Decision Trees

an Introduction

Universiteit Leiden

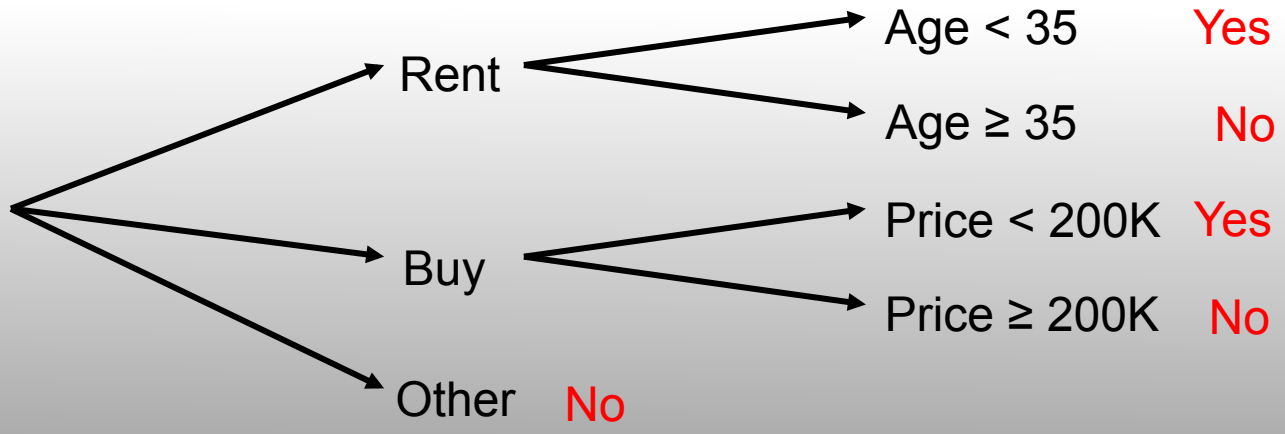
Outline

- Top-Down Decision Tree Construction
- Choosing the Splitting Attribute
- Information Gain and Gain Ratio

Universiteit Leiden

Decision Tree

- An internal node is a test on an attribute
- A branch represents an outcome of the test, e.g., house = Rent
- A leaf node represents a class label or class label distribution
- At each node, one attribute is chosen to split training examples into distinct classes as much as possible
- A new case is classified by following a matching path to a leaf node

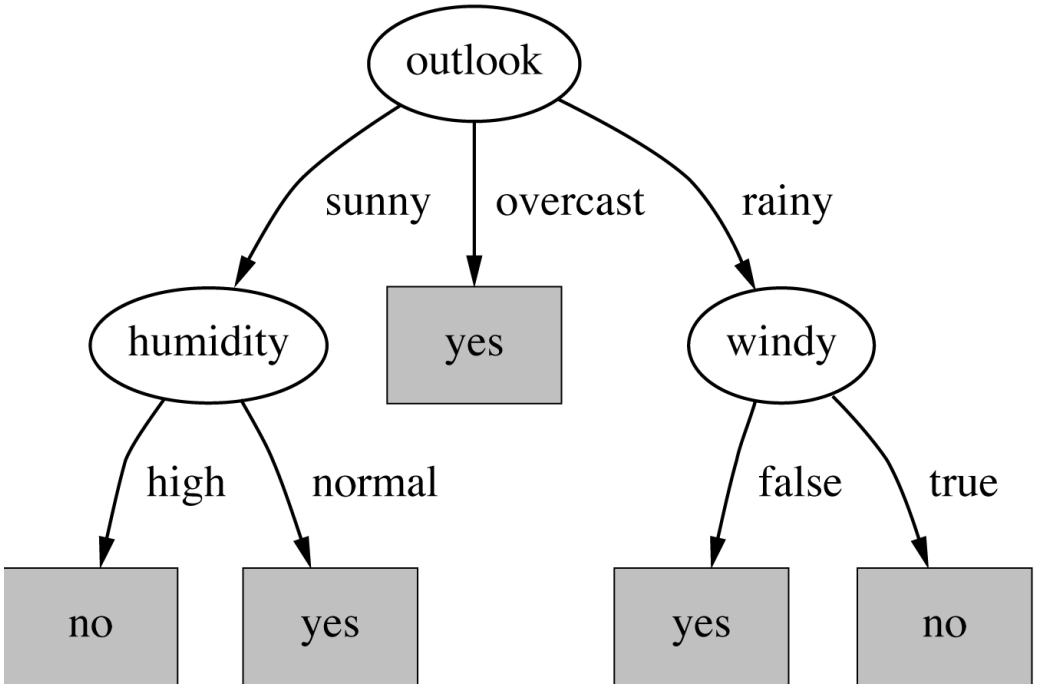


Weather Data: Play tennis or not

Outlook	Temperature	Humidity	Windy	Play?
sunny	hot	high	false	No
sunny	hot	high	true	No
overcast	hot	high	false	Yes
rain	mild	high	false	Yes
rain	cool	normal	false	Yes
rain	cool	normal	true	No
overcast	cool	normal	true	Yes
sunny	mild	high	false	No
sunny	cool	normal	false	Yes
rain	mild	normal	false	Yes
sunny	mild	normal	true	Yes
overcast	mild	high	true	Yes
overcast	hot	normal	false	Yes
rain	mild	high	true	No

Universiteit Leiden

Example Tree for “Play tennis”



Building Decision Tree [Q93]

- Top-down tree construction
  - At start, all training examples are at the root
  - Partition the examples recursively by choosing one attribute each time.
- Bottom-up tree pruning
  - Remove subtrees or branches, in a bottom-up manner, to improve the estimated accuracy on new cases
  - Discussed next week

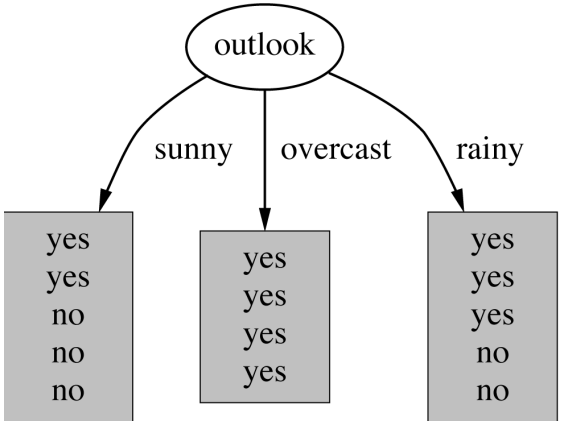
Choosing the Splitting Attribute

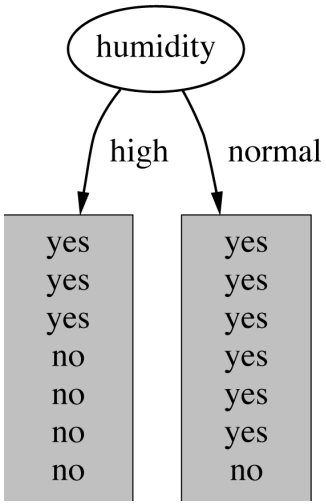
- At each node, available attributes are evaluated on the basis of separating the classes of the training examples
- A *goodness function* is used for this purpose
- Typical goodness functions:
  - information gain (ID3/C4.5)
  - information gain ratio
  - gini index

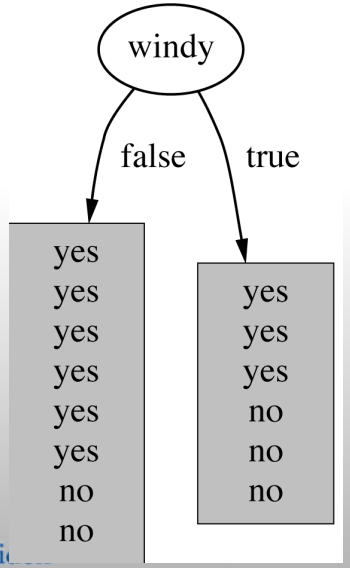
A criterion for attribute selection

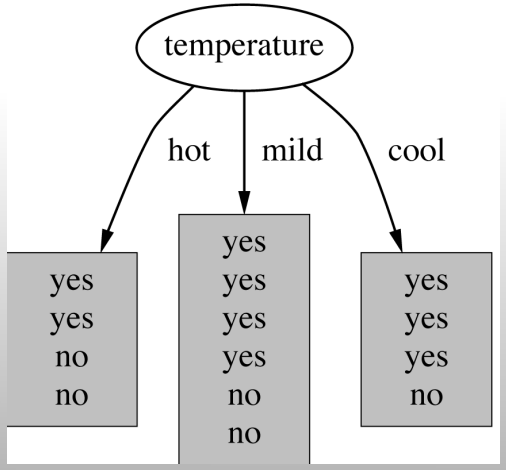
- Which is the best attribute?
  - The one which will result in the smallest tree
  - Heuristic: choose the attribute that produces the "purest" nodes
- Popular *impurity criterion*: *information gain*
  - Information gain increases with the average purity of the subsets that an attribute produces
  - Information gain uses entropy H(*p*)
- Strategy: choose attribute that results in greatest information gain

Which attribute to select?

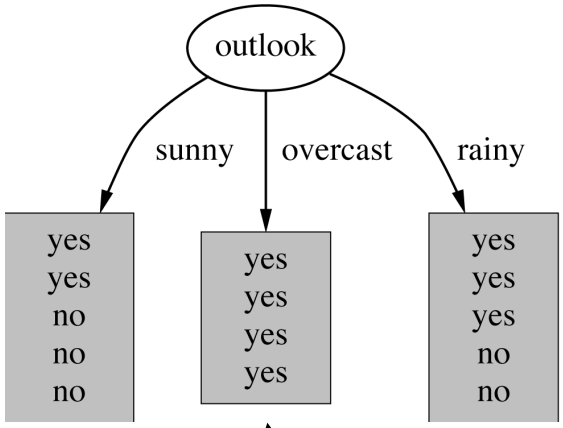






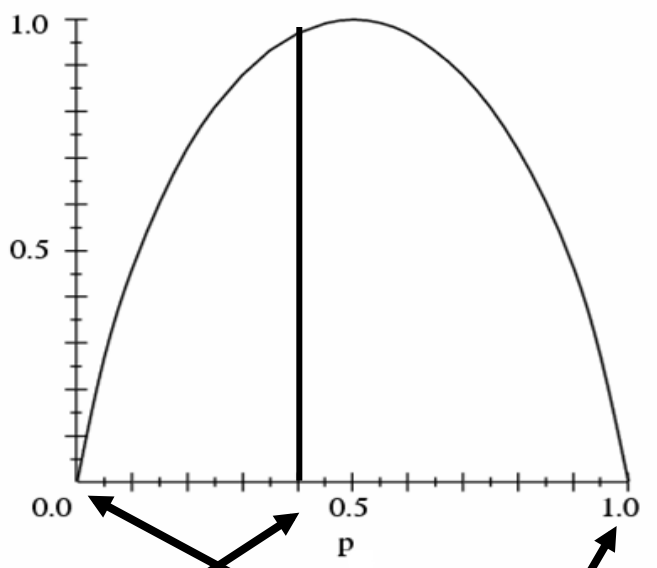


Consider entropy H(*p*)



pure, 100% yes

not pure at all, 40% yes



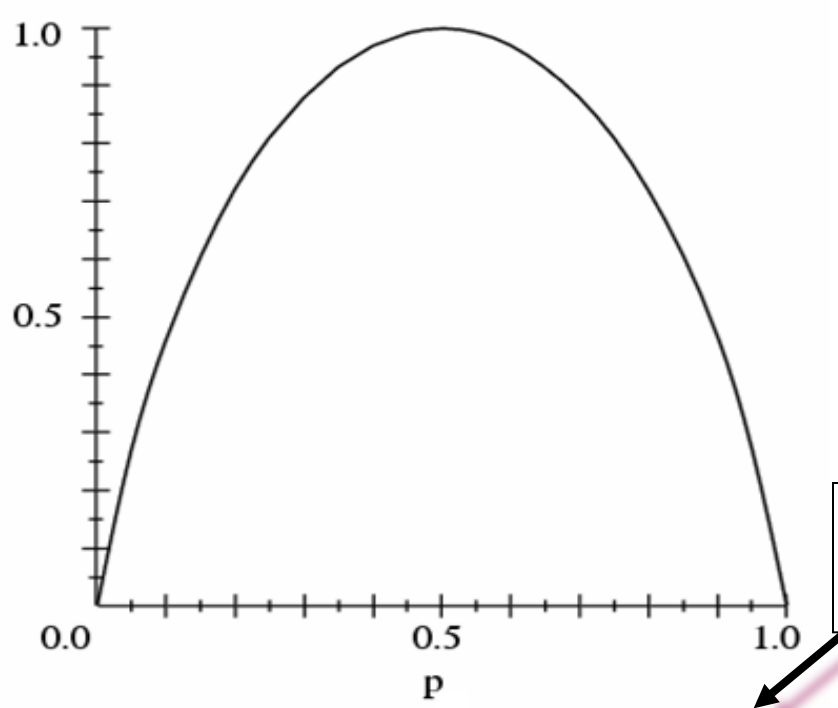
pure, 100% yes

done

not pure at all, 40% yes

allmost 1 bit of information required to distinguish yes and no

Entropy



$\log(p)$ is the 2-log of p

Entropy: $H(p) = -p\log(p) - (1-p)\log(1-p)$

$H(0) = 0$ pure node, distribution is skewed

$\text{entropy}(p_1, p_2, \ldots, p_n) = -p_1\log p_1 - p_2\log p_2 \ldots - p_n\log p_n$

$H(0.5) = 1$ mixed node, equal distribution

Universiteit Leiden

Example: attribute "Outlook"

- "Outlook" = "Sunny":

$$\text{info}([2,3]) = \text{entropy}(2/5,3/5) = -2/5\log(2/5 \quad \text{71 bits}$$

Note: log(0) is not defined, but we evaluate 0*log(0) as zero

- "Outlook" = "Overcast":

$$\text{info}([4,0]) = \text{entropy}(1,0) = -1\log(1) - 0\log(0) = 0 \text{ bits}$$

- "Outlook" = "Rainy":

$$\text{info}([3,2]) = \text{entropy}(3/5,2/5) = -3/5\log(3/5) - 2/5\log(2/5) = 0.971 \text{ bits}$$

- Expected information for "Outlook":

$$\text{info}([3,2],[4,0],[3,2]) = (5/14)\times 0.971 + (4/14)\times 0 + (5/14)\times 0.971$$
$$= 0.693 \text{ bits}$$

Computing the information gain

- Information gain:

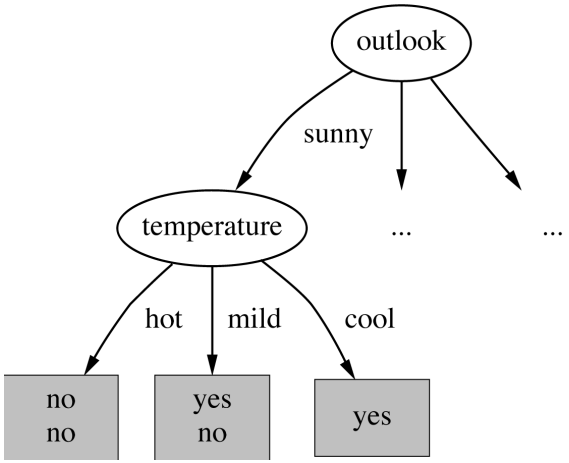
(information before split) – (information after split)

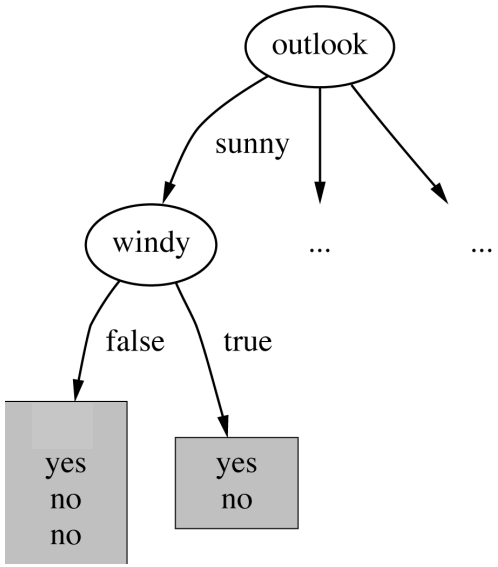
$$\text{gain("Outlook")} = \text{info}([9,5]) - \text{info}([2,3],[4,0],[3,2]) = 0.940 - 0.693$$
$$= 0.247 \text{ bits}$$

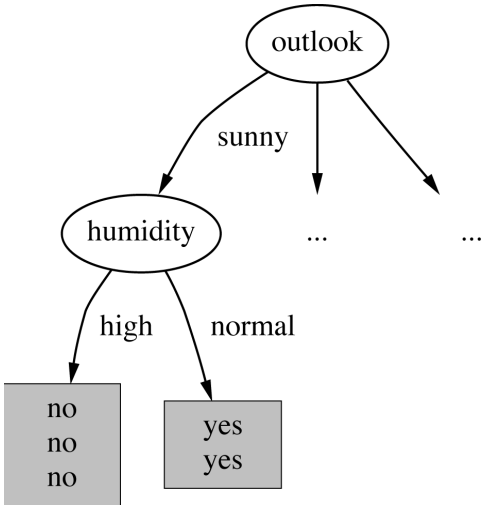
- Information gain for attributes from weather data:

$$\text{gain("Outlook")} = 0.247 \text{ bits}$$
$$\text{gain("Temperature")} = 0.029 \text{ bits}$$
$$\text{gain("Humidity")} = 0.152 \text{ bits}$$
$$\text{gain("Windy")} = 0.048 \text{ bits}$$

Continuing to split







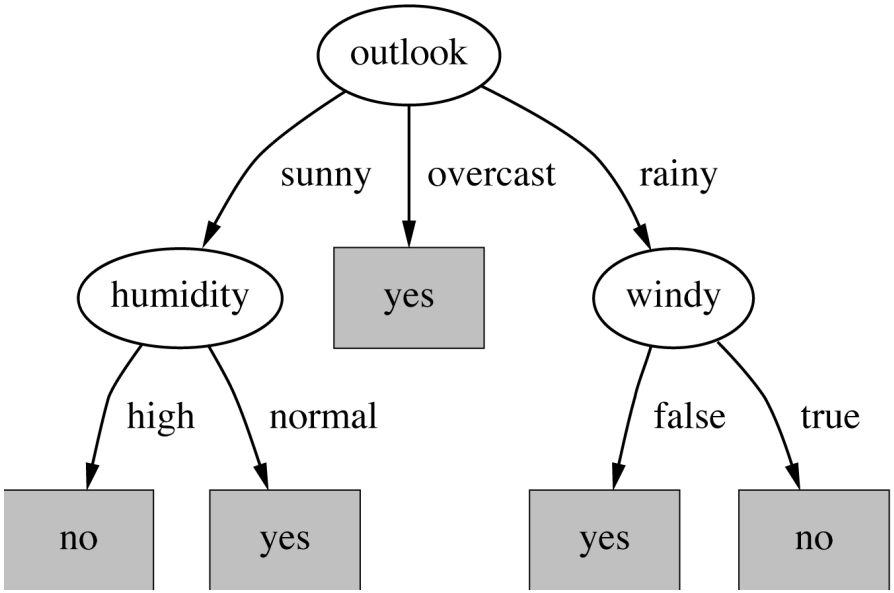
$$\text{gain("Temperature")} = 0.571\,\text{bits}$$

$$\text{gain("Windy")} = 0.020\,\text{bits}$$

$$\text{gain("Humidity")} = 0.971\,\text{bits}$$

Universiteit Leiden

The final decision tree



- Note: not all leaves need to be pure; sometimes identical instances have different classes
  - ⇒ Splitting stops when data can't be split any further

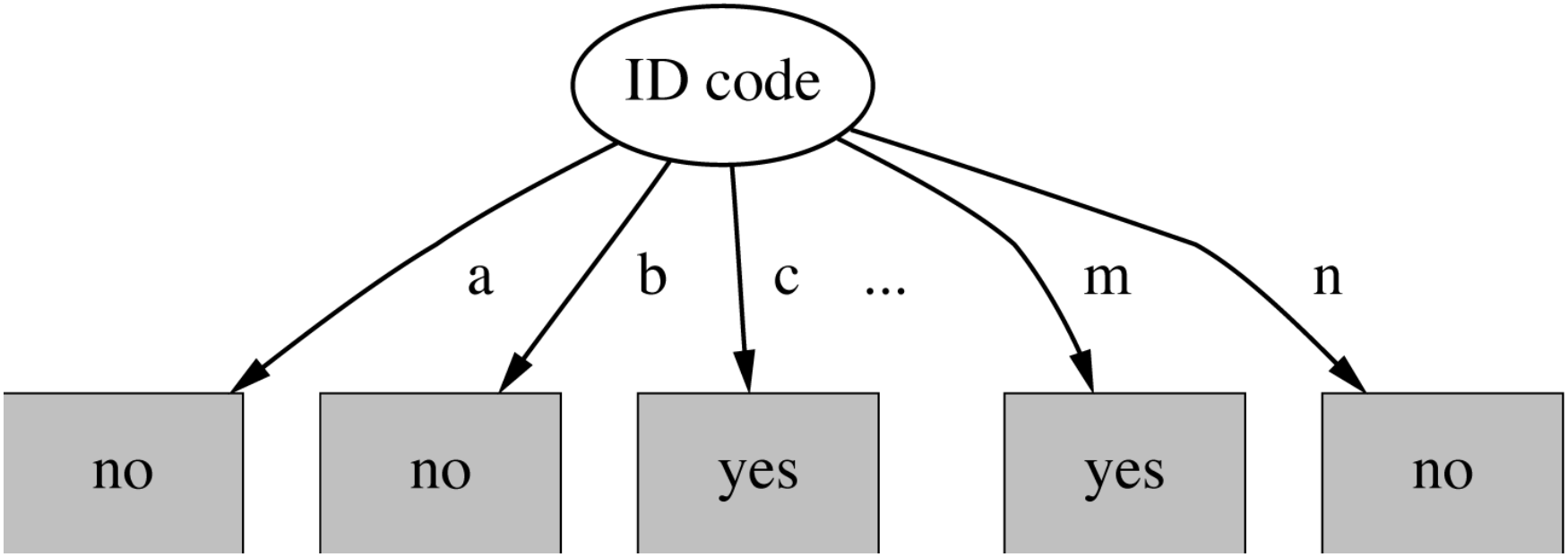
Highly-branching attributes

- Problematic: attributes with a large number of values (extreme case: customer ID)
- Subsets are more likely to be pure if there is a large number of values
  - Information gain is biased towards choosing attributes with a large number of values
  - This may result in *overfitting* (selection of an attribute that is non-optimal for prediction)

Weather Data with ID

ID	Outlook	Temperature	Humidity	Windy	Play?
A	sunny	hot	high	false	No
B	sunny	hot	high	true	No
C	overcast	hot	high	false	Yes
D	rain	mild	high	false	Yes
E	rain	cool	normal	false	Yes
F	rain	cool	normal	true	No
G	overcast	cool	normal	true	Yes
H	sunny	mild	high	false	No
I	sunny	cool	normal	false	Yes
J	rain	mild	normal	false	Yes
K	sunny	mild	normal	true	Yes
L	overcast	mild	high	true	Yes
M	overcast	hot	normal	false	Yes
N	rain	mild	high	true	No

Split for ID attribute



Entropy of split = 0 (since each leaf node is “pure”, having only one case.

Information gain is maximal for ID

Gain ratio

- *Gain ratio*: a modification of the information gain that reduces its bias on high-branch attributes
- Gain ratio should be
  - Large when data is evenly spread
  - Small when all data belong to one branch
- Gain ratio takes number and size of branches into account when choosing an attribute
  - It corrects the information gain by taking the *intrinsic information* of a split into account (i.e. how much info do we need to tell which branch an instance belongs to)

Gain Ratio and Intrinsic Info.

- Intrinsic information: entropy of distribution of instances into branches

$$IntrinsicInfo(S,A) \equiv -\sum \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}.$$

- *Gain ratio* (Quinlan '86) normalizes info gain by:

$$GainRatio(S,A) = \frac{Gain(S,A)}{IntrinsicInfo(S,A)}.$$

Universiteit Leiden

Computing the gain ratio

- Example: intrinsic information for ID

$$\text{info}([1,1,\ldots,1]) = 14 \times (-1/14 \times \log 1/14) = 3.807 \text{ bits}$$

- Importance of attribute decreases as intrinsic information gets larger
- Example:

$$\text{gain\_ratio}(\text{"ID\_code"}) = \frac{0.940 \text{ bits}}{3.807 \text{ bits}} = 0.246$$

Gain ratios for weather data

Outlook		**Temperature**	
Info:	0.693	Info:	0.911
Gain: 0.940-0.693	0.247	Gain: 0.940-0.911	0.029
Split info: info([5,4,5])	1.577	Split info: info([4,6,4])	1.362
Gain ratio: 0.247/1.577	0.156	Gain ratio: 0.029/1.362	0.021

Humidity		**Windy**	
Info:	0.788	Info:	0.892
Gain: 0.940-0.788	0.152	Gain: 0.940-0.892	0.048
Split info: info([7,7])	1.000	Split info: info([8,6])	0.985
Gain ratio: 0.152/1	0.152	Gain ratio: 0.048/0.985	0.049

More on the gain ratio

- "Outlook" still comes out top
- However: "ID" has greater gain ratio
  - Standard fix: *ad hoc* test to prevent splitting on that type of attribute
- Problem with gain ratio: it may overcompensate
  - May choose an attribute just because its intrinsic information is very low
  - Standard fix:
    - First, only consider attributes with greater than average information gain
    - Then, compare them on gain ratio