Learning by Asking Questions: Decision Trees

Piyush Rai

Machine Learning (CS771A)

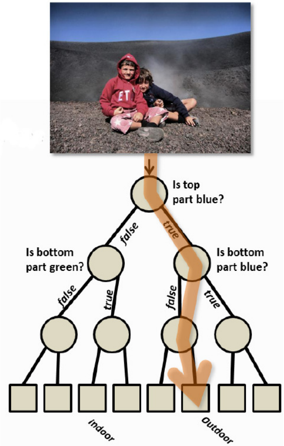
Aug 5, 2016

A Classification Problem

Indoor or Outdoor ?

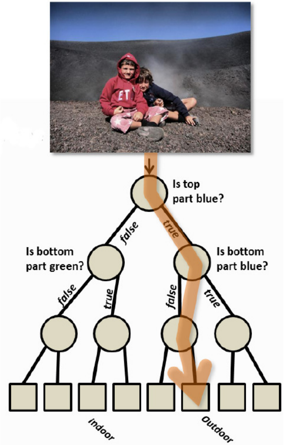
Pic credit: "Decision Forests: A Unified Framework" by Criminisi *et al*

Predicting by Asking Questions



Pic credit: "Decision Forests: A Unified Framework" by Criminisi *et al*

Predicting by Asking Questions

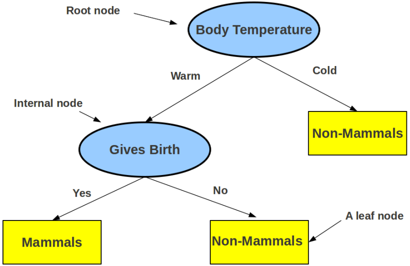


How can we learn this tree using labeled training data?

Pic credit: "Decision Forests: A Unified Framework" by Criminisi *et al*

Decision Tree

- Defined by a hierarchy of rules (in form of a tree)

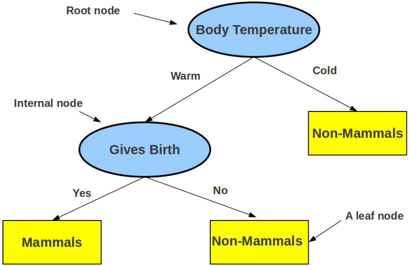


- Rules form the **internal nodes** of the tree (topmost internal node = **root**)
- Each internal node tests the value of some feature and "splits" data across the outgoing branches

[1] Breiman, Leo; Friedman, J. H.; Olshen, R. A.; Stone, C. J. (1984). Classification and regression trees

Decision Tree

- Defined by a hierarchy of rules (in form of a tree)

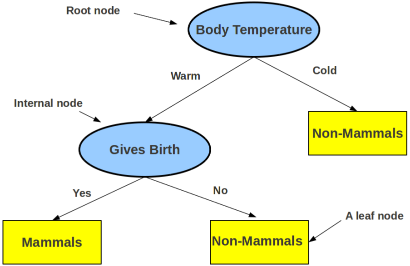


- Rules form the **internal nodes** of the tree (topmost internal node = **root**)
- Each internal node tests the value of some feature and "splits" data across the outgoing branches
- Note: The tree need not be a binary tree

[1] Breiman, Leo; Friedman, J. H.; Olshen, R. A.; Stone, C. J. (1984). Classification and regression trees

Decision Tree

- Defined by a hierarchy of rules (in form of a tree)

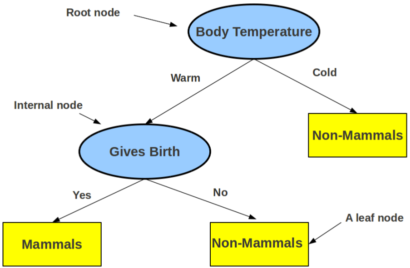


- Rules form the **internal nodes** of the tree (topmost internal node = **root**)
- Each internal node tests the value of some feature and "splits" data across the outgoing branches
- Note: The tree need not be a binary tree
- (Labeled) Training data is used to construct the Decision Tree[1] (DT)

[1] Breiman, Leo; Friedman, J. H.; Olshen, R. A.; Stone, C. J. (1984). Classification and regression trees

Decision Tree

- Defined by a hierarchy of rules (in form of a tree)

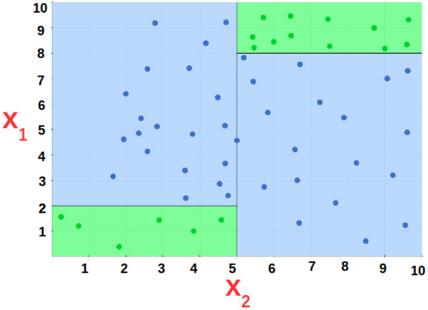


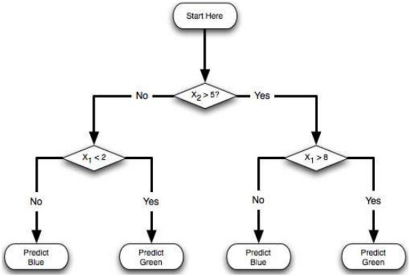
- Rules form the **internal nodes** of the tree (topmost internal node = **root**)
- Each internal node tests the value of some feature and "splits" data across the outgoing branches
- Note: The tree need not be a binary tree
- (Labeled) Training data is used to construct the Decision Tree[1] (DT)
- The DT can then be used to predict label $\mathbf{y}$ of a test example $\boldsymbol{x}$

[1] Breiman, Leo; Friedman, J. H.; Olshen, R. A.; Stone, C. J. (1984). Classification and regression trees

Decision Tree: An Example

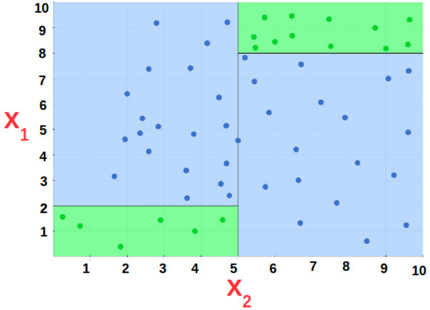
- Identifying the region - blue or green - a point lies in (binary classification)
  - Each point has 2 features: its co-ordinates $\{x_1, x_2\}$ on the 2D plane
  - Left: Training data, Right: A DT constructed using this data

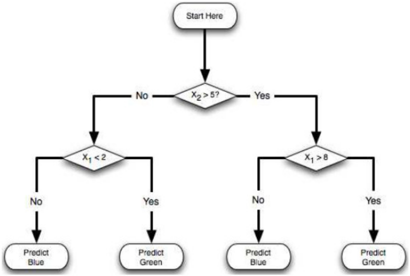




Decision Tree: An Example

- Identifying the region - blue or green - a point lies in (binary classification)
    - Each point has 2 features: its co-ordinates $\{x_1, x_2\}$ on the 2D plane
    - Left: Training data, Right: A DT constructed using this data





- The DT can be used to predict the region (blue/green) of a new test point
    - By testing the features of the test point
    - In the order defined by the DT (first x_2 and then x_1)

Decision Tree: Another Example

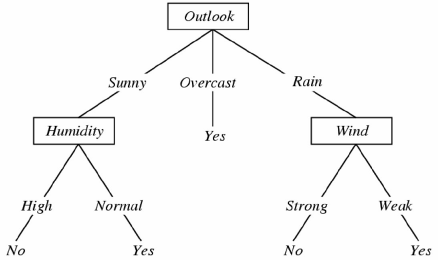
- Deciding whether to play or not to play Tennis on a Saturday
  - A binary classification problem (play vs no-play)
  - Each input (a Saturday) has 4 features: Outlook, Temp., Humidity, Wind

Pic credit: Tom Mitchell

Decision Tree: Another Example

- Deciding whether to play or not to play Tennis on a Saturday
  - A binary classification problem (play vs no-play)
  - Each input (a Saturday) has 4 features: Outlook, Temp., Humidity, Wind
  - Left: Training data, Right: A decision tree constructed using this data

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

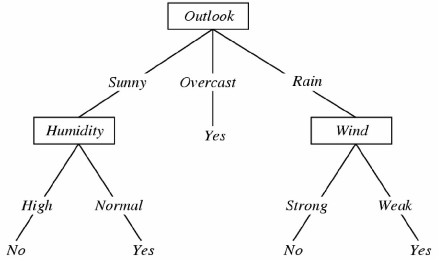


Pic credit: Tom Mitchell

Decision Tree: Another Example

- Deciding whether to play or not to play Tennis on a Saturday
  - A binary classification problem (play vs no-play)
  - Each input (a Saturday) has 4 features: Outlook, Temp., Humidity, Wind
  - Left: Training data, Right: A decision tree constructed using this data

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



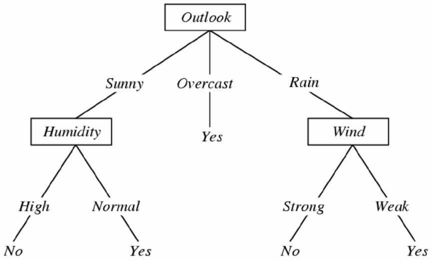
- The DT can be used to predict play vs no-play for a *new* Saturday
  - By testing the features of that Saturday
  - In the order defined by the DT

Pic credit: Tom Mitchell

Decision Tree Construction

- Now let's look at the playing Tennis example

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

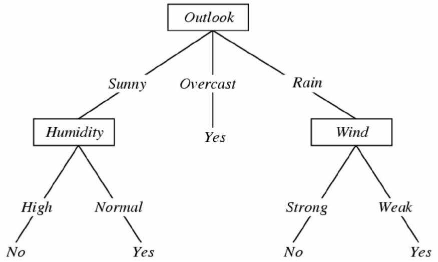


- Question: Why does it make more sense to test the feature "outlook" first?

Decision Tree Construction

- Now let's look at the playing Tennis example

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

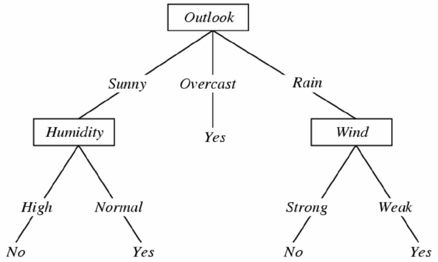


- Question: Why does it make more sense to test the feature "outlook" first?
- Answer: Of all the 4 features, it's most informative

Decision Tree Construction

- Now let's look at the playing Tennis example

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

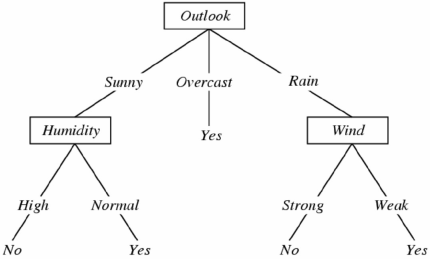


- Question: Why does it make more sense to test the feature "outlook" first?
- Answer: Of all the 4 features, it's most informative
- We will see shortly how to quantity the informativeness

Decision Tree Construction

- Now let's look at the playing Tennis example

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



- Question: Why does it make more sense to test the feature "outlook" first?
- Answer: Of all the 4 features, it's most informative
- We will see shortly how to quantity the informativeness
- **Analogy:** Playing the game 20 Questions (the most useful questions first)

Entropy

- Entropy is a measure of randomness/uncertainty of a set

Entropy

- Entropy is a measure of randomness/uncertainty of a set
- Assume our data is a set S of examples with C many classes

Entropy

- Entropy is a measure of randomness/uncertainty of a set
- Assume our data is a set S of examples with C many classes
- p_c is the probability that a random element of S belongs to class c
    - .. basically, the fraction of elements of S belonging to class c

Entropy

- Entropy is a measure of randomness/uncertainty of a set
- Assume our data is a set S of examples with C many classes
- p_c is the probability that a random element of S belongs to class c
  - .. basically, the fraction of elements of S belonging to class c
- Probability vector $p = [p_1, p_2, \ldots, p_C]$ is the class distribution of the set S

Entropy

- Entropy is a measure of randomness/uncertainty of a set
- Assume our data is a set S of examples with C many classes
- p_c is the probability that a random element of S belongs to class c
  - .. basically, the fraction of elements of S belonging to class c
- Probability vector $p = [p_1, p_2, \ldots, p_C]$ is the class distribution of the set S
- Entropy of the set S

$$H(S) = -\sum_{c \in C} p_c \log_2 p_c$$

Entropy

- Entropy is a measure of randomness/uncertainty of a set
- Assume our data is a set S of examples with C many classes
- p_c is the probability that a random element of S belongs to class c
  - .. basically, the fraction of elements of S belonging to class c
- Probability vector $p = [p_1, p_2, \ldots, p_C]$ is the class distribution of the set S
- Entropy of the set S

$$H(S) = -\sum_{c \in C} p_c \log_2 p_c$$

- If a set S of examples (or any subset of it) has..
  - Some dominant classes $\Longrightarrow$ small entropy of the class distribution

Entropy

- Entropy is a measure of randomness/uncertainty of a set
- Assume our data is a set S of examples with C many classes
- p_c is the probability that a random element of S belongs to class c
  - .. basically, the fraction of elements of S belonging to class c
- Probability vector $p = [p_1, p_2, \ldots, p_C]$ is the class distribution of the set S
- Entropy of the set S

$$H(S) = -\sum_{c \in C} p_c \log_2 p_c$$

- If a set S of examples (or any subset of it) has..
  - Some dominant classes $\Longrightarrow$ small entropy of the class distribution
  - Equiprobable classes $\Longrightarrow$ high entropy of the class distribution

Entropy

- Entropy is a measure of randomness/uncertainty of a set
- Assume our data is a set S of examples with C many classes
- p_c is the probability that a random element of S belongs to class c
  - .. basically, the fraction of elements of S belonging to class c
- Probability vector $p = [p_1, p_2, \ldots, p_C]$ is the class distribution of the set S
- Entropy of the set S

$$H(S) = -\sum_{c \in C} p_c \log_2 p_c$$

- If a set S of examples (or any subset of it) has..
  - Some dominant classes $\Longrightarrow$ small entropy of the class distribution
  - Equiprobable classes $\Longrightarrow$ high entropy of the class distribution
- We can assess informativeness of each feature by looking at how much it reduces the entropy of the class distribution

Information Gain

- Let's assume each element of S has a set of features
- Information Gain (IG) on knowing the value of some feature 'F'

$$IG(S,F) = H(S) - \sum_{f \in F} \frac{|S_f|}{|S|} H(S_f)$$

- S_f denotes the subset of elements of S for which feature F has value f

Information Gain

- Let's assume each element of S has a set of features
- Information Gain (IG) on knowing the value of some feature 'F'

$$IG(S, F) = H(S) - \sum_{f \in F} \frac{|S_f|}{|S|} H(S_f)$$

- S_f denotes the subset of elements of S for which feature F has value f
- $IG(S, F)$ = entropy of S minus the weighted sum of entropy of its children

Information Gain

- Let's assume each element of S has a set of features
- Information Gain (IG) on knowing the value of some feature 'F'

$$IG(S,F) = H(S) - \sum_{f \in F} \frac{|S_f|}{|S|} H(S_f)$$

- S_f denotes the subset of elements of S for which feature F has value f
- $IG(S,F)$ = entropy of S minus the weighted sum of entropy of its children
- $IG(S,F)$: Increase in our certainty about S once we know the value of F

Information Gain

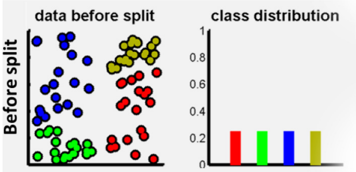
- Let's assume each element of S has a set of features
- Information Gain (IG) on knowing the value of some feature 'F'

$$IG(S, F) = H(S) - \sum_{f \in F} \frac{|S_f|}{|S|} H(S_f)$$

- S_f denotes the subset of elements of S for which feature F has value f
- $IG(S, F)$ = entropy of S minus the weighted sum of entropy of its children
- $IG(S, F)$: Increase in our certainty about S once we know the value of F
- $IG(S, F)$ denotes the no. of bits saved while encoding S once we know the value of the feature F

Entropy and Information Gain: Pictorially

Assume we have a 4-class problem. Each point has 2 features



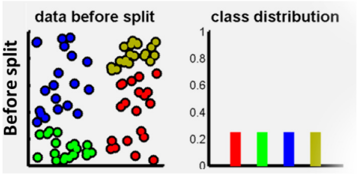
(a)

Pic credit: "Decision Forests: A Unified Framework" by Criminisi *et al*

Entropy and Information Gain: Pictorially

Assume we have a 4-class problem. Each point has 2 features

Which feature should we test (i.e., split on) first?



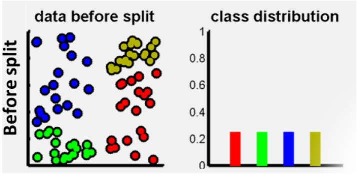
(a)

Pic credit: "Decision Forests: A Unified Framework" by Criminisi *et al*

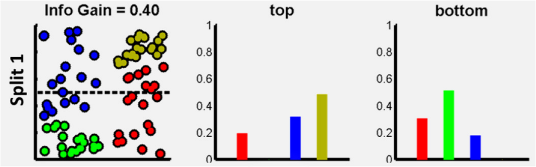
Entropy and Information Gain: Pictorially

Assume we have a 4-class problem. Each point has 2 features

Which feature should we test (i.e., split on) first?



(a)



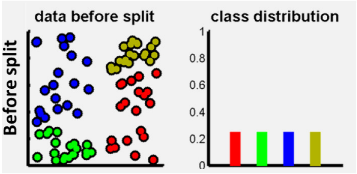
(b)

Pic credit: "Decision Forests: A Unified Framework" by Criminisi *et al*

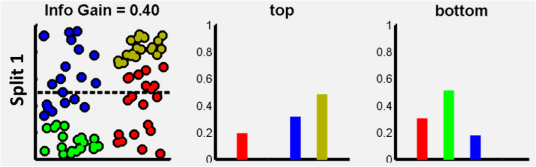
Entropy and Information Gain: Pictorially

Assume we have a 4-class problem. Each point has 2 features

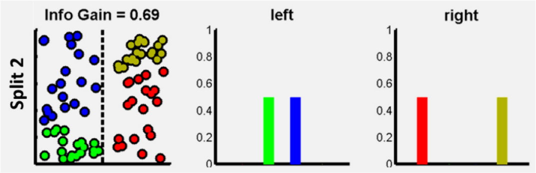
Which feature should we test (i.e., split on) first?



(a)



(b)



(c)

Pic credit: "Decision Forests: A Unified Framework" by Criminisi *et al*

Computing Information Gain

- Coming back to playing tennis..
- Let's begin with the root node of the DT and compute IG of each feature
- Consider feature "wind" $\in$ {weak,strong} and its IG w.r.t. the root node

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

Computing Information Gain

- Coming back to playing tennis..
- Let's begin with the root node of the DT and compute IG of each feature
- Consider feature "wind" $\in$ {weak,strong} and its IG w.r.t. the root node

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

- Root node: $S = [9+, 5-]$ (all training data: 9 play, 5 no-play)
- Entropy: $H(S) = -(9/14)\log_2(9/14) - (5/14)\log_2(5/14) = 0.94$

Computing Information Gain

- Coming back to playing tennis..
- Let's begin with the root node of the DT and compute IG of each feature
- Consider feature "wind" $\in$ {weak,strong} and its IG w.r.t. the root node

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

- Root node: $S = [9+, 5-]$ (all training data: 9 play, 5 no-play)
- Entropy: $H(S) = -(9/14)\log_2(9/14) - (5/14)\log_2(5/14) = 0.94$
- $S_{weak} = [6+, 2-] \implies H(S_{weak}) = 0.811$

Computing Information Gain

- Coming back to playing tennis..
- Let's begin with the root node of the DT and compute *IG* of each feature
- Consider feature "wind" $\in$ {weak,strong} and its *IG* w.r.t. the root node

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

- Root node: $S = [9+, 5-]$ (all training data: 9 play, 5 no-play)
- Entropy: $H(S) = -(9/14)\log_2(9/14) - (5/14)\log_2(5/14) = 0.94$
- $S_{weak} = [6+, 2-] \implies H(S_{weak}) = 0.811$
- $S_{strong} = [3+, 3-] \implies H(S_{strong}) = 1$

Computing Information Gain

- Coming back to playing tennis..
- Let's begin with the root node of the DT and compute *IG* of each feature
- Consider feature "wind" $\in$ {weak,strong} and its *IG* w.r.t. the root node

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

- Root node: $S = [9+, 5-]$ (all training data: 9 play, 5 no-play)
- Entropy: $H(S) = -(9/14)\log_2(9/14) - (5/14)\log_2(5/14) = 0.94$
- $S_{weak} = [6+, 2-] \implies H(S_{weak}) = 0.811$
- $S_{strong} = [3+, 3-] \implies H(S_{strong}) = 1$

$$\begin{aligned} IG(S, \text{wind}) &= H(S) - \frac{|S_{weak}|}{|S|} H(S_{weak}) - \frac{|S_{strong}|}{|S|} H(S_{strong}) \\ &= 0.94 - 8/14 * 0.811 - 6/14 * 1 \\ &= 0.048 \end{aligned}$$

Choosing the most informative feature

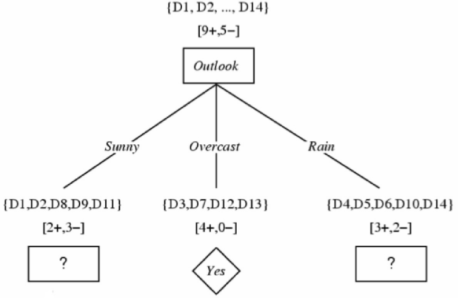
- At the root node, the information gains are:
  - $IG(S, \text{wind}) = 0.048$ (we already saw)
  - $IG(S, \text{outlook}) = 0.246$
  - $IG(S, \text{humidity}) = 0.151$
  - $IG(S, \text{temperature}) = 0.029$

Choosing the most informative feature

- At the root node, the information gains are:
  - $IG(S, \text{wind}) = 0.048$ (we already saw)
  - $IG(S, \text{outlook}) = 0.246$
  - $IG(S, \text{humidity}) = 0.151$
  - $IG(S, \text{temperature}) = 0.029$
- "outlook" has the maximum $IG \implies$ chosen as the root node

Choosing the most informative feature

- At the root node, the information gains are:
  - $IG(S, \text{wind}) = 0.048$ (we already saw)
  - $IG(S, \text{outlook}) = 0.246$
  - $IG(S, \text{humidity}) = 0.151$
  - $IG(S, \text{temperature}) = 0.029$
- "outlook" has the maximum $IG \implies$ chosen as the root node



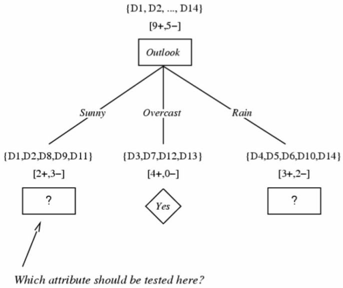
Growing The Tree

- How to decide which feature to test next ?

Growing The Tree

- How to decide which feature to test next ?
- **Rule:** Iterate - for each child node, select the feature with the highest IG

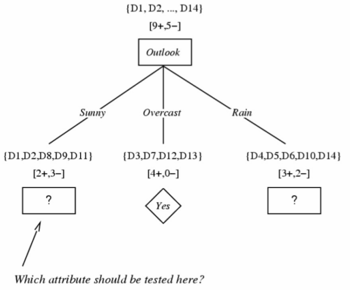
day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



Growing The Tree

- How to decide which feature to test next ?
- **Rule:** Iterate - for each child node, select the feature with the highest IG

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

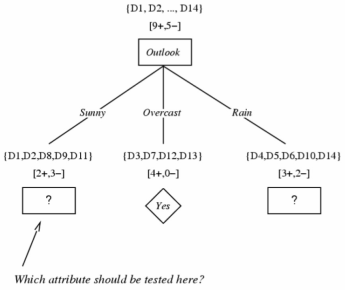


- For level-2, left node: $S = [2+, 3-]$ (days 1,2,8,9,11)
- Compute the Information Gain for each feature (except **outlook**)

Growing The Tree

- How to decide which feature to test next ?
- **Rule:** Iterate - for each child node, select the feature with the highest IG

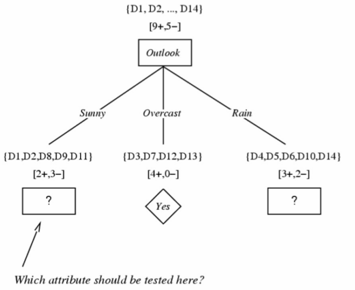
day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



- For level-2, left node: $S = [2+, 3-]$ (days 1,2,8,9,11)
- Compute the Information Gain for each feature (except **outlook**)
- The feature with the highest Information Gain should be chosen for this node

Growing The Tree

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

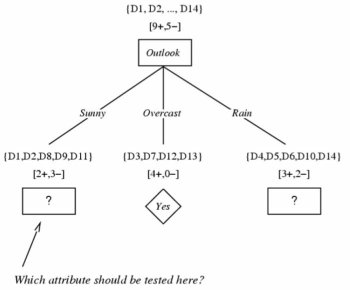


- For this node ($S = [2+, 3-]$), the IG for the feature **temperature**:

$$IG(S, \text{temperature}) \quad = \quad H(S) - \sum_{v \in \{hot, mild, cool\}} \frac{|S_v|}{|S|} H(S_v)$$

Growing The Tree

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



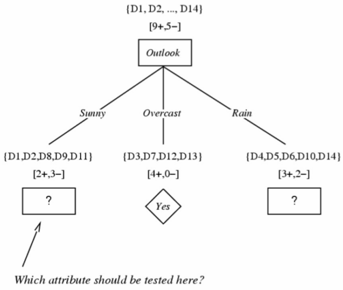
- For this node ($S = [2+, 3-]$), the IG for the feature **temperature**:

$$IG(S, \text{temperature}) \quad = \quad H(S) - \sum_{v \in \{hot, mild, cool\}} \frac{|S_v|}{|S|} H(S_v)$$

- $S = [2+, 3-] \implies H(S) = -(2/5) * \log_2(2/5) - (3/5) * \log_2(3/5) = 0.971$

Growing The Tree

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



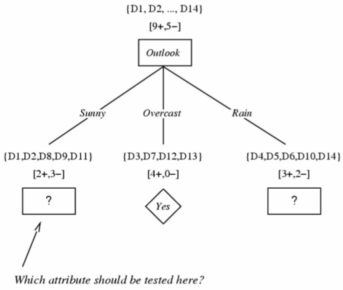
- For this node ($S = [2+, 3-]$), the IG for the feature **temperature**:

$$IG(S, \text{temperature}) \quad = \quad H(S) - \sum_{v \in \{hot, mild, cool\}} \frac{|S_v|}{|S|} H(S_v)$$

- $S = [2+, 3-] \implies H(S) = -(2/5) * \log_2(2/5) - (3/5) * \log_2(3/5) = 0.971$
- $S_{hot} = [0+, 2-] \implies H(S_{hot}) = -0 * \log_2(0) - (2/2) * \log_2(2/2) = 0$

Growing The Tree

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



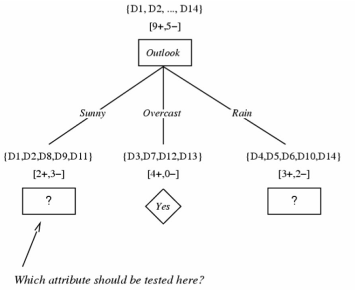
- For this node ($S = [2+, 3-]$), the IG for the feature **temperature**:

$$IG(S, \text{temperature}) \quad = \quad H(S) - \sum_{v \in \{hot, mild, cool\}} \frac{|S_v|}{|S|} H(S_v)$$

- $S = [2+, 3-] \implies H(S) = -(2/5) * \log_2(2/5) - (3/5) * \log_2(3/5) = 0.971$
- $S_{hot} = [0+, 2-] \implies H(S_{hot}) = -0 * \log_2(0) - (2/2) * \log_2(2/2) = 0$
- $S_{mild} = [1+, 1-] \implies H(S_{mild}) = -(1/2) * \log_2(1/2) - (1/2) * \log_2(1/2) = 1$

Growing The Tree

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



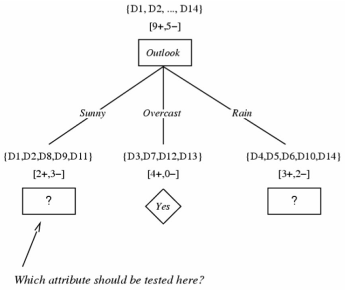
- For this node ($S = [2+, 3-]$), the IG for the feature **temperature**:

$$IG(S, \text{temperature}) \quad = \quad H(S) - \sum_{v \in \{hot, mild, cool\}} \frac{|S_v|}{|S|} H(S_v)$$

- $S = [2+, 3-] \implies H(S) = -(2/5) * \log_2(2/5) - (3/5) * \log_2(3/5) = 0.971$
- $S_{hot} = [0+, 2-] \implies H(S_{hot}) = -0 * \log_2(0) - (2/2) * \log_2(2/2) = 0$
- $S_{mild} = [1+, 1-] \implies H(S_{mild}) = -(1/2) * \log_2(1/2) - (1/2) * \log_2(1/2) = 1$
- $S_{cool} = [1+, 0-] \implies H(S_{cool}) = -(1/1) * \log_2(1/1) - (0/1) * \log_2(0/1) = 0$

Growing The Tree

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



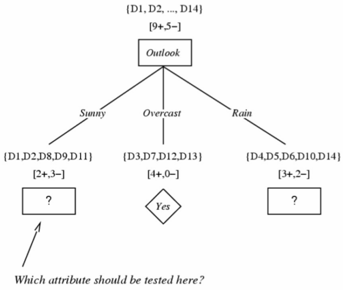
- For this node ($S = [2+, 3-]$), the IG for the feature **temperature**:

$$IG(S, \text{temperature}) \quad = \quad H(S) - \sum_{v \in \{hot, mild, cool\}} \frac{|S_v|}{|S|} H(S_v)$$

- $S = [2+, 3-] \implies H(S) = -(2/5) * \log_2(2/5) - (3/5) * \log_2(3/5) = 0.971$
- $S_{hot} = [0+, 2-] \implies H(S_{hot}) = -0 * \log_2(0) - (2/2) * \log_2(2/2) = 0$
- $S_{mild} = [1+, 1-] \implies H(S_{mild}) = -(1/2) * \log_2(1/2) - (1/2) * \log_2(1/2) = 1$
- $S_{cool} = [1+, 0-] \implies H(S_{cool}) = -(1/1) * \log_2(1/1) - (0/1) * \log_2(0/1) = 0$
- $IG(S, \text{temperature}) = 0.971 - 2/5 * 0 - 2/5 * 1 - 1/5 * 0 = 0.570$

Growing The Tree

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



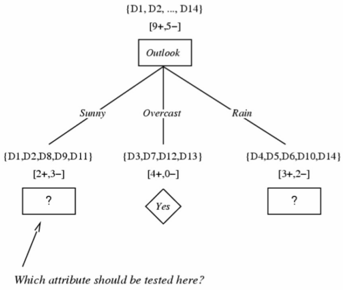
- For this node ($S = [2+, 3-]$), the IG for the feature **temperature**:

$$IG(S, \text{temperature}) = H(S) - \sum_{v \in \{hot, mild, cool\}} \frac{|S_v|}{|S|} H(S_v)$$

- $S = [2+, 3-] \implies H(S) = -(2/5) * \log_2(2/5) - (3/5) * \log_2(3/5) = 0.971$
- $S_{hot} = [0+, 2-] \implies H(S_{hot}) = -0 * \log_2(0) - (2/2) * \log_2(2/2) = 0$
- $S_{mild} = [1+, 1-] \implies H(S_{mild}) = -(1/2) * \log_2(1/2) - (1/2) * \log_2(1/2) = 1$
- $S_{cool} = [1+, 0-] \implies H(S_{cool}) = -(1/1) * \log_2(1/1) - (0/1) * \log_2(0/1) = 0$
- $IG(S, \text{temperature}) = 0.971 - 2/5 * 0 - 2/5 * 1 - 1/5 * 0 = 0.570$
- Likewise we can compute: $IG(S, \text{humidity}) = 0.970$, $IG(S, \text{wind}) = 0.019$

Growing The Tree

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



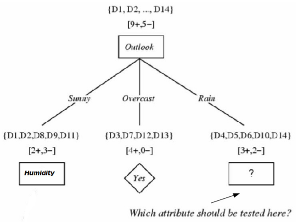
- For this node ($S = [2+, 3-]$), the IG for the feature **temperature**:

$$IG(S, \text{temperature}) \quad = \quad H(S) - \sum_{v \in \{hot, mild, cool\}} \frac{|S_v|}{|S|} H(S_v)$$

- $S = [2+, 3-] \Longrightarrow H(S) = -(2/5) * \log_2(2/5) - (3/5) * \log_2(3/5) = 0.971$
- $S_{hot} = [0+, 2-] \Longrightarrow H(S_{hot}) = -0 * \log_2(0) - (2/2) * \log_2(2/2) = 0$
- $S_{mild} = [1+, 1-] \Longrightarrow H(S_{mild}) = -(1/2) * \log_2(1/2) - (1/2) * \log_2(1/2) = 1$
- $S_{cool} = [1+, 0-] \Longrightarrow H(S_{cool}) = -(1/1) * \log_2(1/1) - (0/1) * \log_2(0/1) = 0$
- $IG(S, \text{temperature}) = 0.971 - 2/5 * 0 - 2/5 * 1 - 1/5 * 0 = 0.570$
- Likewise we can compute: $IG(S, \text{humidity}) = 0.970$, $IG(S, \text{wind}) = 0.019$
- Therefore, **we choose "humidity"** (with highest $IG = 0.970$) for the level-2 left node

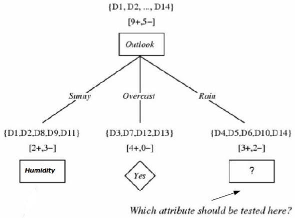
Growing The Tree

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



Growing The Tree

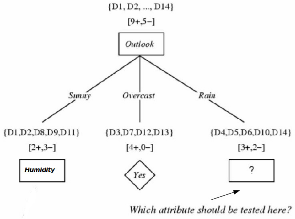
day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



- Level-2, middle node: no need to grow (already a leaf)

Growing The Tree

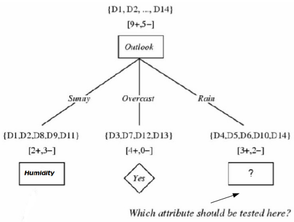
day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



- Level-2, middle node: no need to grow (already a leaf)
- Level-2, right node: repeat the same exercise!

Growing The Tree

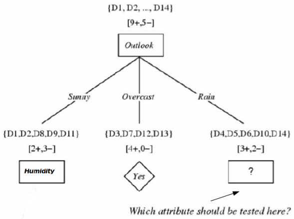
day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



- Level-2, middle node: no need to grow (already a leaf)
- Level-2, right node: repeat the same exercise!
    - Compute *IG* for each feature (except outlook)

Growing The Tree

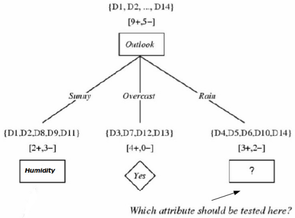
day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



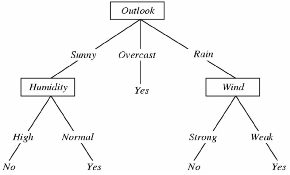
- Level-2, middle node: no need to grow (already a leaf)
- Level-2, right node: repeat the same exercise!
  - Compute *IG* for each feature (except outlook)
  - **Exercise:** Verify that wind has the highest *IG* for this node

Growing The Tree

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

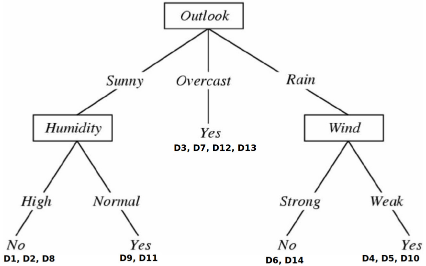


- Level-2, middle node: no need to grow (already a leaf)
- Level-2, right node: repeat the same exercise!
  - Compute *IG* for each feature (except outlook)
  - **Exercise:** Verify that wind has the highest *IG* for this node
- Level-2 expansion gives us the following tree:



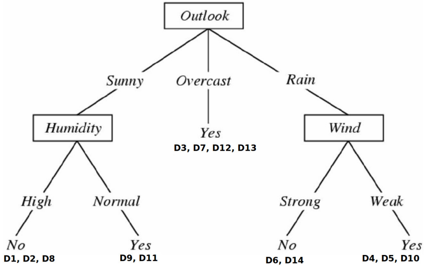
Growing The Tree: Stopping Criteria

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



Growing The Tree: Stopping Criteria

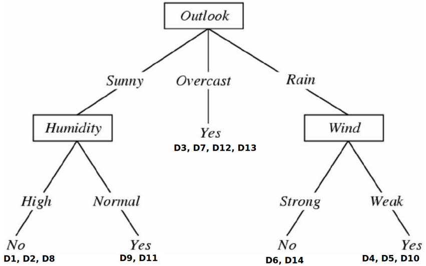
day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



- Stop expanding a node further when:

Growing The Tree: Stopping Criteria

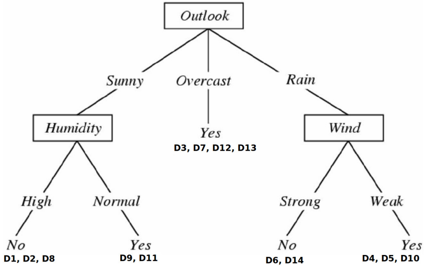
day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



- Stop expanding a node further when:
  - It consist of examples all having the same label (the node becomes "pure")

Growing The Tree: Stopping Criteria

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



- Stop expanding a node further when:
  - It consist of examples all having the same label (the node becomes "pure")
  - Or we run out of features to test!

Decision Tree Learning Algorithm

A recursive algorithm:

$\mathrm{DT}(\textit{Examples}, \textit{Labels}, \textit{Features})$:

Decision Tree Learning Algorithm

A recursive algorithm:

$\mathrm{DT}(\mathit{Examples}, \mathit{Labels}, \mathit{Features})$:

- If all examples are positive, return a single node tree *Root* with label $= +$

Decision Tree Learning Algorithm

A recursive algorithm:

$\mathrm{DT}(\mathit{Examples}, \mathit{Labels}, \mathit{Features})$:

- If all examples are positive, return a single node tree *Root* with label $= +$
- If all examples are negative, return a single node tree *Root* with label $=$ -

Decision Tree Learning Algorithm

A recursive algorithm:

$\mathrm{DT}(\mathit{Examples}, \mathit{Labels}, \mathit{Features})$:

- If all examples are positive, return a single node tree *Root* with label $= +$
- If all examples are negative, return a single node tree *Root* with label $=$ -
- If all features exhausted, return a single node tree *Root* with majority label

Decision Tree Learning Algorithm

A recursive algorithm:

$\mathrm{DT}(\mathit{Examples}, \mathit{Labels}, \mathit{Features})$:

- If all examples are positive, return a single node tree *Root* with label $= +$
- If all examples are negative, return a single node tree *Root* with label = -
- If all features exhausted, return a single node tree *Root* with majority label
- Otherwise, let F be the feature having the highest information gain
- $\mathit{Root} \leftarrow F$

Decision Tree Learning Algorithm

A recursive algorithm:

$\mathrm{DT}(\mathit{Examples}, \mathit{Labels}, \mathit{Features})$:

- If all examples are positive, return a single node tree *Root* with label $= +$
- If all examples are negative, return a single node tree *Root* with label = -
- If all features exhausted, return a single node tree *Root* with majority label
- Otherwise, let F be the feature having the highest information gain
- $\mathit{Root} \leftarrow F$
- For each possible value f of F

Decision Tree Learning Algorithm

A recursive algorithm:

$\mathrm{DT}(\mathit{Examples}, \mathit{Labels}, \mathit{Features})$:

- If all examples are positive, return a single node tree *Root* with label $= +$
- If all examples are negative, return a single node tree *Root* with label $=$ -
- If all features exhausted, return a single node tree *Root* with majority label
- Otherwise, let F be the feature having the highest information gain
- $\mathit{Root} \leftarrow F$
- For each possible value f of F
  - Add a tree branch below *Root* corresponding to the test $F = f$

Decision Tree Learning Algorithm

A recursive algorithm:

DT(*Examples*, *Labels*, *Features*):

- If all examples are positive, return a single node tree *Root* with label = +
- If all examples are negative, return a single node tree *Root* with label = -
- If all features exhausted, return a single node tree *Root* with majority label
- Otherwise, let F be the feature having the highest information gain
- $Root \leftarrow F$
- For each possible value f of F
  - Add a tree branch below *Root* corresponding to the test $F = f$
  - Let $Examples_f$ be the set of examples with feature F having value f
  - Let $Labels_f$ be the corresponding labels

Decision Tree Learning Algorithm

A recursive algorithm:

$\mathrm{DT}(\textit{Examples}, \textit{Labels}, \textit{Features})$:

- If all examples are positive, return a single node tree *Root* with label $= +$
- If all examples are negative, return a single node tree *Root* with label = -
- If all features exhausted, return a single node tree *Root* with majority label
- Otherwise, let F be the feature having the highest information gain
- $Root \leftarrow F$
- For each possible value f of F
    - Add a tree branch below *Root* corresponding to the test $F = f$
    - Let $Examples_f$ be the set of examples with feature F having value f
    - Let $Labels_f$ be the corresponding labels
    - If $Examples_f$ is empty, add a leaf node below this branch with label $=$ most common label in *Examples*

Decision Tree Learning Algorithm

A recursive algorithm:

$\mathrm{DT}(\mathit{Examples}, \mathit{Labels}, \mathit{Features})$:

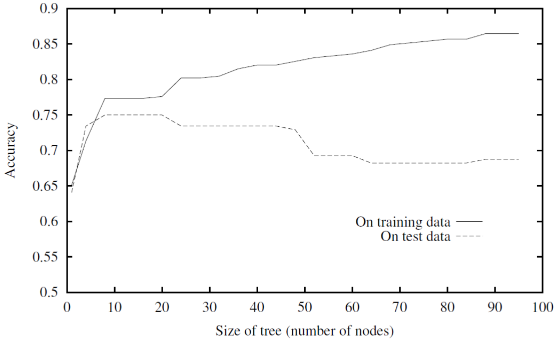
- If all examples are positive, return a single node tree *Root* with label $= +$
- If all examples are negative, return a single node tree *Root* with label $=$ -
- If all features exhausted, return a single node tree *Root* with majority label
- Otherwise, let F be the feature having the highest information gain
- $\mathit{Root} \leftarrow F$
- For each possible value f of F
    - Add a tree branch below *Root* corresponding to the test $F = f$
    - Let $\mathit{Examples}_f$ be the set of examples with feature F having value f
    - Let Labels_f be the corresponding labels
    - If $\mathit{Examples}_f$ is empty, add a leaf node below this branch with label $=$ most common label in *Examples*
    - Otherwise, add the following subtree below this branch:

$$\mathrm{DT}(\mathit{Examples}_f, \mathit{Labels}_f, \mathit{Features} - \{F\})$$

    - Note: $\mathit{Features} - \{F\}$ removes feature F from the feature set *Features*

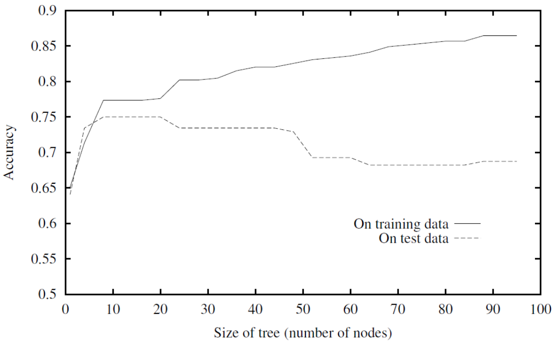
Overfitting in Decision Trees

- Overfitting Illustration



Overfitting in Decision Trees

- Overfitting Illustration



- High training accuracy doesn't necessarily imply high test accuracy

Avoiding Overfitting: Decision Tree Pruning

- Desired: a DT that is not too big in size, yet fits the training data reasonably

- Mainly two approaches

Avoiding Overfitting: Decision Tree Pruning

- Desired: a DT that is not too big in size, yet fits the training data reasonably
- Mainly two approaches
  - Prune while building the tree (**stopping early**)

Avoiding Overfitting: Decision Tree Pruning

- Desired: a DT that is not too big in size, yet fits the training data reasonably
- Mainly two approaches
  - Prune while building the tree (**stopping early**)
  - Prune after building the tree (**post-pruning**)

Avoiding Overfitting: Decision Tree Pruning

- Desired: a DT that is not too big in size, yet fits the training data reasonably
- Mainly two approaches
  - Prune while building the tree (**stopping early**)
  - Prune after building the tree (**post-pruning**)
- Criteria for judging which nodes could potentially be pruned
  - Use a validation set (separate from the training set)

Avoiding Overfitting: Decision Tree Pruning

- Desired: a DT that is not too big in size, yet fits the training data reasonably
- Mainly two approaches
  - Prune while building the tree (**stopping early**)
  - Prune after building the tree (**post-pruning**)
- Criteria for judging which nodes could potentially be pruned
  - Use a validation set (separate from the training set)
    - Prune each possible node that doesn't hurt the accuracy on the validation set

Avoiding Overfitting: Decision Tree Pruning

- Desired: a DT that is not too big in size, yet fits the training data reasonably
- Mainly two approaches
  - Prune while building the tree (**stopping early**)
  - Prune after building the tree (**post-pruning**)
- Criteria for judging which nodes could potentially be pruned
  - Use a validation set (separate from the training set)
    - Prune each possible node that doesn't hurt the accuracy on the validation set
    - Greedily remove the node that improves the validation accuracy the most

Avoiding Overfitting: Decision Tree Pruning

- Desired: a DT that is not too big in size, yet fits the training data reasonably
- Mainly two approaches
  - Prune while building the tree (**stopping early**)
  - Prune after building the tree (**post-pruning**)
- Criteria for judging which nodes could potentially be pruned
  - Use a validation set (separate from the training set)
    - Prune each possible node that doesn't hurt the accuracy on the validation set
    - Greedily remove the node that improves the validation accuracy the most
    - Stop when the validation set accuracy starts worsening

Avoiding Overfitting: Decision Tree Pruning

- Desired: a DT that is not too big in size, yet fits the training data reasonably
- Mainly two approaches
    - Prune while building the tree (**stopping early**)
    - Prune after building the tree (**post-pruning**)
- Criteria for judging which nodes could potentially be pruned
    - Use a validation set (separate from the training set)
        - Prune each possible node that doesn't hurt the accuracy on the validation set
        - Greedily remove the node that improves the validation accuracy the most
        - Stop when the validation set accuracy starts worsening
    - Minimum Description Length (MDL): more details when we cover Model Selection

Decision Tree Extensions

- Real-valued features can be dealt with using thresholding

[2] Breiman, Leo; Friedman, J. H.; Olshen, R. A.; Stone, C. J. (1984). Classification and regression trees

Decision Tree Extensions

- Real-valued features can be dealt with using thresholding
- Real-valued labels (Regression Trees[2]) by re-defining entropy or using other criteria (how similar to each other are the $\boldsymbol{y}$'s at any node)

[2] Breiman, Leo; Friedman, J. H.; Olshen, R. A.; Stone, C. J. (1984). Classification and regression trees

Decision Tree Extensions

- Real-valued features can be dealt with using thresholding
- Real-valued labels (Regression Trees[2]) by re-defining entropy or using other criteria (how similar to each other are the $\boldsymbol{y}$'s at any node)
- Other criteria for judging feature informativeness
  - Gini-index, misclassification rate

[2] Breiman, Leo; Friedman, J. H.; Olshen, R. A.; Stone, C. J. (1984). Classification and regression trees

Decision Tree Extensions

- Real-valued features can be dealt with using thresholding
- Real-valued labels (Regression Trees[2]) by re-defining entropy or using other criteria (how similar to each other are the $\boldsymbol{y}$'s at any node)
- Other criteria for judging feature informativeness
  - Gini-index, misclassification rate
- More sophisticated decision rules at the internal nodes (anything that splits the data into homogeneous groups; e.g., a machine learning classifier)

[2] Breiman, Leo; Friedman, J. H.; Olshen, R. A.; Stone, C. J. (1984). Classification and regression trees

Decision Tree Extensions

- Real-valued features can be dealt with using thresholding
- Real-valued labels (Regression Trees[2]) by re-defining entropy or using other criteria (how similar to each other are the $\boldsymbol{y}$'s at any node)
- Other criteria for judging feature informativeness
  - Gini-index, misclassification rate
- More sophisticated decision rules at the internal nodes (anything that splits the data into homogeneous groups; e.g., a machine learning classifier)
- Handling features with differing costs

[2] Breiman, Leo; Friedman, J. H.; Olshen, R. A.; Stone, C. J. (1984). Classification and regression trees

Some Aspects about Decision Trees

Some key strengths:

- Simple and each to interpret
- Do not make any assumption about distribution of data
- Easily handle different types of features (real, categorical/nominal, etc.)
- Very fast at test time (just need to check the features, starting the root node and following the DT until you reach a leaf node)
- Multiple DTs can be combined via ensemble methods (e.g., Decision Forest)
  - Each DT can be constructed using a (random) small subset of features

Some Aspects about Decision Trees

Some key strengths:

- Simple and each to interpret
- Do not make any assumption about distribution of data
- Easily handle different types of features (real, categorical/nominal, etc.)
- Very fast at test time (just need to check the features, starting the root node and following the DT until you reach a leaf node)
- Multiple DTs can be combined via ensemble methods (e.g., Decision Forest)
  - Each DT can be constructed using a (random) small subset of features

Some key weaknesses:

- Learning the optimal DT is NP-Complete. The existing algorithms are heuristics (e.g., greedy selection of features)
- Can be unstable if some labeled examples are noisy
- Can sometimes become very complex unless some pruning is applied

Next Class:
Learning as Optimization