Decision Trees

COMP5318 Machine Learning and Data Mining

semester 1, 2021, week 5a

Irena Koprinska

Reference: Witten ch.4.3 and ch.6.1, Tan ch.3.3

THE UNIVERSITY OF SYDNEY

Outline

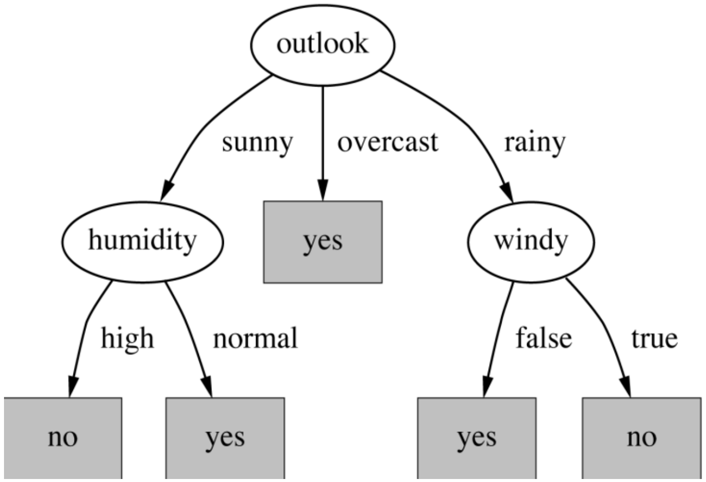
- Constructing decision tree
- Entropy and information gain
- Pruning decision trees
- Dealing with numeric attributes
- Dealing with highly branching attributes – gain ratio

Decision trees

- The most popular and well researched ML and DM method
- Developed in parallel
  - in ML by Ross Quinlan (University of Sydney)
    - ID3 algorithm, 1986; C4.5, 1993; other versions
  - in statistics by Breiman et al.
    - CART algorithm, 1984
- The most popular version is C4.5

What is a decision tree?

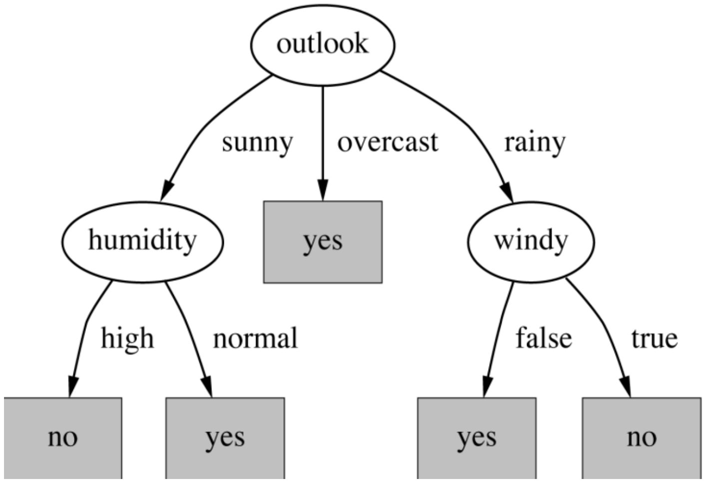
- each *non-leaf node* the corresponds to a test for the values of an attribute
- each *branch* corresponds to an attribute value
- each *leaf node* assigns a class



outlook	temp.	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

What is a decision tree? (2)

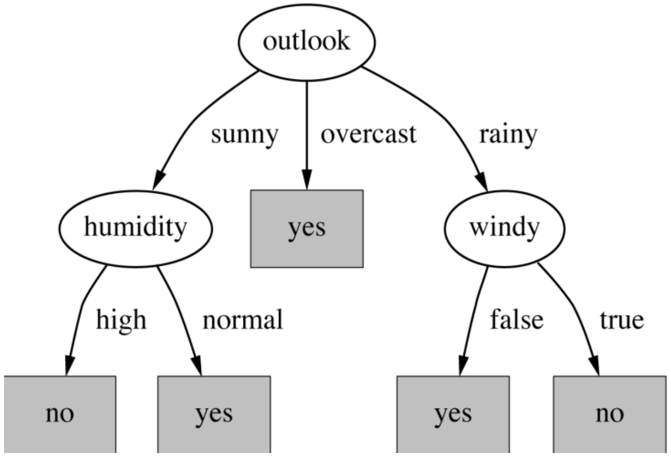
- To predict the class for a new example:
  - start from the root and test the values of the attributes, until you reach a leaf node; return the class of the leaf node
- What would be the prediction for this new example?
  - outlook=rainy, temperature=cool, humidity=high, windy=false



outlook	temp.	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

Constructing decision trees

- Strategy: top-down learning using recursive *divide-and-conquer* process:
  - First: Select the best attribute for root node and create branch for each possible attribute value
  - Then: Split examples into subsets, one for each branch extending from the node
  - Finally: Repeat recursively for each branch, using only the examples that reach the branch
- Stop if all examples have the same class
  - Make a leaf node for this class



Example

- Assume that we know how to select the best attribute

name	Uni degree	work exp. >3y	sex	class
David	yes	no	M	reject
Anna	no	yes	F	reject
Simon	yes	yes	M	accept
Kate	yes	yes	F	accept

Uni degree

yes

name	Uni degree	work exp. >3y	sex	class
David	yes	no	M	reject
Simon	yes	yes	M	accept
Kate	yes	yes	F	accept

no

name	Uni degree	work exp. >3y	sex	class
Anna	no	yes	F	reject

reject

Example (2)

name	Uni degree	work exp. >3y	sex	class
David	yes	no	M	reject
Simon	yes	yes	M	accept
Kate	yes	yes	F	accept

work exp. >3y

yes

name	Uni degree	work exp. >3y	sex	class
Simon	yes	yes	M	accept
Kate	yes	yes	F	accept

accept

no

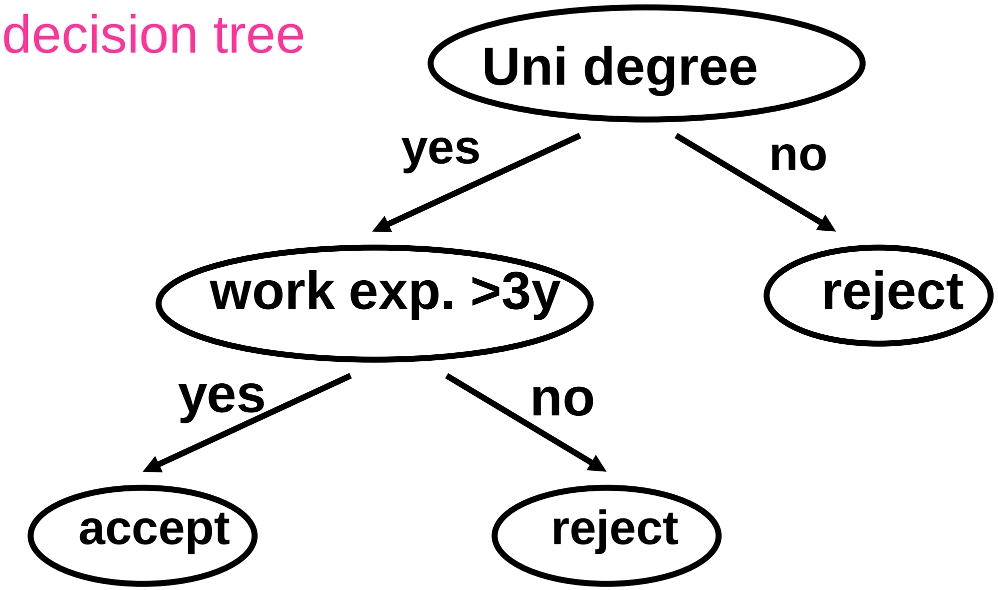
name	Uni degree	work exp. >3y	sex	class
David	yes	no	M	reject

reject

Example (3)

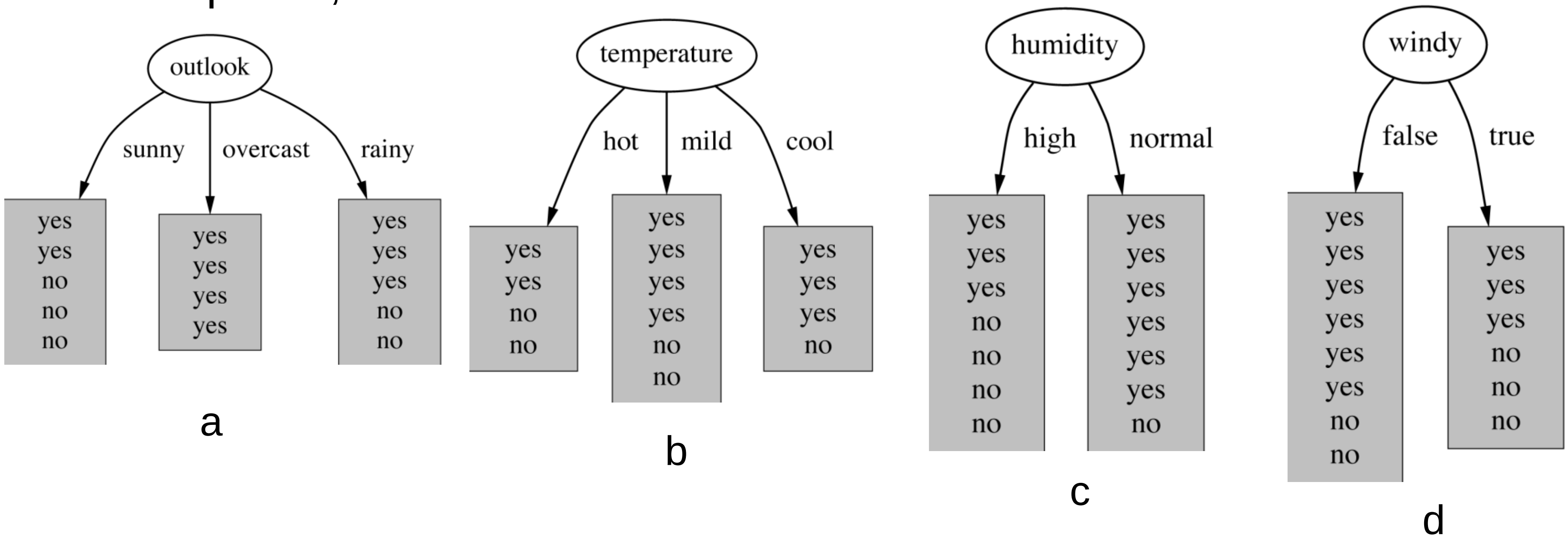
name	Uni degree	work exp. >3y	sex	class
David	yes	no	M	reject
Anna	no	yes	F	reject
Simon	yes	yes	M	accept
Kate	yes	yes	F	accept

Final decision tree



Which attribute to select?

- Four options, which one is the best choice?



- A leaf node with only 1 class (yes or no) will not have to be split further and the recursive process will terminate
- We would like this to happen as soon as possible as we seek small trees
- If we have a measure of *purity* of each node, we can choose the attribute that produces the purest partitions

Entropy

- The measure of purity that we will use is called *information gain*
- It is based on another measure called *entropy*

- Given a set of examples with their class, entropy measures the *homogeneity* (purity) of this set with respect to the class
- The smaller the entropy, the greater the purity of the data set

- Entropy is used also in signal compression, information theory and physics

Entropy - definition

- Entropy H(S) of data set S:

$$H(S) = I(S) = -\sum_i P_i . \log_2 P_i$$

P_i - proportion of examples that belong to class *i*

- Different notation used in textbooks, we will use H(S) and I(S)

- For our example: weather data - 9 yes and 5 no examples:

$$H(S) = -P_{yes}\log_2 P_{yes} - P_{no}\log_2 P_{no} = I(P_{yes}, P_{no}) = I(\frac{9}{14}, \frac{5}{14}) = -\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14} = 0.940\ bits$$

- The entropy is measured in bits

- When calculating entropy, we will assumed that $\log_2 0 = 0$

Range of entropy for binary tasks

- 2 classes: yes and no
- on x: p, the proportion of positive examples
- (the proportion of negative examples will be 1-p)
- on y: the entropy H(S)

$$H(S) = I(p,(1-p)) =$$
$$= -p\log_2 p - (1-p)\log_2(1-p)$$

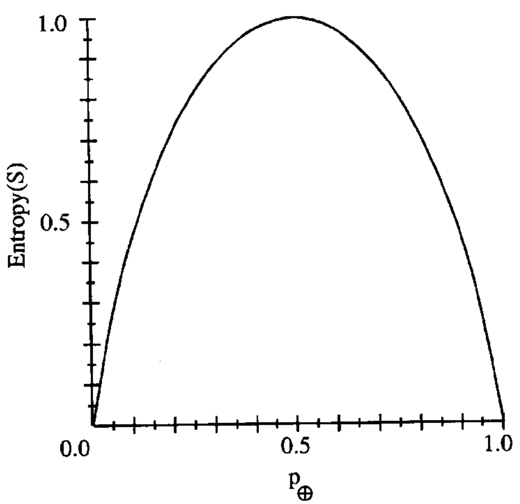


Image from Tom Mitchell, Machine Learning, McGraw Hill, 1997

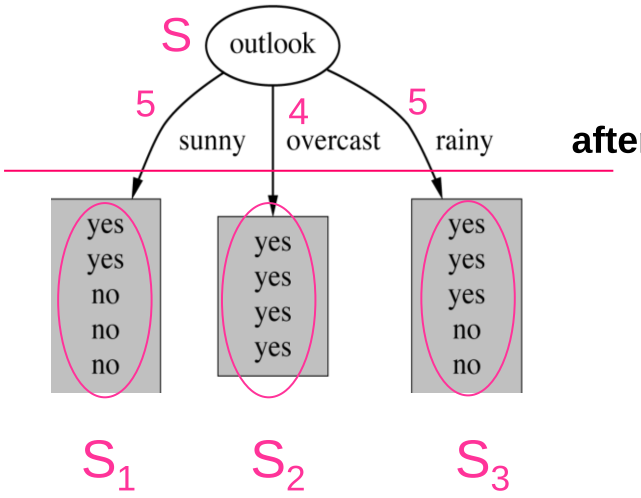
- H(S) $\in$ [0,1]
- H(S)=0 => all elements of S belong to the same class (max purity, min value of entropy)
- H(S)=1 => equal number of yes & no (min purity, max value of entropy)

Information gain

- *Information gain* measures the reduction in entropy caused by using an attribute to partition the set of training examples
- The best attribute is the one with the highest information gain (i.e. with the biggest reduction in entropy)

- It is a difference of 2 entropies: Gain = *T1 –T2*
- *T1* is the entropy of the set of examples S associated with the parent node before the split
- T2 is the remaining entropy in S, after S is split by the attribute (e.g. outlook)
- The larger the difference, the higher the information gain
- The best attribute is the one with highest information gain
  - it reduces most the entropy of the parent node

Before splitting: 9 yes & 5 no



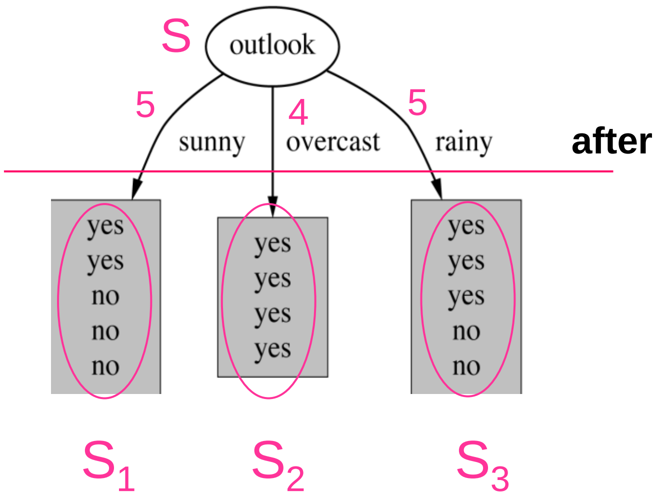
Calculating information gain

- Let's calculate the information gain of the attribute *outlook*
- *T1* is the entropy of the set of examples S before the split:

$$T1 = H(S) = I(\frac{9}{14}, \frac{5}{14}) = 0.940\ bits$$

- T2 is the remaining entropy in S, after S is split by the attribute
- It takes into consideration the entropies of the child nodes and the distribution of the examples along each child node
  - e.g. for a split on outlook, it will consider the entropies of S1, S2 and S3 and the proportion of examples following each branch (5/14, 4/14, 5/15):

Before splitting: 9 yes & 5 no

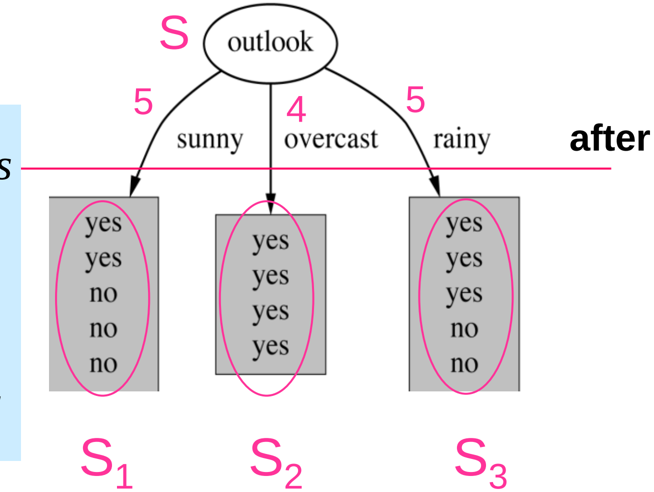


$$T2 = H(S \mid outlook) = \frac{5}{14} \cdot H(S_1) + \frac{4}{14} \cdot H(S_2) + \frac{5}{14} \cdot H(S_3)$$

Calculating information gain (2)

$$T2 = H(S \mid outlook) = \frac{5}{14} \cdot H(S_1) + \frac{4}{14} \cdot H(S_2) + \frac{5}{14} \cdot H(S_3)$$

Before splitting: 9 yes & 5 no



$$H(S \mid outlook = sunny) = I(\frac{2}{5}, \frac{3}{5}) = -\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5} = 0.971 bits$$

$$H(S \mid outlook = overcast) = I(\frac{4}{4}, \frac{0}{4}) = -\frac{4}{4}\log_2\frac{4}{4} - \frac{0}{4}\log_2\frac{0}{4} = 0 bits$$

$$H(S \mid outlook = rainy) = I(\frac{3}{5}, \frac{2}{5}) = -\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5} = 0.971 bits$$

$$H(S \mid outlook) = \frac{5}{14} \cdot 0.971 + \frac{4}{14} \cdot 0 + \frac{5}{14} \cdot 0.971 = 0.693 bits$$

Gain(S|outlook)=H(S)-H(S|outlook)= 0.940-0.693=0.247 bits

Calculating information gain (3)

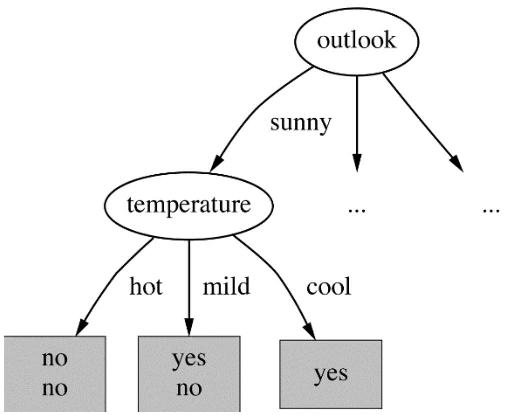
$$Gain(S|outlook)=H(S)-H(S|outlook)= 0.940\text{-}0.693=0.247 \text{ bits}$$

- Similarly, the information gain for the other three attributes is:

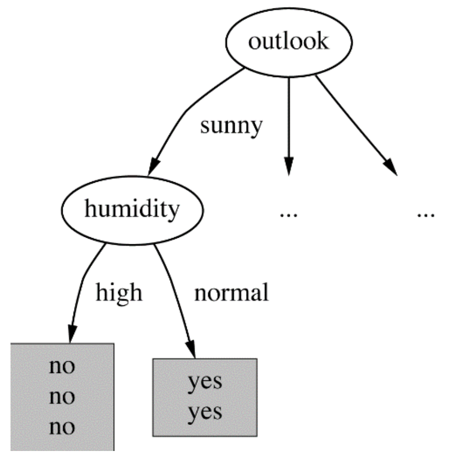
$$Gain(S|temperature)=0.029 \ bits$$
$$Gain(S|humidity)=0.152 \ bits$$
$$Gain(S|windy)=0.048 \ bits$$

- => we select outlook as it has the highest information gain

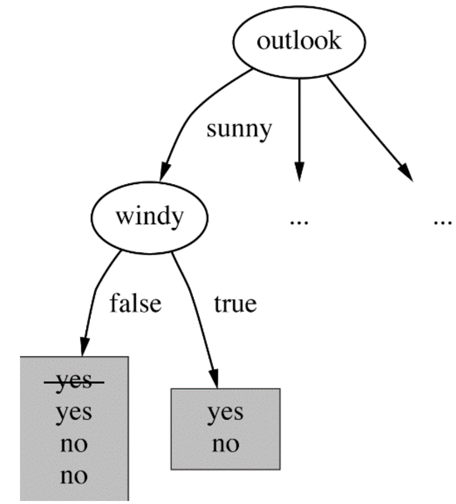
Continuing to split and final tree



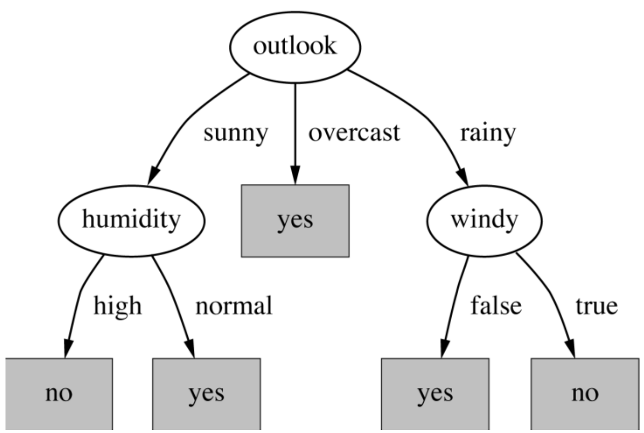
Gain(S, temperature)=0.571 bits



Gain(S, humidity)=0.971 bits



Gain(S, windy)=0.020 bits



Final DT

Task

- Ex.1 from the theoretical tutorial exercises (t5.pdf)
- Given is the following set of training examples:

shape	color	class
circle	blue	+
circle	blue	+
square	blue	-
triangle	blue	-
square	red	+
square	blue	-
square	red	+
circle	red	+

You may use this table to calculate information gain:

x	y	-(x/y)* log_2(x/y)	x	y	-(x/y)* log_2(x/y	x	y	-(x/y)* log_2(x/y	x	y	-(x/y)* log_2(x/y
1	2	0.50	4	5	0.26	6	7	0.19	5	9	0.47
1	3	0.53	1	6	0.43	1	8	0.38	7	9	0.28
2	3	0.39	5	6	0.22	3	8	0.53	8	9	0.15
1	4	0.5	1	7	0.40	5	8	0.42	1	10	0.33
3	4	0.31	2	7	0.52	7	8	0.17	3	10	0.52
1	5	0.46	3	7	0.52	1	9	0.35	7	10	0.36
2	5	0.53	4	7	0.46	2	9	0.48	9	10	0.14
3	5	0.44	5	7	0.35	4	9	0.52			

a) What is the entropy of this collection of training examples?
b) What is the information gain of the attribute *shape*?

Solution a)

- a) What is the entropy of this collection of training examples?

shape	color	class
circle	blue	+
circle	blue	+
square	blue	-
triangle	blue	-
square	red	+
square	blue	-
square	red	+
circle	red	+

You may use this table to calculate information gain:

x	y	-(x/y)* $\log_2$(x/y)	x	y	-(x/y)* $\log_2$(x/y	x	y	-(x/y)* $\log_2$(x/y	x	y	-(x/y)* $\log_2$(x/y
1	2	0.50	4	5	0.26	6	7	0.19	5	9	0.47
1	3	0.53	1	6	0.43	1	8	0.38	7	9	0.28
2	3	0.39	5	6	0.22	3	8	0.53	8	9	0.15
1	4	0.5	1	7	0.40	5	8	0.42	1	10	0.33
3	4	0.31	2	7	0.52	7	8	0.17	3	10	0.52
1	5	0.46	3	7	0.52	1	9	0.35	7	10	0.36
2	5	0.53	4	7	0.46	2	9	0.48	9	10	0.14
3	5	0.44	5	7	0.35	4	9	0.52			

H(S)=I(5/8, 3/8) = -5/8 log(5/8)-3/8 log(3/8) = 0.42 + 0.53 = 0.95 bits

Solution b)

- b) What is the information gain of the attribute *shape*?

shape	color	class
circle	blue	+
circle	blue	+
square	blue	-
triangle	blue	-
square	red	+
square	blue	-
square	red	+
circle	red	+

You may use this table to calculate information gain:

x	y	-(x/y)* $\log_2$(x/y)	x	y	-(x/y)* $\log_2$(x/y	x	y	-(x/y)* $\log_2$(x/y	x	y	-(x/y)* $\log_2$(x/y
1	2	0.50	4	5	0.26	6	7	0.19	5	9	0.47
1	3	0.53	1	6	0.43	1	8	0.38	7	9	0.28
2	3	0.39	5	6	0.22	3	8	0.53	8	9	0.15
1	4	0.5	1	7	0.40	5	8	0.42	1	10	0.33
3	4	0.31	2	7	0.52	7	8	0.17	3	10	0.52
1	5	0.46	3	7	0.52	1	9	0.35	7	10	0.36
2	5	0.53	4	7	0.46	2	9	0.48	9	10	0.14
3	5	0.44	5	7	0.35	4	9	0.52			

Split on *shape*:

$H(S_{circle}) = I(3/3, 0/3) = -3/3 \log(3/3)-0/3 \log(0/3) = 0 + 0 = 0$ bits

$H(S_{square}) = I(2/4, 2/4) = -2/4 \log(2/4)-2/4 \log(2/4) = 0.5 + 0.5 = 1$ bit

$H(S_{triangle}) = I(0/1, 1/1) = -0/1 \log(0/1)-1/1 \log(1/1)= 0 + 0 = 0$ bits

$H(S|shape) = 3/8*0 + 4/8*1 + 1/8*0 = 0.5$ bits

$gain(shape) = 0.95 - 0.5 = 0.45$ bits

Pruning Decision Trees

Pruning decision trees

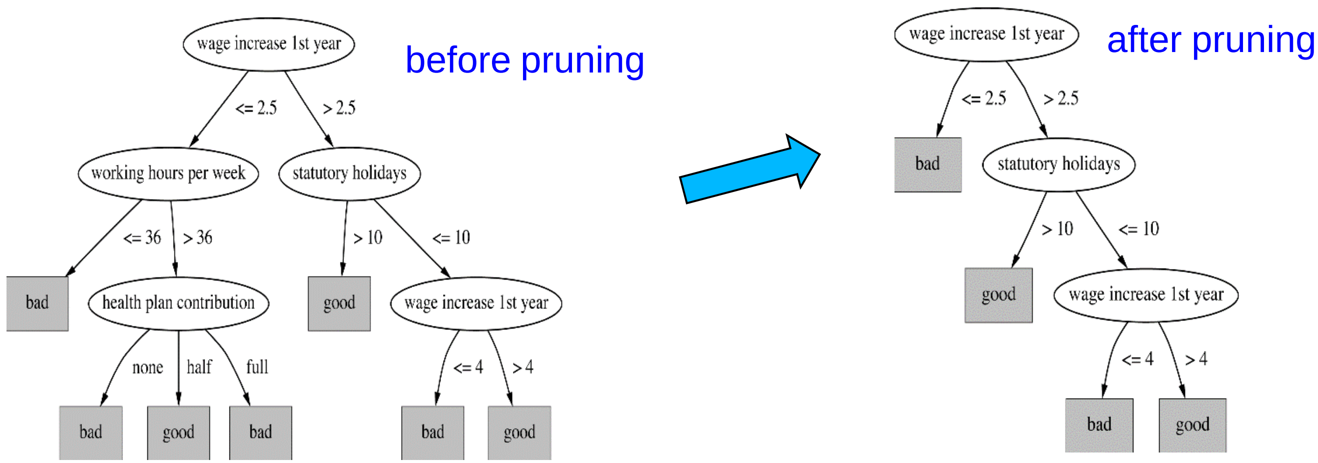
- If we grow the decision tree to perfectly classify the training set, the tree may become too specific and overfit the data
- Overfitting – high accuracy on the training data but low accuracy on new data
  - The tree has become too specific, mainly memorizing data, instead of extracting patterns
- When does overfitting occurs in decision trees?
  - Training data is too small -> not enough representative examples to build a model that can generalize well on new data
  - Noise in the training data, e.g. incorrectly labelled examples -> the decision tree learns them by adding new braches and making the tree overly specific
- Use tree pruning to avoid overfitting

Strategies for pruning decision trees

- Two main strategies
  - Pre-pruning - stop growing the tree earlier, before it reaches the point where it perfectly classifies the training data
  - Post-pruning – fully grow the tree, allowing it to perfectly cover the training data, and then prune it
- Post-pruning is preferred in practice
- Different post-pruning methods, e.g.:
  - sub-tree replacement
  - sub-tree raising
  - converting the tree to rules and then pruning them
- How much to prune? Use a validation set to decide
  - => the available data is split into 3 sets: training, validation and test set
    - training set - to build the tree
    - validation set - to evaluate the impact of pruning and decide when to stop
    - test data – to evaluate how good the tree is

Pruning by sub-tree replacement - idea

- Bottom-up – from the bottom of the tree to the root
- Each non-leaf node is a candidate for pruning, for each node:
  - Remove the sub-tree rooted at it
  - Replace it with a leaf with class=majority class of examples that go along the candidate node
  - Compare the new tree with the old tree by calculating the accuracy on the validation set for both
  - If the accuracy of the new tree is better or the same as the accuracy of the old tree, keep the new tree (i.e. prune the candidate node)
  - For more information see Witten ch.6.1



Numerical Attributes

Discretizing numeric attributes

- Numerical attributes need to be discretized, i.e. converted into nominal
- Standard method: binary splits, e.g. temp<45
- Unlike nominal attributes, every numerical attribute has many possible splits
- Discretization procedure:
  - Sort the examples according the values of the numerical attribute
  - Split points – whenever the class changes, halfway
  - Evaluate information gain or other measure for every possible split point and choose the best one
  - Information gain for best split point = information gain for the attribute

Example

- Values of *temperature*:

64	65	68	69	70	71	72	73	74	75	80	81	83	85
yes	no	yes	yes	yes	no	no	no	yes	yes	no	yes	yes	no

- 7 possible splits; let's consider the split between 70 and 71
- Calculate Information gain for:
  - temperature < 70.5 : 4 yes & 1 no
  - temperature =>70.5 : 4 yes & 5 no

$$H(S) = -\frac{8}{14}\log_2\frac{8}{14} - \frac{6}{14}\log_2\frac{6}{14} = 0.985\,bits$$

$$H(S_{temp<70.5}) = -\frac{4}{5}\log_2\frac{4}{5} - \frac{1}{5}\log_2\frac{1}{5} = 0.722\,bits$$

$$H(S_{temp=>70.5}) = -\frac{4}{9}\log_2\frac{4}{9} - \frac{5}{9}\log_2\frac{5}{9} = 0.991\,bits$$

$$H(S\,|\,temp70.5) = \frac{5}{14}0.722 + \frac{9}{14}0.991 = 0.895\,bits$$

$$Gain(S\,|\,temp70.5) = 0.985 - 0.895 = 0.09\,bits$$

outlook	temp.	humidity	windy	play
sunny	85	high	false	no
sunny	80	high	true	no
overcast	83	high	false	yes
rainy	70	high	false	yes
rainy	68	normal	false	yes
rainy	65	normal	true	no
overcast	64	normal	true	yes
sunny	73	high	false	no
sunny	69	normal	false	yes
rainy	74	normal	false	yes
sunny	75	normal	true	yes
overcast	72	high	true	yes
overcast	81	normal	false	yes
rainy	71	high	true	no

Alternatives to information gain

Highly branching attributes are problematic

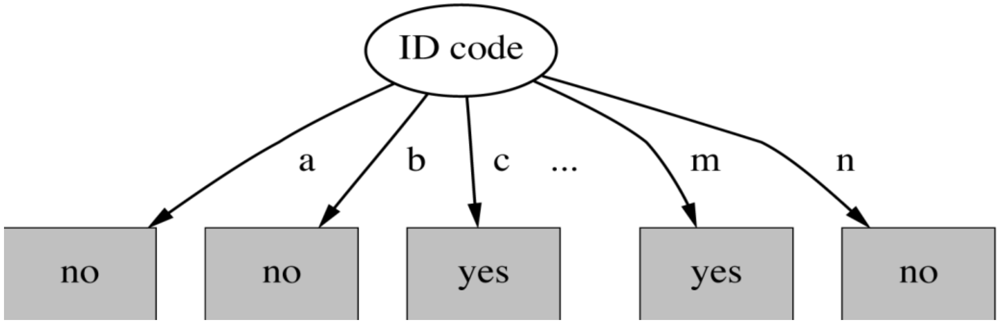
- If an attribute is highly-branching (with a large number of values), information gain will select it!
- Reason: highly-branching attributes are more likely to create pure subsets
  - => pure subsets have low entropy -> high information gain
- Example: imagine using ID code as one of the attributes; it will split the training data into 1-example subsets, its information gain will be high and it will be selected as the best attribute
- This is likely to lead to overfitting

Example – weather data with ID code

- If an attribute is highly-branching (with a large number of values), information gain will select it

Weather data with ID code

ID code	outlook	temp.	humidity	windy	play
a	sunny	hot	high	false	no
b	sunny	hot	high	true	no
c	overcast	hot	high	false	yes
d	rainy	mild	high	false	yes
e	rainy	cool	normal	false	yes
f	rainy	cool	normal	true	no
g	overcast	cool	normal	true	yes
h	sunny	mild	high	false	no
i	sunny	cool	normal	false	yes
j	rainy	mild	normal	false	yes
k	sunny	mild	normal	true	yes
l	overcast	mild	high	true	yes
m	overcast	hot	normal	false	yes
n	rainy	mild	high	true	no



- All single instance subsets have entropy=0
- This means the information gain is maximal for the ID code attribute (namely 0.940 bits)

Gain ratio

- *Gain ratio* is a modification of information gain that reduces its bias towards highly branching attributes
- It takes into account the number of branches when choosing an attribute and penalizes highly-branching attributes

Decision trees - summary

- Very popular ML technique
- Top-down learning using recursive divide-and-conquer process
- Easy to implement
- Interpretable
  - The produced tree is easy to visualize and understand by non-experts and clients
  - Interpretability increases the trust in using the machine learning model in practice
- Uses pruning to prevent overfitting
- Numeric attributes are converted into nominal (binary split)
- Selecting the best attribute – information gain, gain ratio, others
- Variations: purity can be measured in different ways, e.g. CART uses Gini Index not entropy