Lecture 2: Decision Trees

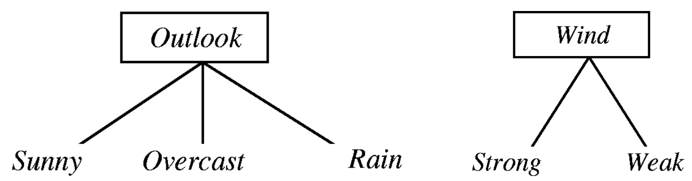
DD2421

Atsuto Maki

Spring, 2020

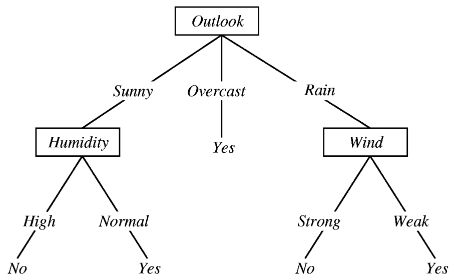
Basic Idea: Test the attributes (features) sequentially
= Ask questions about the target/status sequentially

Example: building a concept of whether someone would like to play tennis.



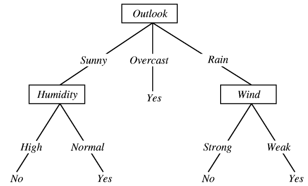
Useful also (but not limited to) when nominal data are involved, e.g. in medical diagnosis, credit risk analysis etc.

The whole analysis strategy can be seen as a tree.



(T. Mitchell, Machine Learning)

Each leaf node bears a category label, and the test pattern is assigned the category of the leaf node reached.



What does the tree encode?

$$(\text{Sunny} \wedge \text{Normal Humidity}) \vee (\text{Cloudy}) \vee (\text{Rainy} \wedge \text{Weak Wind})$$

Logical expressions of the conjunction of decisions along the path.

Arbitrary boolean functions can be represented!

Training: we need to grow a tree from scratch given a set of labeled training data.

How to grow/construct the tree automatically?

1. Choose the best question (according to the information gain), and split the input data into subsets
2. Terminate: call branches with a unique class labels leaves (no need for further quesitons)
3. Grow: recursively extend other branches (with subsets bearing mixtures of labels)

Entropy

How to measure information gain?

The Shannon information content of an outcome is:

$$\log_2 \frac{1}{p_i}$$

(p_i : probability for event i)

The Entropy — measure of uncertainty (unpredictability)

$$\text{Entropy} = \sum_i -p_i \log_2 p_i$$

is a sensible measure of expected information content.

Entropy

Example: tossing a coin
$p_{\text{head}} = 0.5; \quad p_{\text{tail}} = 0.5$

$$\text{Entropy} = \sum_i -p_i \log_2 p_i =$$

$$= -0.5 \log_2 0.5 - 0.5 \log_2 0.5 \quad = -0.5 \underbrace{\log_2 0.5}_{-1} - 0.5 \underbrace{\log_2 0.5}_{-1} =$$

$$= 1$$

The result of a coin-toss has 1 bit of information

Entropy

Example: rolling a die
$p_1 = \frac{1}{6}; \quad p_2 = \frac{1}{6}; \ldots \quad p_6 = \frac{1}{6}$

$$\text{Entropy} = \sum_i -p_i \log_2 p_i =$$

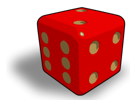
$$= 6 \times \left(-\frac{1}{6} \log_2 \frac{1}{6}\right) =$$

$$= -\log_2 \frac{1}{6} = \log_2 6 \approx 2.58$$

The result of a die-roll has 2.58 bit of information

Entropy

Example: rolling a fake die
$p_1 = 0.1; \ldots \quad p_5 = 0.1; \quad p_6 = 0.5$



$$\text{Entropy} = \sum_i -p_i \log_2 p_i =$$

$$= -5 \cdot 0.1 \log_2 0.1 - 0.5 \log_2 0.5 =$$

$$\approx 2.16$$

A real die is more unpredictable (2.58 bit) than a fake (2.16 bit)

Entropy

Unpredictability of a dataset (think of a subset at a node)

- 100 examples, 42 positive

$$-\frac{58}{100}\log_2\frac{58}{100}-\frac{42}{100}\log_2\frac{42}{100}=0.981$$

- 100 examples, 3 positive

$$-\frac{97}{100}\log_2\frac{97}{100}-\frac{3}{100}\log_2\frac{3}{100}=0.194$$

Back to the decision trees

Smart idea:
Ask about the attribute which maximizes the expected reduction of the entropy.

Information gain
Ask about attribute A for a data set S that has Entropy $\mathrm{Ent}(S)$, and get subsets S_v according to the value of A

$$\mathrm{Gain}=\underbrace{\mathrm{Ent}(S)}_{\text{before}}-\underbrace{\sum_{v\in\mathrm{Values}(A)}\frac{|S_v|}{|S|}}_{\substack{\text{weighted}\\ \text{sum}}}\underbrace{\mathrm{Ent}(S_v)}_{\text{after}}$$

What is the entropy of this binary dataset (attributes=$\{A,B,C,D\}$, $n=25$)?

$\mathrm{Ent}=-\frac{12}{25}\log_2\frac{12}{25}-\frac{13}{25}\log_2\frac{13}{25}\approx\mathbf{0.9988}$

$A=\bullet$: $\frac{3}{6}$ positive $\rightarrow$ 1.0
$A=\circ$: $\frac{9}{19}$ positive $\rightarrow$ 0.9980
Expected: $\frac{6}{25}\cdot 1.0+\frac{19}{25}\cdot 0.9980\approx\mathbf{0.9985}$

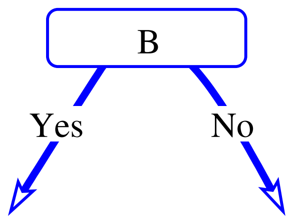
$B=\bullet$: $\frac{9}{11}$ positive $\rightarrow$ 0.684
$B=\circ$: $\frac{3}{14}$ positive $\rightarrow$ 0.750
Expected: **0.721**

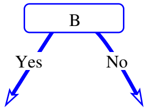
$C=\bullet$: $\frac{6}{12}$ positive $\rightarrow$ 1.0
$C=\circ$: $\frac{6}{13}$ positive $\rightarrow$ 0.9957
Expected: **0.9977**

$D=\bullet$: $\frac{3}{5}$ positive $\rightarrow$ 0.9710
$D=\circ$: $\frac{9}{20}$ positive $\rightarrow$ 0.9928
Expected: **0.9884**

A	B	C	D	
○	●	●	○	+
●	●	○	○	+
○	○	○	○	
○	○	●	●	+
○	●	○	○	+
●	○	●	○	
○	●	●	○	+
○	○	○	○	
●	○	○	○	
○	●	●	○	+
○	○	○	●	+
○	○	●	○	
●	●	●	○	+
○	●	○	●	
○	○	○	○	
○	○	●	○	
○	●	●	●	
○	●	○	○	+
○	○	●	○	
●	○	○	○	
○	●	○	○	+
○	○	●	●	+
●	●	○	○	+
○	○	○	○	
○	○	●	○	

$\mathrm{Gain}(A)=0.9988-0.9985=\mathbf{0.0003}$
$\mathrm{Gain}(B)=0.9988-0.7210=\mathbf{0.2778}$
$\mathrm{Gain}(C)=0.9988-0.9977=\mathbf{0.0011}$
$\mathrm{Gain}(D)=0.9988-0.9884=\mathbf{0.0104}$
Attribute B gives most information gain





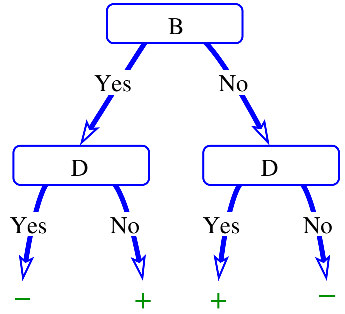
Examples where
$B = \bullet$

A	B	C	D	
○	●	●	○	+
●	●	○	○	+
○	●	○	○	+
○	●	●	○	+
○	●	●	○	+
●	●	●	○	+
○	●	○	●	
○	●	●	●	
○	●	○	○	+
○	●	○	○	+
●	●	○	○	+

Examples where
$B = \circ$

A	B	C	D	
○	○	○	○	
○	○	●	●	+
●	○	●	○	
○	○	○	○	
●	○	○	○	
○	○	○	●	+
○	○	●	○	
○	○	○	○	
○	○	●	○	
●	○	○	○	
○	○	●	○	
○	○	●	●	+
○	○	○	○	
○	○	●	○	

Decision Trees
Unpredictability
Overfitting
Applications
Entropy
Information gain
Gini impurity



Atsuto Maki
Lecture 2: Decision Trees

Decision Trees
Unpredictability
Overfitting
Applications
Entropy
Information gain
Gini impurity

Greedy approach to choose a question:
Choose the attribute which tells us most about the answer

In sum, we need to find good questions to ask.
(more than one attribute could be involved in one question)

Atsuto Maki
Lecture 2: Decision Trees

Decision Trees
Unpredictability
Overfitting
Applications
Entropy
Information gain
Gini impurity

Gini impurity: Another definition of predictability (impurity).

$$\sum_i p_i(1 - p_i) = 1 - \sum_i p_i^2$$

(p_i : probability for event i)

The expected error rate at a node, N, if the category label is randomly selected from the class distribution present at N.

Similar to the entropy but more strongly peaked at equal probabilities.

Atsuto Maki
Lecture 2: Decision Trees

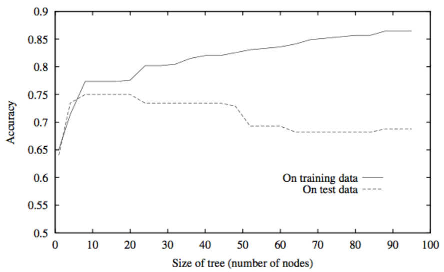
Overfitting

When the learned models are overly specialized for the training samples.

Good results on training data, but generalizes poorly.

When does this occur?

- Non-representative sample
- Noisy examples
- Too complex model



(T. Mitchell, Machine Learning)

What can be done about it?
Choose a simpler model and accept some errors for the training examples

Which hypothesis should be preferred when several are compatible with the data?

Occam's principle (Occam's razor)

William from Ockham, Theologian and Philosopher (1288–1348)

"Entities should not be multiplied beyond necessity"

The simplest explanation compatible with data tends to be the right one

Separate the available data into two sets of examples

- *Training* set T: to form the learned model
- *Validation* set V: to evaluate the accuracy of this model

The motivations:

- The training may be misled by random errors, but the validation set is unlikely to exhibit the same random fluctuations
- The validation set to provide a safety check against overfitting the spurious characteristics of the training set

(V need be large enough to provide statistically meaningful instances)

Reduced-Error Pruning

Split data into *training* and *validation* set

Do until further pruning is harmful:

- Evaluate impact on *validation* set of pruning each possible node (plus those below it)
- Greedily remove the one that most improves *validation* set accuracy

Produces smallest version of most accurate subtree

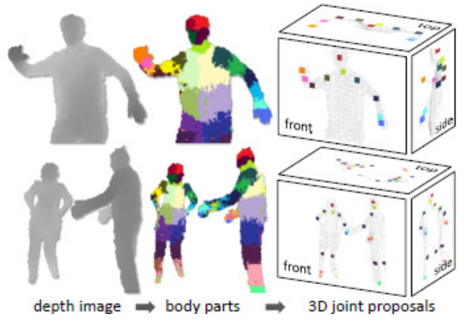
Possible ways of improving/extending the decision trees

- Avoid overfitting
  - Stop growing when data split not statistically significant
  - Grow full tree, then post-prune (e.g. Reduced error pruning)

A collection of trees (Ensemble learning: in Lecture 10)

- Bootstrap aggregating (bagging)
- Decision Forests

Example: Human pose estimation



(Picture courtesy of J. Shotton et al)