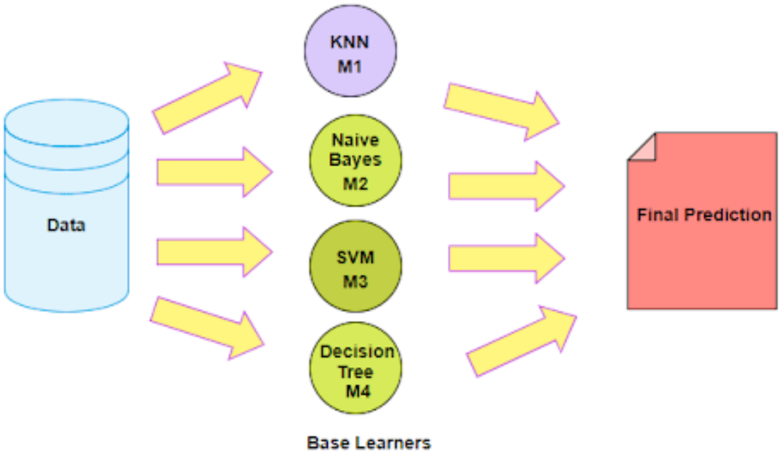
On training data it looks great.

But not for the test data.

The tree is pruned back to the red line where it gives more accurate results on the test data.

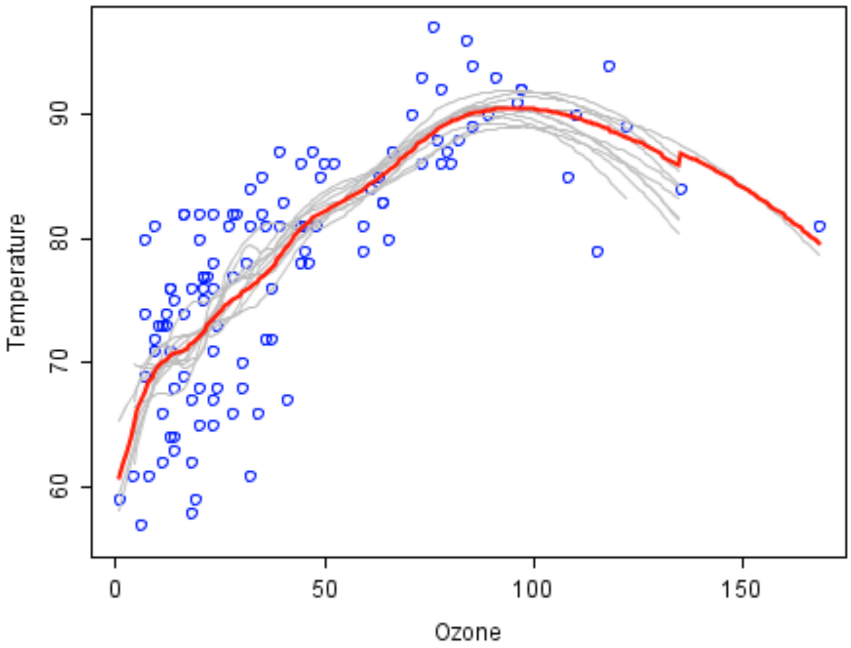
Ensemble Learning

- An ensemble method is a technique that combines the predictions from multiple machine learning algorithms together to make more accurate predictions than any individual model.
- Popular ensemble methods are Bagging, Boosting, and Stacking.
- Ensembling can reduce overfitting without decreasing performance.



Benefits of Ensemble Learning

Relationship between ozone and temperature (data from Rousseeuw and Leroy (1986)



Ensemble average of 100 regression models, each trained on a subset of the original dataset (blue markers).

Individual predictors (gray lines) wiggle a lot and are clearly overfitting.

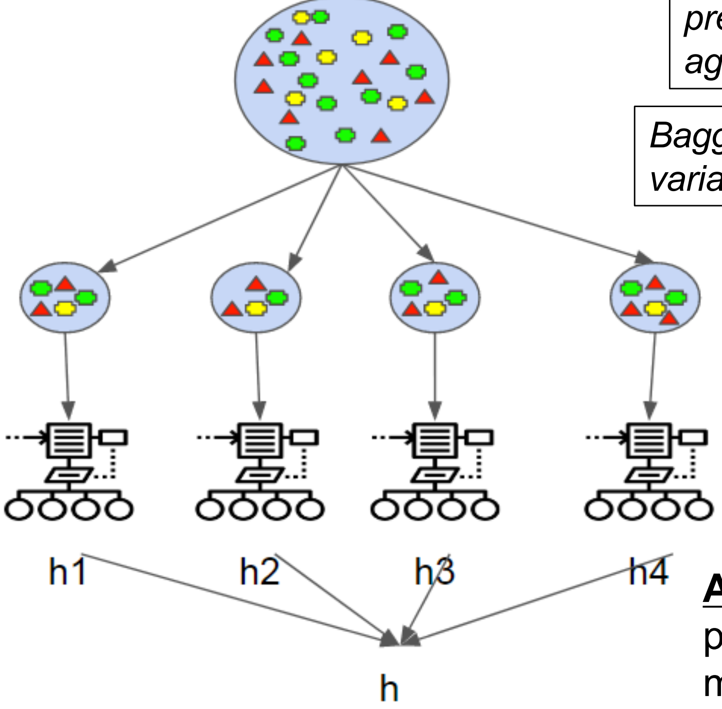
The averaged ensemble predictor (red line) is more stable and less overfitting.

Bagging – Bootstrap Aggregating

Machine Learning Ensemble meta-algorithm

"Bagging predictors is a method for generating multiple versions of a predictor and using these to get an aggregated predictor." - Leo Breiman

Bagging improves accuracy, reduces variance, and helps to prevent overfitting

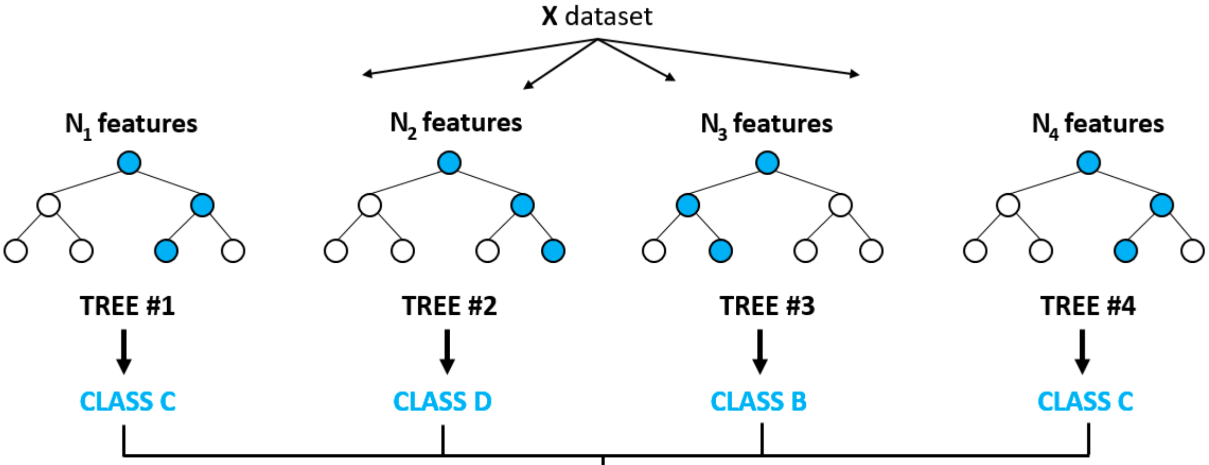


<u>Bootstrapping</u>: Random sampling with replacement

Train <u>multiple</u> decision trees & search <u>all</u> features to find best feature to split on for each tree

<u>Aggregating</u>: Combine multiple predictions via averaging or majority voting

Random Forest



Random Forest is an ensemble of decision trees, trained on different data subsets of the data, whose results are aggregated into one final result.

MAJORITY VOTING

FINAL CLASS

What is the difference between Bagging and Random Forests?

Random Forests reduce variance and limit overfitting without substantially increasing error due to bias.

Performance of Learning Algorithms

- Sources of error
  - High Bias – Learning algorithm is not adequate and is not able to fit the training data (underfitting)
  - High Variance – Algorithm is sensitive to small fluctuations in the training data (overfitting)
  - Irreducible Error – Due to inherent noise in the data
- Bias-Variance Tradeoff – Algorithms with a lower bias have higher variance and vice versa.
- We want a learning algorithm that:
  - captures the regularities in its training data, but also generalizes well to unseen data.
  - has low bias and low variance

Bias-Variance Tradeoff

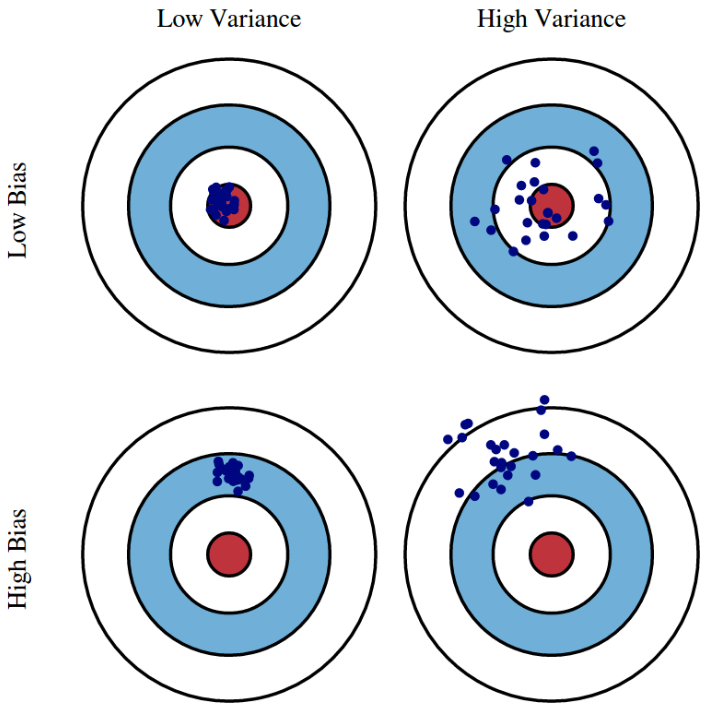
Precision vs. Accuracy

Low Variance
High Precision

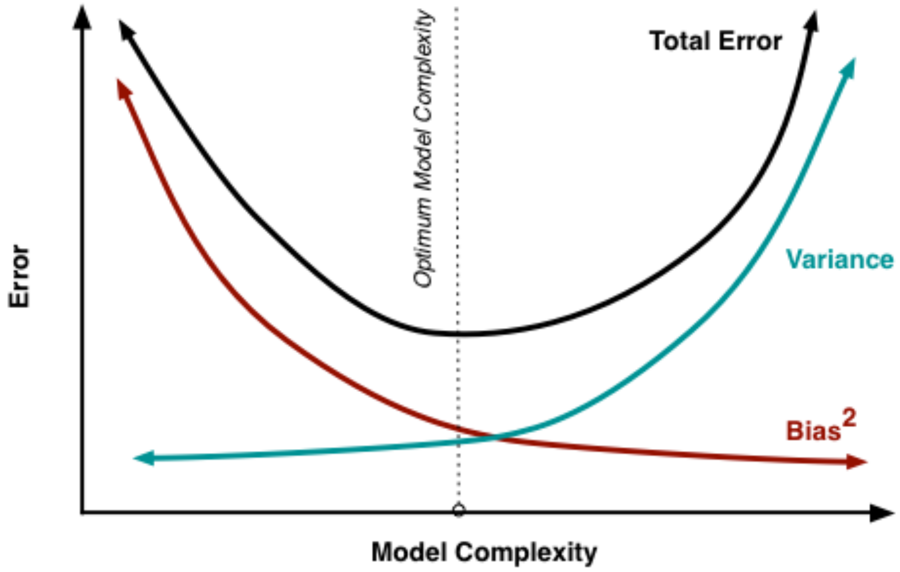
High Variance
Low Precision

Low Bias
High Accuracy

High Bias
Low Accuracy



The Bias squared-Variance Curve



- A curve of squared bias vs variance showing the inverse correlation that is typical of the relation between the two as the model gets more complex.
- It is not uncommon for the resulting Total Error to follow some variant of the U-shape shown in the figure above.

Thank you!

- Please send me your feedback and any questions you may have.
- The best way to contact me is via email:

  mehul.motani@gmail.com
- Thanks for listening!