Automatic learning of decision trees

Lecture 2.1

Data mining task

- Looking for *hidden* patterns, structures, models
- Task: generate a decision tree model from tabular data
- Teach computer to generate the model *automatically,* and then to use the model to make an autonomous decision or to assist us with the decision

Supervised learning

- Input (a set of attributes) and the output (*target* or *class* attribute) are given as a collection of historical records
- Goal: learn the function which maps input to output
- Output is provided by a friendly teacher – *supervised* learning
- When the model has been learned, we can use it to predict a class label of a new record – *predictive* data mining

Decision tree induction (ID3 algorithm*)

- Normal procedure: top down in a recursive divide-and-conquer fashion
  - First: an attribute is selected for root node and an outgoing edge (a branch) is created for each possible attribute value
  - Then: the instances are split into subsets (one for each branch extending from the node)
  - Finally: the same procedure is repeated recursively for each branch, using only instances that reach that branch
- Process stops if all instances have the same class label

*ID3 – Iterative Dichotomizer

Weather dataset

Outlook	Temp	Play
Sunny	30	Yes
Overcast	15	No
Sunny	16	Yes
Rainy	27	Yes
Overcast	25	Yes
Overcast	17	No
Rainy	17	No
Rainy	35	Yes

Weather $\xrightarrow{?}$ Play (Yes, No)

Categorizing numeric attributes

Temp		Temp
30		Hot
15		Chilly
16		Chilly
27	→	Warm
25		Warm
17		Chilly
17		Chilly
35		Hot

Decision tree from the weather dataset

Outlook	Temp	Play
Sunny	Hot	Yes
Overcast	Chilly	No
Sunny	Chilly	Yes
Rainy	Warm	Yes
Overcast	Warm	Yes
Overcast	Chilly	No
Rainy	Chilly	No
Rainy	Hot	Yes

Sunny → Yes

Overcast → Yes No

Rainy → Yes No

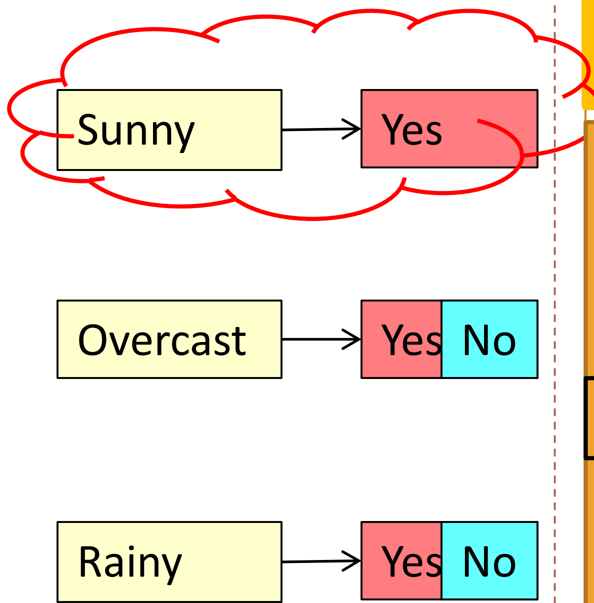
Weather $\xrightarrow{?}$ Play (Yes, No)

- Decision trees
- Supervised learning
- Tree induction algorithm
- Algorithm design issues
- Applications of decision trees

Observations about outlook

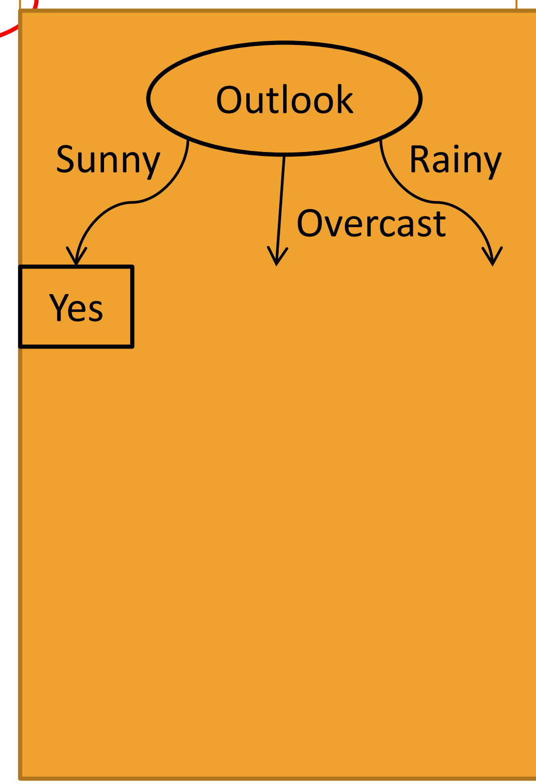
Decision tree from the weather dataset

Outlook	Temp	Play
Sunny	Hot	Yes
Overcast	Chilly	No
Sunny	Chilly	Yes
Rainy	Warm	Yes
Overcast	Warm	Yes
Overcast	Chilly	No
Rainy	Chilly	No
Rainy	Hot	Yes



Weather $\xrightarrow{?}$ Play (Yes, No)

- Decision trees
- Supervised learning
- Tree induction algorithm



- Observations about outlook: if it is sunny, always play

Decision tree from the weather dataset

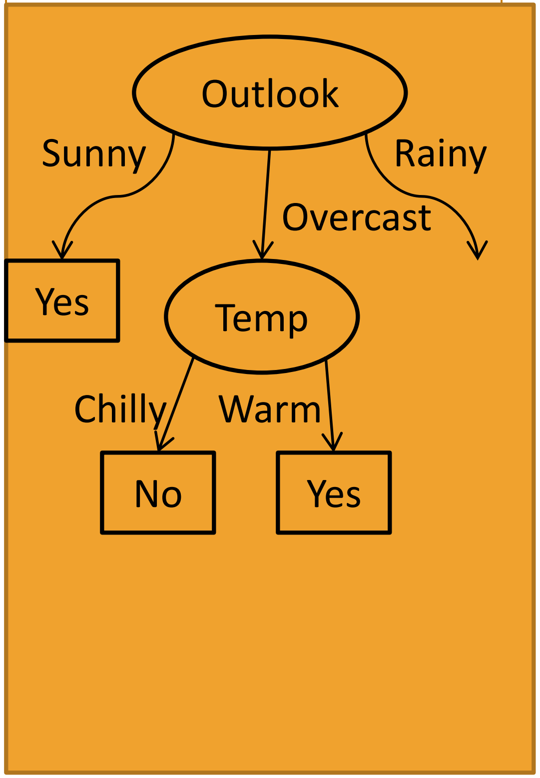
Outlook	Temp	Play
Sunny	Hot	Yes
Overcast	Chilly	No
Sunny	Chilly	Yes
Rainy	Warm	Yes
Overcast	Warm	Yes
Overcast	Chilly	No
Rainy	Chilly	No
Rainy	Hot	Yes

Overcast, Chilly → No

Overcast, Warm → Yes

Weather $\xrightarrow{?}$ Play (Yes, No)

- Decision trees
- Supervised learning
- Tree induction algorithm



- Adding temperature to overcast to arrive to a decision

Decision tree from the weather dataset

Outlook	Temp	Play
Sunny	Hot	Yes
Overcast	Chilly	No
Sunny	Chilly	Yes
Rainy	Warm	Yes
Overcast	Warm	Yes
Overcast	Chilly	No
Rainy	Chilly	No
Rainy	Hot	Yes

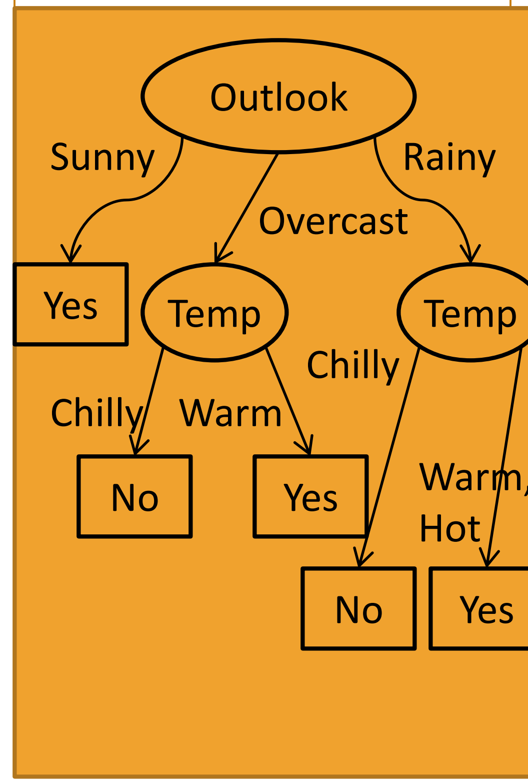
Rainy, Warm → Yes

Rainy, Chilly → No

Rainy, Hot → Yes

Weather $\xrightarrow{?}$ Play (Yes, No)

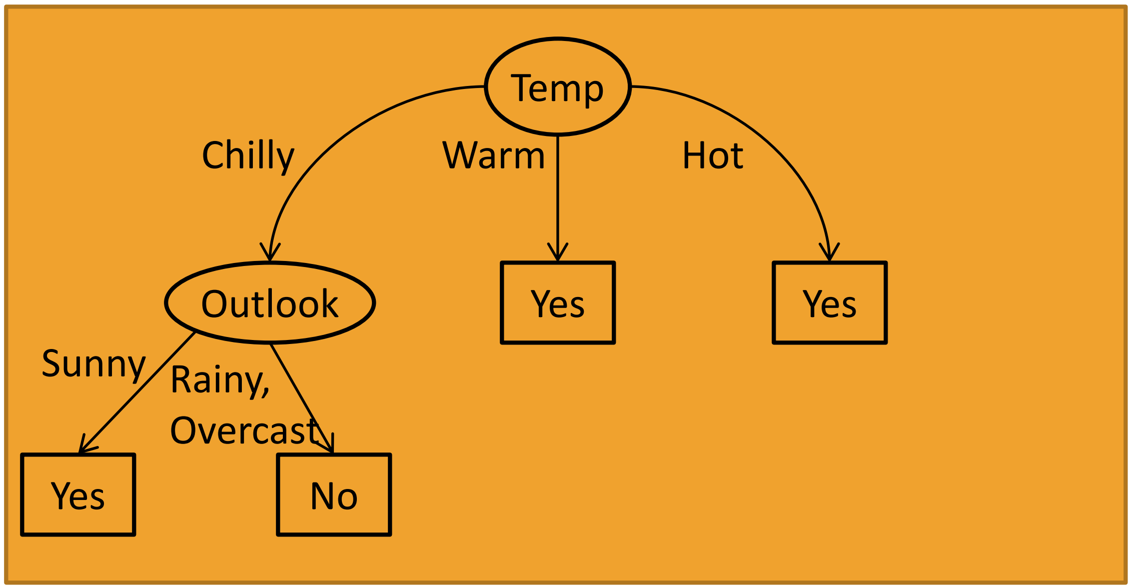
- Decision trees
- Supervised learning
- Tree induction algorithm



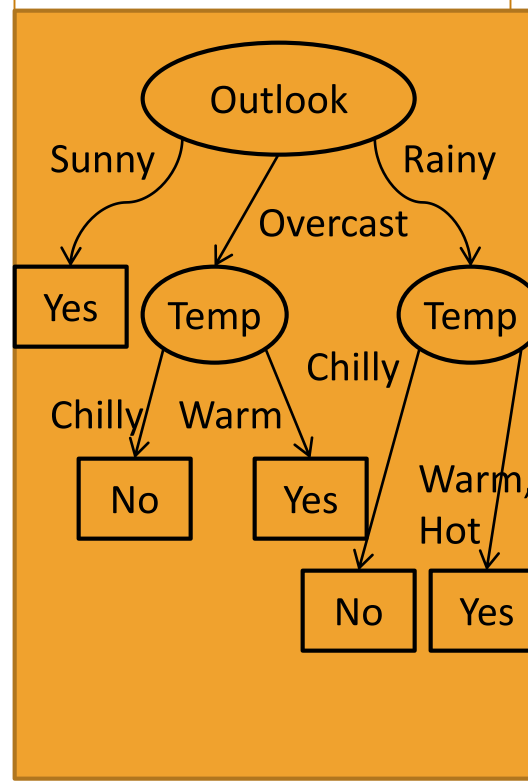
- Adding temperature to rainy to arrive to a decision

Variations

- There are many different trees which fit the same data

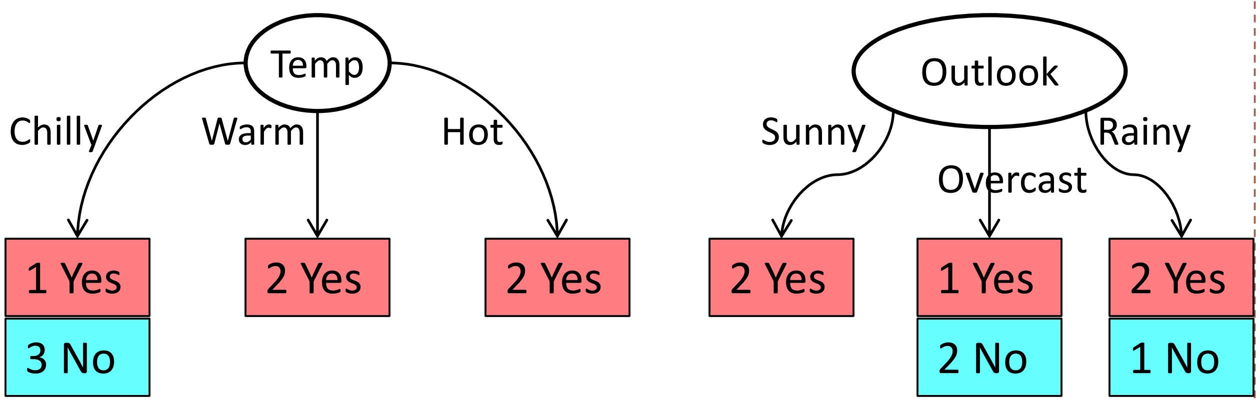


- Decision trees
- Supervised learning
- Tree induction algorithm



Design issues

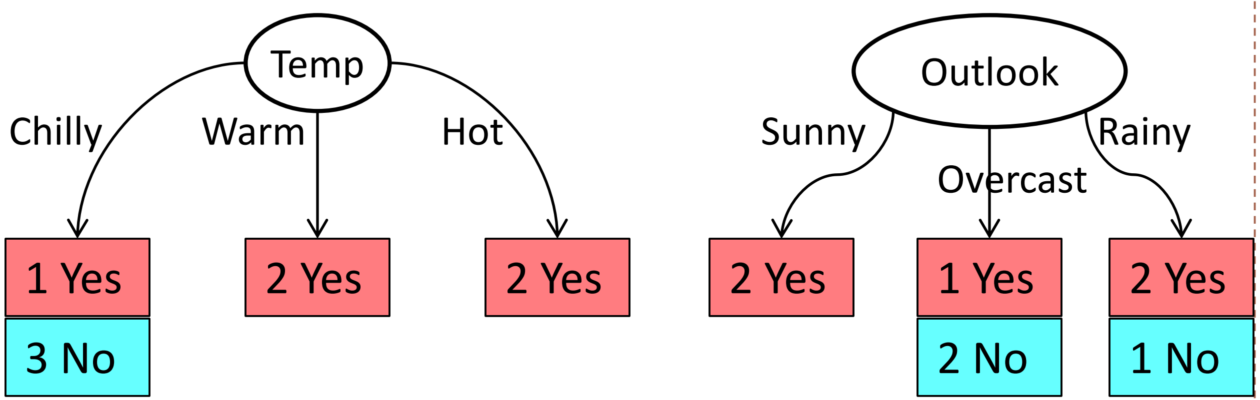
- What attribute to select at each step for splitting?



- Decision trees
- Supervised learning
- Tree induction algorithm
- Algorithm design issues
  - **Best splitting attribute**
- Applications of decision trees

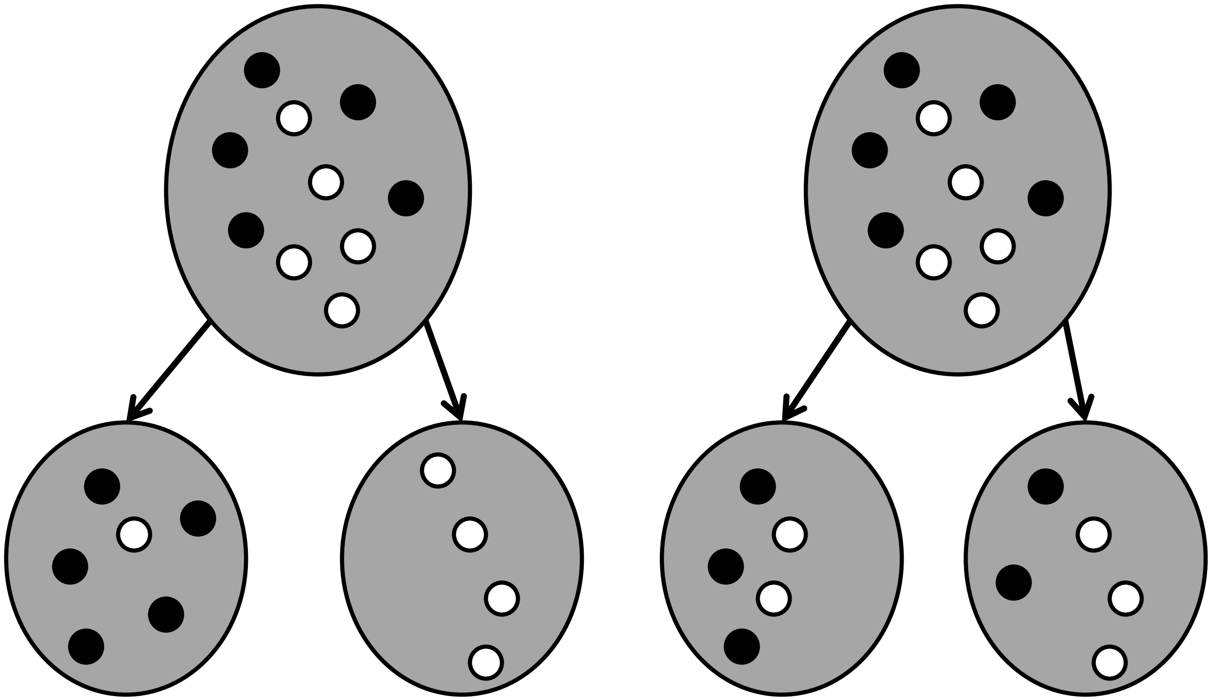
Best splitting attribute: intuition

- The one which divides records into most class-homogenous groups – into nodes with the highest possible *purity*



Purity

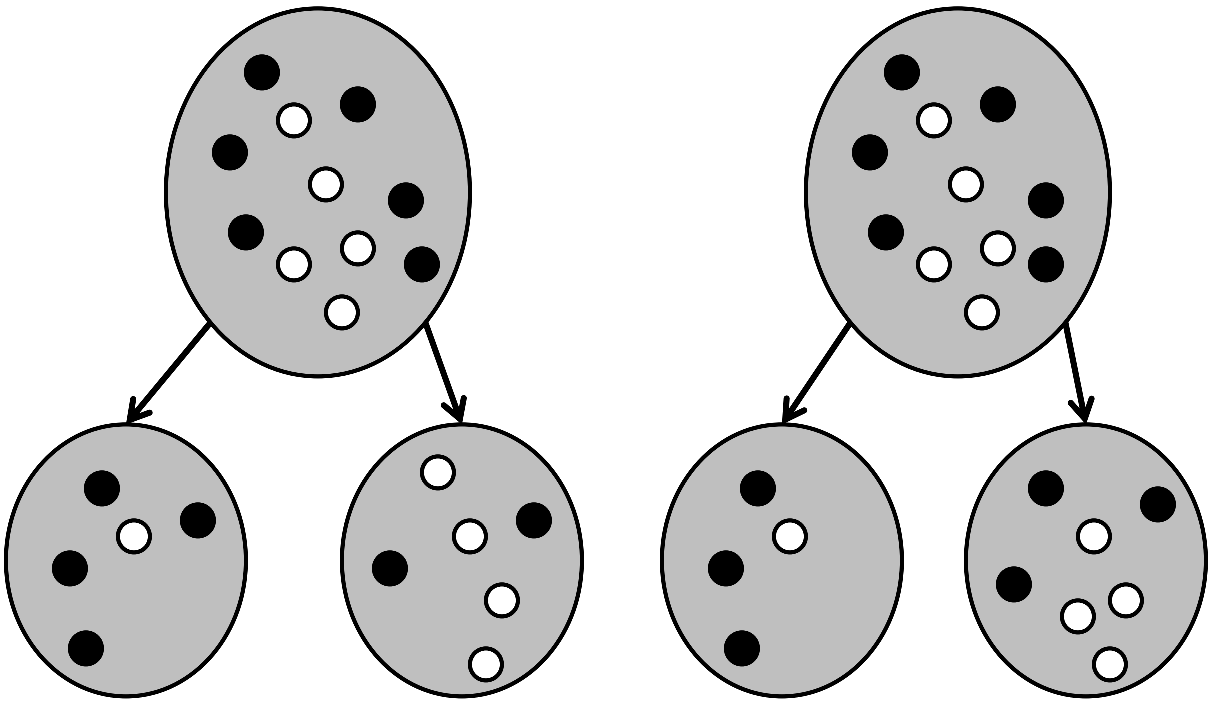
- Which split is better?



- Decision trees
- Supervised learning
- Tree induction algorithm
- Algorithm design issues
  - Best splitting attribute
- Applications of decision trees

Purity

- And now?

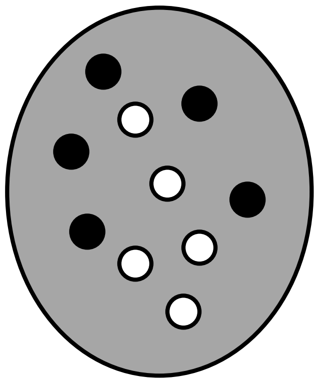


- Decision trees
- Supervised learning
- Tree induction algorithm
- Algorithm design issues
  - Best splitting attribute
- Applications of decision trees

We need a measure of node purity

Purity measure: *GINI* score

- The probability that two items chosen at random are in the same class: the sum of squares of the proportions of the classes

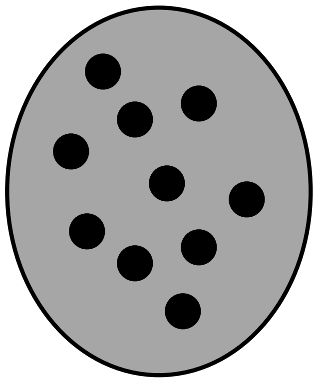


A node with evenly mixed classes has GINI:
$0.5^2+0.5^2=0.5$

The chance of picking the same class twice by random selection is: the probability of picking 2 white dots twice (0.5^2) or picking 2 black dots twice (0.5^2).

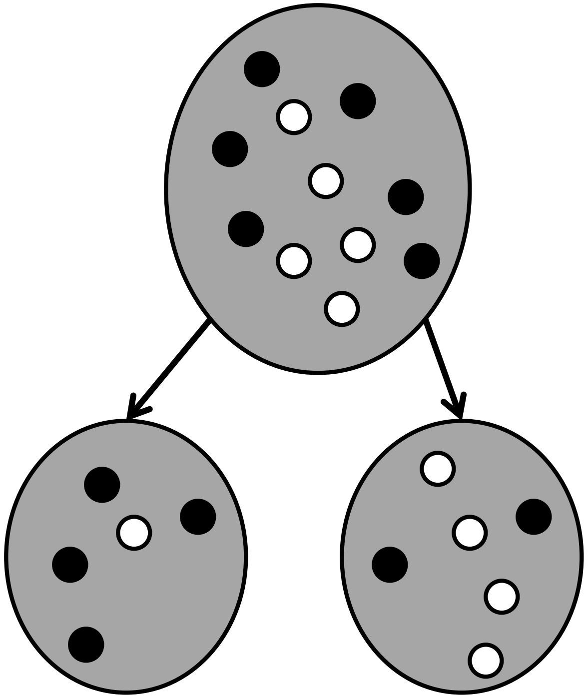
Purity measure: *GINI* score

- The probability that two items chosen at random are in the same class: the sum of squares of the proportions of the classes



A node with one homogenous class has GINI: 1.0
(The chance of picking the same class twice is 100%)

Best split with GINI score

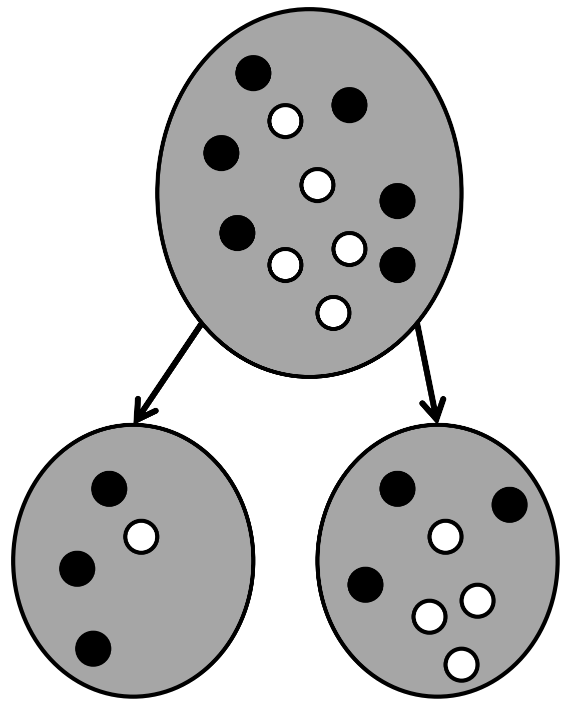


GINI(4,1)=$1/5^2+4/5^2=0.04+0.64=0.68$

GINI(2,4)=$2/6^2+4/6^2=0.11+0.44=0.55$

We take a *weighted average*:
5/11*0.68 + 6/11*0.55=0.31+0.3=0.61

Best split with GINI score



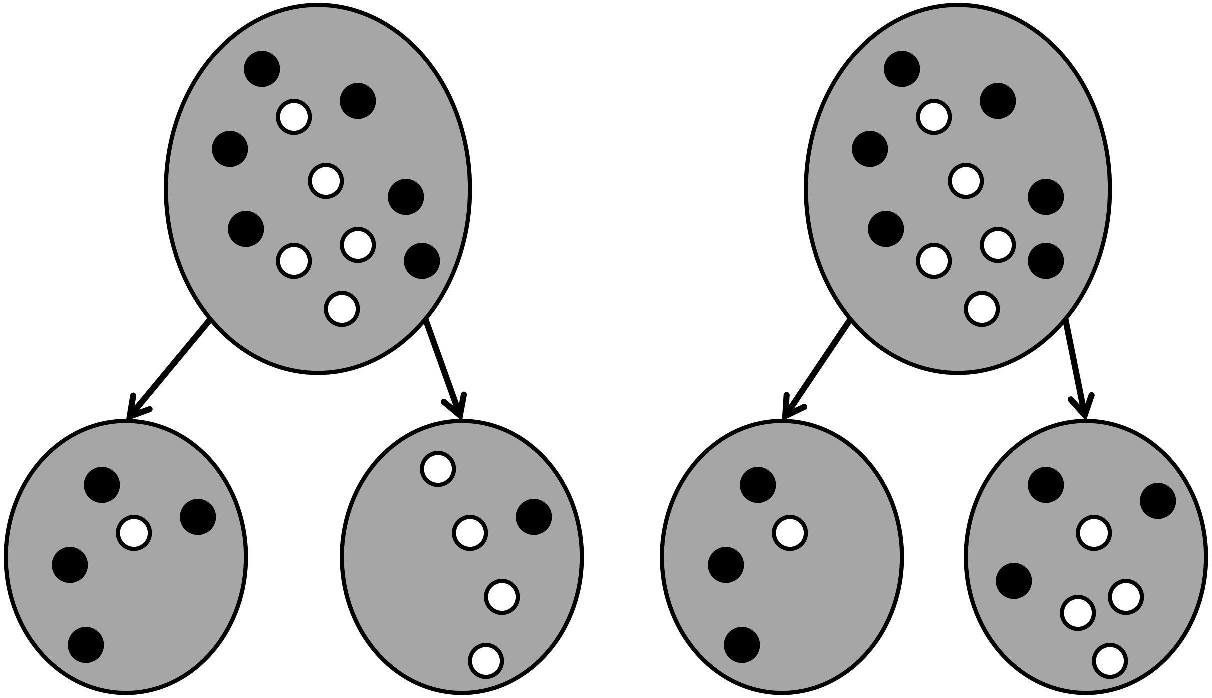
GINI(3,1)=3/4^2+1/4^2=0.56+0.06=0.62

GINI(3,4)=3/7^2+4/7^2=0.18+0.33=0.51

We take a *weighted average*:
4/11*0.62 + 7/11*0.51=0.23+0.32=0.55

- Decision trees
- Supervised learning
- Tree induction algorithm
- Algorithm design issues
  - **Best splitting attribute**
- Applications of decision trees

Comparing average GINI scores



GINI = 0.61

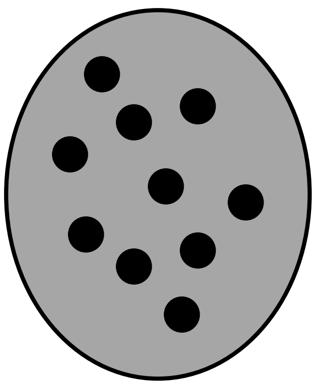
GINI = 0.55

- Decision trees
- Supervised learning
- Tree induction algorithm
- Algorithm design issues
  - Best splitting attribute
- Applications of decision trees

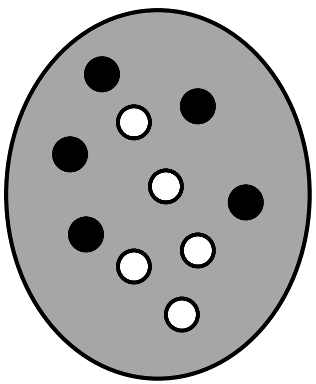
The bigger the GINI score, the better

Purity measure: *Entropy*

- In information theory *entropy* is a measure of how disorganized the system is



A node with one homogenous class has entropy: 0
(very organized)



A node with evenly mixed population has the largest entropy: 1.0
(most disorganized)

- Decision trees
- Supervised learning
- Tree induction algorithm
- Algorithm design issues
  - Best splitting attribute
- Applications of decision trees

The smaller the entropy, the better

Bits

- We are watching a set of independent random samples of X
- We see that X has four possible values

P(X=A) = 1/4	P(X=B) = 1/4	P(X=C) = 1/4	P(X=D) = 1/4

- So we might see: BAACBADCDADDDA…
- We transmit data over a binary serial link. We can encode each reading with two bits (e.g. A=00, B=01, C=10, D = 11)

0100001001001110110011111100…

Fewer Bits

- Someone tells us that the probabilities are not equal

$P(X=A) = 1/2$	$P(X=B) = 1/4$	$P(X=C) = 1/8$	$P(X=D) = 1/8$

- It's possible…

…to invent a coding for your transmission that only uses

1.75 bits on average l. Here is one.

A	0
B	10
C	110
D	111

General Case

- Suppose X can have one of *m* values…

$P(X=V_1) = p_1$	$P(X=V_2) = p_2$	….	$P(X=V_m) = p_m$

- What's the smallest possible number of bits, on average, per symbol, needed to transmit a stream of symbols drawn from X's distribution? It's

$$entropy(p_1,\ldots,p_m) = -p_1 \log_2 p_1 - \ldots p_m \log_2 p_m$$

- Well, Shannon got to this formula by setting down several desirable properties for uncertainty, and then finding it.

Tree node entropy

- Suppose class attribute X in a given tree node occurs in the following proportions

$P(X=V_1) = p_1$	$P(X=V_2) = p_2$	....	$P(X=V_m) = p_m$

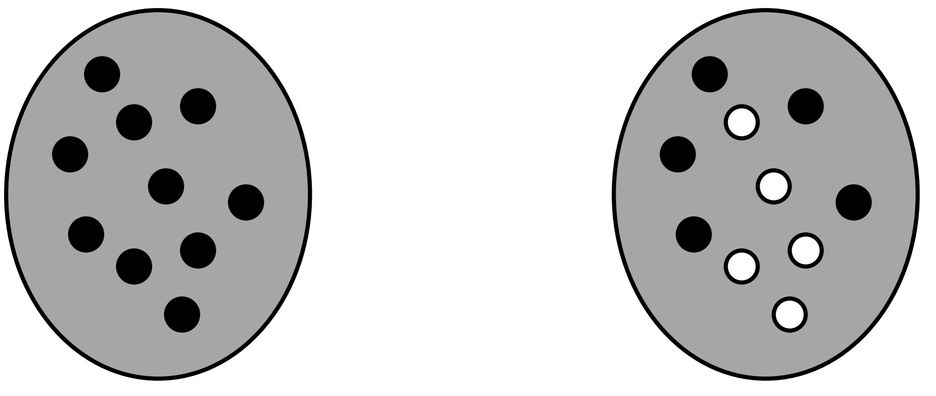
- By finding entropy of the node, we evaluate how many bits are needed to encode this node

$$entropy(p_1, \ldots, p_m) = -p_1 \log_2 p_1 - \ldots p_m \log_2 p_m$$

- The smaller the number of bits to encode the entire tree, the better: the *minimum description length* (MDL) principle

Computing entropy of a node

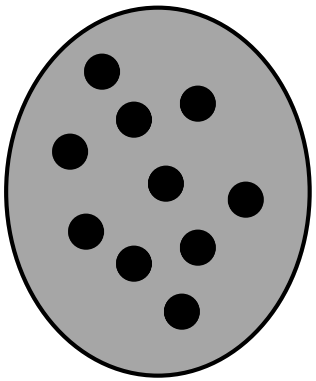
- Compute entropy of a node



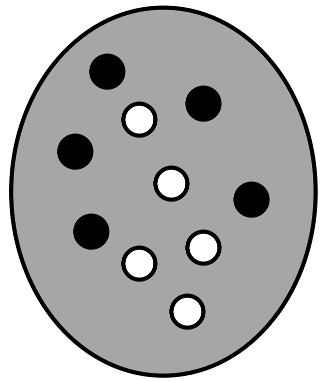
Entropy(10,0)=
-10/10*log (10/10)-0*log (0)
=0

=0 in this formula

Computing entropy of a node



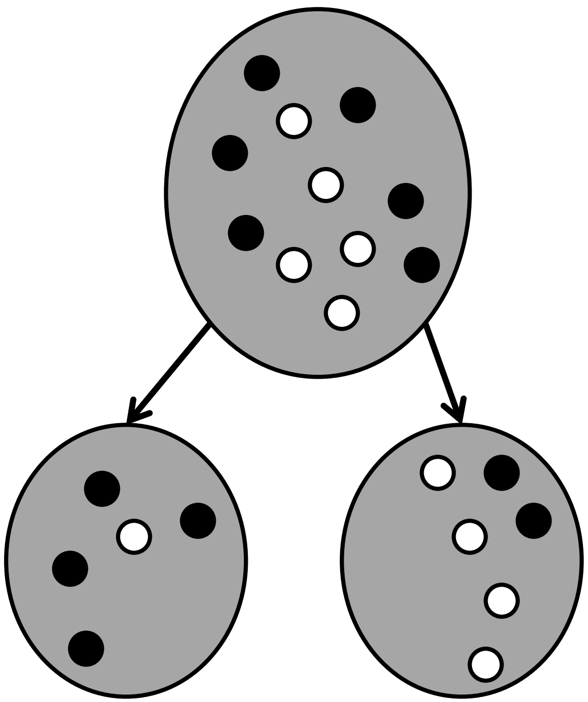
Entropy(10,0)=0



Entropy(5,5)=
-5/10*log 5/10 -5/10*log(5/10)
=1

- Decision trees
- Supervised learning
- Tree induction algorithm
- Algorithm design issues
  - Best splitting attribute
- Applications of decision trees

Best split with Entropy reduction



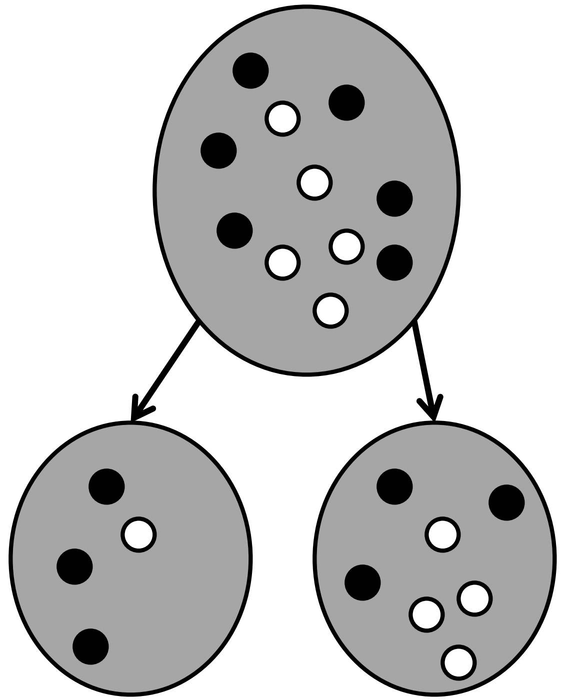
Entropy(4,1)=-4/5 log 4/5-1/5 log 1/5= 0.26+0.46=0.72

Entropy(2,4)=-2/6 log 2/6 - 4/6 log 4/6=0.53+0.39=0.92

We take a *weighted average*:
5/11*0.72 + 6/11*0.92=0.33+0.5=0.83

- Decision trees
- Supervised learning
- Tree induction algorithm
- Algorithm design issues
  - **Best splitting attribute**
- Applications of decision trees

Best split with Entropy reduction

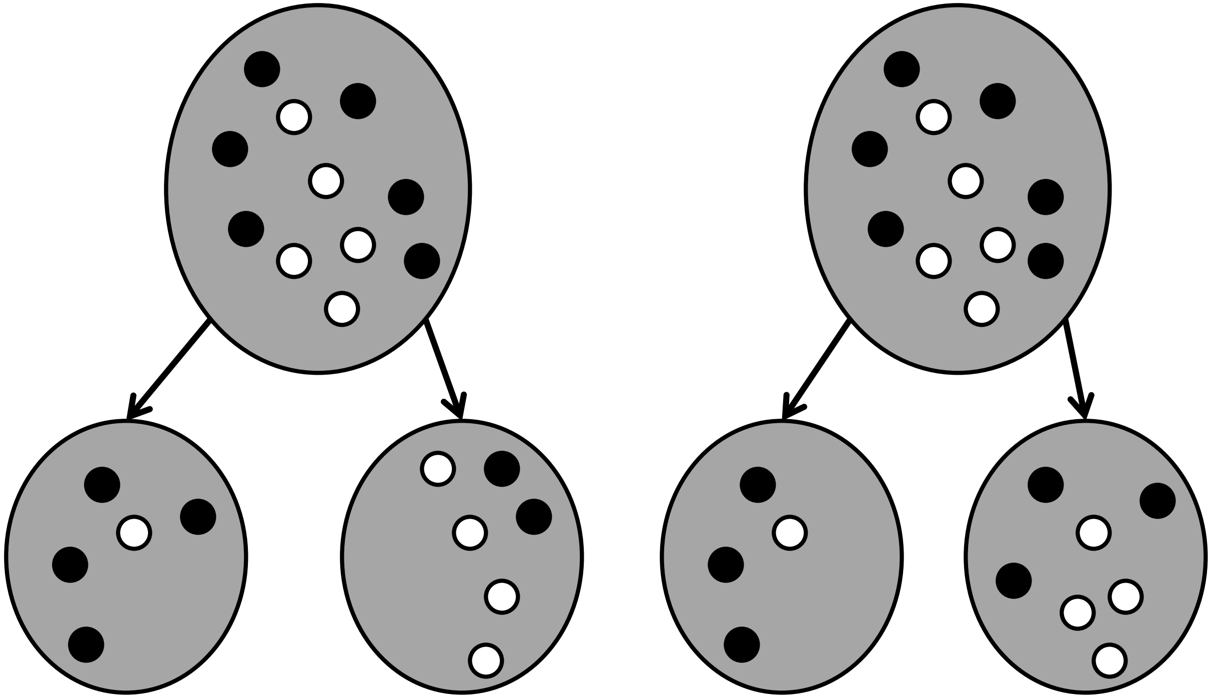


Entropy(3,1)=-3/4 log 3/4-1/4 log 1/4= 0.31+0.5=0.81

Entropy(3,4)=-3/7 log 3/7 - 4/7 log 4/7=0.52+0.46=0.98

We take a *weighted average*:
4/11*0.81 + 7/11*0.98=0.295+0.63=0.92

Comparing average entropies



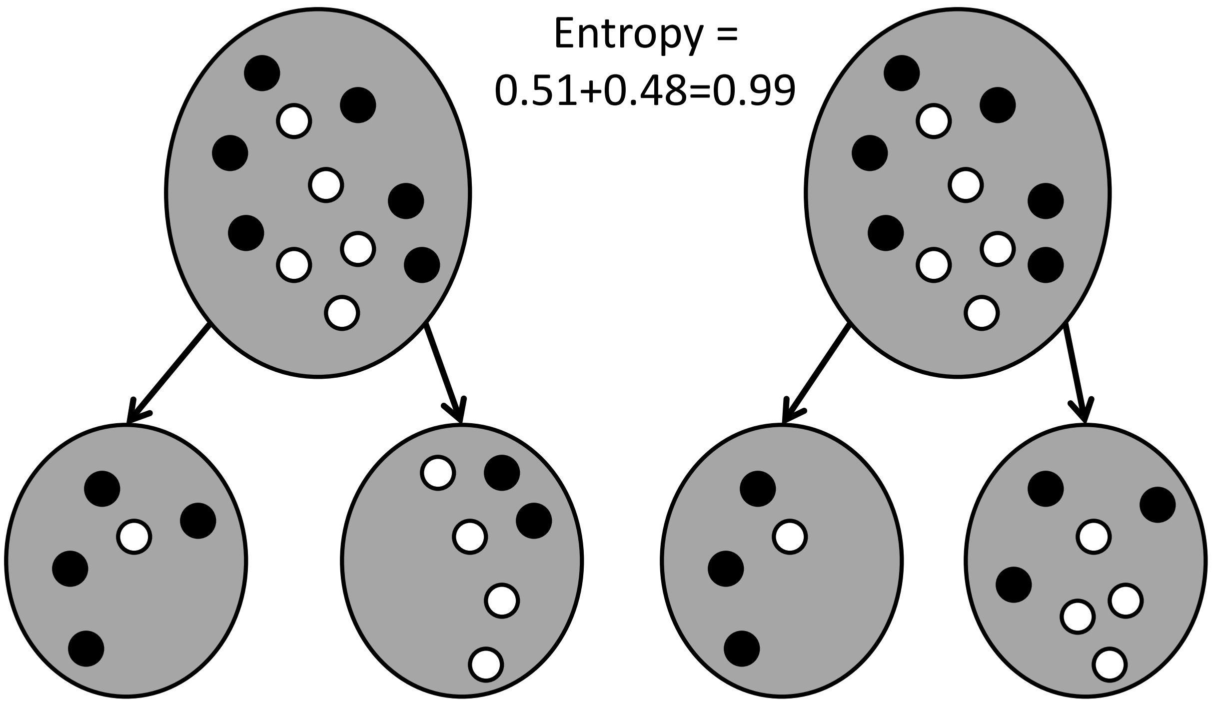
Entropy = 0.83

Entropy=0.92

- Decision trees
- Supervised learning
- Tree induction algorithm
- Algorithm design issues
  - **Best splitting attribute**
- Applications of decision trees

The smaller the entropy, the better

Entropy reduction or *information gain*



Entropy = 0.83

Entropy=0.92

- Decision trees
- Supervised learning
- Tree induction algorithm
- Algorithm design issues
  - **Best splitting attribute**
- Applications of decision trees

In this case, it might be better not to split at all, since the information gain is small

To split or not to split?

- Not to split: when the node consists of elements of the same class
- Not to split: when the node consists of elements which have the same attribute values, except the class attribute
- Not to split: when there is no information gain (no entropy reduction). Not to split when information gain is insignificant

When to use the decision tree classifier

High performance (use decision trees)

- The factors of the decision are not less important than the classification accuracy
- Attributes with nominal values (not numeric) and with low cardinality*
- Categorical class labels with low cardinality*
- There is a set of objective rules underlying the data

Bad performance (use something else)

- Continuous numeric attributes, ordinal attributes
- Hierarchical relationships between classes
- High-cardinality attributes
- Numeric value prediction

- Decision trees
- Supervised learning
- Tree induction algorithm
- Algorithm design issues
  - Best splitting attribute
  - When to stop splitting
- Applications of decision trees

*cardinality - the number of possible distinct values