BASIC METHODS 2 - ID3

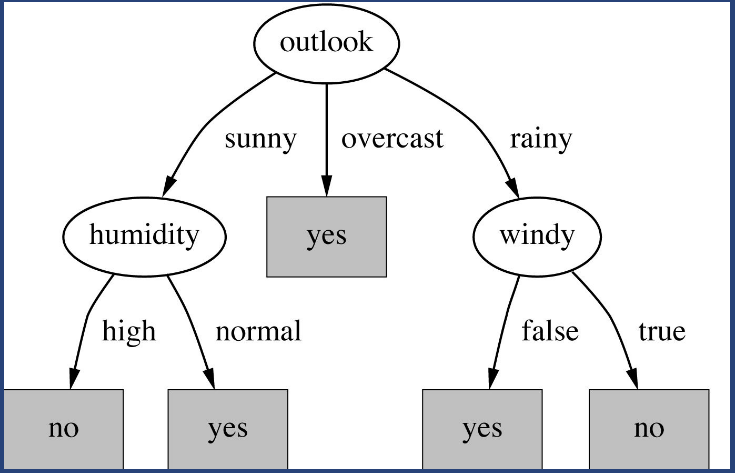
Gréta Pataki

29 September, 2021

Decision trees

- Classification and regression trees (categorical & numerical data handling)

- Splits dataset into small subsets, final result: tree with
  - root node
  - decision nodes: branches -> possible values for the attribure
  - leaf nodes: represents a classification



- Which feature splits the data better (which is the best attribute)?

ID3 (Iterative Dichotomiser 3)

- Core algorithm for building decision trees
  - top-down, greedy search to test each attribute at every node of the tree

- Which is the best attribute?
  - the one which will result in the smallest tree
  - choose the attribute that produces the "purest" nodes
  - information gain (IG):
    - [information before splitting] – [information after splitting]
    - is used to construct a tree
  - best attribute: gives maximum IG (minimum entropy)

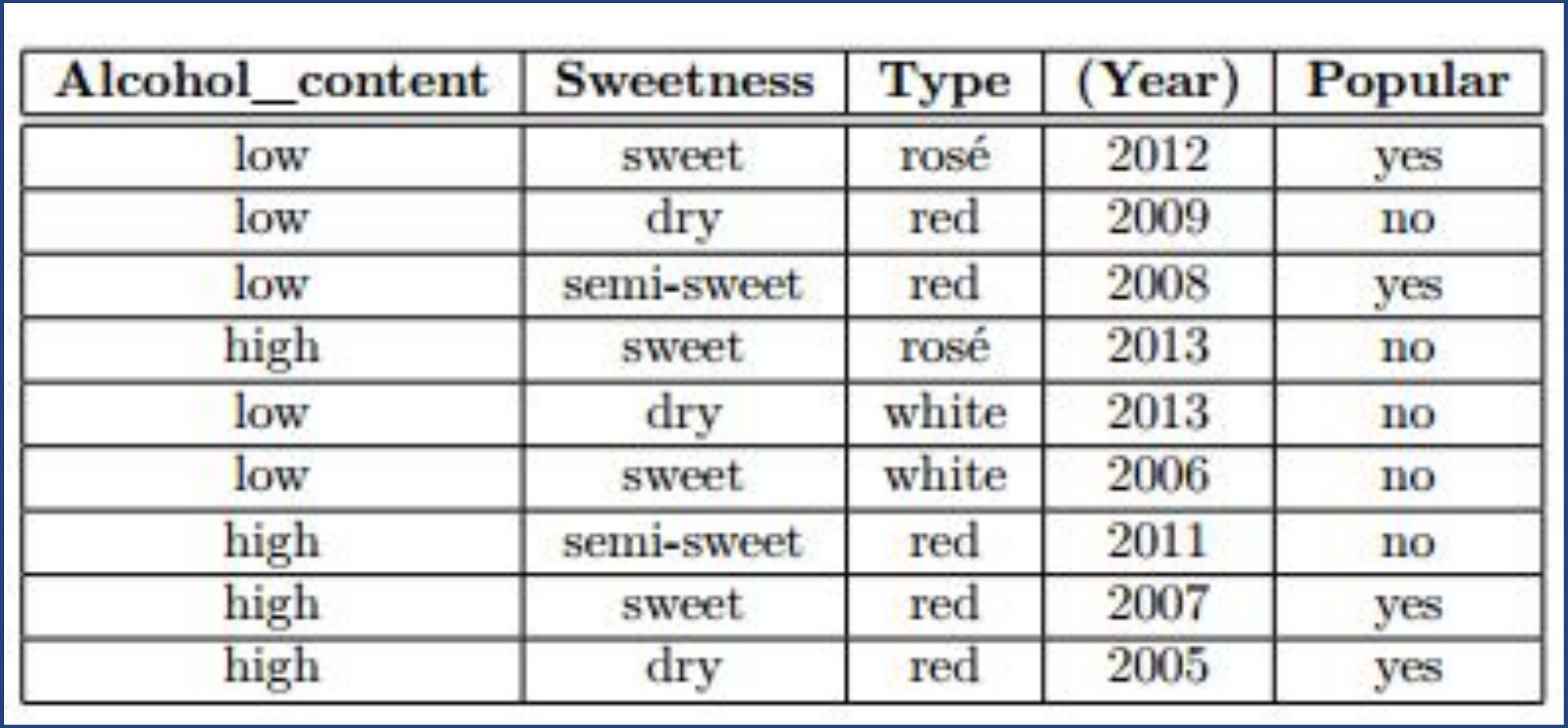
- Entropy: measure of randomness
  - unbiased coin toss (head and tail is equally likely): E = 1
  - biased (2 head): E = 0
  - ID3 uses entropy to calculate the homogeneity of a sample

Entropy

- Information is measured in *bits*
  - – Given a probability distribution, the info required to predict an event is the distribution's *entropy*
  - – Entropy gives the information required in bits (this can involve fractions of bits!)
- Formula for computing the entropy:

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

Wine dataset



Alcohol_content	Sweetness	Type	(Year)	Popular
low	sweet	rosé	2012	yes
low	dry	red	2009	no
low	semi-sweet	red	2008	yes
high	sweet	rosé	2013	no
low	dry	white	2013	no
low	sweet	white	2006	no
high	semi-sweet	red	2011	no
high	sweet	red	2007	yes
high	dry	red	2005	yes

IG: [information before splitting] – [information after splitting]

- Calculate for each attribute

Alcohol_content

- Information before splitting:

  Info[4,5] = entropy(4/9,5/9)

Alcohol_content

- Information before splitting:

  Info[4,5] = entropy(4/9,5/9)

$$H(X) = -p \log_2 p - (1-p) \log_2 (1-p)$$

Alcohol_content

- Information before splitting:

Info[4,5] = entropy(4/9,5/9) = -4/9

$$H(X) = -p \log_2 p - (1-p) \log_2 (1-p)$$

Alcohol_content

- Information before splitting:

Info[4,5] = entropy(4/9,5/9) = -4/9*log(4/9)

$$H(X) = -p \log_2 p - (1-p) \log_2 (1-p)$$

Alcohol_content

- Information before splitting:

Info[4,5] = entropy(4/9,5/9) = -4/9*log(4/9) – 5/9

$$H(X) = -p\log_2 p - (1-p)\log_2(1-p)$$

Alcohol_content

- Information before splitting:

Info[4,5] = entropy(4/9,5/9) = -4/9*log(4/9) – 5/9*log(5/9)

$$H(X) = -p \log_2 p - (1-p) \log_2 (1-p)$$

Alcohol_content

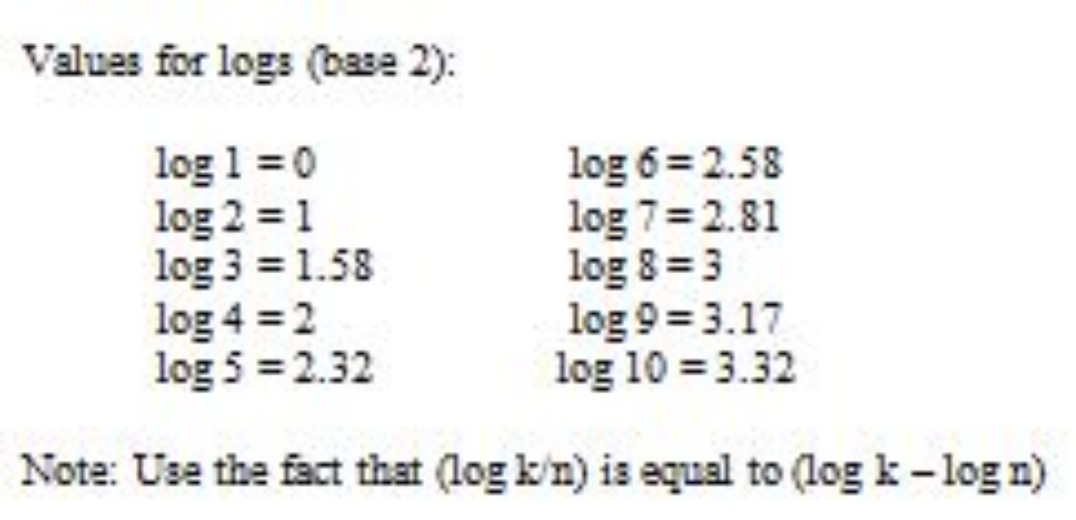
- Information before splitting:

  $\mathrm{Info}[4,5] = \mathrm{entropy}(4/9,5/9) = -4/9*\log(4/9) – 5/9*\log(5/9)$

Alcohol_content

- Information before splitting:

Info[4,5] = entropy(4/9,5/9) = -4/9*log(4/9) – 5/9*log(5/9)



Values for logs (base 2):

log 1 = 0	log 6 = 2.58
log 2 = 1	log 7 = 2.81
log 3 = 1.58	log 8 = 3
log 4 = 2	log 9 = 3.17
log 5 = 2.32	log 10 = 3.32

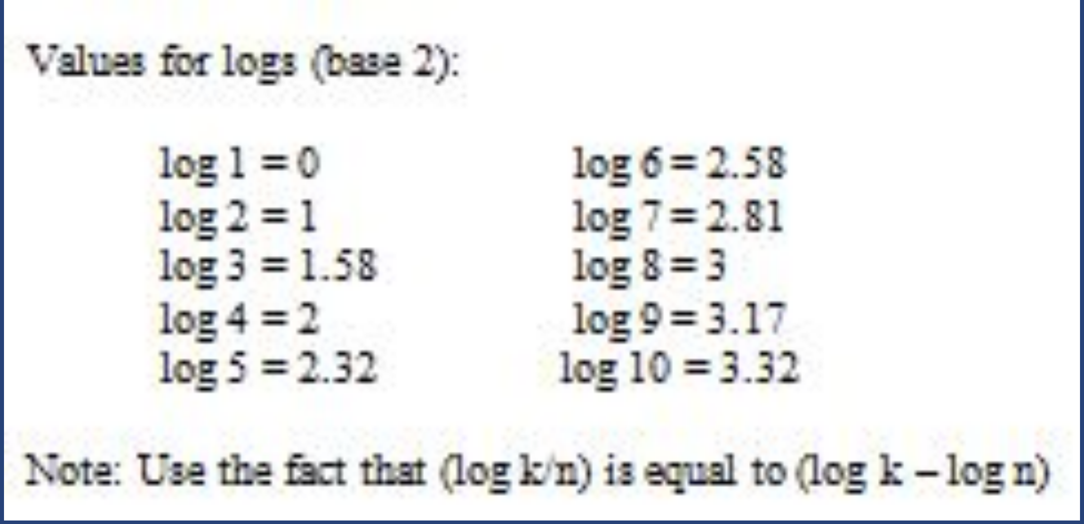
Note: Use the fact that (log k/n) is equal to (log k – log n)

Alcohol_content

- Information before splitting:

Info[4,5] = entropy(4/9,5/9) = -4/9*log(4/9) – 5/9*log(5/9) =

= -4/9 *



Values for logs (base 2):

log 1 = 0	log 6 = 2.58
log 2 = 1	log 7 = 2.81
log 3 = 1.58	log 8 = 3
log 4 = 2	log 9 = 3.17
log 5 = 2.32	log 10 = 3.32

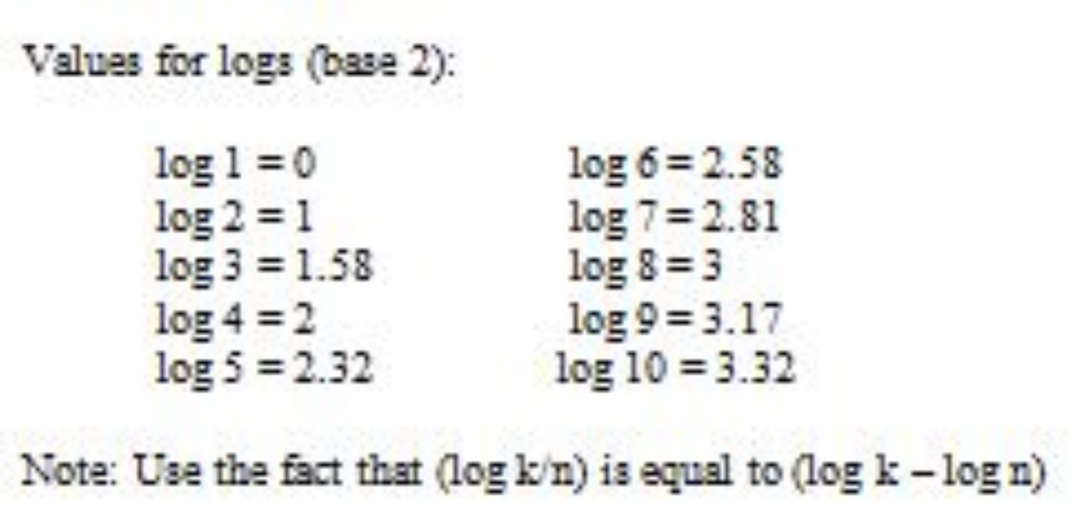
Note: Use the fact that (log k/n) is equal to (log k – log n)

Alcohol_content

- Information before splitting:

Info[4,5] = entropy(4/9,5/9) = -4/9*log(4/9) – 5/9*log(5/9) =

= -4/9 * (2 -



Values for logs (base 2):

log 1 = 0
log 2 = 1
log 3 = 1.58
log 4 = 2
log 5 = 2.32

log 6 = 2.58
log 7 = 2.81
log 8 = 3
log 9 = 3.17
log 10 = 3.32

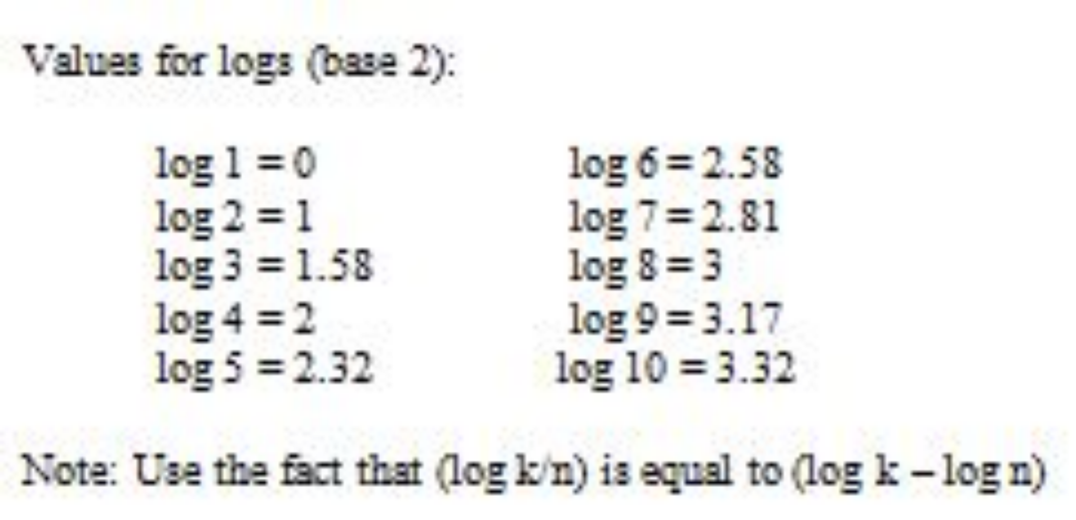
Note: Use the fact that (log k/n) is equal to (log k – log n)

Alcohol_content

- Information before splitting:

$$\text{Info}[4,5] = \text{entropy}(4/9,5/9) = -4/9*\log(4/9) – 5/9*\log(5/9) =$$

$$= -4/9 * (2 - 3.17)$$



Values for logs (base 2):

log 1 = 0
log 2 = 1
log 3 = 1.58
log 4 = 2
log 5 = 2.32

log 6 = 2.58
log 7 = 2.81
log 8 = 3
log 9 = 3.17
log 10 = 3.32

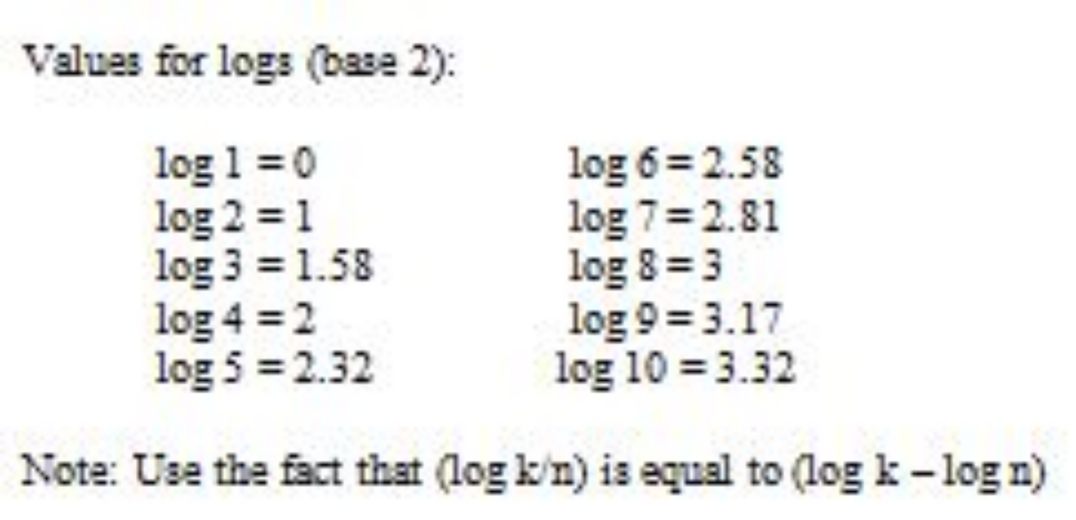
Note: Use the fact that (log k/n) is equal to (log k – log n)

Alcohol_content

- Information before splitting:

Info[4,5] = entropy(4/9,5/9) = -4/9*log(4/9) – 5/9*log(5/9) =

= -4/9 * (2 - 3.17) –



Values for logs (base 2):

log 1 = 0
log 2 = 1
log 3 = 1.58
log 4 = 2
log 5 = 2.32

log 6 = 2.58
log 7 = 2.81
log 8 = 3
log 9 = 3.17
log 10 = 3.32

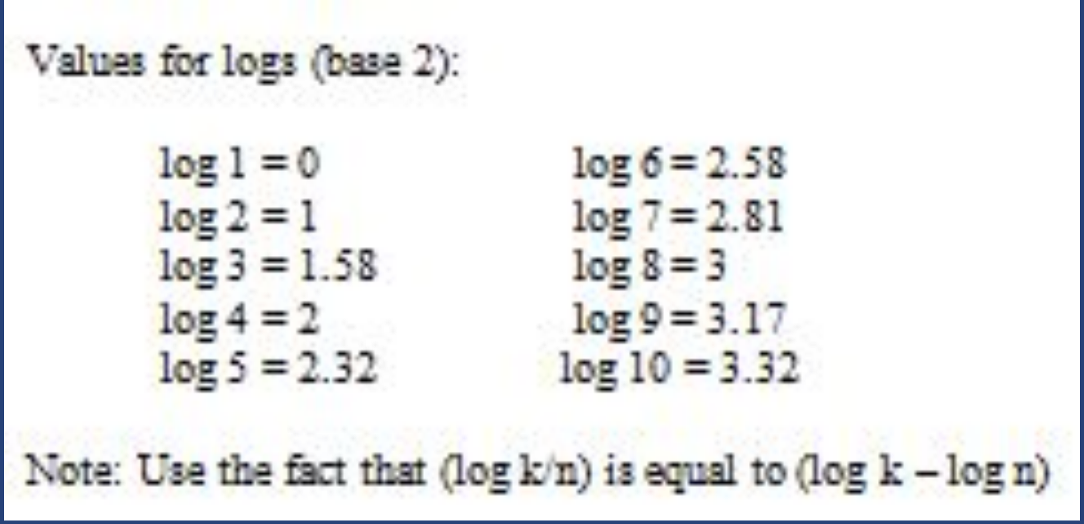
Note: Use the fact that (log k/n) is equal to (log k – log n)

Alcohol_content

- Information before splitting:

Info[4,5] = entropy(4/9,5/9) = -4/9*log(4/9) – 5/9*log(5/9) =

= -4/9 * (2 - 3.17) – 5/9 *



Values for logs (base 2):

log 1 = 0
log 2 = 1
log 3 = 1.58
log 4 = 2
log 5 = 2.32

log 6 = 2.58
log 7 = 2.81
log 8 = 3
log 9 = 3.17
log 10 = 3.32

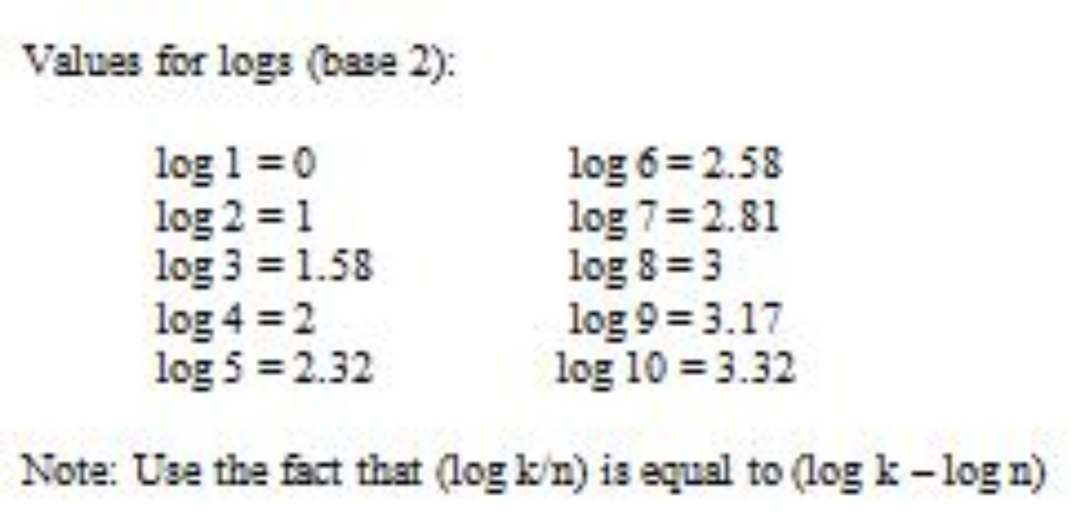
Note: Use the fact that (log k/n) is equal to (log k – log n)

Alcohol_content

- Information before splitting:

$$Info[4,5] = entropy(4/9,5/9) = -4/9*\log(4/9) – 5/9*\log(5/9) =$$

$$= -4/9 * (2 - 3.17) – 5/9 * (2.32$$



Values for logs (base 2):

$\log 1 = 0$
$\log 2 = 1$
$\log 3 = 1.58$
$\log 4 = 2$
$\log 5 = 2.32$

$\log 6 = 2.58$
$\log 7 = 2.81$
$\log 8 = 3$
$\log 9 = 3.17$
$\log 10 = 3.32$

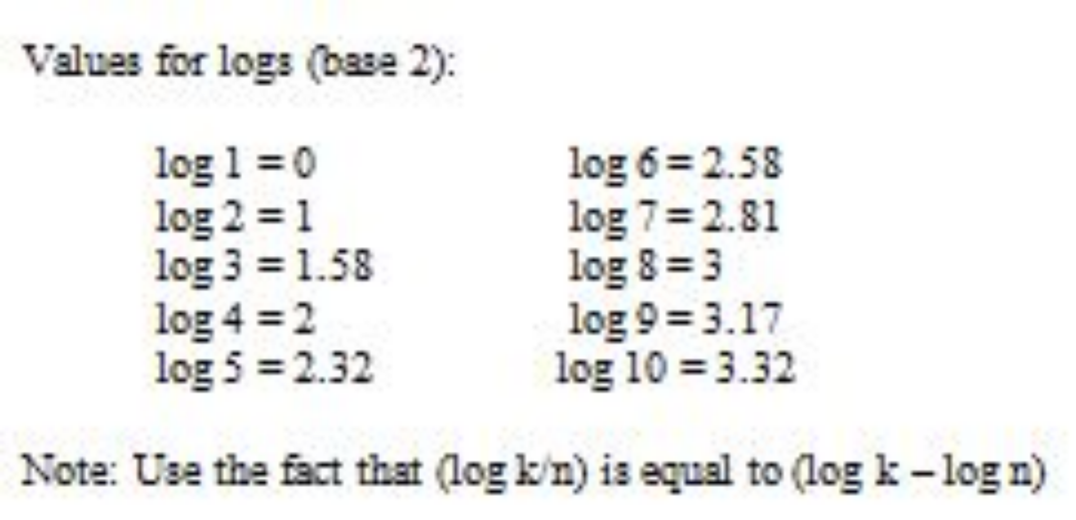
Note: Use the fact that (log k/n) is equal to (log k – log n)

Alcohol_content

- Information before splitting:

$$\text{Info}[4,5] = \text{entropy}(4/9,5/9) = -4/9*\log(4/9) – 5/9*\log(5/9) =$$

$$= -4/9 * (2 - 3.17) – 5/9 * (2.32 - 3.17)$$



Values for logs (base 2):

log 1 = 0
log 2 = 1
log 3 = 1.58
log 4 = 2
log 5 = 2.32

log 6 = 2.58
log 7 = 2.81
log 8 = 3
log 9 = 3.17
log 10 = 3.32

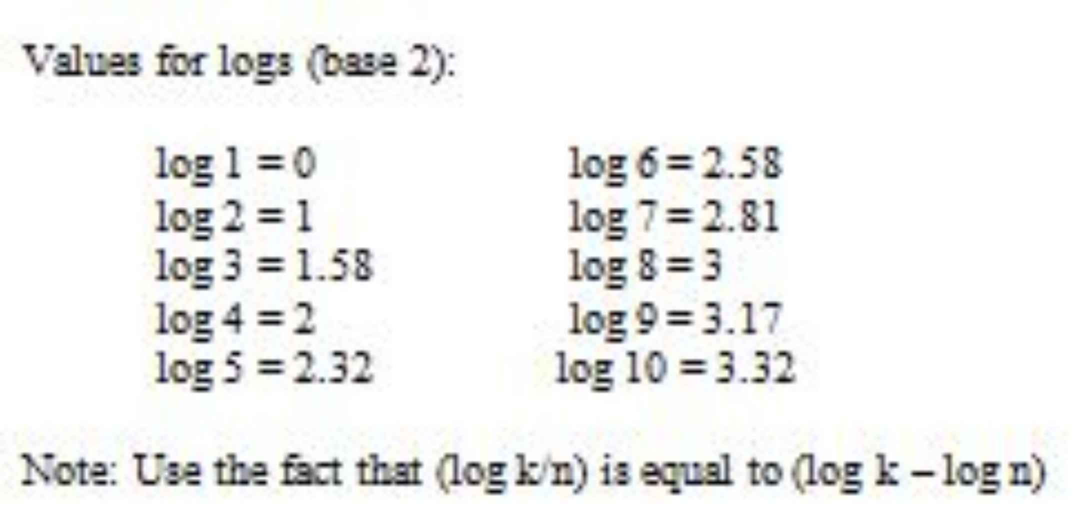
Note: Use the fact that (log k/n) is equal to (log k – log n)

Alcohol_content

- Information before splitting:

Info[4,5] = entropy(4/9,5/9) = -4/9*log(4/9) – 5/9*log(5/9) =

= -4/9 * (2 - 3.17) – 5/9 * (2.32 - 3.17) = 0.99



Values for logs (base 2):

log 1 = 0
log 2 = 1
log 3 = 1.58
log 4 = 2
log 5 = 2.32

log 6 = 2.58
log 7 = 2.81
log 8 = 3
log 9 = 3.17
log 10 = 3.32

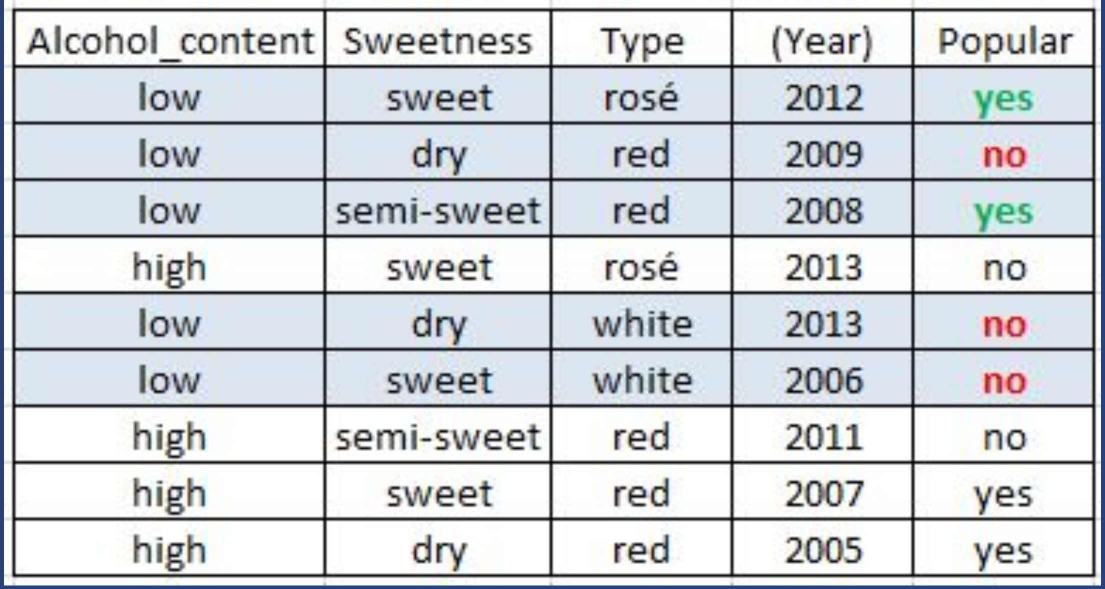
Note: Use the fact that (log k/n) is equal to (log k – log n)

Alcohol_content

- *Alcohol_content* = low:

Alcohol_content

- *Alcohol_content* = low:

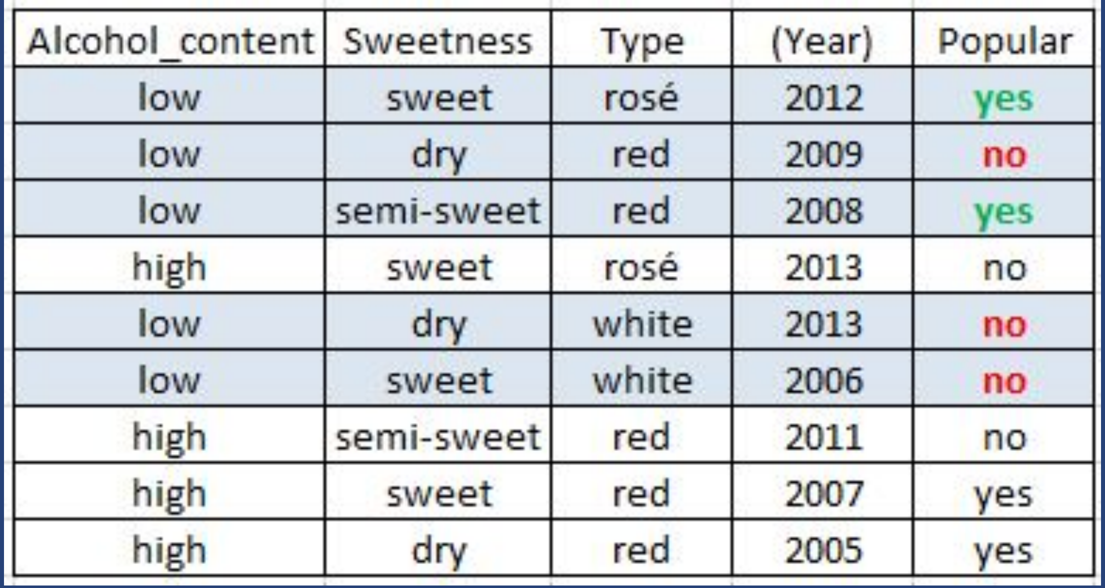


Alcohol_content	Sweetness	Type	(Year)	Popular
low	sweet	rosé	2012	yes
low	dry	red	2009	no
low	semi-sweet	red	2008	yes
high	sweet	rosé	2013	no
low	dry	white	2013	no
low	sweet	white	2006	no
high	semi-sweet	red	2011	no
high	sweet	red	2007	yes
high	dry	red	2005	yes

Alcohol_content

- *Alcohol_content* = low:

  Info([2,3]) = entropy(2/5, 3/5) =

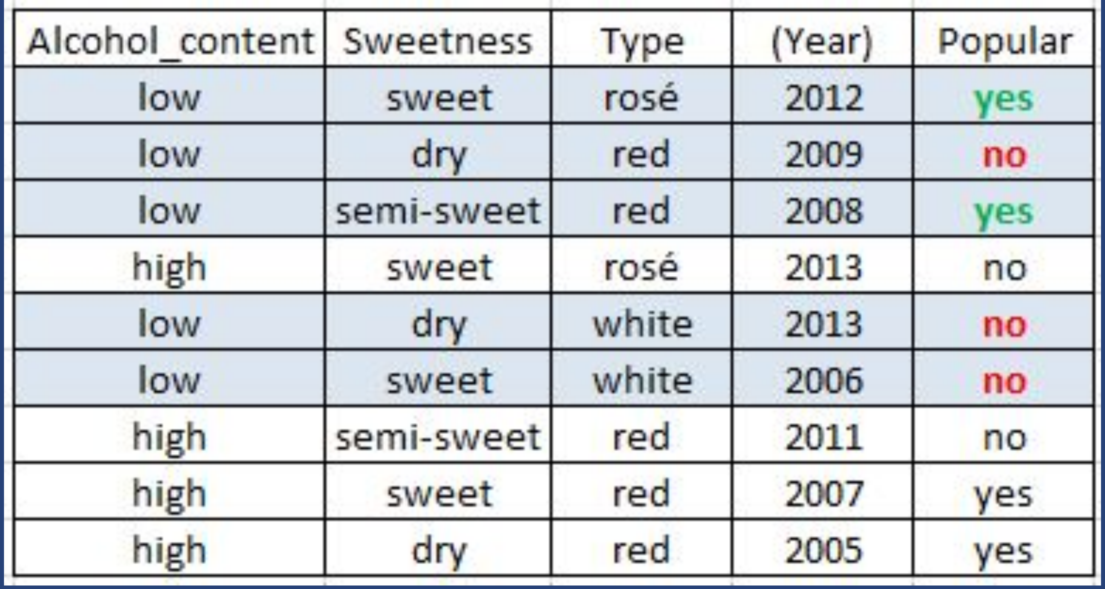


Alcohol_content	Sweetness	Type	(Year)	Popular
low	sweet	rosé	2012	yes
low	dry	red	2009	no
low	semi-sweet	red	2008	yes
high	sweet	rosé	2013	no
low	dry	white	2013	no
low	sweet	white	2006	no
high	semi-sweet	red	2011	no
high	sweet	red	2007	yes
high	dry	red	2005	yes

Alcohol_content

- *Alcohol_content* = low:

  Info([2,3]) = entropy(2/5, 3/5) = -2/5 log(2/5) - 3/5 log(3/5) = 0.972 bits



Alcohol_content	Sweetness	Type	(Year)	Popular
low	sweet	rosé	2012	yes
low	dry	red	2009	no
low	semi-sweet	red	2008	yes
high	sweet	rosé	2013	no
low	dry	white	2013	no
low	sweet	white	2006	no
high	semi-sweet	red	2011	no
high	sweet	red	2007	yes
high	dry	red	2005	yes

Alcohol_content

- *Alcohol_content* = low:

  Info([2,3]) = entropy(2/5, 3/5) = -2/5 log(2/5) - 3/5 log(3/5) = 0.972 bits

Alcohol_content

- *Alcohol_content* = low:

  Info([2,3]) = entropy(2/5, 3/5) = -2/5 log(2/5) - 3/5 log(3/5) = 0.972 bits

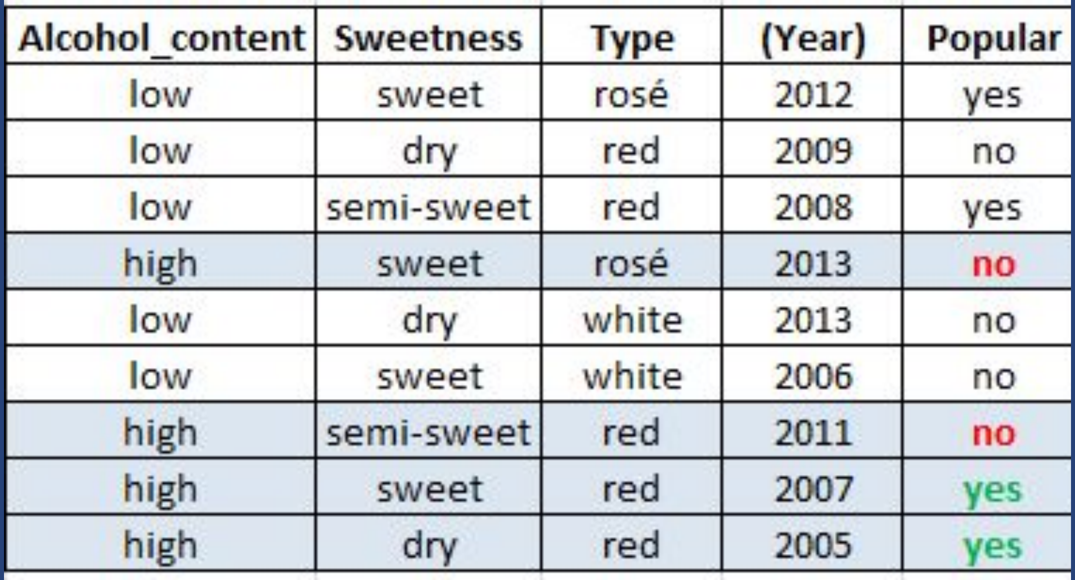
- *Alcohol_content* = high:

Alcohol_content

- *Alcohol_content* = low:

  Info([2,3]) = entropy(2/5, 3/5) = -2/5 log(2/5) - 3/5 log(3/5) = 0.972 bits

- *Alcohol_content* = high:



Alcohol_content	Sweetness	Type	(Year)	Popular
low	sweet	rosé	2012	yes
low	dry	red	2009	no
low	semi-sweet	red	2008	yes
high	sweet	rosé	2013	no
low	dry	white	2013	no
low	sweet	white	2006	no
high	semi-sweet	red	2011	no
high	sweet	red	2007	yes
high	dry	red	2005	yes

Alcohol_content

- *Alcohol_content* = low:

  Info([2,3]) = entropy(2/5, 3/5) = -2/5 log(2/5) - 3/5 log(3/5) = 0.972 bits

- *Alcohol_content* = high:

  Info([2,2]) = entropy(2/4, 2/4) =

Alcohol_content

- *Alcohol_content* = low:

  $\text{Info}([2,3]) = \text{entropy}(2/5, 3/5) = -2/5 \log(2/5) - 3/5 \log(3/5) = 0.972$ bits

- *Alcohol_content* = high:

  $\text{Info}([2,2]) = \text{entropy}(2/4, 2/4) = -2/4 \log(2/4) - 2/4 \log(2/4) = 1$ bits

Alcohol_content

- *Alcohol_content* = low:

  Info([2,3]) = entropy(2/5, 3/5) = -2/5 log(2/5) - 3/5 log(3/5) = 0.972 bits

- *Alcohol_content* = high:

  Info([2,2]) = entropy(2/4, 2/4) = -2/4 log(2/4) - 2/4 log(2/4) = 1 bits

- Expected information for attribute:

  Info([2,3], [2,2]) =

Alcohol_content

- *Alcohol_content* = low:

  Info([2,3]) = entropy(2/5, 3/5) = -2/5 log(2/5) - 3/5 log(3/5) = 0.972 bits

- *Alcohol_content* = high:

  Info([2,2]) = entropy(2/4, 2/4) = -2/4 log(2/4) - 2/4 log(2/4) = 1 bits

- Expected information for attribute:

  Info([2,3], [2,2]) = 5/9 * 0.972 + 4/9 * 1 = 0.98 bits

Alcohol_content

- *Alcohol_content* = low:

  $\text{Info}([2,3]) = \text{entropy}(2/5, 3/5) = -2/5 \log(2/5) - 3/5 \log(3/5) = 0.972$ bits

- *Alcohol_content* = high:

  $\text{Info}([2,2]) = \text{entropy}(2/4, 2/4) = -2/4 \log(2/4) - 2/4 \log(2/4) = 1$ bits

- Expected information for attribute:

  $\text{Info}([2,3], [2,2]) = 5/9 * 0.972 + 4/9 * 1 = 0.98$ bits

- Information gain: information before splitting – information after splitting

Alcohol_content

- *Alcohol_content* = low:

  Info([2,3]) = entropy(2/5, 3/5) = -2/5 log(2/5) - 3/5 log(3/5) = 0.972 bits

- *Alcohol_content* = high:

  Info([2,2]) = entropy(2/4, 2/4) = -2/4 log(2/4) - 2/4 log(2/4) = 1 bits

- Expected information for attribute:

  Info([2,3], [2,2]) = 5/9 * 0.972 + 4/9 * 1 = 0.98 bits

- Information gain: information before splitting – information after splitting

  gain(*Alcohol_content*) =

Alcohol_content

- *Alcohol_content* = low:

  Info([2,3]) = entropy(2/5, 3/5) = -2/5 log(2/5) - 3/5 log(3/5) = 0.972 bits

- *Alcohol_content* = high:

  Info([2,2]) = entropy(2/4, 2/4) = -2/4 log(2/4) - 2/4 log(2/4) = 1 bits

- Expected information for attribute:

  Info([2,3], [2,2]) = 5/9 * 0.972 + 4/9 * 1 = 0.98 bits

- Information gain: information before splitting – information after splitting

  gain(*Alcohol_content*) = info([4,5]) – info([2,3],[2,2])

Alcohol_content

- *Alcohol_content* = low:

  Info([2,3]) = entropy(2/5, 3/5) = -2/5 log(2/5) - 3/5 log(3/5) = 0.972 bits

- *Alcohol_content* = high:

  Info([2,2]) = entropy(2/4, 2/4) = -2/4 log(2/4) - 2/4 log(2/4) = 1 bits

- Expected information for attribute:

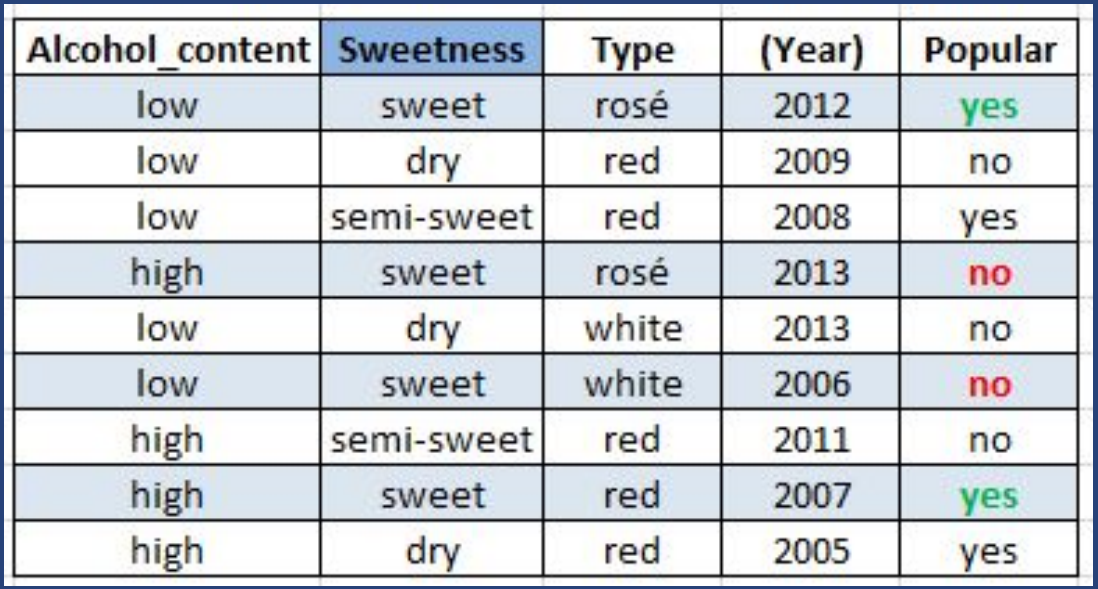
  Info([2,3], [2,2]) = 5/9 * 0.972 + 4/9 * 1 = 0.98 bits

- Information gain: information before splitting – information after splitting

  gain(*Alcohol_content*) = info([4,5]) – info([2,3],[2,2]) = 0.99 – 0.98 = 0.01 bits

Sweetness

- *Sweetness = sweet*:

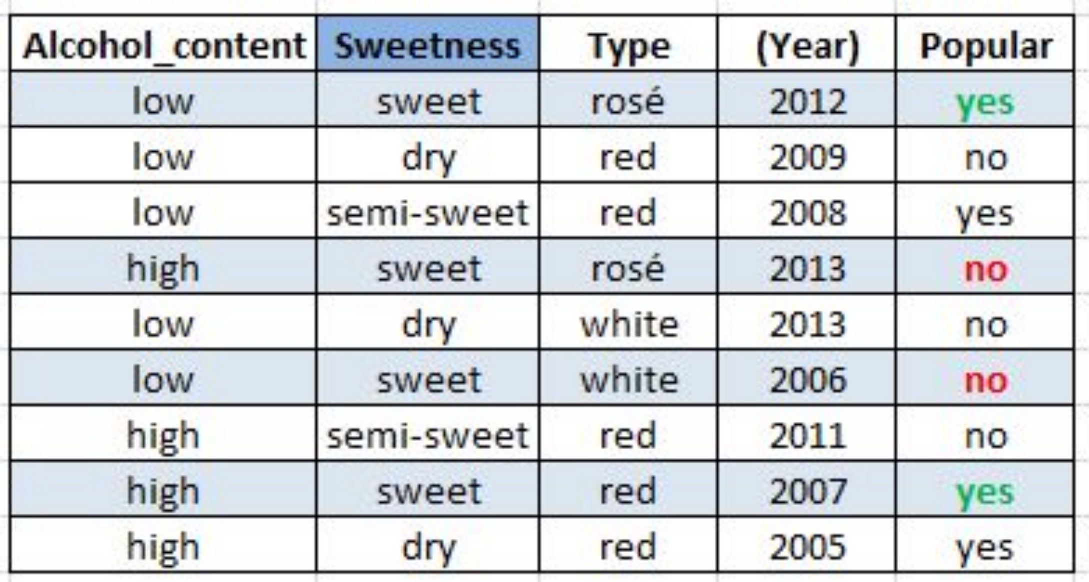


Alcohol_content	Sweetness	Type	(Year)	Popular
low	sweet	rosé	2012	yes
low	dry	red	2009	no
low	semi-sweet	red	2008	yes
high	sweet	rosé	2013	no
low	dry	white	2013	no
low	sweet	white	2006	no
high	semi-sweet	red	2011	no
high	sweet	red	2007	yes
high	dry	red	2005	yes

Sweetness

- *Sweetness = sweet*:

Info([2,2]) = entropy(2/4,2/4)=

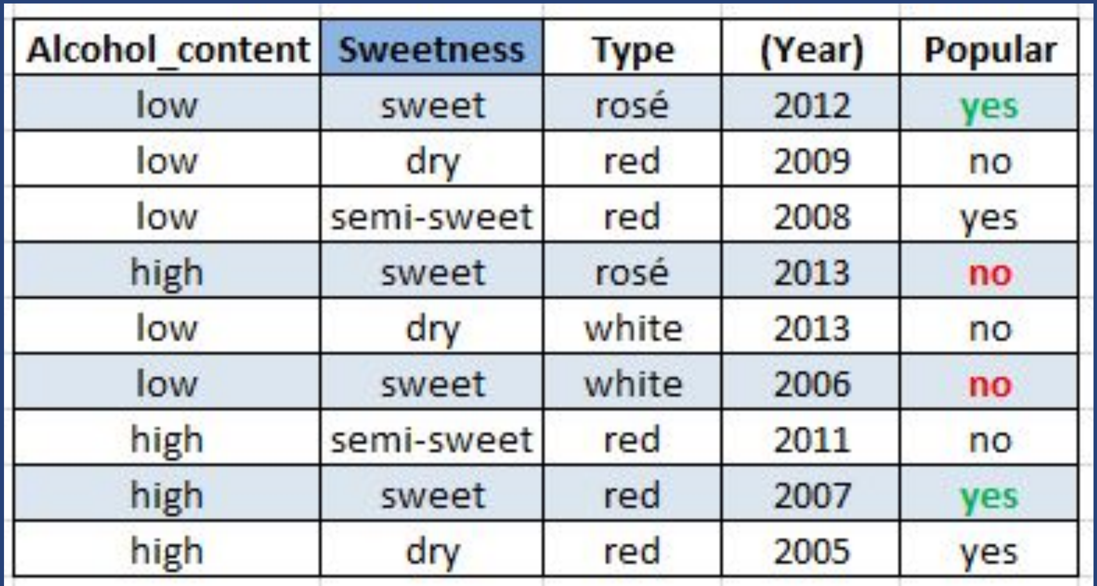


Alcohol_content	Sweetness	Type	(Year)	Popular
low	sweet	rosé	2012	yes
low	dry	red	2009	no
low	semi-sweet	red	2008	yes
high	sweet	rosé	2013	no
low	dry	white	2013	no
low	sweet	white	2006	no
high	semi-sweet	red	2011	no
high	sweet	red	2007	yes
high	dry	red	2005	yes

Sweetness

- *Sweetness = sweet*:

$\text{Info}([2,2]) = \text{entropy}(2/4,2/4) = -2/4 \log(2/4) - 2/4 \log(2/4) = 1$ bits



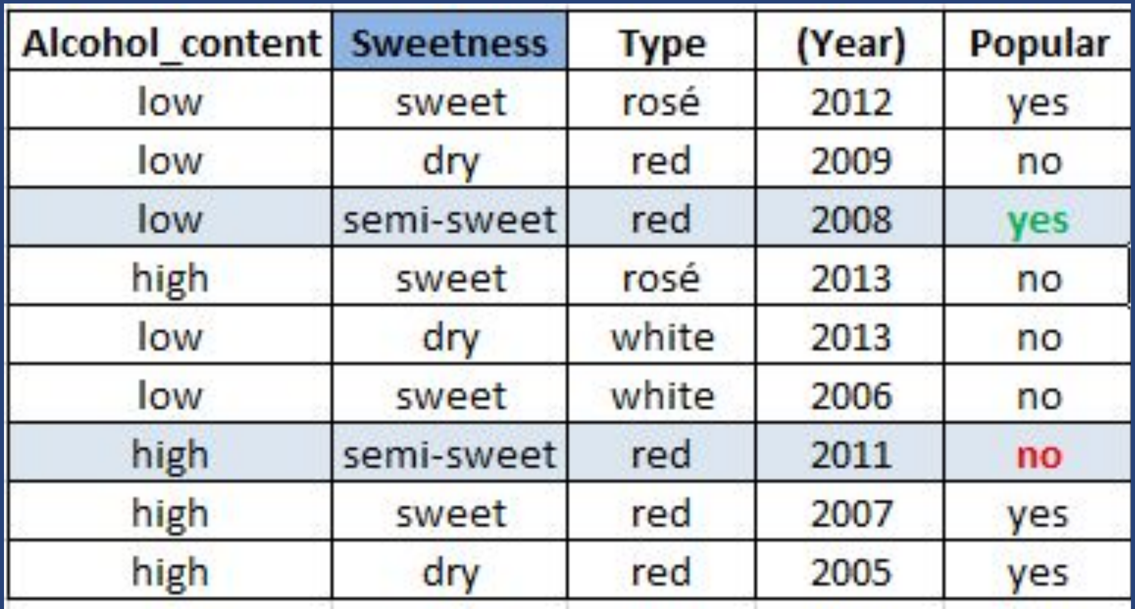
Alcohol_content	Sweetness	Type	(Year)	Popular
low	sweet	rosé	2012	yes
low	dry	red	2009	no
low	semi-sweet	red	2008	yes
high	sweet	rosé	2013	no
low	dry	white	2013	no
low	sweet	white	2006	no
high	semi-sweet	red	2011	no
high	sweet	red	2007	yes
high	dry	red	2005	yes

Sweetness

- *Sweetness = sweet*:

Info([2,2]) = entropy(2/4,2/4)= -2/4 log(2/4)-2/4log(2/4) = 1 bits

- *Sweetness* = semi-sweet



Alcohol_content	Sweetness	Type	(Year)	Popular
low	sweet	rosé	2012	yes
low	dry	red	2009	no
low	semi-sweet	red	2008	yes
high	sweet	rosé	2013	no
low	dry	white	2013	no
low	sweet	white	2006	no
high	semi-sweet	red	2011	no
high	sweet	red	2007	yes
high	dry	red	2005	yes

Sweetness

- *Sweetness = sweet*:

  Info([2,2]) = entropy(2/4,2/4)= -2/4 log(2/4)-2/4log(2/4) = 1 bits

- *Sweetness* = semi-sweet

  Info([1,1]) = entropy(1/2,1/2)= -1/2 log(1/2)-1/2log(1/2) = 1 bits

Sweetness

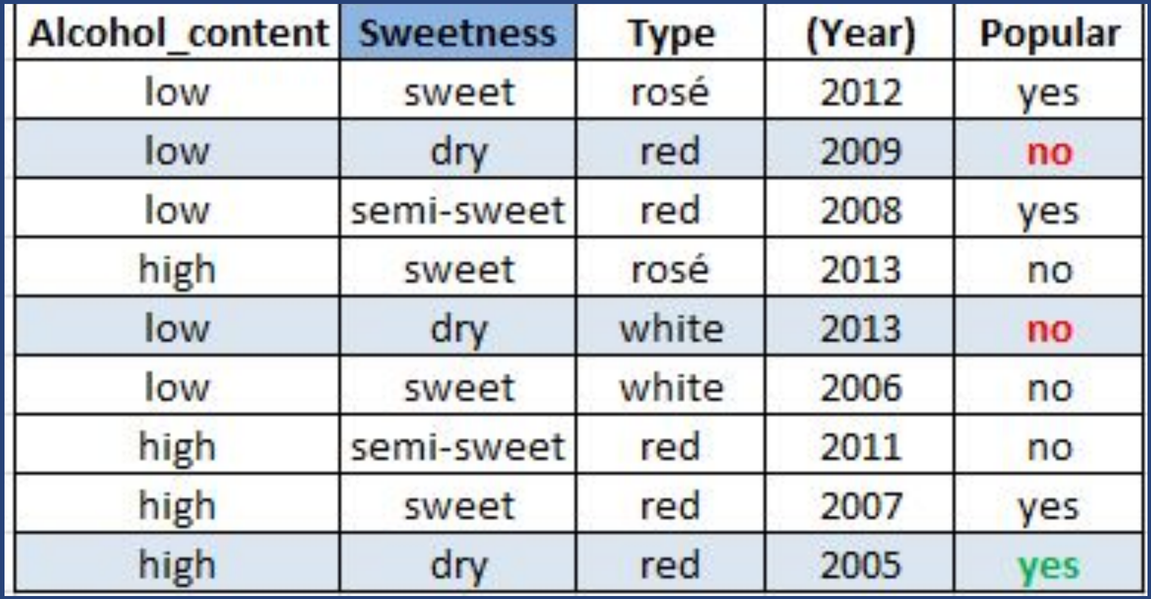
- *Sweetness = sweet*:

  Info([2,2]) = entropy(2/4,2/4)= -2/4 log(2/4)-2/4log(2/4) = 1 bits

- *Sweetness* = semi-sweet

  Info([1,1]) = entropy(1/2,1/2)= -1/2 log(1/2)-1/2log(1/2) = 1 bits

- *Sweetness* = dry



Alcohol_content	Sweetness	Type	(Year)	Popular
low	sweet	rosé	2012	yes
low	dry	red	2009	no
low	semi-sweet	red	2008	yes
high	sweet	rosé	2013	no
low	dry	white	2013	no
low	sweet	white	2006	no
high	semi-sweet	red	2011	no
high	sweet	red	2007	yes
high	dry	red	2005	yes

Sweetness

- *Sweetness = sweet*:

  Info([2,2]) = entropy(2/4,2/4)= -2/4 log(2/4)-2/4log(2/4) = 1 bits

- *Sweetness* = semi-sweet

  Info([1,1]) = entropy(1/2,1/2)= -1/2 log(1/2)-1/2log(1/2) = 1 bits

- *Sweetness* = dry

  Info([1,2]) = entropy(1/3,2/3)= -1/3 log(1/3)-2/3log(2/3) = 0. 913 bits

Sweetness

- *Sweetness = sweet*:

  Info([2,2]) = entropy(2/4,2/4)= -2/4 log(2/4)-2/4log(2/4) = 1 bits

- *Sweetness* = semi-sweet

  Info([1,1]) = entropy(1/2,1/2)= -1/2 log(1/2)-1/2log(1/2) = 1 bits

- *Sweetness* = dry

  Info([1,2]) = entropy(1/3,2/3)= -1/3 log(1/3)-2/3log(2/3) = 0. 913 bits

- Expected information for attribute:

  Info([2,2], [1,1], [1,2]) = 4/9 * 1 + 2/9 * 1 + 3/9 *0.913 = 0.971 bits

Sweetness

- *Sweetness = sweet*:

  Info([2,2]) = entropy(2/4,2/4)= -2/4 log(2/4)-2/4log(2/4) = 1 bits

- *Sweetness* = semi-sweet

  Info([1,1]) = entropy(1/2,1/2)= -1/2 log(1/2)-1/2log(1/2) = 1 bits

- *Sweetness* = dry

  Info([1,2]) = entropy(1/3,2/3)= -1/3 log(1/3)-2/3log(2/3) = 0. 913 bits

- Expected information for attribute:

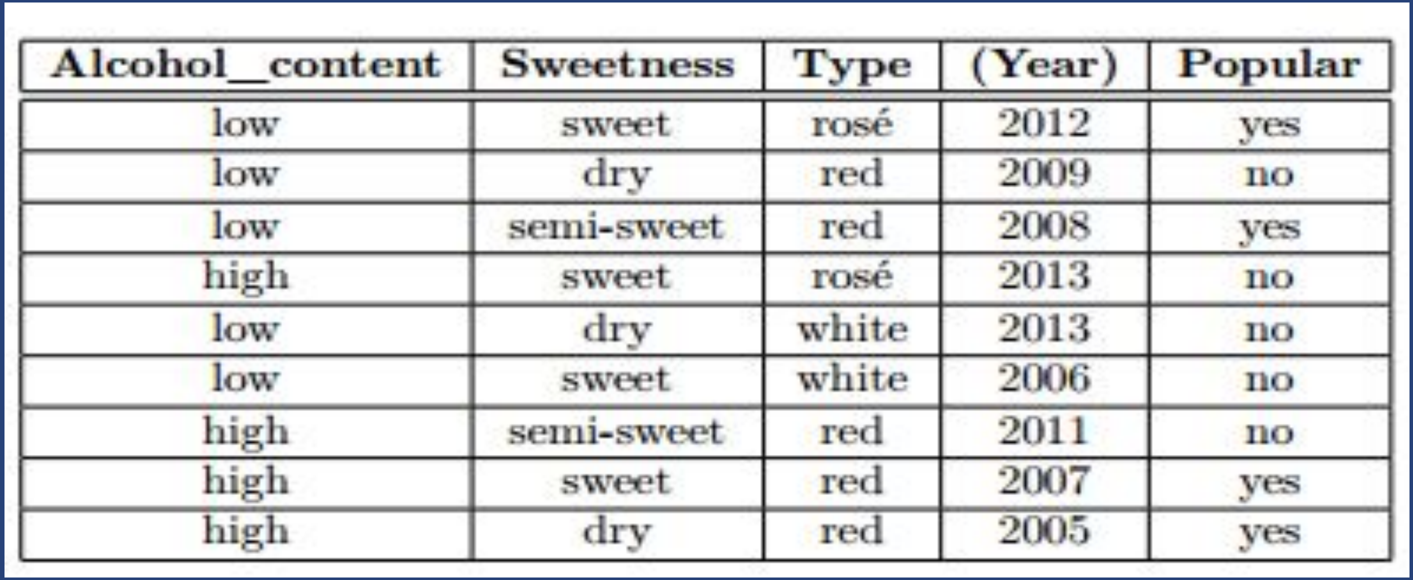
  Info([2,2], [1,1], [1,2]) = 4/9 * 1 + 2/9 * 1 + 3/9 *0.913 = 0.971 bits

- Information gain: information before splitting – information after splitting

  gain(*Sweetness*) = info([4,5]) – info([2,2],[1,1],[1,2]) = 0.99 – 0.971 = 0.019 bits

Type

- *Type = rosé:*
- *Type* = red:
- Type = white:



Alcohol_content	Sweetness	Type	(Year)	Popular
low	sweet	rosé	2012	yes
low	dry	red	2009	no
low	semi-sweet	red	2008	yes
high	sweet	rosé	2013	no
low	dry	white	2013	no
low	sweet	white	2006	no
high	semi-sweet	red	2011	no
high	sweet	red	2007	yes
high	dry	red	2005	yes

- Expected information for attribute:
- Information gain: information before splitting – information after splitting

  gain(*Type*) =

Type

- *Type = rosé:*

  Info([1,1]) = entropy(1/2,1/2)= -1/2 log(1/2)-1/2log(1/2) = 1 bits

- *Type* = red:

- Type = white:

- Expected information for attribute:

- Information gain: information before splitting – information after splitting

  gain(*Type*) =

Type

- *Type = rosé:*

  Info([1,1]) = entropy(1/2,1/2)= -1/2 log(1/2) - 1/2log(1/2) = 1 bits

- *Type* = red:

  Info([3,2]) = = entropy(3/5,2/5)= -3/5log(3/5) - 2/5log(2/5) = 0.972

- Type = white:

- Expected information for attribute:

- Information gain: information before splitting – information after splitting

  gain(*Type*) =

Type

- *Type = rosé:*

  Info([1,1]) = entropy(1/2,1/2)= -1/2 log(1/2) - 1/2log(1/2) = 1 bits

- *Type* = red:

  Info([3,2]) = = entropy(3/5,2/5)= -3/5log(3/5) - 2/5log(2/5)= 0.972

- Type = white:

  Info([0,2]) = entropy(0/2,2/2)= -0/2 log(0/2) - 2/2log(2/2) = 0 bits

- Expected information for attribute:

- Information gain: information before splitting – information after splitting

  gain(*Type*) =

Type

- *Type = rosé:*

  Info([1,1]) = entropy(1/2,1/2)= -1/2 log(1/2) - 1/2log(1/2) = 1 bits

- *Type* = red:

  Info([3,2]) = = entropy(3/5,2/5)= -3/5 log(3/5) - 2/5 log(2/5)= 0.972

- Type = white:

  Info([0,2]) = entropy(0/2,2/2)= -0/2 log(0/2) - 2/2 log(2/2) = 0 bits

- Expected information for attribute:

  Info([1,1], [3,2], [0,2]) = 2/9 * 1 + 5/9 * 0.972 + 2/9 *0 = 0.762 bits

- Information gain: information before splitting – information after splitting

  gain(*Type*) =

Type

- *Type = rosé:*

  Info([1,1]) = entropy(1/2,1/2)= -1/2 log(1/2)-1/2log(1/2) = 1 bits

- *Type* = red:

  Info([3,2]) = = entropy(3/5,2/5)= -3/5 log(3/5)-2/5 log(2/5)= 0.972

- Type = white:

  Info([0,2]) = entropy(0/2,2/2)= -0/2 log(0/2)-2/2 log(2/2) = 0 bits

- Expected information for attribute:

  Info([1,1], [3,2], [0,2]) = 2/9 * 1 + 5/9 * 0.972 + 2/9 *0 = 0.762 bits

- Information gain: information before splitting – information after splitting

  gain(*Type*) = info([4,5]) – info([1,1],[3,2],[0,2]) =

Type

- *Type = rosé:*

  Info([1,1]) = entropy(1/2,1/2)= -1/2 log(1/2)-1/2log(1/2) = 1 bits

- *Type* = red:

  Info([3,2]) = = entropy(3/5,2/5)= -3/5 log(3/5)-2/5 log(2/5)= 0.972

- Type = white:

  Info([0,2]) = entropy(0/2,2/2)= -0/2 log(0/2)-2/2 log(2/2) = 0 bits

- Expected information for attribute:

  Info([1,1], [3,2], [0,2]) = 2/9 * 1 + 5/9 * 0.972 + 2/9 *0 = 0.762 bits

- Information gain: information before splitting – information after splitting

  gain(*Type*) = info([4,5]) – info([1,1],[3,2],[0,2]) = 0.99 –

Type

- *Type = rosé:*

  Info([1,1]) = entropy(1/2,1/2)= -1/2 log(1/2)-1/2log(1/2) = 1 bits

- *Type* = red:

  Info([3,2]) = = entropy(3/5,2/5)= -3/5 log(3/5)-2/5 log(2/5)= 0.972

- Type = white:

  Info([0,2]) = entropy(0/2,2/2)= -0/2 log(0/2)-2/2 log(2/2) = 0 bits

- Expected information for attribute:

  Info([1,1], [3,2], [0,2]) = 2/9 * 1 + 5/9 * 0.972 + 2/9 *0 = 0.762 bits

- Information gain: information before splitting – information after splitting

  gain(*Type*) = info([4,5]) – info([1,1],[3,2],[0,2]) = 0.99 – 0.762 =

Type

- *Type = rosé:*

  Info([1,1]) = entropy(1/2,1/2)= -1/2 log(1/2)-1/2log(1/2) = 1 bits

- *Type* = red:

  Info([3,2]) = = entropy(3/5,2/5)= -3/5 log(3/5)-2/5 log(2/5)= 0.972

- Type = white:

  Info([0,2]) = entropy(0/2,2/2)= -0/2 log(0/2)-2/2 log(2/2) = 0 bits

- Expected information for attribute:

  Info([1,1], [3,2], [0,2]) = 2/9 * 1 + 5/9 * 0.972 + 2/9 *0 = 0.762 bits

- Information gain: information before splitting – information after splitting

  gain(*Type*) = info([4,5]) – info([1,1],[3,2],[0,2]) = 0.99 – 0.762 = 0.227 bits

Information gain

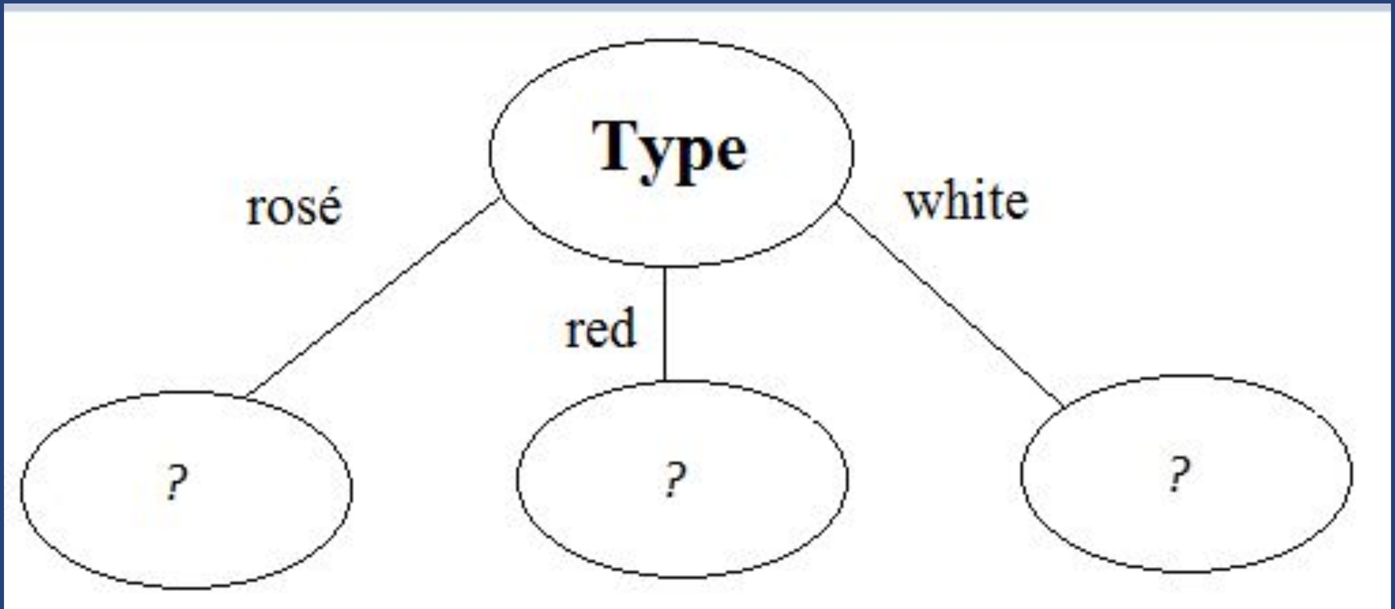
- gain(*Alcohol_content*) = 0.01 bits
- gain(*Sweetness*) = 0.019 bits
- gain(*Type*) = 0.227 bits

Information gain

- gain(*Alcohol_content*) = 0.01 bits
- gain(*Sweetness*) = 0.019 bits
- gain(Type) = 0.227 bits

Information gain

- gain(*Alcohol_content*) = 0.01 bits
- gain(*Sweetness*) = 0.019 bits
- gain(Type) = 0.227 bits



Type = rosé

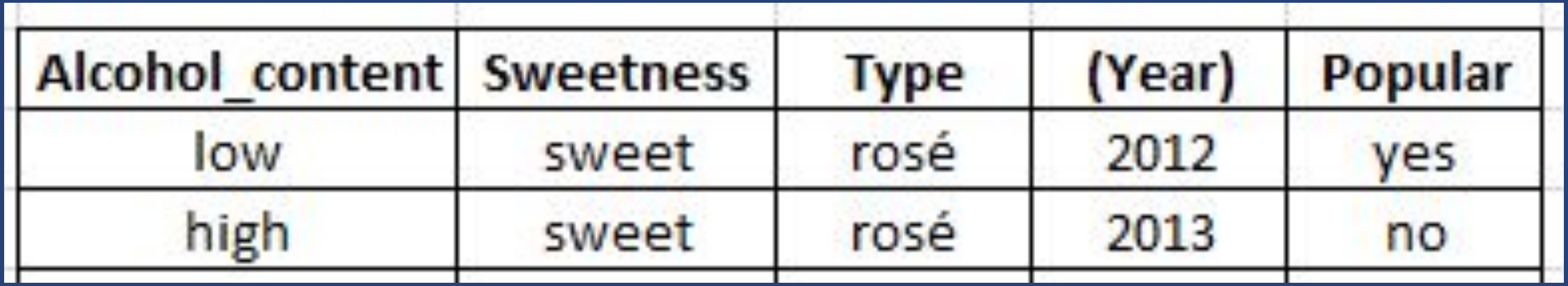
- Information before splitting: Info[1,1] = 1 bits

- *Alcohol_content* = low: Info([1,0]) = 0 bits

- *Alcohol_content* = high: Info([0,1]) = 0 bits

- Info([1,0], [0,1]) = 1/2 * 0 + 1/2 * 0 = 0 bits

- gain(*Alcohol_content*) = 1 – 0 = 1 bits



Alcohol_content	Sweetness	Type	(Year)	Popular
low	sweet	rosé	2012	yes
high	sweet	rosé	2013	no

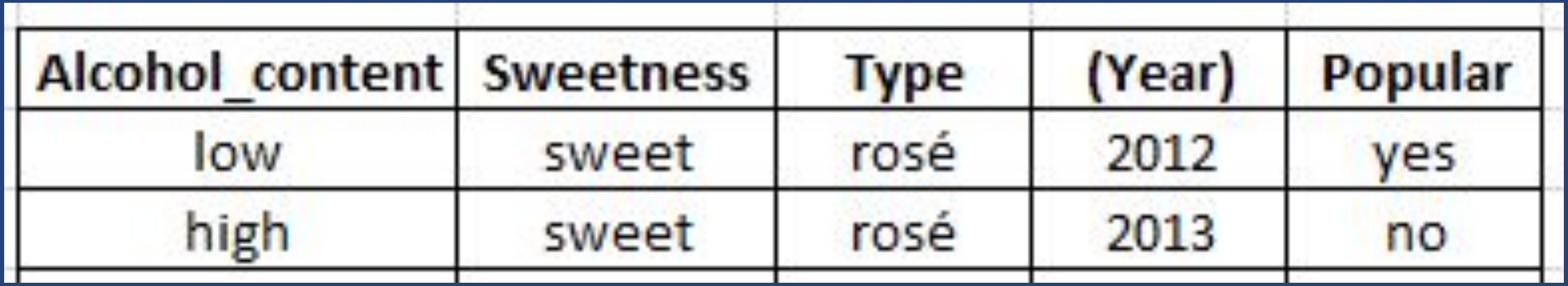
Type = rosé

- Information before splitting: Info[1,1] = 1 bits

- *Sweetness* = sweet: Info([1,1]) = 1 bits
- *Sweetness* = semi-sweet: Info([0,0]) = 0 bits
- *Sweetness* = dry: Info([0,0]) = 0 bits

- Info([1,1], [0,0]),[0,0]) = 1 bits

- gain(*Sweetness*) = 1– 1 = 0 bits



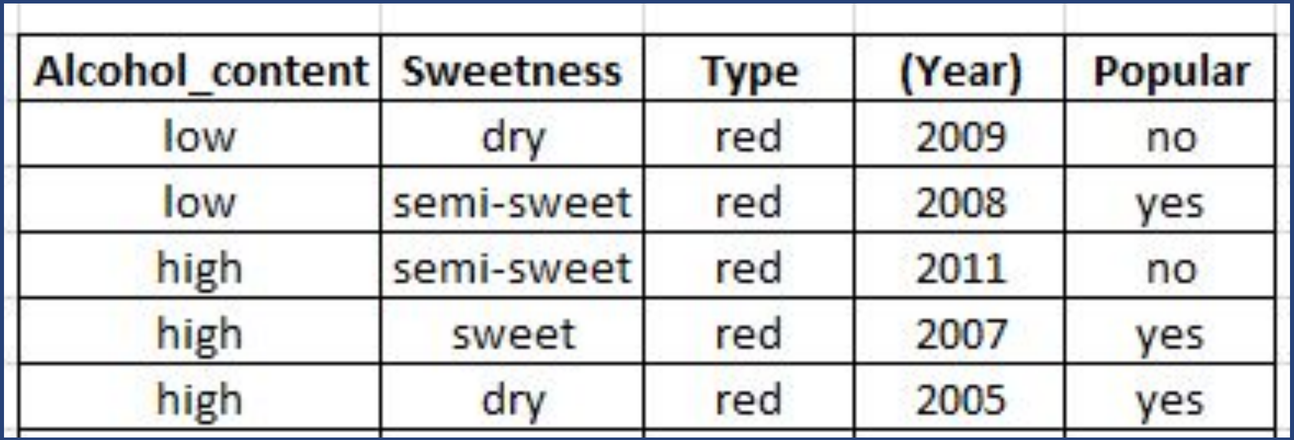
Alcohol_content	Sweetness	Type	(Year)	Popular
low	sweet	rosé	2012	yes
high	sweet	rosé	2013	no

Type = rosé

- gain(*Alcohol_content*) = 1 bits
- gain(*Sweetness*) = 0 bits

Type = red

- Information before splitting: Info[3,2] =0.972 bits
- *Alcohol_content* = low: Info([1,1]) = 1 bits
- *Alcohol_content* = high: Info([2,1]) = 0.913 bits
- Info([1,1], [2,1]) = 2/5 * 1 + 3/5 * 0.913 = 0.95 bits
- gain(*Alcohol_content*) = 0.972– 0.95 = 0.022 bits



Alcohol_content	Sweetness	Type	(Year)	Popular
low	dry	red	2009	no
low	semi-sweet	red	2008	yes
high	semi-sweet	red	2011	no
high	sweet	red	2007	yes
high	dry	red	2005	yes

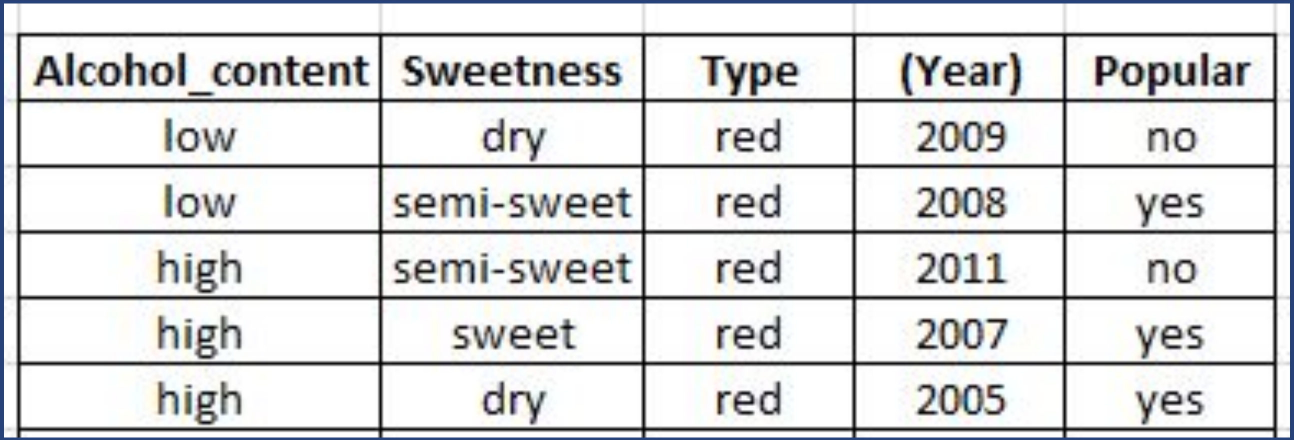
Type = red

- Information before splitting: Info[3,2] = 0.972 bits

- *Sweetness* = sweet: Info([1,0]) = 0 bits
- *Sweetness* = semi-sweet: Info([1,1]) = 1 bits
  Sweetness = dry: Info([1,1]) = 1 bits

- Info([1,0], [1,1]),[1,1]) = 1/5*0 + 2/5 * 1 + 2/5 *1 = 0.8 bits

- gain(*Sweetness*) = 0.972– 0.8 = 0.172 bits

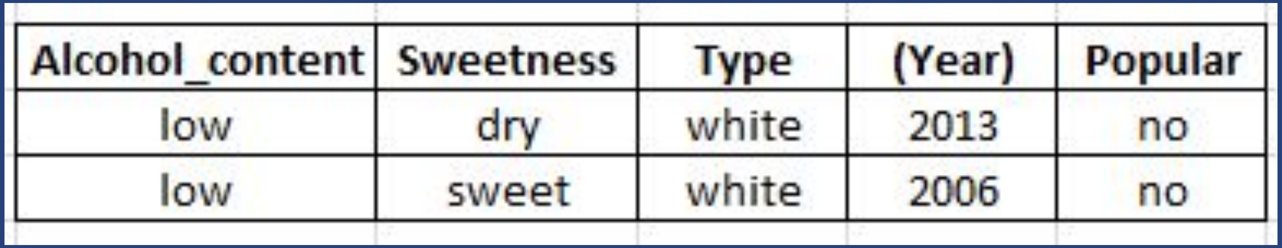


Alcohol_content	Sweetness	Type	(Year)	Popular
low	dry	red	2009	no
low	semi-sweet	red	2008	yes
high	semi-sweet	red	2011	no
high	sweet	red	2007	yes
high	dry	red	2005	yes

Type = red

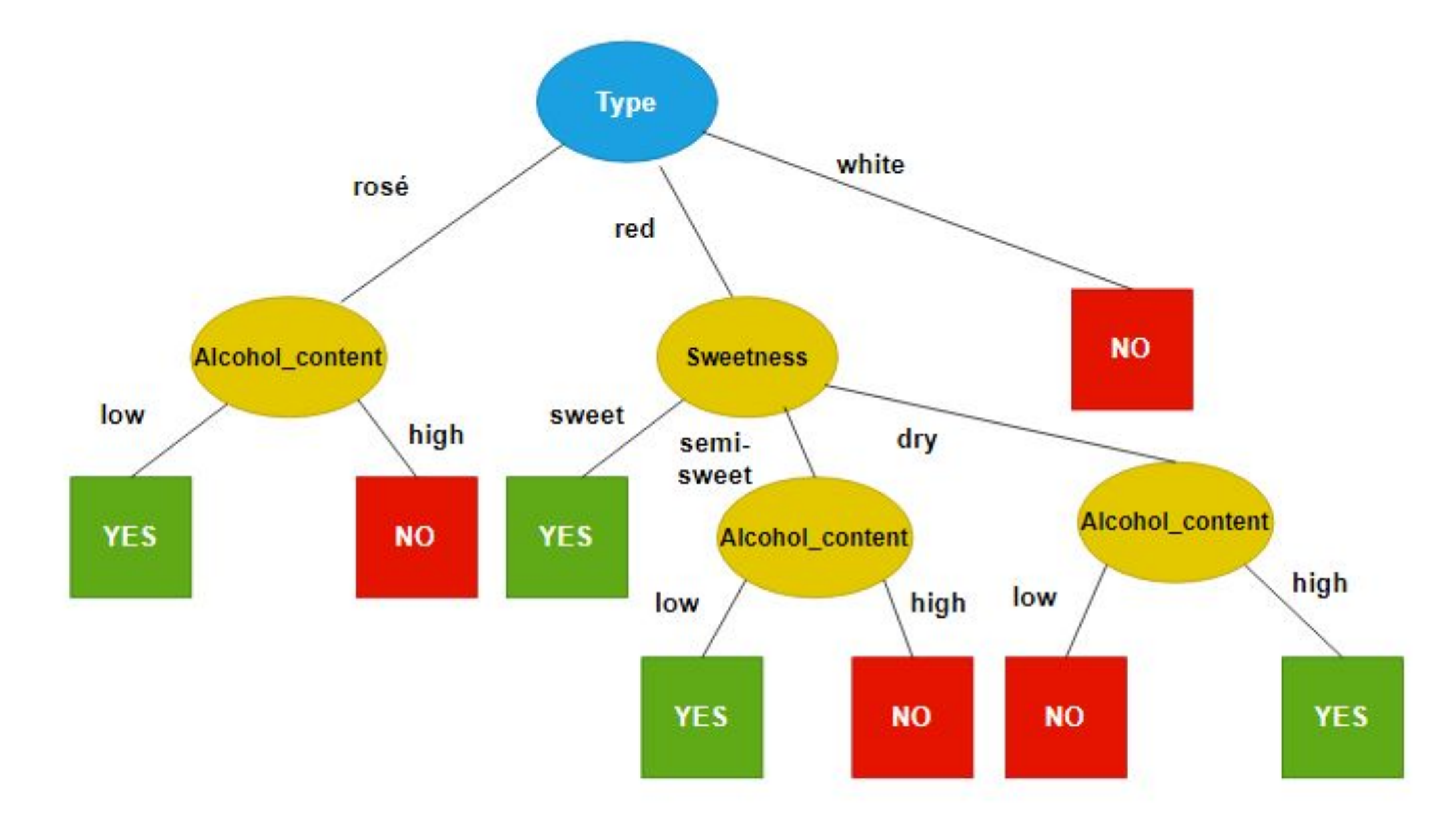
- gain(*Alcohol_content*) = 0.022 bits
- gain(*Sweetness*) = 0.172 bits

Type = white

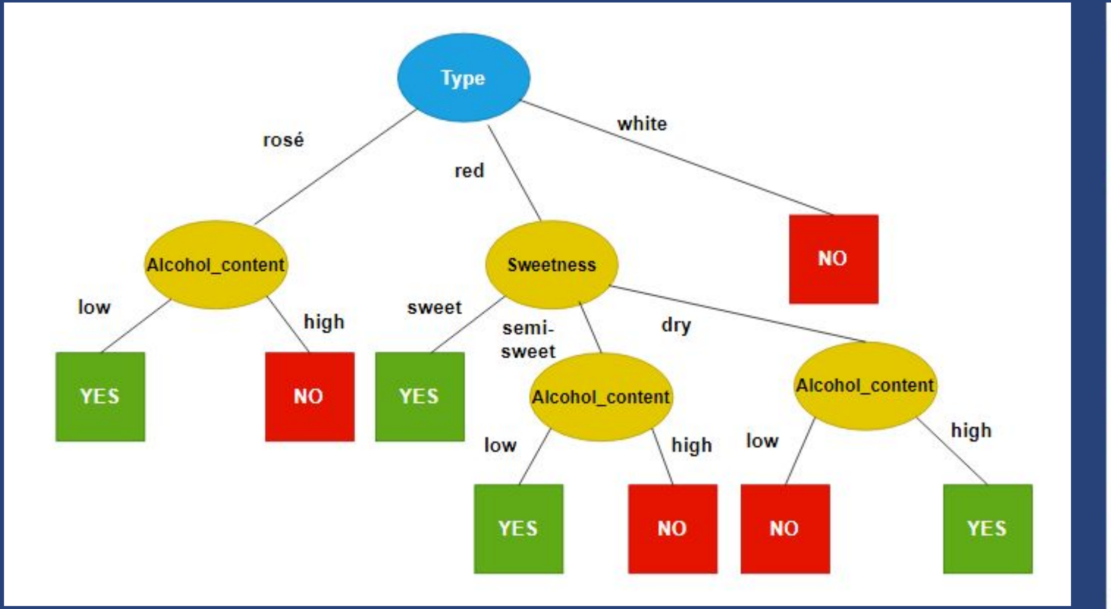


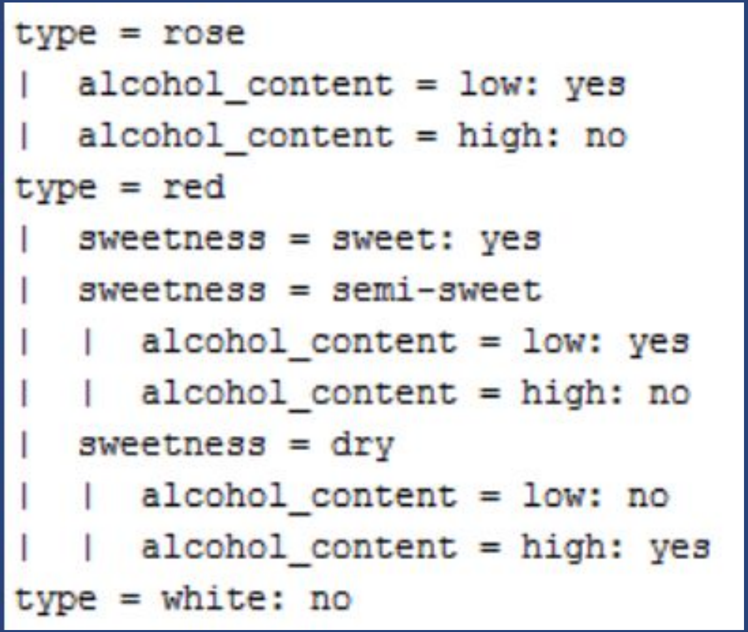
Alcohol_content	Sweetness	Type	(Year)	Popular
low	dry	white	2013	no
low	sweet	white	2006	no

Final tree


Type
rosé
red
white
NO
Alcohol_content
low
YES
high
NO
Sweetness
sweet
YES
semi-sweet
Alcohol_content
low
YES
high
NO
dry
Alcohol_content
low
NO
high
YES

Final tree





```
type = rose
|   alcohol_content = low: yes
|   alcohol_content = high: no
type = red
|   sweetness = sweet: yes
|   sweetness = semi-sweet
|   |   alcohol_content = low: yes
|   |   alcohol_content = high: no
|   sweetness = dry
|   |   alcohol_content = low: no
|   |   alcohol_content = high: yes
type = white: no
```