Learning Decision Trees

Machine Learning
Fall 2017

THE UNIVERSITY OF UTAH

Some slides from Tom Mitchell, Dan Roth and others

Key issues in machine learning

- Modeling

  How to formulate your problem as a machine learning problem? How to represent data? Which algorithms to use? What learning protocols?

- Representation

  Good hypothesis spaces and good features

- Algorithms
  - What is a good learning algorithm?
  - What is success?
  - Generalization vs overfitting
  - The computational question: How long will learning take?

This lecture: Learning Decision Trees

1. **Representation**: What are decision trees?

2. **Algorithm**: Learning decision trees
   – The ID3 algorithm: A greedy heuristic

3. Some extensions

This lecture: Learning Decision Trees

1. **Representation**: What are decision trees?

2. **Algorithm**: Learning decision trees
   – The ID3 algorithm: A greedy heuristic

3. Some extensions

Will I play tennis today?

- **Features**
  - Outlook: {Sun, Overcast, Rain}
  - Temperature: {Hot, Mild, Cool}
  - Humidity: {High, Normal, Low}
  - Wind: {Strong, Weak}

- **Labels**
  - Binary classification task: Y = {+, -}

Will I play tennis today?

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

Outlook: S(unny), O(vercast), R(ainy)

Temperature: H(ot), M(edium), C(ool)

Humidity: H(igh), N(ormal), L(ow)

Wind: S(trong), W(eak)

Basic Decision Tree Learning Algorithm

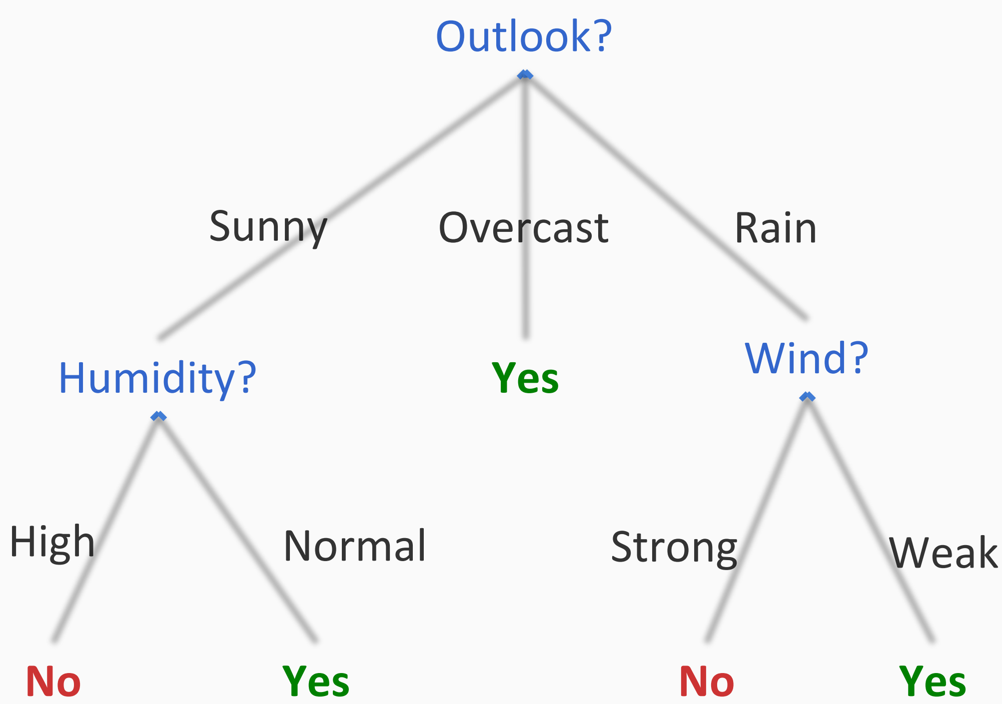
- Data is processed in Batch (i.e. all the data available)

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

Basic Decision Tree Learning Algorithm

- Data is processed in Batch (i.e. all the data available)
- Recursively build a decision tree top down.

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-



Basic Decision Tree Algorithm: ID3

Input:
S the set of Examples
Label is the target attribute (the prediction)
Attributes is the set of measured attributes

ID3(S, Attributes, Label):

1. If all examples have same label:

   Return a single node tree with the label

Basic Decision Tree Algorithm: ID3

Input:

S the set of Examples

Label is the target attribute (the prediction)

Attributes is the set of measured attributes

ID3(S, Attributes, Label):

1. If all examples have same label:

   Return a single node tree with the label

2. Otherwise

Basic Decision Tree Algorithm: ID3

Input:
S the set of Examples
Label is the target attribute (the prediction)
Attributes is the set of measured attributes

ID3(S, Attributes, Label):

1. If all examples are have same label:
   Return a single node tree with the label
2. Otherwise
   1. Create a `Root node` for tree

   4. Return `Root node`

Basic Decision Tree Algorithm: ID3

Input:
S the set of Examples
Label is the target attribute (the prediction)
Attributes is the set of measured attributes

ID3(S, Attributes, Label):

1. If all examples are have same label:
   Return a single node tree with the label
2. Otherwise
   1. Create a `Root node` for tree
   2. A = attribute in Attributes that ***best*** classifies S

   4. Return `Root node`

Basic Decision Tree Algorithm: ID3

Input:
S the set of Examples
Label is the target attribute (the prediction)
Attributes is the set of measured attributes

ID3(S, Attributes, Label):

1. If all examples are have same label:
   Return a single node tree with the label
2. Otherwise
   1. Create a `Root node` for tree
   2. A = attribute in Attributes that ***best*** classifies S
   3. for each possible value v of that A can take:

   4. Return `Root node`

Basic Decision Tree Algorithm: ID3

Input:
S the set of Examples
Label is the target attribute (the prediction)
Attributes is the set of measured attributes

ID3(S, Attributes, Label):

1. If all examples are have same label:
   Return a single node tree with the label
2. Otherwise
   1. Create a `Root node` for tree
   2. A = attribute in Attributes that ***best*** classifies S
   3. for each possible value v of that A can take:
      1. Add a new tree branch corresponding to A=v
   4. Return `Root node`

Basic Decision Tree Algorithm: ID3

Input:
S the set of Examples
Label is the target attribute (the prediction)
Attributes is the set of measured attributes

ID3(S, Attributes, Label):

1. If all examples are have same label:
   Return a single node tree with the label
2. Otherwise
   1. Create a `Root node` for tree
   2. A = attribute in Attributes that ***best*** classifies S
   3. for each possible value v of that A can take:
      1. Add a new tree branch corresponding to A=v
      2. Let S_v be the subset of examples in S with A=v
   4. Return `Root node`

Basic Decision Tree Algorithm: ID3

Input:
S the set of Examples
Label is the target attribute (the prediction)
Attributes is the set of measured attributes

ID3(S, Attributes, Label):

1. If all examples are have same label:
   Return a single node tree with the label
2. Otherwise
   1. Create a `Root node` for tree
   2. A = attribute in Attributes that *best* classifies S
   3. for each possible value v of that A can take:
      1. Add a new tree branch corresponding to A=v
      2. Let S_v be the subset of examples in S with A=v
      3. if S_v is empty:
   4. Return `Root node`

Basic Decision Tree Algorithm: ID3

Input:
S the set of Examples
Label is the target attribute (the prediction)
Attributes is the set of measured attributes

ID3(S, Attributes, Label):

1. If all examples are have same label:
   Return a single node tree with the label
2. Otherwise
   1. Create a `Root node` for tree
   2. A = attribute in Attributes that ***best*** classifies S
   3. for each possible value v that A can take:
      1. Add a new tree branch corresponding to A=v
      2. Let S_v be the subset of examples in S with A=v
      3. if S_v is empty:
         add leaf node with the common value of Label in S why?
   4. Return `Root node`

Basic Decision Tree Algorithm: ID3

Input:
S the set of Examples
Label is the target attribute (the prediction)
Attributes is the set of measured attributes

ID3(S, Attributes, Label):

1. If all examples are have same label:
   Return a single node tree with the label
2. Otherwise
   1. Create a `Root node` for tree
   2. A = attribute in Attributes that ***best*** classifies S
   3. for each possible value v of that A can take:
      1. Add a new tree branch corresponding to A=v
      2. Let S_v be the subset of examples in S with A=v
      3. if S_v is empty:
         add leaf node with the common value of Label in S why?
         For generalization at test time
   4. Return `Root node`

Basic Decision Tree Algorithm: ID3

Input:
S the set of Examples
Label is the target attribute (the prediction)
Attributes is the set of measured attributes

ID3(S, Attributes, Label):

1. If all examples are have same label:
   Return a single node tree with the label
2. Otherwise
   1. Create a `Root node` for tree
   2. A = attribute in Attributes that ***best*** classifies S
   3. for each possible value v of that A can take:
      1. Add a new tree branch corresponding to A=v
      2. Let S_v be the subset of examples in S with A=v
      3. if S_v is empty:
         add leaf node with the common value of Label in S — why?
         For generalization at test time
         Else:
         below this branch add the subtree ID3(S_v, Attributes - {A}, Label)
   4. Return `Root node`

Picking the Root Attribute

- Goal: Have the resulting decision tree **as small as possible** (*Occam's Razor*)
  - But, finding the minimal decision tree consistent with data is NP-hard

- The recursive algorithm is a greedy heuristic search for a simple tree, but cannot guarantee optimality

- The main decision in the algorithm is the selection of the next attribute to split on

Picking the Root Attribute

Consider data with two Boolean attributes (A,B).

< (A=0,B=0), – >: 50 examples

< (A=0,B=1), – >: 50 examples

< (A=1,B=0), – >: 0 examples

< (A=1,B=1), + >: 100 examples

Picking the Root Attribute

Consider data with two Boolean attributes (A,B).

< (A=0,B=0), – >: 50 examples

< (A=0,B=1), – >: 50 examples

< (A=1,B=0), – >: 0 examples

< (A=1,B=1), + >: 100 examples

What should be the first attribute we select?

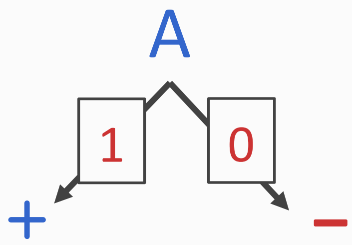
Picking the Root Attribute

Consider data with two Boolean attributes (A,B).

< (A=0,B=0), – >: 50 examples
< (A=0,B=1), – >: 50 examples
< (A=1,B=0), – >: 0 examples
< (A=1,B=1), + >: 100 examples

What should be the first attribute we select?

Splitting on A: we get purely labeled nodes.



Picking the Root Attribute

Consider data with two Boolean attributes (A,B).

< (A=0,B=0), – >: 50 examples

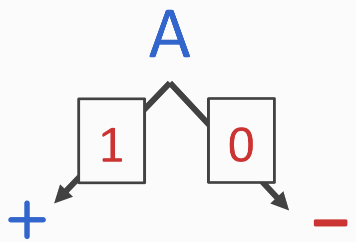
< (A=0,B=1), – >: 50 examples

< (A=1,B=0), – >: 0 examples

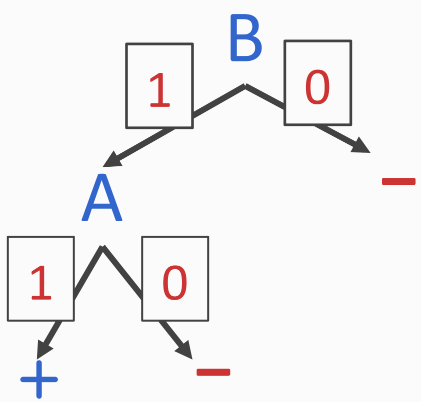
< (A=1,B=1), + >: 100 examples

What should be the first attribute we select?

Splitting on A: we get purely labeled nodes.



Splitting on B: we don't get purely labeled nodes.



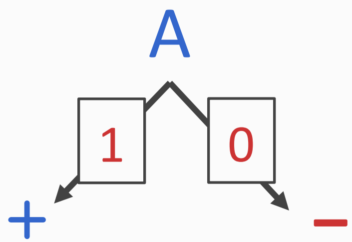
Picking the Root Attribute

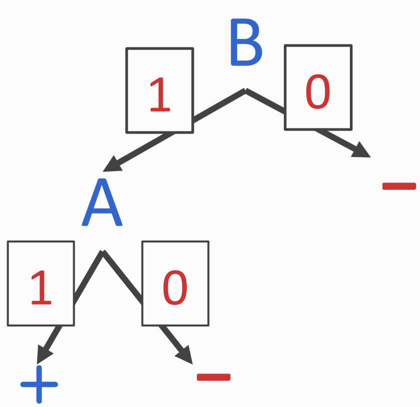
Consider data with two Boolean attributes (A,B).

< (A=0,B=0), – >: 50 examples
< (A=0,B=1), – >: 50 examples
< (A=1,B=0), – >: 0 examples
< (A=1,B=1), + >: 100 examples

What should be the first attribute we select?

Splitting on A: we get purely labeled nodes.





Splitting on B: we don't get purely labeled nodes.

What if we have: <(A=1,B=0), - >: 3 examples

Picking the Root Attribute

Consider data with two Boolean attributes (A,B).

< (A=0,B=0), – >: 50 examples

< (A=0,B=1), – >: 50 examples

< (A=1,B=0), – >: ~~0 examples~~ 3 examples

< (A=1,B=1), + >: 100 examples

Which attribute should we choose?

Picking the Root Attribute

Consider data with two Boolean attributes (A,B).

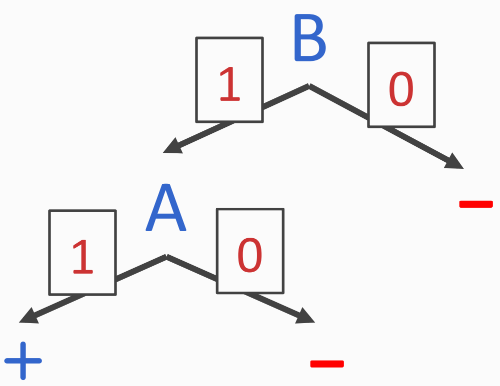
< (A=0,B=0), – >: 50 examples

< (A=0,B=1), – >: 50 examples

< (A=1,B=0), – >: ~~0 examples~~ 3 examples

< (A=1,B=1), + >: 100 examples

Which attribute should we choose?



Picking the Root Attribute

Consider data with two Boolean attributes (A,B).

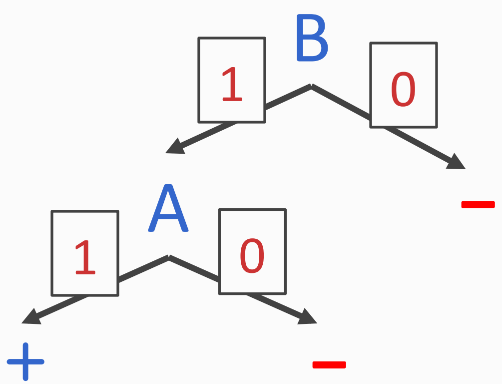
< (A=0,B=0), – >: 50 examples

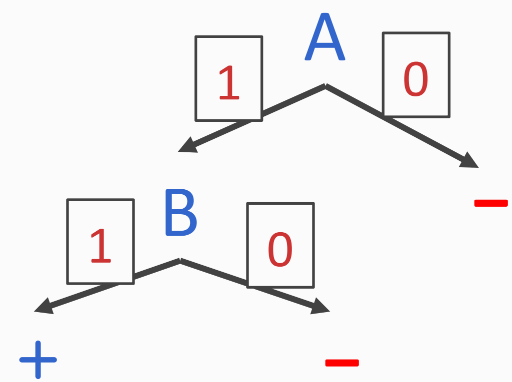
< (A=0,B=1), – >: 50 examples

< (A=1,B=0), – >: ~~0 examples~~ 3 examples

< (A=1,B=1), + >: 100 examples

Which attribute should we choose?



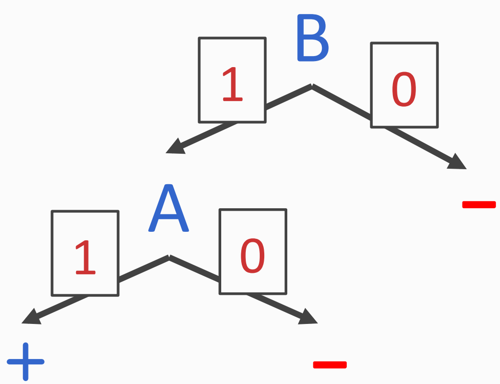


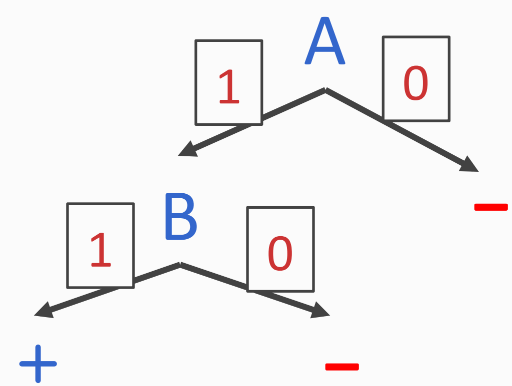
Picking the Root Attribute

Consider data with two Boolean attributes (A,B).

< (A=0,B=0), – >: 50 examples
< (A=0,B=1), – >: 50 examples
< (A=1,B=0), – >: ~~0 examples~~ 3 examples
< (A=1,B=1), + >: 100 examples

Which attribute should we choose? Trees looks structurally similar!



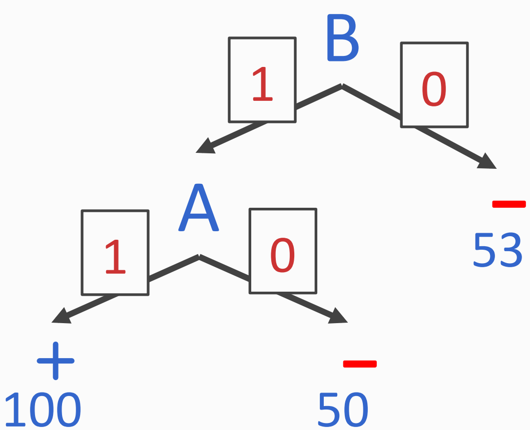


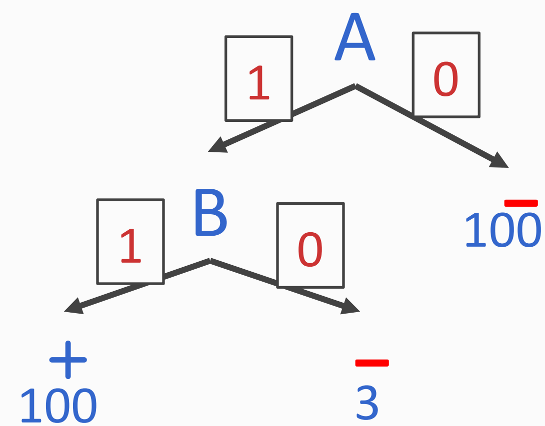
Picking the Root Attribute

Consider data with two Boolean attributes (A,B).

< (A=0,B=0), – >: 50 examples
< (A=0,B=1), – >: 50 examples
< (A=1,B=0), – >: ~~0 examples~~ 3 examples
< (A=1,B=1), + >: 100 examples

Which attribute should we choose? Trees looks structurally similar!





Picking the Root Attribute

Consider data with two Boolean attributes (A,B).

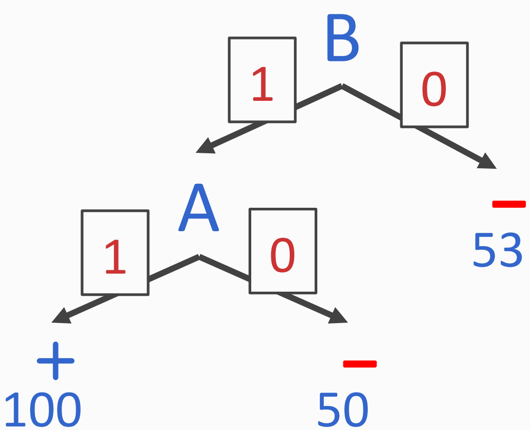
< (A=0,B=0), – >: 50 examples

< (A=0,B=1), – >: 50 examples

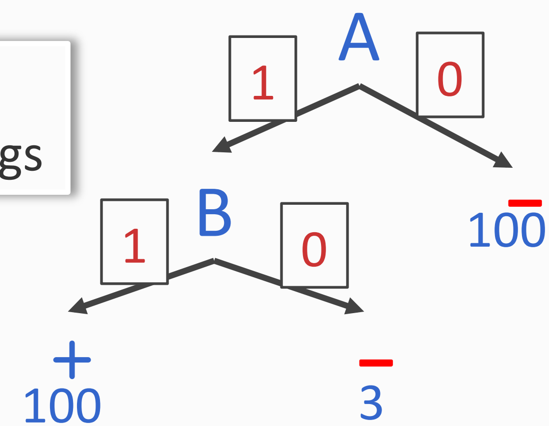
< (A=1,B=0), – >: ~~0 examples~~ 3 examples

< (A=1,B=1), + >: 100 examples

Which attribute should we choose? Trees looks structurally similar!



Advantage A. But…
Need a way to quantify things



Picking the Root Attribute

Goal: Have the resulting decision tree **as small as possible** (*Occam's Razor*)

- The main decision in the algorithm is the selection of the next attribute for splitting the data

- We want attributes that split the examples to sets that are relatively pure in one label
  - This way we are closer to a leaf node.

- The most popular heuristic is information gain, originated with the ID3 system of Quinlan

Reminder: Entropy

Entropy (impurity, disorder) of a set of examples S with respect to binary classification is

$$Entropy(S) = H(S) = -p_+ \log(p_+) - p_- \log(p_-)$$

- The proportion of positive examples is p_+
- The proportion of negative examples is p_-

Reminder: Entropy

Entropy (impurity, disorder) of a set of examples S with respect to binary classification is

$$Entropy(S) = H(S) = -p_+ \log(p_+) - p_- \log(p_-)$$

- The proportion of positive examples is p_+
- The proportion of negative examples is p_-

In general, for a discrete probability distribution with K possible values, with probabilities $\{p_1, p_2, \cdots, p_K\}$ the entropy is given by

$$H(\{p_1, p_2, \cdots, p_K\}) = -\sum_{i=1}^{K} p_i \log(p_i)$$

Reminder: Entropy

Entropy (impurity, disorder) of a set of examples S with respect to binary classification is

$$Entropy(S) = H(S) = -p_+ \log(p_+) - p_- \log(p_-)$$

- The proportion of positive examples is p_+
- The proportion of negative examples is p_-

- If all examples belong to the same category, entropy = 0
- If $p_+ = p_- = 0.5$, entropy = 1

Entropy can be viewed as the number of bits required, on average, to encode the class of labels. If the probability for + is 0.5, a single bit is required for each example; if it is 0.8: can use less then 1 bit.

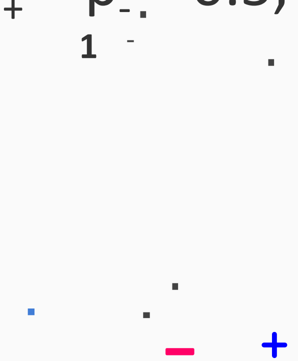
Reminder: Entropy

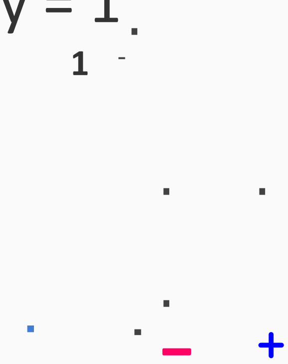
Entropy (impurity, disorder) of a set of examples S with respect to binary classification is

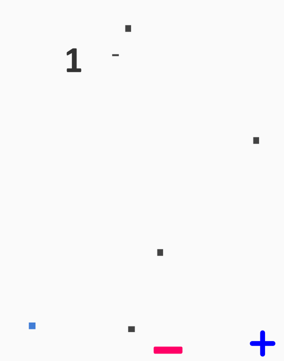
$$Entropy(S) = H(S) = -p_+ \log(p_+) - p_- \log(p_-)$$

- The proportion of positive examples is p_+
- The proportion of negative examples is p_-

- If all examples belong to the same category, entropy = 0
- If $p_+ = p_- = 0.5$, entropy = 1



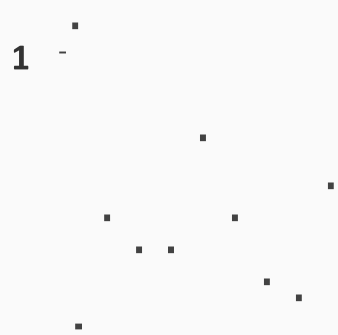


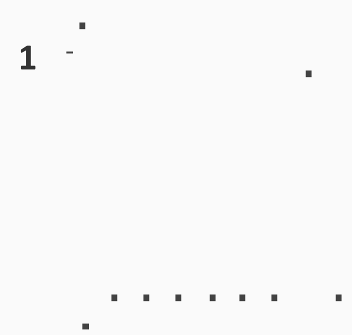


Reminder: Entropy

Entropy (impurity, disorder) of a set of examples S with respect to binary classification is

The uniform distribution has the highest entropy

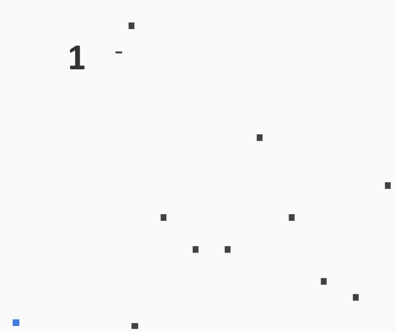


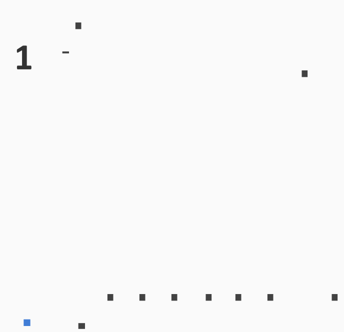


Reminder: Entropy

Entropy (impurity, disorder) of a set of examples S with respect to binary classification is

The uniform distribution has the highest entropy





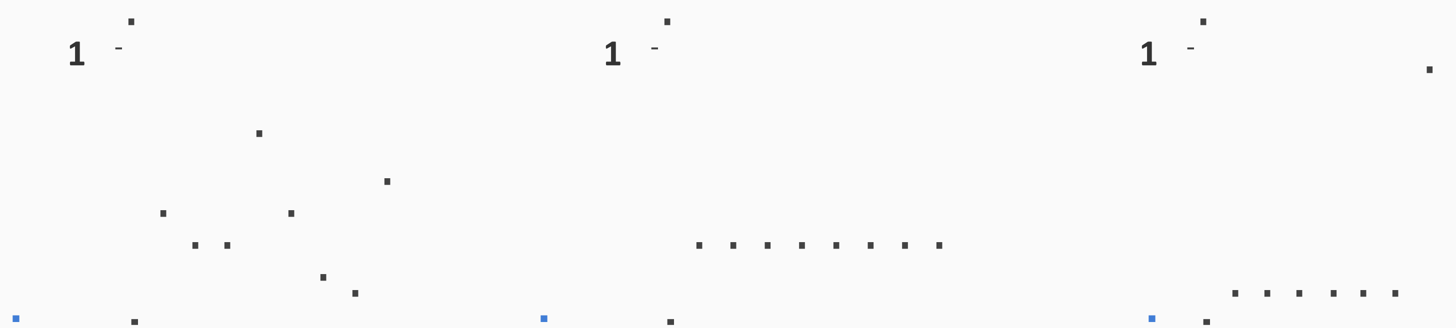
Reminder: Entropy

High Entropy – High level of Uncertainty

Low Entropy – No Uncertainty.

Entropy (impurity, disorder) of a set of examples S with respect to binary classification is

The uniform distribution has the highest entropy



Picking the Root Attribute

Goal: Have the resulting decision tree **as small as possible** (*Occam's Razor*)

- The main decision in the algorithm is the selection of the next attribute for splitting the data

- We want attributes that split the examples to sets that are relatively pure in one label
  - This way we are closer to a leaf node.

- The most popular heuristic is information gain, originated with the ID3 system of Quinlan

Information Gain

The *information gain* of an attribute A is the expected reduction in entropy caused by partitioning on this attribute

$$Gain(S, A) = Entropy(S) - \sum_{v \in \mathrm{Values}(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

S_v: the subset of examples where the value of attribute A is set to value v

Information Gain

The *information gain* of an attribute A is the expected reduction in entropy caused by partitioning on this attribute

$$Gain(S, A) = Entropy(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

S_v: the subset of examples where the value of attribute A is set to value v

Entropy of partitioning the data is calculated by weighing the entropy of each partition by its size relative to the original set

– Partitions of low entropy (imbalanced splits) lead to high gain

Information Gain

The *information gain* of an attribute A is the expected reduction in entropy caused by partitioning on this attribute

$$Gain(S, A) = Entropy(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

S_v: the subset of examples where the value of attribute A is set to value v

Entropy of partitioning the data is calculated by weighing the entropy of each partition by its size relative to the original set

- Partitions of low entropy (imbalanced splits) lead to high gain

Go back to check which of the A, B splits is better

Information Gain

- High Entropy – High level of Uncertainty
- Low Entropy – No Uncertainty.

The *information gain* of an attribute A is the expected reduction in entropy caused by partitioning on this attribute

$$Gain(S, A) = Entropy(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

S_v: the subset of examples where the value of attribute A is set to value v

Entropy of partitioning the data is calculated by weighing the entropy of each partition by its size relative to the original set

– Partitions of low entropy (imbalanced splits) lead to high gain

Go back to check which of the A, B splits is better

Will I play tennis today?

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

Outlook: S(unny), O(vercast), R(ainy)

Temperature: H(ot), M(edium), C(ool)

Humidity: H(igh), N(ormal), L(ow)

Wind: S(trong), W(eak)

Will I play tennis today?

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

Current entropy:
$p = 9/14$
$n = 5/14$

$H(\text{Play?}) = -(9/14)\log_2(9/14) - (5/14)\log_2(5/14)$
≈ 0.94

Information Gain: Outlook

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

Information Gain: Outlook

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

Outlook = sunny: 5 of 14 examples

$p = 2/5$ $n = 3/5$ $\mathbf{H_S = 0.971}$

Information Gain: Outlook

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

Outlook = sunny: 5 of 14 examples

$p = 2/5$ $n = 3/5$ $\mathbf{H_S = 0.971}$

Outlook = overcast: 4 of 14 examples

$p = 4/4$ $n = 0$ $\mathbf{H_o = 0}$

Information Gain: Outlook

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

Outlook = sunny: 5 of 14 examples

$p = 2/5$ $n = 3/5$ $\mathbf{H_S = 0.971}$

Outlook = overcast: 4 of 14 examples

$p = 4/4$ $n = 0$ $\mathbf{H_o = 0}$

Outlook = rainy: 5 of 14 examples

$p = 3/5$ $n = 2/5$ $\mathbf{H_R = 0.971}$

Expected entropy:

$(5/14)\times0.971 + (4/14)\times0 + (5/14)\times0.971$

$= \mathbf{0.694}$

Information gain:

$0.940 - 0.694 = \mathbf{0.246}$

Information Gain: Humidity

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

Information Gain: Humidity

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

Humidity = High:

$p = 3/7$ $n = 4/7$ $\mathbf{H_h = 0.985}$

Information Gain: Humidity

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

Humidity = High:

$p = 3/7$ $n = 4/7$ $\mathbf{H_h = 0.985}$

Humidity = Normal:

$p = 6/7$ $n = 1/7$ $\mathbf{H_o = 0.592}$

Expected entropy:

(7/14)×0.985 + (7/14)×0.592= **0.7885**

Information Gain: Humidity

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

Humidity = High:

$p = 3/7 \quad n = 4/7$ $\mathbf{H_h = 0.985}$

Humidity = Normal:

$p = 6/7 \quad n = 1/7$ $\mathbf{H_o = 0.592}$

Expected entropy:

$(7/14)\times0.985 + (7/14)\times0.592 =$ **0.7885**

Information gain:

$0.940 - 0.7885 =$ **0.1515**

Which feature to split on?

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

Information gain:

Outlook: 0.246
Humidity: 0.151
Wind: 0.048
Temperature: 0.029

Which feature to split on?

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

Information gain:

Outlook: 0.246
Humidity: 0.151
Wind: 0.048
Temperature: 0.029

→ Split on Outlook

An Illustrative Example

Outlook

^

Gain(S,Humidity)=0.151
Gain(S,Wind) = 0.048
Gain(S,Temperature) = 0.029
Gain(S,Outlook) = 0.246

An Illustrative Example

Outlook

Sunny	Overcast	Rain
1,2,8,9,11	3,7,12,13	4,5,6,10,14
2+,3-	4+,0-	3+,2-
?	Yes	?

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

An Illustrative Example

Outlook

Sunny	Overcast	Rain
1,2,8,9,11	3,7,12,13	4,5,6,10,14
2+,3-	4+,0-	3+,2-
?	Yes	?

Continue until:

- Every attribute is included in path, or,
- All examples in the leaf have same label

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

An Illustrative Example

Outlook

Gain(S_{sunny}, Humidity) = .97-(3/5) 0-(2/5) 0 = .97

Gain(S_{sunny},Temp) = .97- 0-(2/5) 1 = .57

Gain(S_{sunny}, wind) = .97-(2/5) 1 - (3/5) .92= .02

Sunny	Overcast	Rain
1,2,8,9,11	3,7,12,13	4,5,6,10,14
2+,3-	4+,0-	3+,2-
?	Yes	?

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes

An Illustrative Example

	Outlook	
Sunny	**Overcast**	**Rain**
1,2,8,9,11	**3,7,12,13**	**4,5,6,10,14**
2+,3-	**4+,0-**	**3+,2-**
?	**Yes**	**?**

An Illustrative Example

Outlook

Sunny	Overcast	Rain
1,2,8,9,11	3,7,12,13	4,5,6,10,14
2+,3-	4+,0-	3+,2-
Humidity	<u>Yes</u>	?

High	Normal
<u>No</u>	<u>Yes</u>

induceDecisionTree(S)

1. Does S uniquely define a class?

    if all s $\in$ S have the same label y: **return** S;

2. Find the feature with the most information gain:

    i = argmax $_i$ *Gain*(S, X_i)

3. Add children to S:

    for *k* **in** Values(X_i):

        $S_k = \{s \in S \mid x_i = k\}$

        addChild(S, S_k)

        induceDecisionTree(S_k)

    return S;

An Illustrative Example

Outlook

Sunny | Overcast | Rain

1,2,8,9,11 | 3,7,12,13 | 4,5,6,10,14

2+,3- | 4+,0- | 3+,2-

Humidity | Yes | Wind

High | Normal | Strong | Weak

No | Yes | No | Yes

Hypothesis Space in Decision Tree Induction

- Search over decision trees, which can represent all possible discrete functions (has pros and cons)

- Goal: to find the best decision tree

- Finding a minimal decision tree consistent with a set of data is NP-hard.

- ID3 performs a greedy heuristic search: hill climbing without backtracking

- Makes statistically based decisions using all data

History of Decision Tree Research

- Full search decision tree methods to model human concept learning: Hunt et al 60s, psychology
- Quinlan developed the ID3 algorithm, with the information gain heuristic to learn expert systems from examples (late 70s)
- Breiman, Freidman and colleagues in statistics developed CART (Classification And Regression Trees)
- A variety of improvements in the 80s: coping with noise, continuous attributes, missing data, non-axis parallel, etc.
- Quinlan's updated algorithms, C4.5 (1993) and C5 are more commonly used
- Boosting (or Bagging) over DTs is a very good general purpose algorithm

Summary: Learning Decision Trees

1. **Representation**: What are decision trees?
   – A hierarchical data structure that represents data
2. **Algorithm**: Learning decision trees

   The ID3 algorithm: A greedy heuristic

   - If all the examples have the same label, create a leaf with that label
   - Otherwise, find the “most informative” attribute and split the data for different values of that attributes
   - Recurse on the splits