CO3091 - Computational Intelligence and Software Engineering

Lecture 21

Image from: http://cliparts.co/cliparts/8Tx/kkA/8TxKkAEXc.jpg

# Decision Trees — Part I

Leandro L. Minku

# Overview

- What are decision trees in the context of machine learning?
- How to build decision trees?
  - General idea of recursive algorithm to split nodes.
  - Classification and regression problems.
  - Categorical and ordinal input attributes.
- Decision trees topic to be continued in the next two lectures.
  - How to split nodes?
    - Classification and regression problems.
    - Categorical, ordinal and numerical input attributes.
  - How to avoid overfitting?
  - Applications of decision trees.

# Decision Trees

- Decision-support tools that use a tree-like structure.
- In the context of machine learning, decision trees are approaches that use predictive models with tree-like structures.
  - Internal nodes: specify tests to be carried out on a single input attribute, with one branch for each possible outcome of the test.
  - Leaf nodes: indicate the value of the output attribute.

# Example of Decision Tree

Image from: http://www.lstc.org.uk/wp-content/uploads/2015/11/DSC_0446.jpg

# Previous Invitations for Playing Tennis

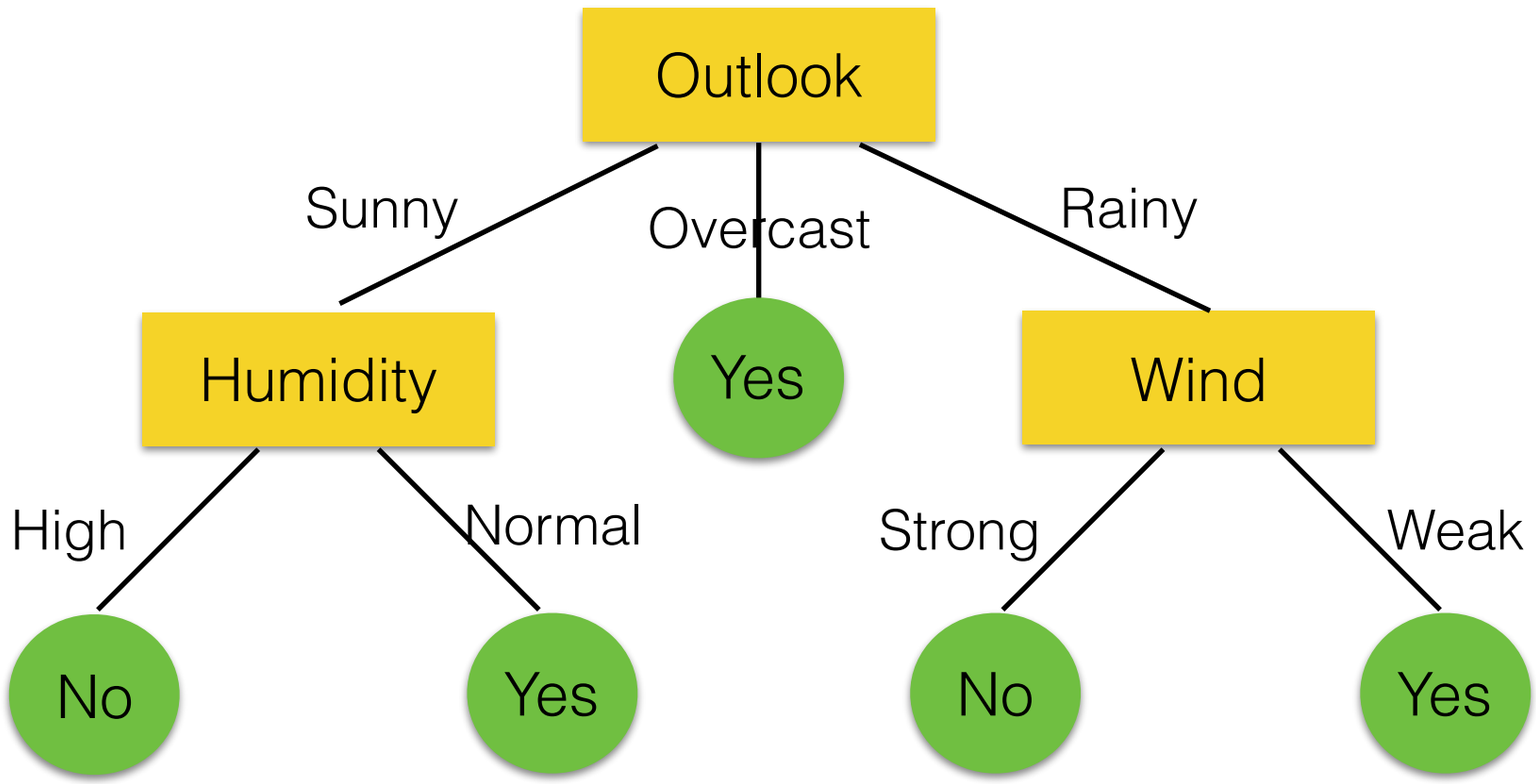
| Day | Outlook | Temperature | Humidity | Wind | Play |
|---|---|---|---|---|---|
| D1 | Sunny | Hot | High | Weak | No |
| D2 | Sunny | Hot | High | Strong | No |
| D3 | Overcast | Hot | High | Weak | Yes |
| D4 | Rain | Mild | High | Weak | Yes |
| D5 | Rain | Cool | Normal | Weak | Yes |
| D6 | Rain | Cool | Normal | Strong | No |
| D7 | Overcast | Cool | Normal | Strong | Yes |
| D8 | Sunny | Mild | High | Weak | No |
| D9 | Sunny | Cool | Normal | Weak | Yes |
| D10 | Rain | Mild | Normal | Weak | Yes |
| D11 | Sunny | Mild | Normal | Strong | Yes |
| D12 | Overcast | Mild | High | Strong | Yes |
| D13 | Overcast | Hot | Normal | Weak | Yes |
| D14 | Rain | Mild | High | Strong | No |

# WEKA Example

| Day | Outlook | Temperature | Humidity | Wind | Play |
|---|---|---|---|---|---|
| D1 | Sunny | Hot | High | Weak | No |
| D2 | Sunny | Hot | High | Strong | No |
| D3 | Overcast | Hot | High | Weak | Yes |
| D4 | Rain | Mild | High | Weak | Yes |
| D5 | Rain | Cool | Normal | Weak | Yes |
| D6 | Rain | Cool | Normal | Strong | No |
| D7 | Overcast | Cool | Normal | Strong | Yes |
| D8 | Sunny | Mild | High | Weak | No |
| D9 | Sunny | Cool | Normal | Weak | Yes |
| D10 | Rain | Mild | Normal | Weak | Yes |
| D11 | Sunny | Mild | Normal | Strong | Yes |
| D12 | Overcast | Mild | High | Strong | Yes |
| D13 | Overcast | Hot | Normal | Weak | Yes |
| D14 | Rain | Mild | High | Strong | No |

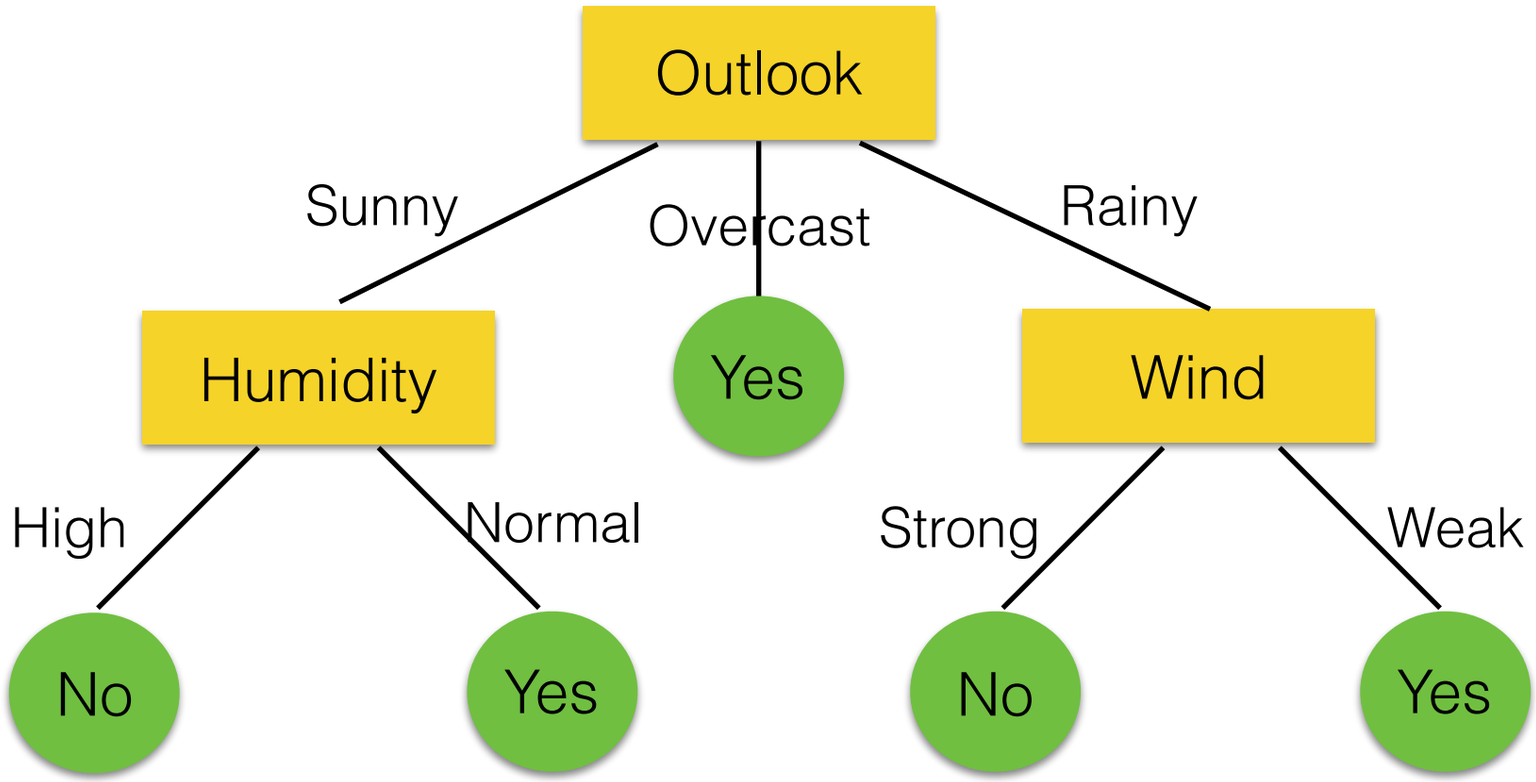
trees -> J48

# Decision Tree Learnt




Internal nodes make splits on input attributes.

Leaves indicate output values.



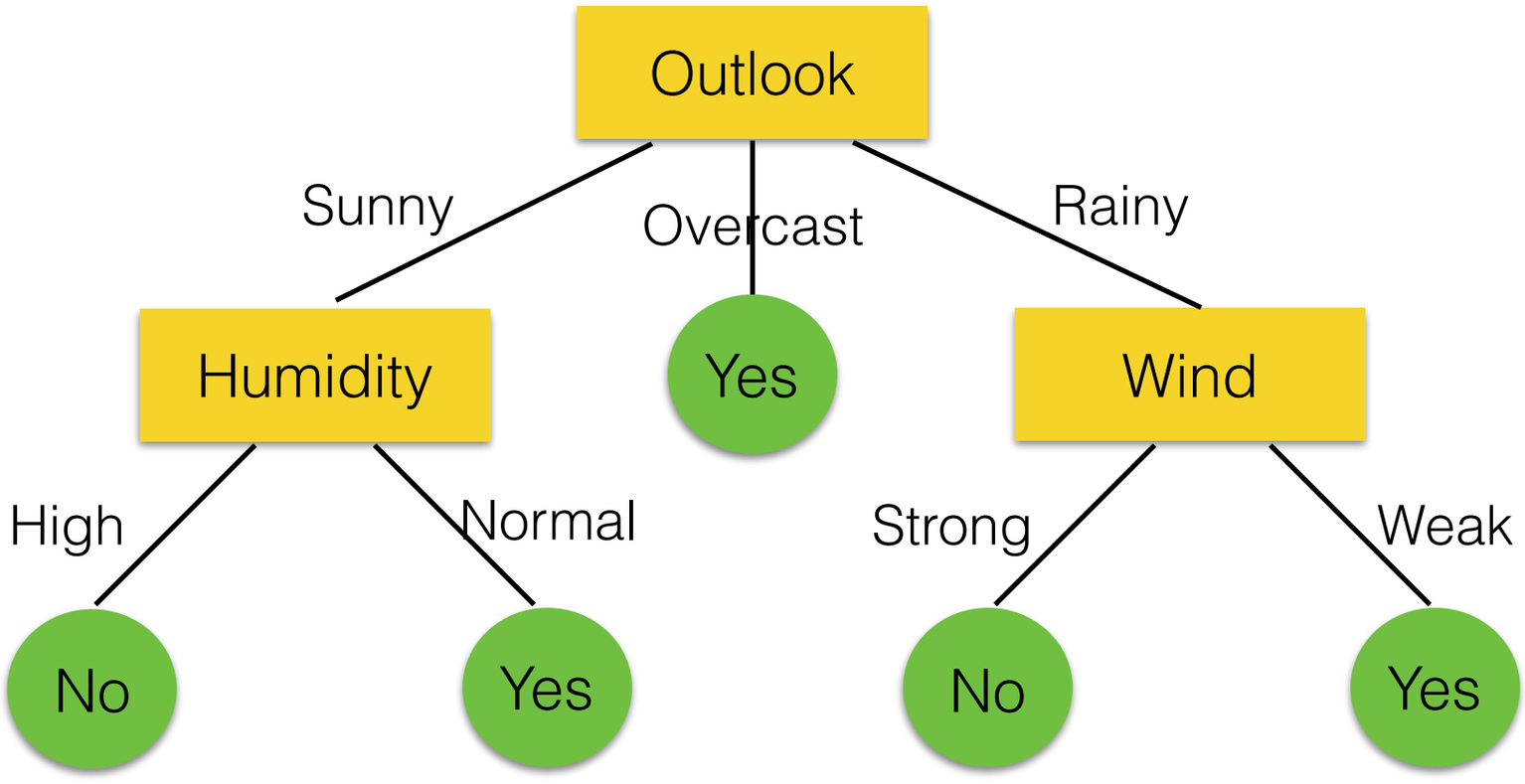

```
J48 pruned tree
------------------

Outlook = Sunny
|   Humidity = High: No (3.0)
|   Humidity = Normal: Yes (2.0)
Outlook = Overcast: Yes (4.0)
Outlook = Rain
|   Wind = Weak: Yes (3.0)
|   Wind = Strong: No (2.0)
```

# Predictions Based on Decision Trees

In order to make a prediction, walk through the tree.

# Example of Decision Tree for Playing Tennis




Irrelevant or redundant attributes don't appear on the tree.

The smallest the tree, the easier to understand it.

What would be the prediction for an instance [outlook=sunny, temperature=hot, humidity=normal,wind=strong]?

# Types of Decision Trees

- According to output attributes:
  - Classification trees.
  - Regression trees.
- Input attributes can be either numerical, categorical or ordinal.

Most common use of decision trees.

# How to Build Decision Trees Based on Training Data?

General idea:

- Create a node and split it based on the input attribute that best separates the training examples associated to that node into different classes or numerical values.
- Once a split is made, create a node for each branch and split it based on the procedure above.

Stopping criteria (incomplete):

- If all training examples associated to a node have the same output value, make this node a leaf node of that output value and stop splitting it.

# How to Build Decision Trees Based on Training Data?

General idea:

- **Create a node and split it based on the input attribute that best separates the training examples associated to that node into different classes or numerical values.**

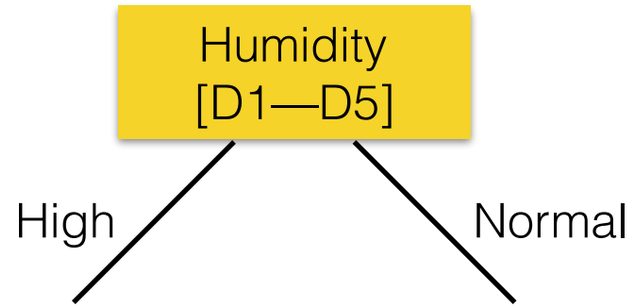
- Once a split is made, create a node for each branch and split it based on the procedure above.

Stopping criteria (incomplete):

- If all training examples associated to a node have the same output value, make this node a leaf node of that output value and stop splitting it.

# Choosing an Attribute to Split On — Basic Idea

| Day | Wind | Humidity | Play |
|---|---|---|---|
| D1 | Strong | High | No |
| D2 | Strong | High | No |
| D3 | Weak | High | No |
| D4 | Strong | Normal | Yes |
| D5 | Strong | Normal | Yes |




Create one branch for each possible value of humidity.

# How to Build Decision Trees Based on Training Data?

General idea:

- Create a node and split it based on the input attribute that best separates the training examples associated to that node into different classes or numerical values.
- **Once a split is made, create a node for each branch and split it based on the procedure above.**

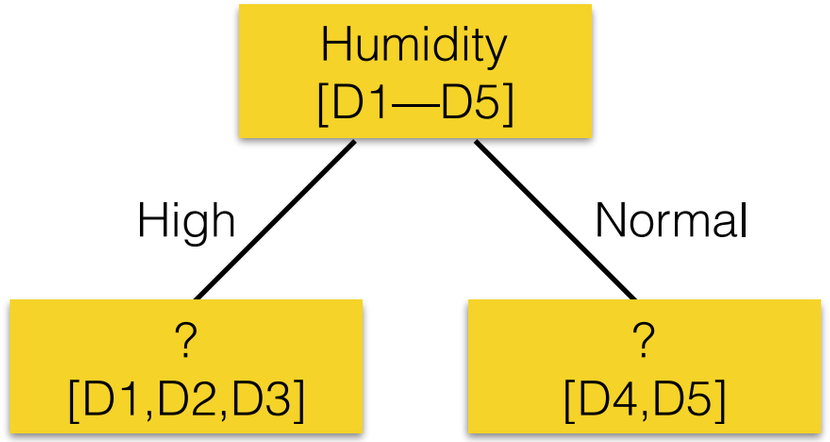
Stopping criteria (incomplete):

- **If all training examples associated to a node have the same output value, make this node a leaf node of that output value and stop splitting it.**

# Training Data Associated To Each New Node

| Day | Wind | Humidity | Play |
|---|---|---|---|
| D1 | Strong | High | No |
| D2 | Strong | High | No |
| D3 | Weak | High | No |

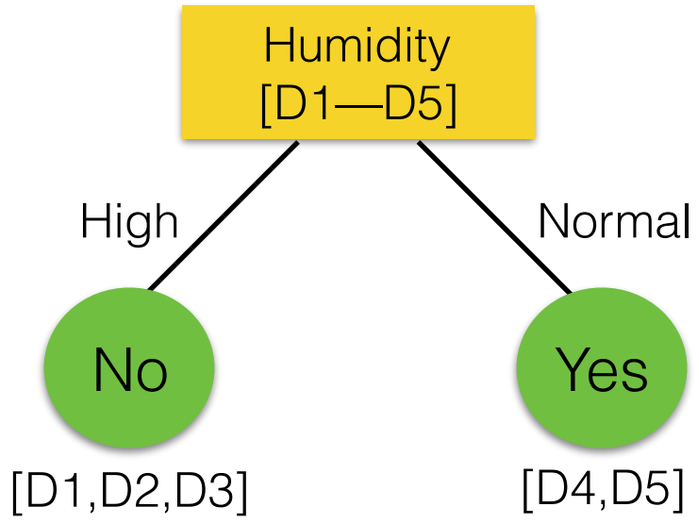
| Day | Wind | Humidity | Play |
|---|---|---|---|
| D4 | Strong | Normal | Yes |
| D5 | Strong | Normal | Yes |



# Creating Leaf Node for Classification Problems

| Day | Wind | Play |
|---|---|---|
| D1 | Strong | No |
| D2 | Strong | No |
| D3 | Weak | No |

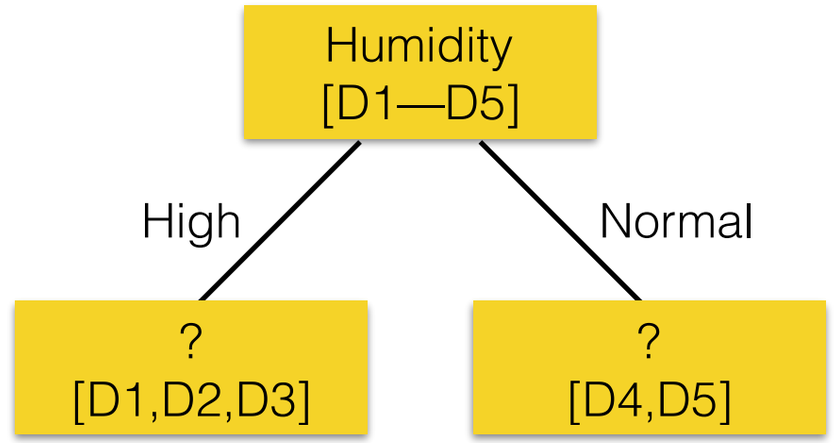
| Day | Wind | Play |
|---|---|---|
| D4 | Strong | Yes |
| D5 | Strong | Yes |



# Another Example

| Day | Wind | Play |
|---|---|---|
| D1 | Strong | No |
| D2 | Strong | No |
| D3 | Weak | Yes |

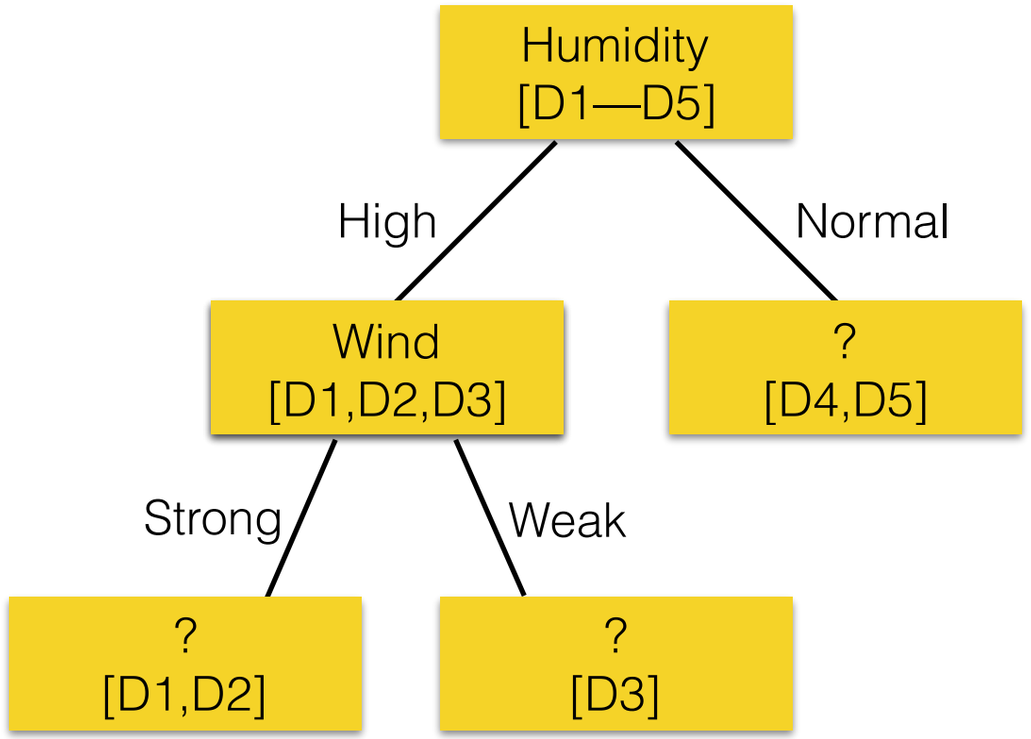
| Day | Wind | Play |
|---|---|---|
| D4 | Strong | Yes |
| D5 | Strong | Yes |



# Further Split New Node

| Day | Wind | Play |
|---|---|---|
| D1 | Strong | No |
| D2 | Strong | No |
| D3 | Weak | Yes |

| Day | Wind | Play |
|---|---|---|
| D4 | Strong | Yes |
| D5 | Strong | Yes |

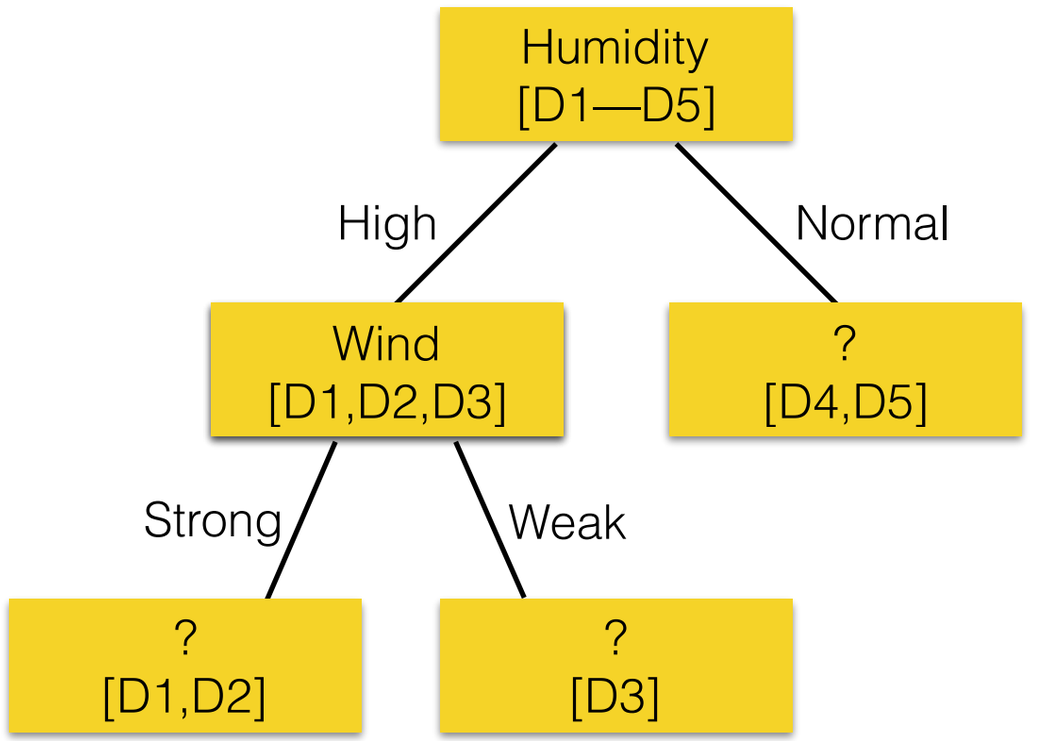


# Training Data Associated To Each New Node

| Day | Wind | Play |
|---|---|---|
| D1 | Strong | No |
| D2 | Strong | No |

| Day | Wind | Play |
|---|---|---|
| D3 | Weak | Yes |

| Day | Wind | Play |
|---|---|---|
| D4 | Strong | Yes |
| D5 | Strong | Yes |

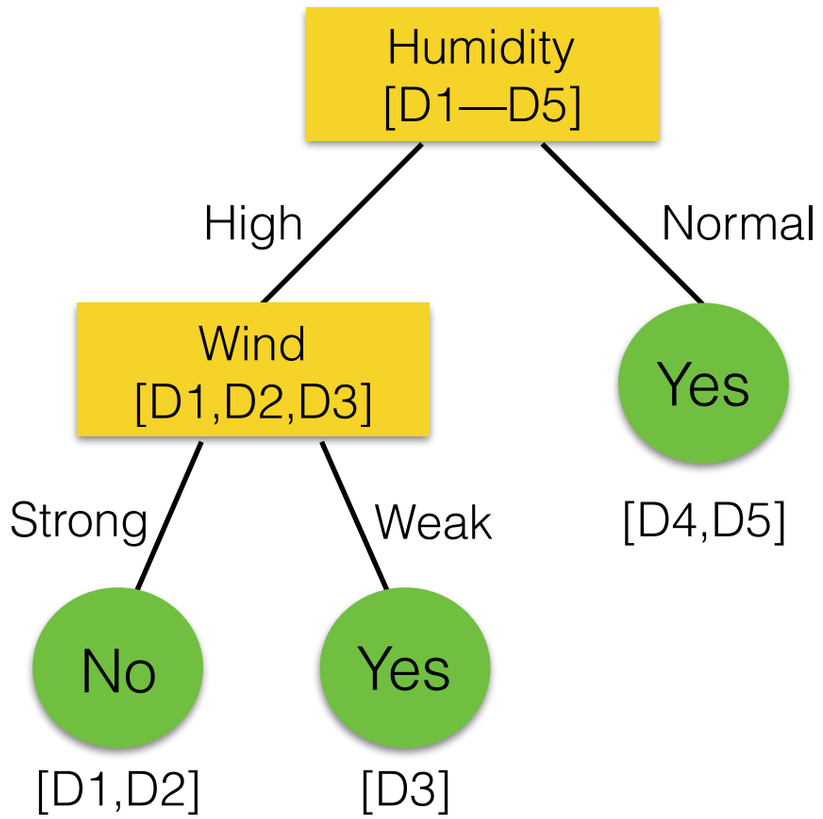


# Creating Leaf Nodes for Classification Problems

| Day | Play |
|---|---|
| D1 | No |
| D2 | No |

| Day | Play |
|---|---|
| D3 | Yes |

| Day | Wind | Play |
|---|---|---|
| D4 | Strong | Yes |
| D5 | Strong | Yes |

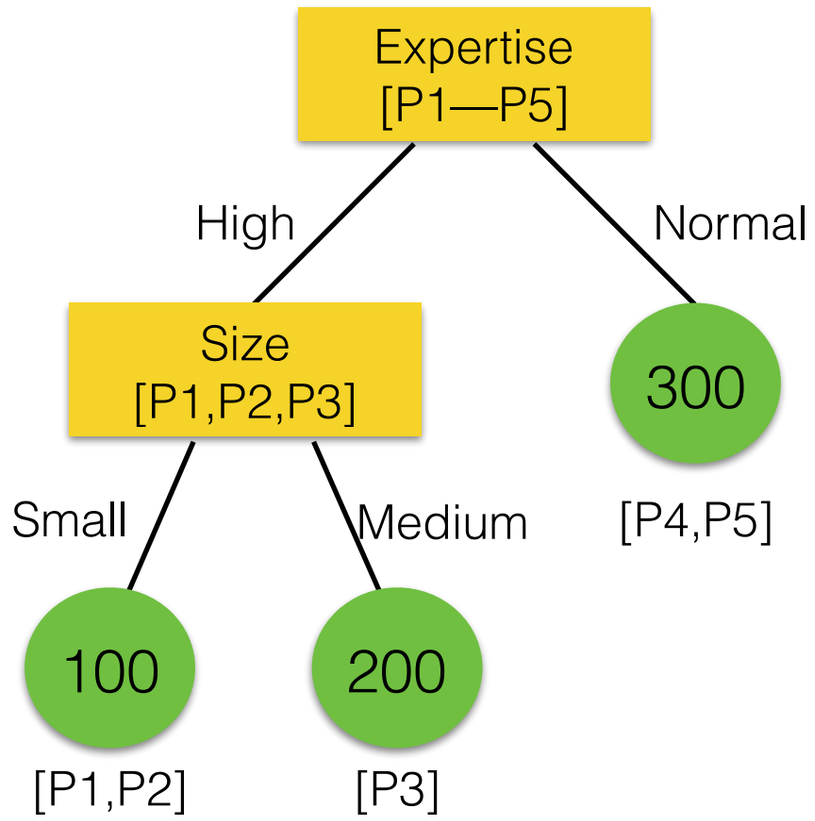


# Creating Leaf Nodes for Regression Problems

| Project | Effort |
|---|---|
| P1 | 100 |
| P2 | 100 |

| Project | Effort |
|---|---|
| P3 | 200 |

| Project | Size | Effort |
|---|---|---|
| P4 | Medium | 300 |
| P5 | Large | 300 |



# How to Build Decision Trees Based on Training Data?

General idea:

- Create a node and split it based on the input attribute that best separates the training examples associated to that node into different classes or numerical values.
- Once a split is made, create a node for each branch and split it based on the procedure above.

Stopping criteria (incomplete):

- If all training examples associated to a node have the same output value, make this node a leaf node of that output value and stop splitting it.

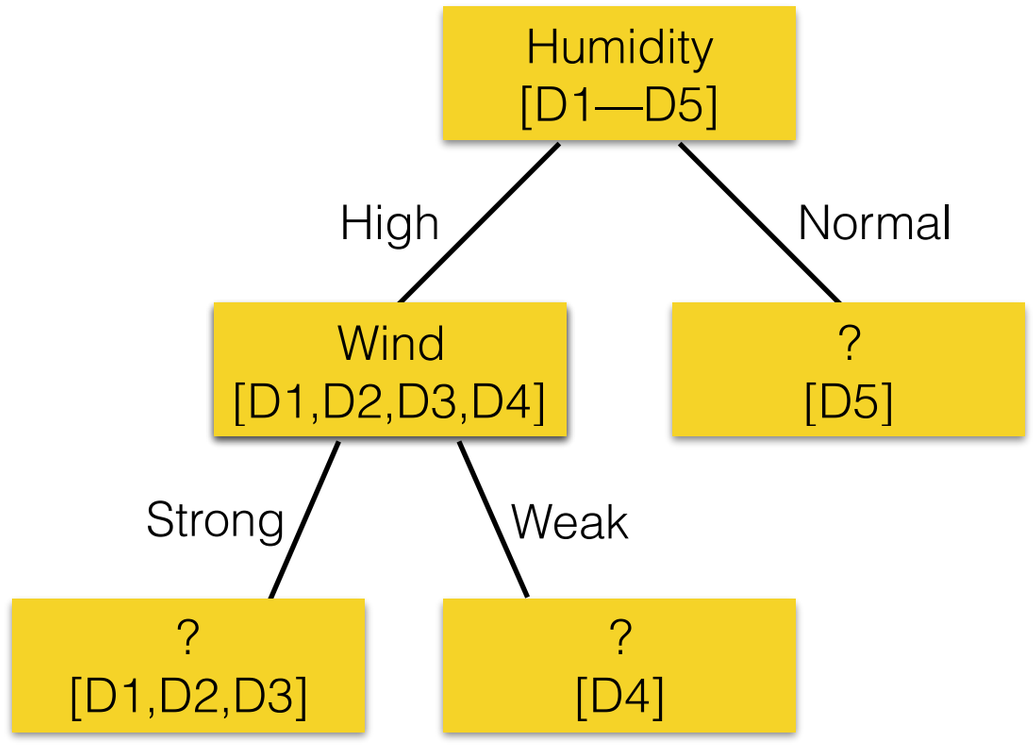
# Heterogeneous Examples With Same Input Attributes

There may be no attributes available for further splitting and the examples associated to a node are not of the same class, e.g., as a result of noise.

| Day | Play |
|---|---|
| D1 | Yes |
| D2 | No |
| D3 | Yes |

| Day | Play |
|---|---|
| D4 | Yes |

| Day | Wind | Play |
|---|---|---|
| D5 | Strong | Yes |



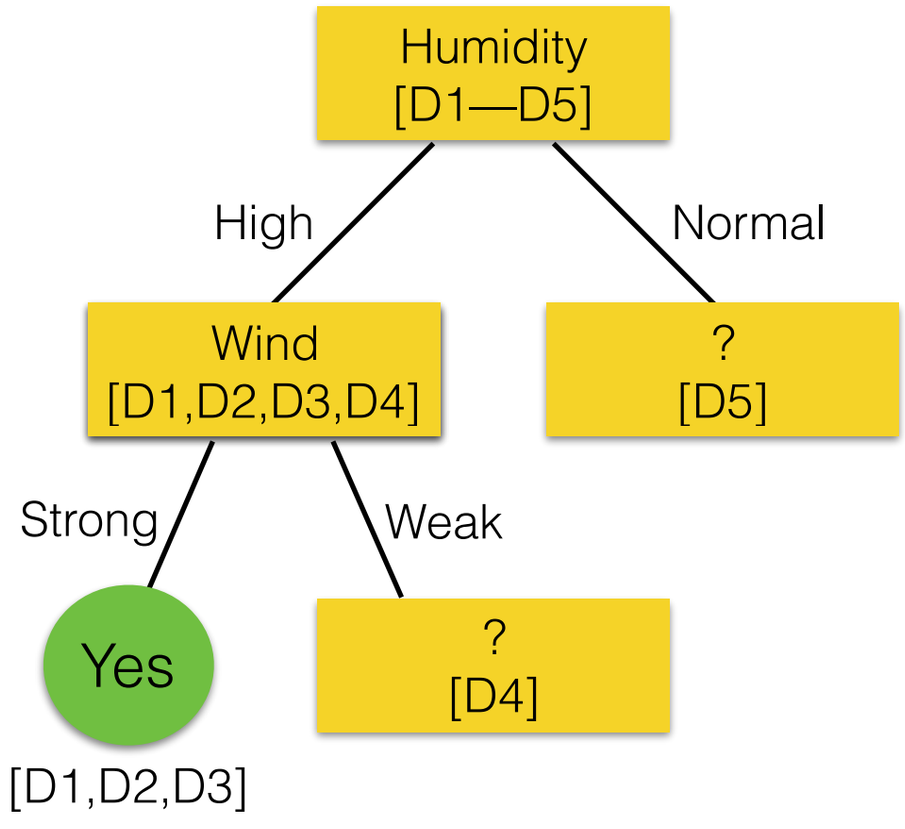
# Creating Leaf Nodes for Classification Problems

Create the leaf node using the **majority** of the outputs of the examples associated to that node.

| Day | Play |
|---|---|
| D1 | Yes |
| D2 | No |
| D3 | Yes |

| Day | Play |
|---|---|
| D4 | Yes |

| Day | Wind | Play |
|---|---|---|
| D5 | Strong | Yes |



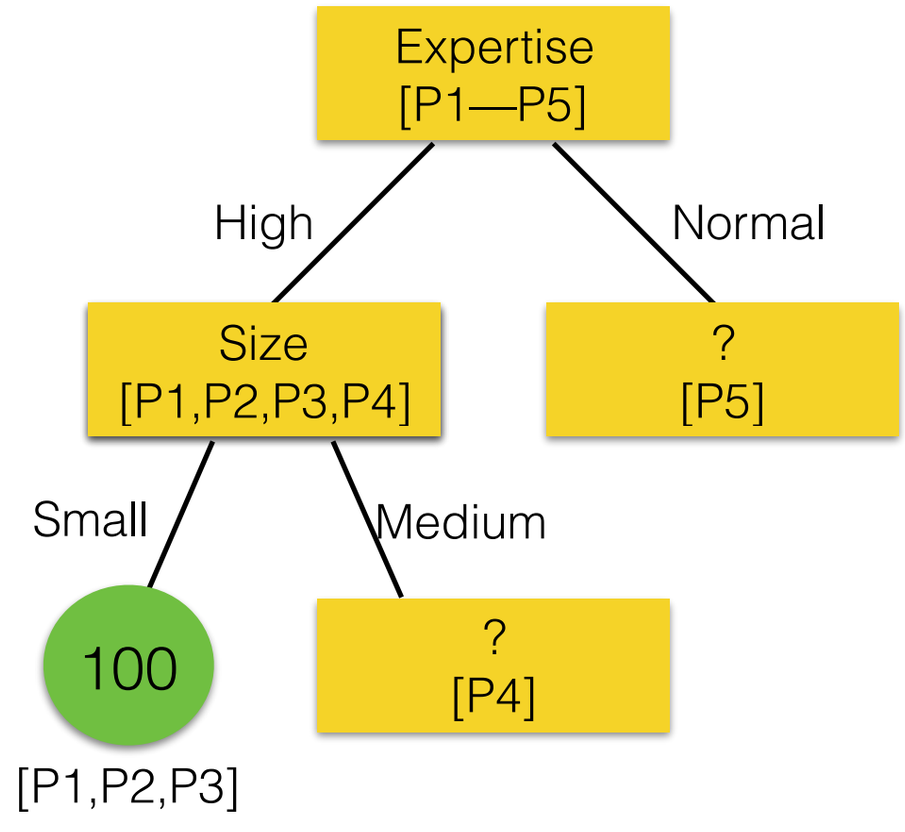
# Creating Leaf Nodes for Regression Problems

Leaf nodes predict the **average** of the outputs of the training examples associated to it.

| Project | Effort |
|---|---|
| D1 | 90 |
| D2 | 100 |
| D3 | 110 |

| Project | Effort |
|---|---|
| D4 | 700 |

| Project | Size | Effort |
|---|---|---|
| D5 | Large | 1000 |



# How to Build Decision Trees Based on Training Data?

General idea:

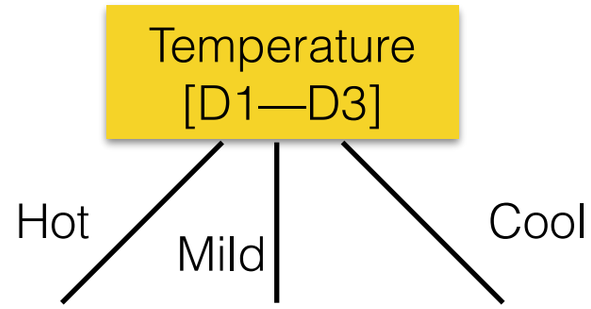
- Create a node and split it based on the input attribute that best separates the training examples associated to that node into different classes or numerical values.
- Once a split is made, create a node for each branch and split it based on the procedure above.

Stopping criteria (incomplete):

- If all training examples associated to a node have the same output value, make this node a leaf node of that output value and stop splitting it.
- **If there are no further attributes to split the node on but there are still examples associated to the node, make this node a leaf node of the majority class (or average of numerical output values) and stop splitting it.**

# Split When There is No Training Data With a Given Attribute Value

| Day | Temperature | Play |
|---|---|---|
| D1 | Hot | No |
| D2 | Hot | No |
| D3 | Cool | Yes |

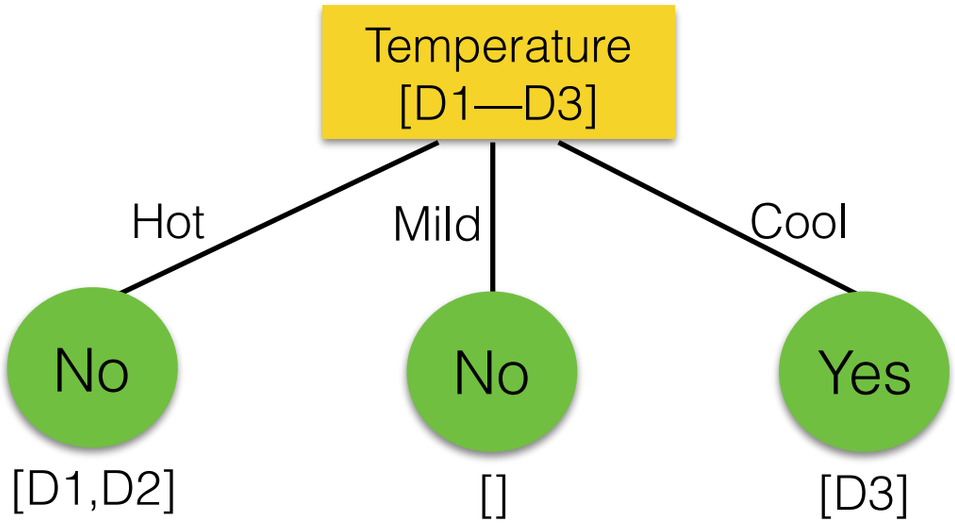



Temperature ∈ {hot, mild, cool}

# Creating Leaf Node for Classification Problems

| Day | Temperature | Play |
|---|---|---|
| D1 | Hot | No |
| D2 | Hot | No |
| D3 | Cool | Yes |

Temperature ∈ {hot, mild, cool}

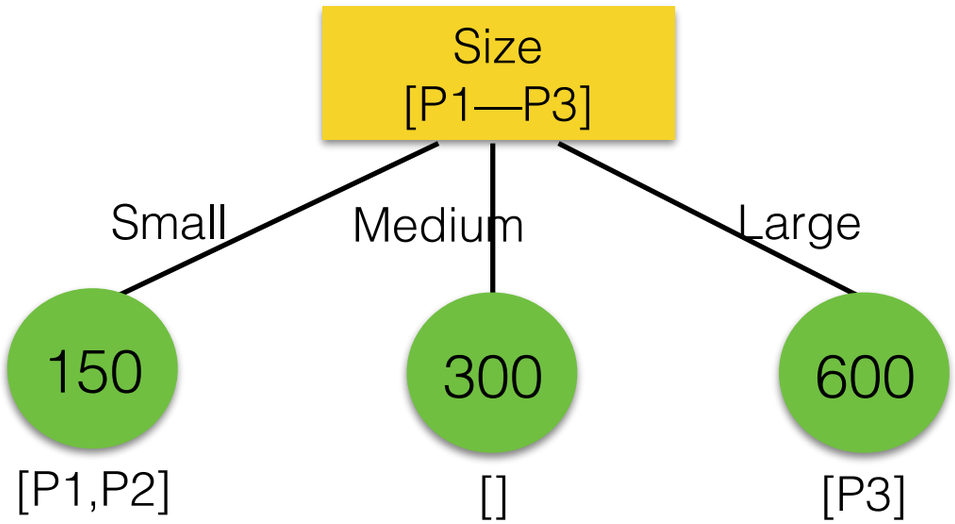



The leaf node for mild predicts the **majority** among the outputs of the examples of its parent node.

# Creating Leaf Node for Regression Problems

| Project | Size | Effort |
|---|---|---|
| P1 | Small | 100 |
| P2 | Small | 200 |
| P3 | Large | 600 |

Size ∈ {small, medium, large}




Leaf nodes predict the **average** of the outputs of the training examples associated to its parent.

# How to Build Decision Trees Based on Training Data?

General idea:

- Create a node and split it based on the input attribute that best separates the training examples associated to that node into different classes or numerical values.
- Once a split is made, create a node for each branch and split it based on the procedure above.

Stopping criteria:

- If all training examples associated to a node have the same output value, make this node a leaf node of that output value and stop splitting it.
- If there are no further attributes to split the node on but there are still examples associated to the node, make this node a leaf node of the majority class (or average of numerical output values) and stop splitting it.
- **If there are no examples associated to a node, make this node a leaf node of the majority class (or average of numerical output values) of its parent and stop splitting it.**

# How to Build Decision Trees Based on Training Data?

General idea:

- Create a node and split it based on the input attribute that best separates the training examples associated to that node into different classes or numerical values.
- Once a split is made, create a node for each branch and split it based on the procedure above.

Stopping criteria:

- If all training examples associated to a node have the same output value, make this node a leaf node of that output value and stop splitting it.
- If there are no further attributes to split the node on but there are still examples associated to the node, make this node a leaf node of the majority class (or average of numerical output values) and stop splitting it.
- If there are no examples associated to a node, make this node a leaf node of the majority class (or average of numerical output values) of its parent and stop splitting it.

# Next Lectures

General idea:

- Create a node and split it based on the **input attribute that best separates the training examples** associated to that node into different classes or numerical values.
- Once a split is made, create a node for each branch and split it based on the procedure above.

How to build decision trees for numerical input attributes.

How to avoid overfitting.

Applications of decision trees.

# Further Reading

Tom Mitchell

Machine Learning

London : McGraw-Hill, 1997

Chapter 3, sections 3.1 to 3.5.

http://www.cs.princeton.edu/courses/archive/spr07/cos424/papers/mitchell-dectrees.pdf

http://content.talisaspire.com/leicester/bundles/571f72e6e7ebb60e4600000a