Machine Learning

Decision Trees Classification

Mustafa Jarrar

Birzeit University

Watch this lecture and download the slides

Course Page: http://www.jarrar.info/courses/AI/
More Online Courses at: http://www.jarrar.info

Acknowledgement:
This lecture is based on (but not limited to) to the lecture notes found in [1,2,3]

Machine Learning

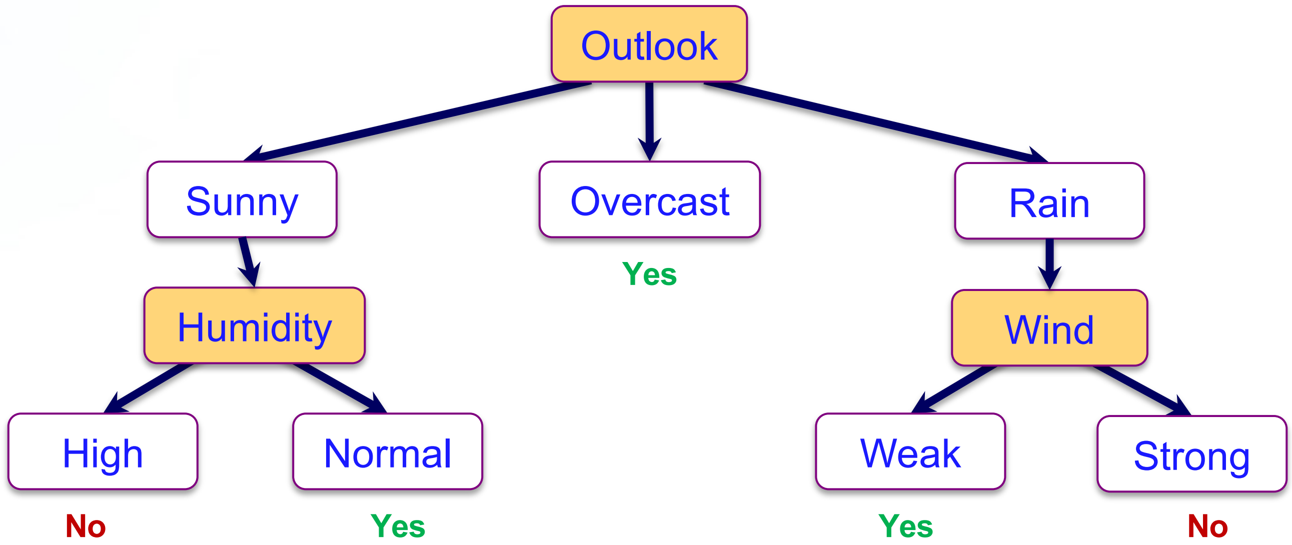
Decision Tree Classification

In this lecture:

- Part 1: **Motivation Example**
- ❑ Part 2: ID3 Algorithm
- ❑ Part 3: Entropy and Information Gain
- ❑ Part 4: Overfitting and Pruning
- ❑ Part 5: Classifying Continuous/Numerical Values
- ❑ Part 6: Pros and Cons of Decision Trees
- ❑ Part 7: Using R and Python to learn Decision Trees

Example of a Decision Tree

Is it a good weather to play outside?



How to learn such a tree from past experience?

Decision Tree – Motivation Example

Given the following training examples, will you play in D15?

Divide and conquer:
- split into subsets
- are they pure?
  (all yes or all no)
- if yes: stop
- if not: repeat

See which subset new data falls into

Training Examples

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

New data:

D15	**Rain**	**High**	**Weak**	**???**

Decision Tree – Motivation Example

9 Yes / 5 No

Outlook

2 / 3

Sunny

Day	Outlook	Humid	Wind
D1	Sunny	High	Weak
D2	Sunny	High	Strong
D8	Sunny	High	Weak
D9	Sunny	Norma	Weak
D11	Sunny	Norma	Strong

4 / 0

Overcast

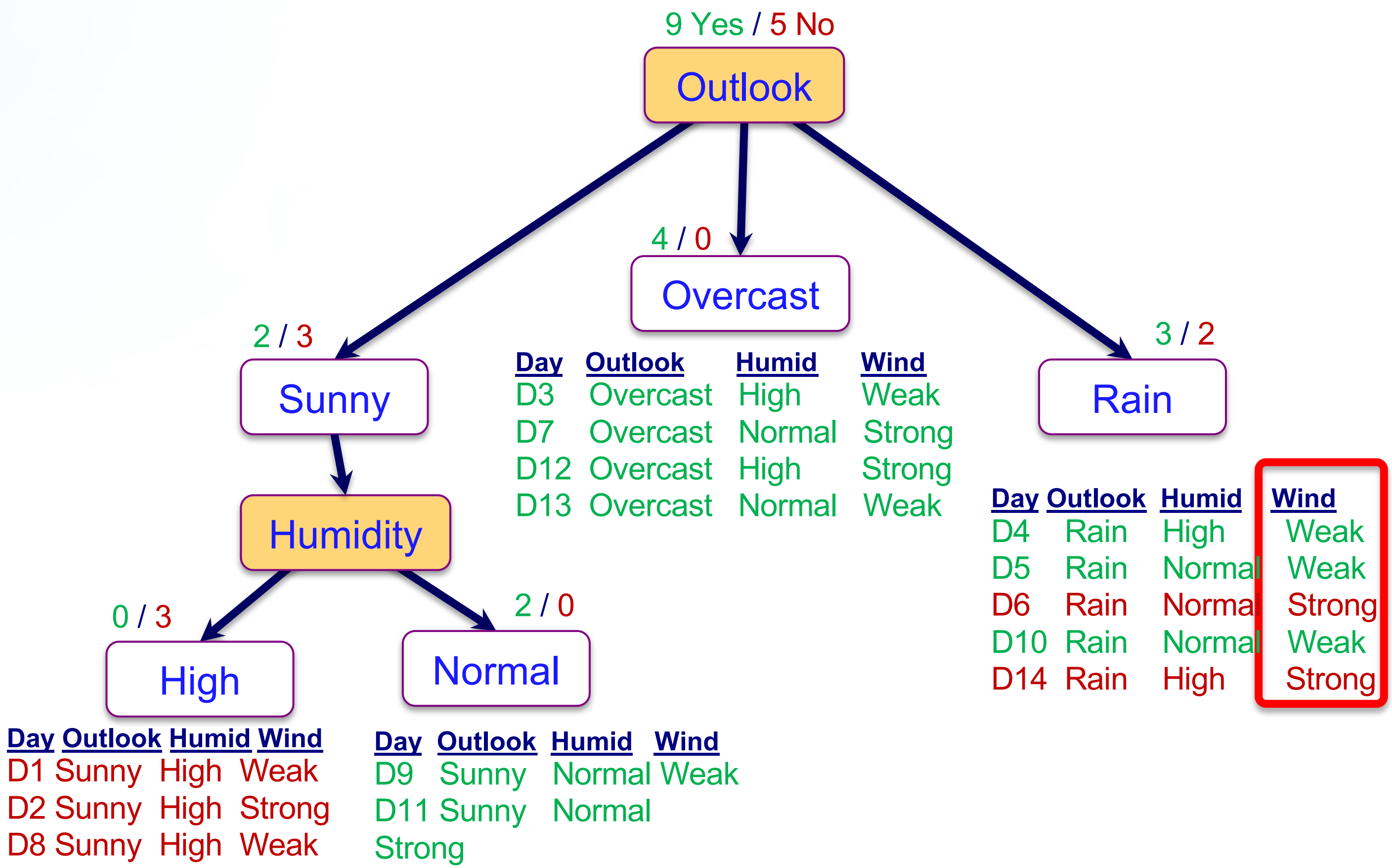
Day	Outlook	Humid	Wind
D3	Overcast	High	Weak
D7	Overcast	Normal	Strong
D12	Overcast	High	Strong
D13	Overcast	Normal	Weak

3 / 2

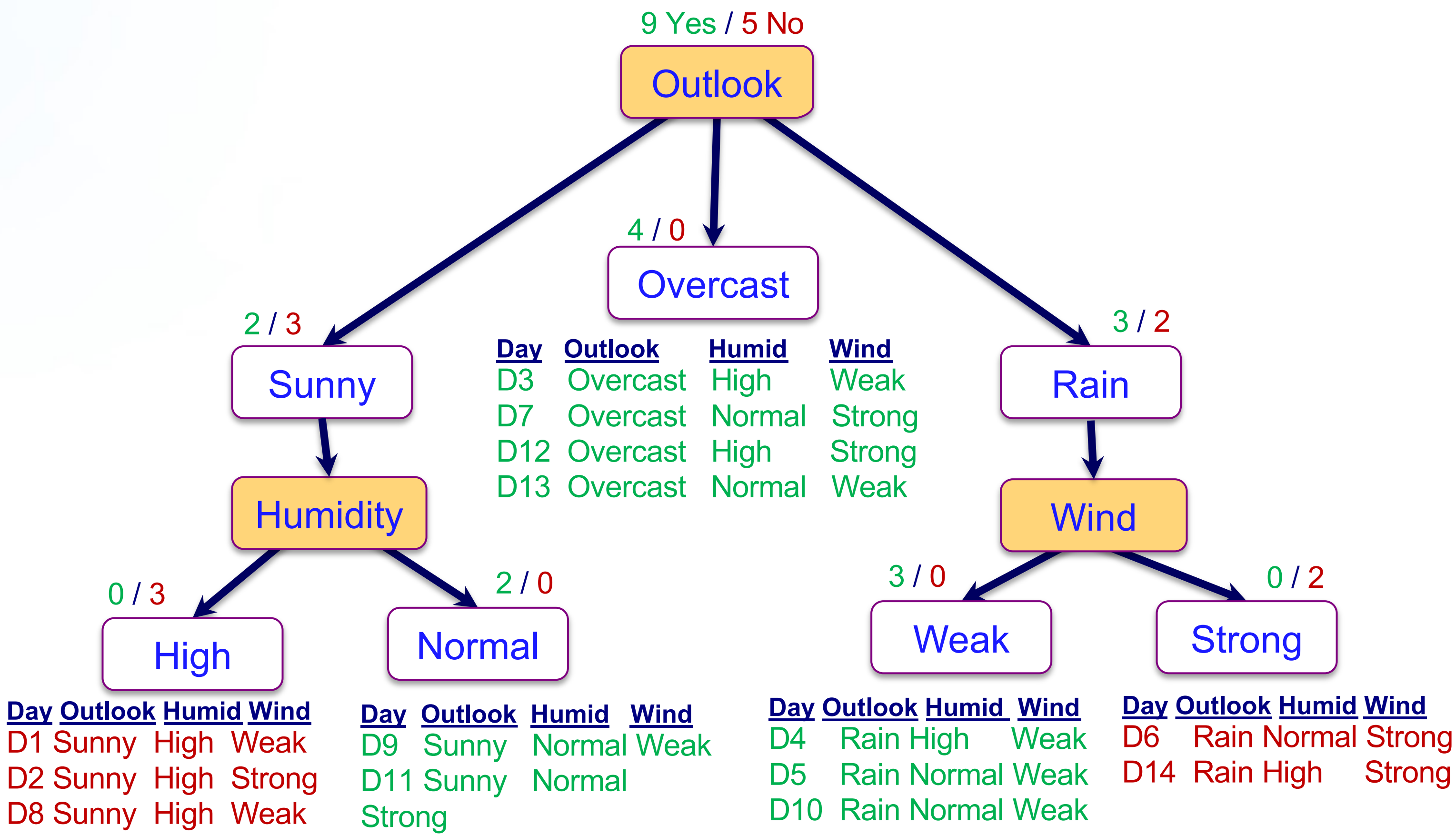
Rain

Day	Outlook	Humid	Wind
D4	Rain	High	Weak
D5	Rain	Normal	Weak
D6	Rain	Normal	Strong
D10	Rain	Normal	Weak
D14	Rain	High	Strong

Decision Tree – Motivation Example

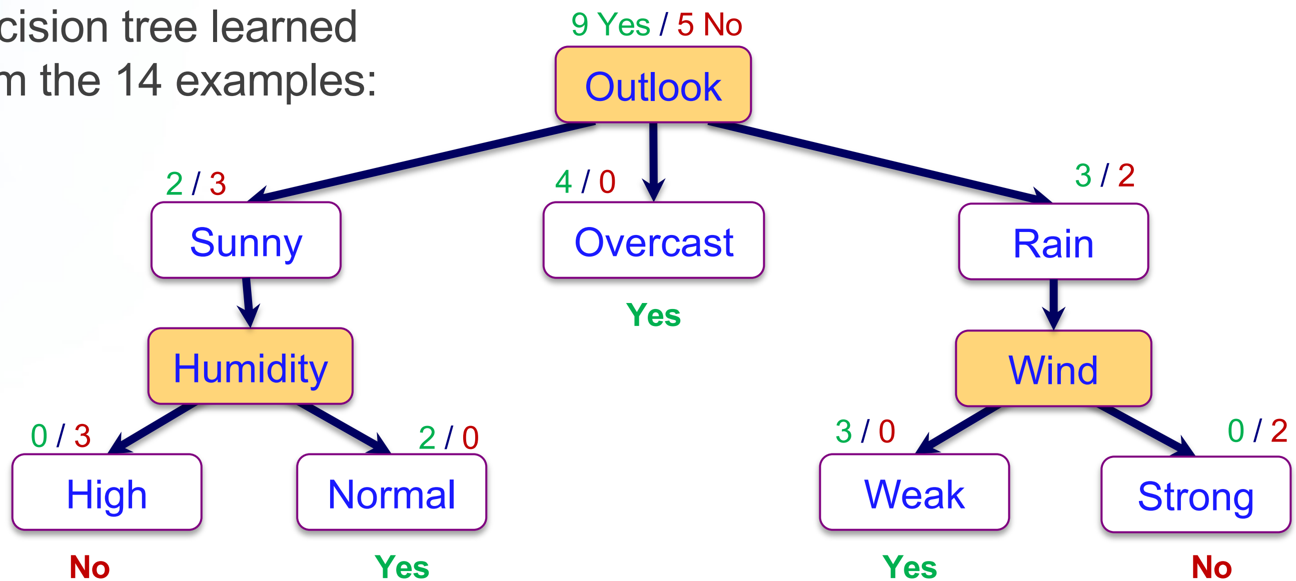


Decision Tree – Motivation Example



Decision Tree – Motivation Example

Decision tree learned from the 14 examples:

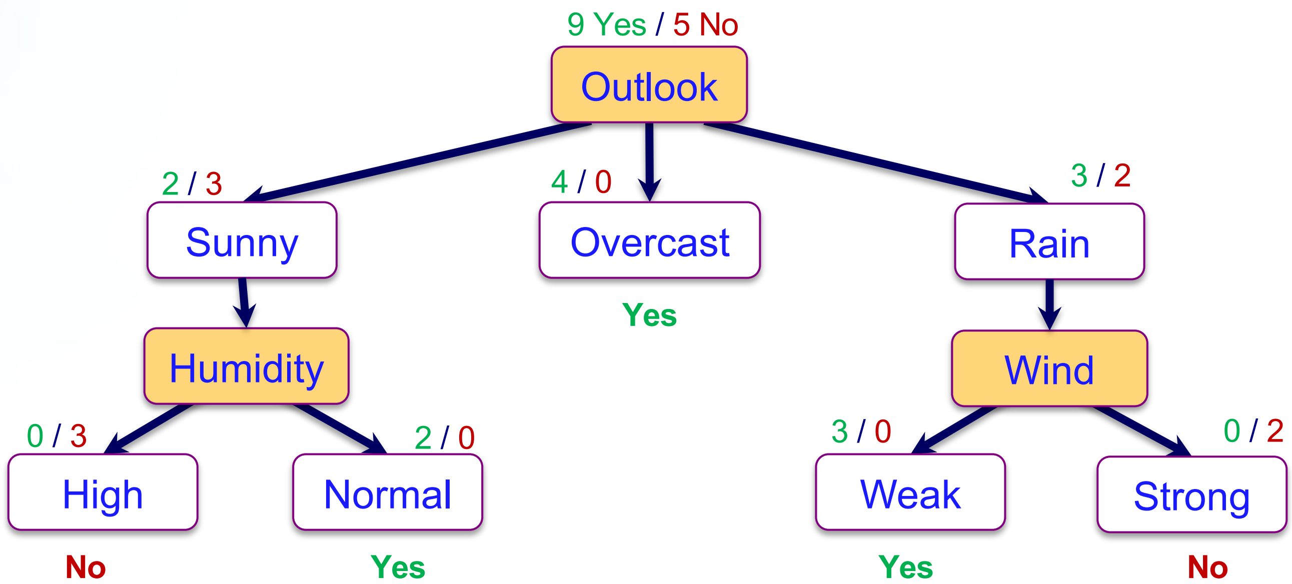


Decision Rule:

Yes ⇔ (Outlook=Overcast) ∨
(Outlook=Sunny ∧ Humidity=Normal) ∨
(Outlook=Rain ∧ Wind=Weak)

Day	Outlook	Humid	Wind		
D15	Rain	High	Weak	???	➔ Play

Decision Trees are Interpretable



Disjunction of conjunctions of constraints on the attribute values of instances i.e., (... ∧ ... ∧ ...) ∨ (... ∧ ... ∧ ...) ∨ ...

Set of **if-then-rules**, each branch represents one if-then-rule

- **if-part**: conjunctions of attribute tests on the nodes
- **then-part**: classification of the branch

Machine Learning

Decision Tree Classification

In this lecture:

- Part 1: Motivation Example
- Part 2: **ID3 Algorithm**
- Part 3: Entropy and Information Gain
- Part 4: Overfitting and Pruning
- Part 5: Classifying Continuous/Numerical Values
- Part 6: Pros and Cons of Decision Trees
- Part 7: Using R and Python to learn Decision Trees

ID3 Algorithm

Based on [1]

Split (node, {examples}):

1. A ← the <u>best attribute</u> for splitting the {examples}
2. Decision attribute for this node ← A
3. For each value of A, create new child node
4. Split training {examples} to child nodes
5. If examples perfectly classified: STOP

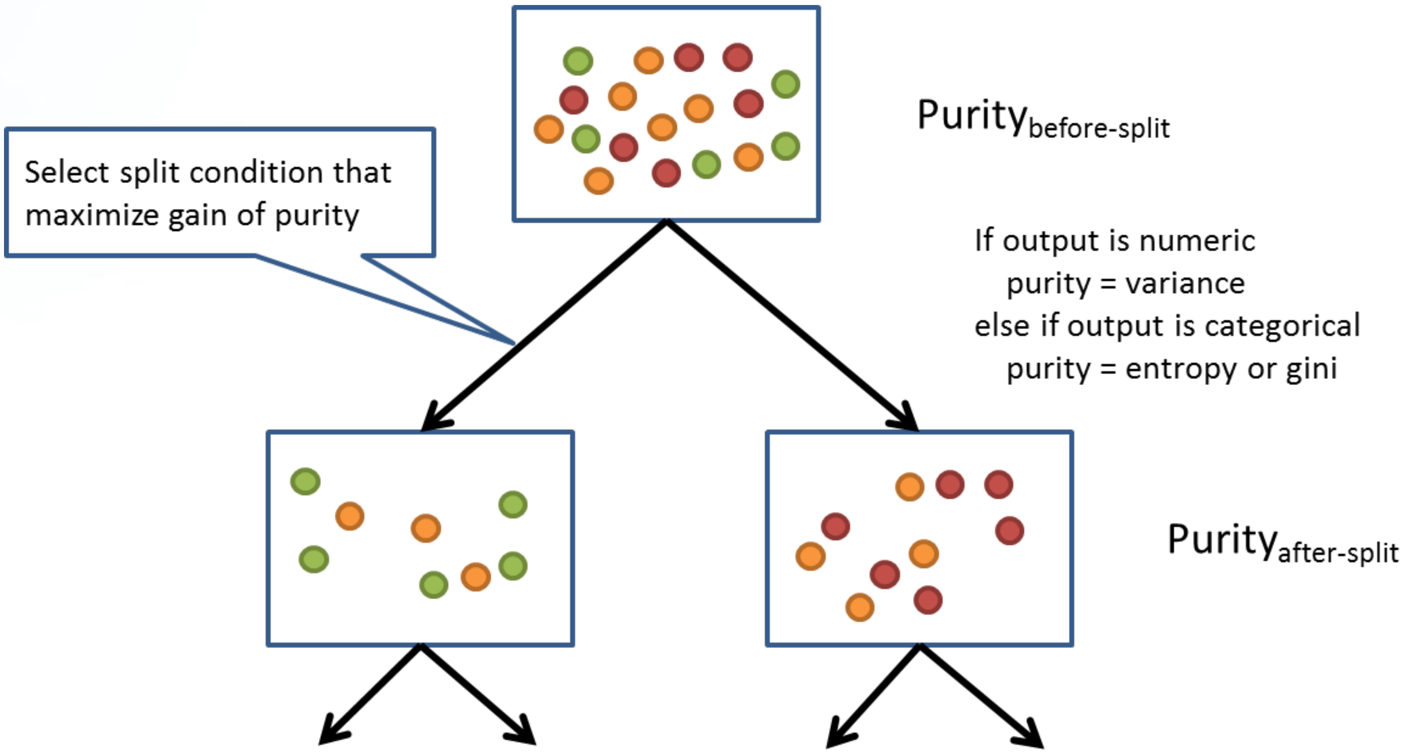
   else: iterate over new child nodes

   Split (child_node, {subset of examples})

- Ross Quinlan (ID3:1986), (C4.5:1993) from machine learning
- Breimanetal (CaRT:1984) from statistics

ID3 Algorithm

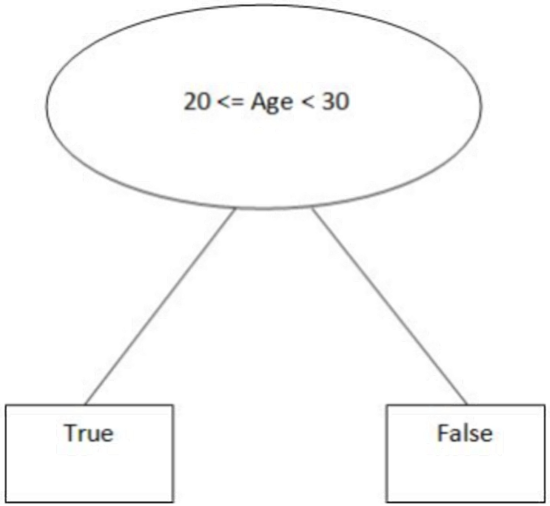
From [Sagarnil Das]



C4.5 (and C5.0) Algorithms

C4.5 (and C5.0) are similar algorithms to construct decision trees but are also able handle continuous values

In case of **Continuous Variables** like age, weight etc?



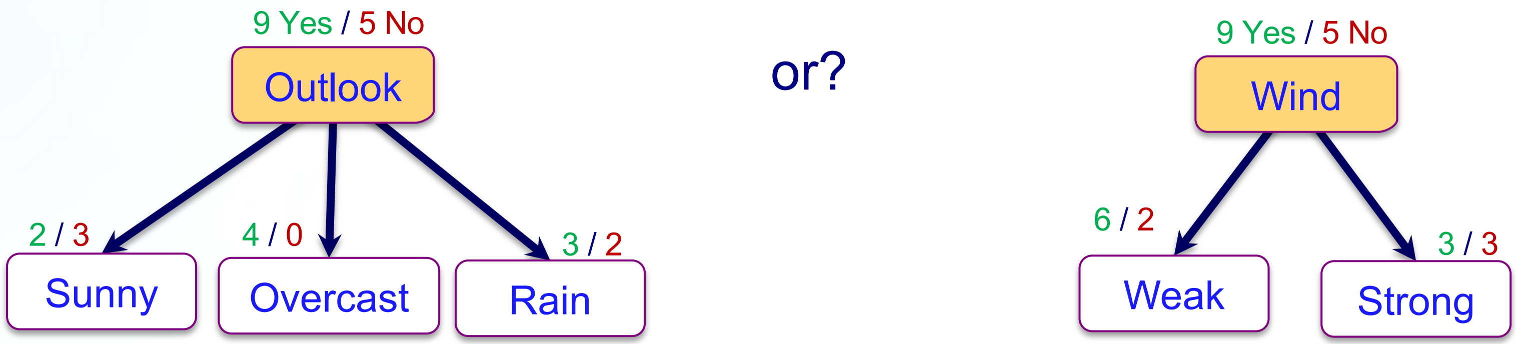
Machine Learning

Decision Tree Classification

In this lecture:

- ❑ **Part 1:** Motivation Example
- ❑ **Part 2:** ID3 Algorithm
- ❑ **Part 3: Entropy and Information Gain**
- ❑ **Part 4:** Overfitting and Pruning
- ❑ **Part 5:** Classifying Continuous/Numerical Values
- ❑ **Part 6:** Pros and Cons of Decision Trees
- ❑ **Part 7:** Using R and Python to learn Decision Trees

Which attribute to split on?



Want to measure "purity" of the split

- more certain about Yes/No after the split
  - pure set (4 yes / 0 no) => completely certain (100%)
  - impure (3 yes / 3 no) => completely uncertain (50%)
- can't use the probability of "yes" given the set, P("yes" | set):
  - must be symmetric: 4 yes / 0 no as pure as 0 yes / 4 no

Entropy

Entropy tells us how much a set of data is pure/impure

For binary classification:

$$\text{Entropy(S)} = \ \text{H}(S) = -p_{\oplus} \log_2 p_{\oplus} \ - \ p_{\ominus} \log_2 p_{\ominus} \text{ bits}$$

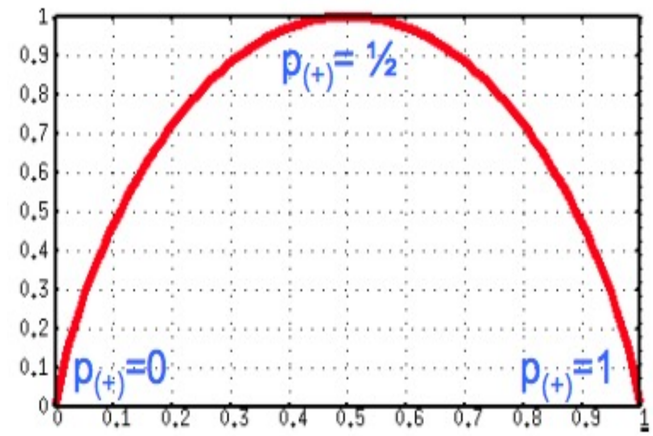
– S … is a sample training examples
– $p_{\oplus}$ proportion of positive examples in S
– $p_{\ominus}$ proportion of negative examples in S
– $p_{\oplus} / p_{\ominus}$ … % of positive/negative examples in S

- impure (3 yes / 3 no):

$$H(S) = -\frac{3}{6}\log_2\frac{3}{6} - \frac{3}{6}\log_2\frac{3}{6} = 1 \text{ bits}$$

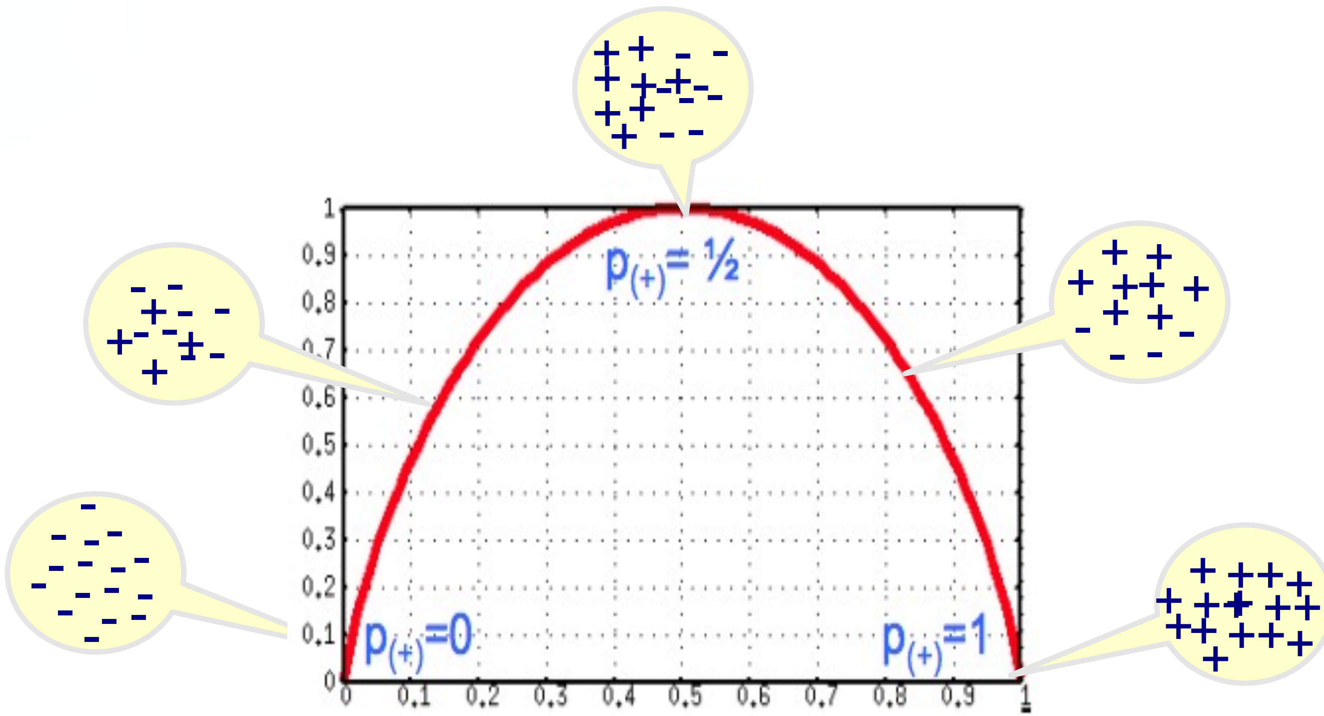
- pure set (4 yes / 0 no):

$$H(S) = -\frac{4}{4}\log_2\frac{4}{4} - \frac{0}{4}\log_2\frac{0}{4} = 0 \text{ bits}$$



Entropy

Entropy tells us how much a set of data is pure/impure



Information Gain

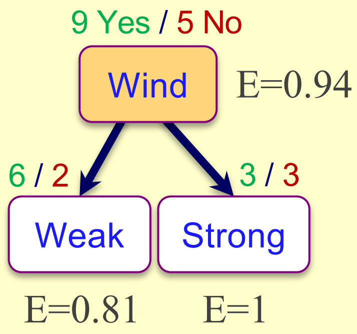
Entropy measures purity at each node, information gain looks at all nodes together and the expected drop in entropy after split.

Gain(S,A) = expected reduction in entropy due to sorting on A

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

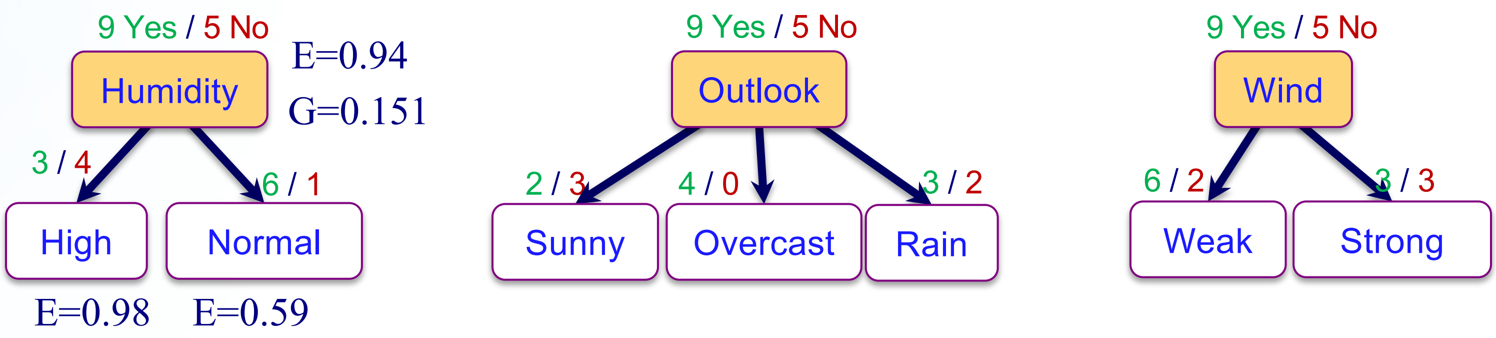
Maximum Gain(S, A) is selected!

Example



$$Gain(\text{S, Wind}) = 0.94 - (8/14) \cdot 0.81 - (6/14) \cdot 1 = 0.048$$

Full Example (Entropy & Information Gain)



$$Entropy(\text{Humidity}) = -9/14 \cdot \log_2(9/14) - 5/14 \cdot \log_2(5/14) = 0.94$$

$$Entropy(\text{High}) = -3/7 \cdot \log_2(3/7) - 4/7 \cdot \log_2(4/7) = 0.98$$

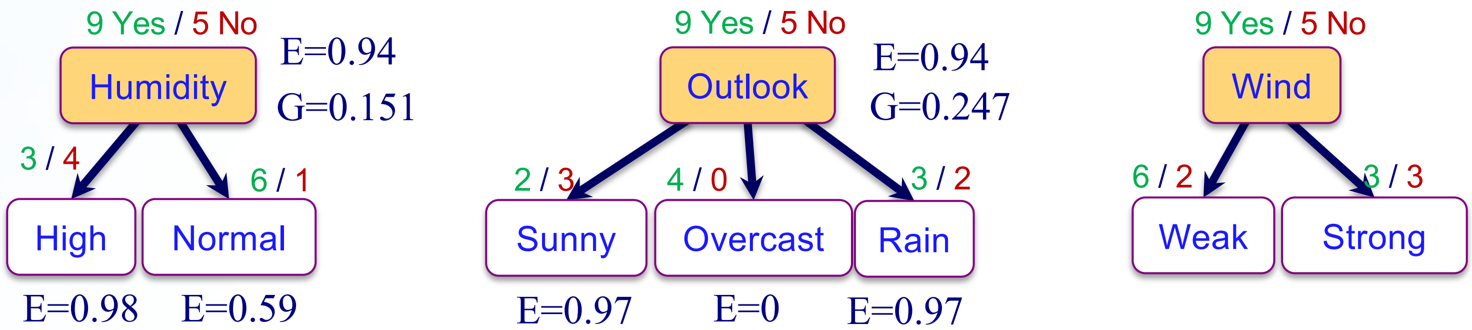
$$Entropy(\text{Normal}) = -6/7 \cdot \log_2(6/7) - 1/7 \cdot \log_2(1/7) = 0.59$$

$$Gain(\text{S, A}) = Entropy(\text{S}) - \sum_{v=\{\text{High, Normal}\}} \frac{|Sv|}{S} \cdot \text{Entropy}(\text{S}_v)$$

$$Gain(\text{S, Humidity}) = 0.94 - (7/14) \cdot 0.98 - (7/14) \cdot 0.59 = 0.151$$

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

Full Example (Entropy & Information Gain)



$$Entropy(\text{Outlook}) = -9/14\cdot\log_2(9/14) - 5/14\cdot\log_2(5/14) = 0.94$$

$$Entropy(\text{Sunny}) = -2/5\cdot\log_2(2/5) - 3/5\cdot\log_2(3/5) = 0.97$$

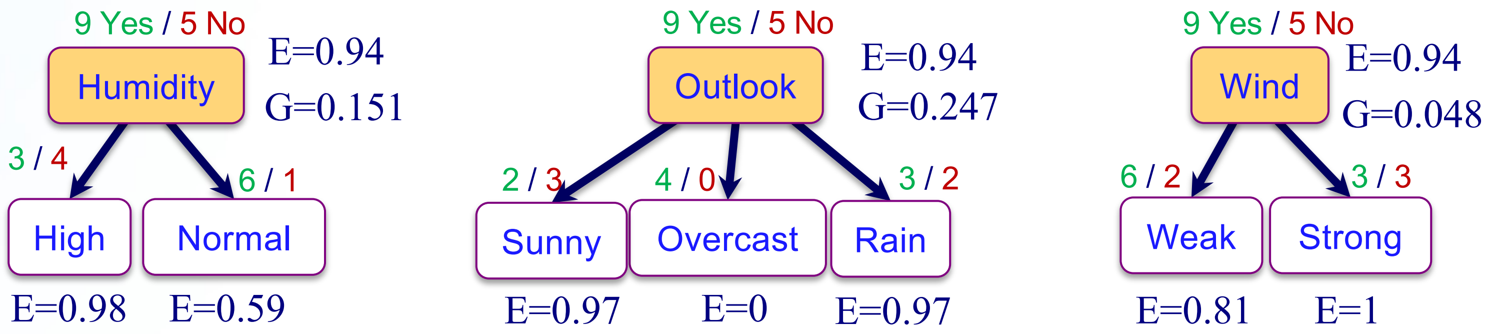
$$Entropy(\text{Overcast}) = -4/4\cdot\log_2(4/4) - 0/4\cdot\log_2(0/4) = 0$$

$$Entropy(\text{Rain}) = -3/5\cdot\log_2(3/5) - 2/5\cdot\log_2(2/5) = 0.97$$

$$Gain(\text{S, Outlook})=0.94 -(5/14)\cdot 0.97 -(4/14)\cdot 0 -(5/14)\cdot 97 = 0.247$$

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

Full Example (Entropy & Information Gain)



$$Entropy(\text{Wind}) = -9/14 \cdot \log_2(9/14) - 5/14 \cdot \log_2(5/14) = 0.94$$

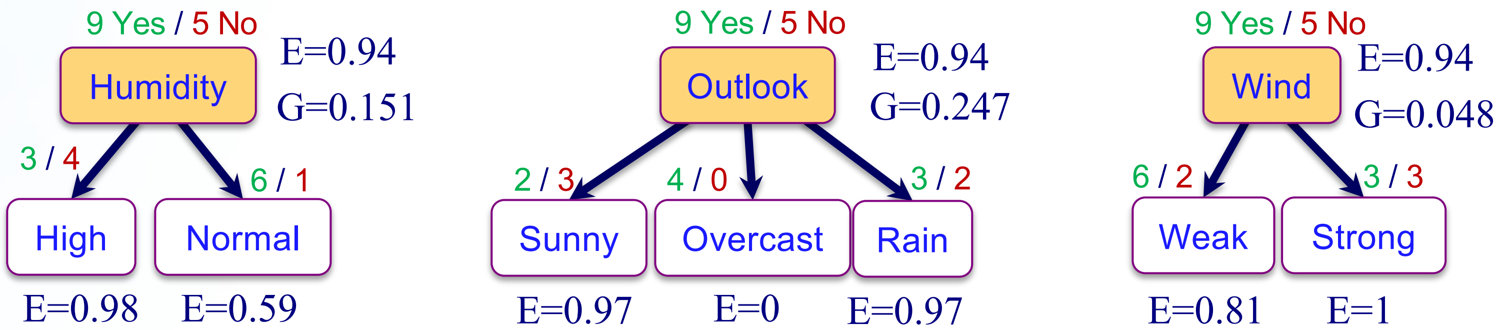
$$Entropy(\text{Weak}) = -6/8 \cdot \log_2(6/8) - 2/8 \cdot \log_2(2/8) = 0.81$$

$$Entropy(\text{Strong}) = -3/6 \cdot \log_2(3/6) - 3/6 \cdot \log_2(3/6) = 1$$

$$Gain(\text{S, Wind}) = 0.94 \quad - (8/14) \cdot 0.81 \;\; - (6/14) \cdot 1 = 0.048$$

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

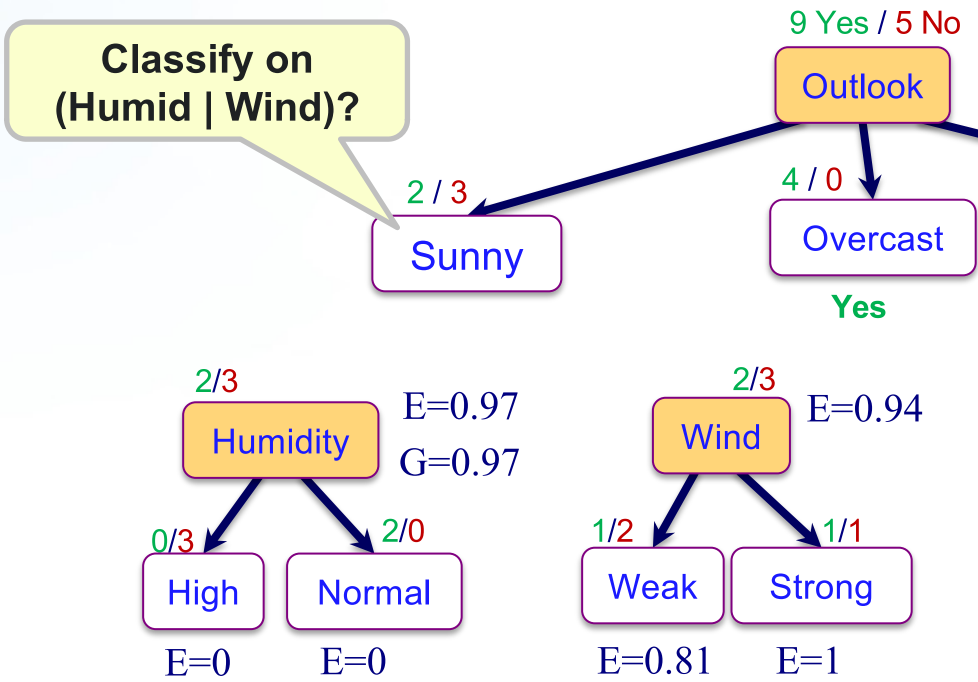
Full Example (Entropy & Information Gain)



The attribute with the largest Information Gain (Outlook 0.247) is selected as the decision node.

Nodes with zero Entropy (e.g., Overcast) does not need splitting

Full Example (Entropy & Information Gain)



Day	Outlook	Humid	Wind
D1	Sunny	High	Weak
D2	Sunny	High	Strong
D8	Sunny	High	Weak
D9	Sunny	Normal	Weak
D11	Sunny	Normal	Strong

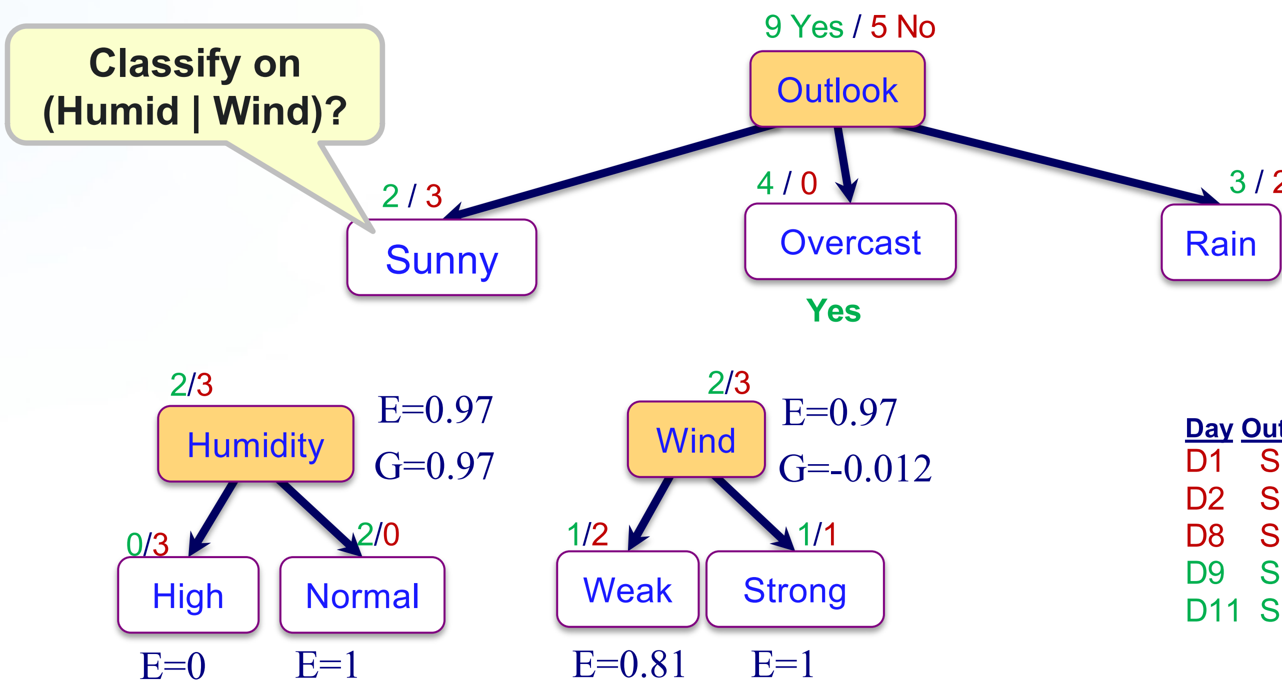
$$Entropy(\text{Humidity}) = -2/5 \cdot \log_2(2/5) - 3/5 \cdot \log_2(3/5) = 0.97$$

$$Entropy(\text{High}) = -0/3 \cdot \log_2(0/3) - 3/3 \cdot \log_2(3/3) = 0$$

$$Entropy(\text{Normal}) = -2/2 \cdot \log_2(2/2) - 0/2 \cdot \log_2(0/2) = 0$$

$$Gain(\text{S, Humidity}) = 0.97 \quad - (3/5) \cdot 0 \ - (2/5) \cdot 0 = 0.97$$

Full Example (Entropy & Information Gain)



Day	Outlook	Humid	Wind
D1	Sunny	High	Weak
D2	Sunny	High	Strong
D8	Sunny	High	Weak
D9	Sunny	Normal	Weak
D11	Sunny	Normal	Strong

$$\mathit{Entropy}(\text{Wind}) = -2/5 \cdot \log_2(2/5) - 3/5 \cdot \log_2(3/5) = 0.97$$

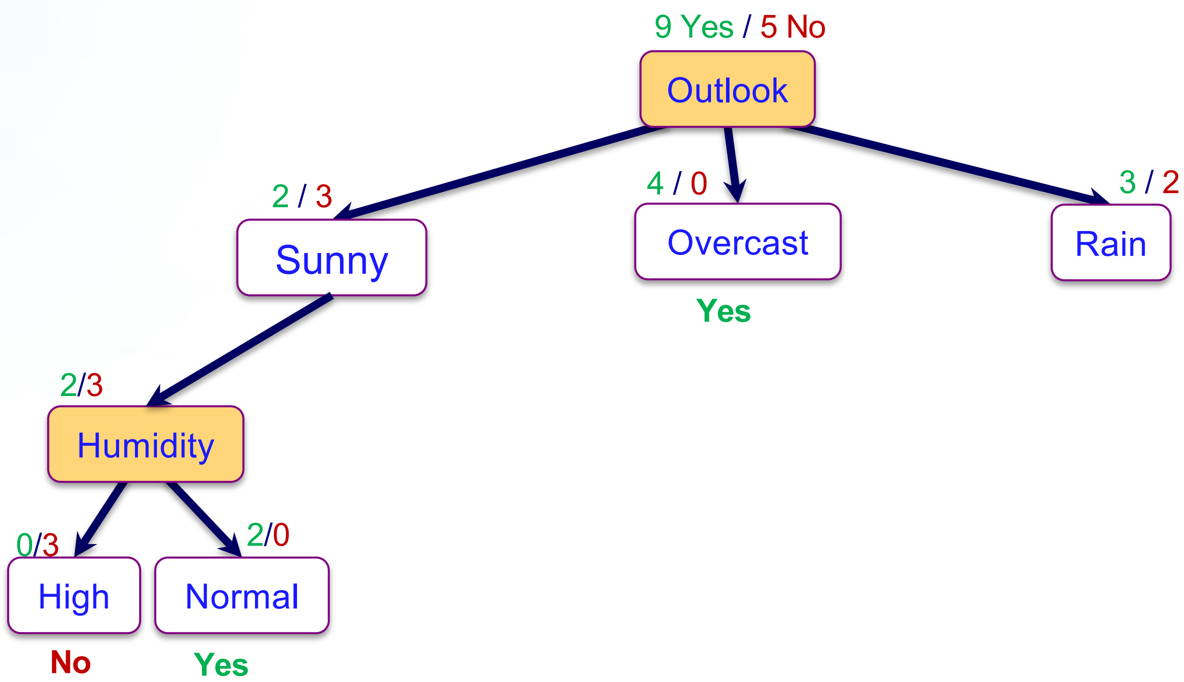
$$\mathit{Entropy}(\text{Weak}) = -1/3 \cdot \log_2(1/3) - 2/3 \cdot \log_2(2/3) = 0.92$$

$$\mathit{Entropy}(\text{Strong}) = -1/2 \cdot \log_2(1/2) - 1/2 \cdot \log_2(1/2) = 1$$

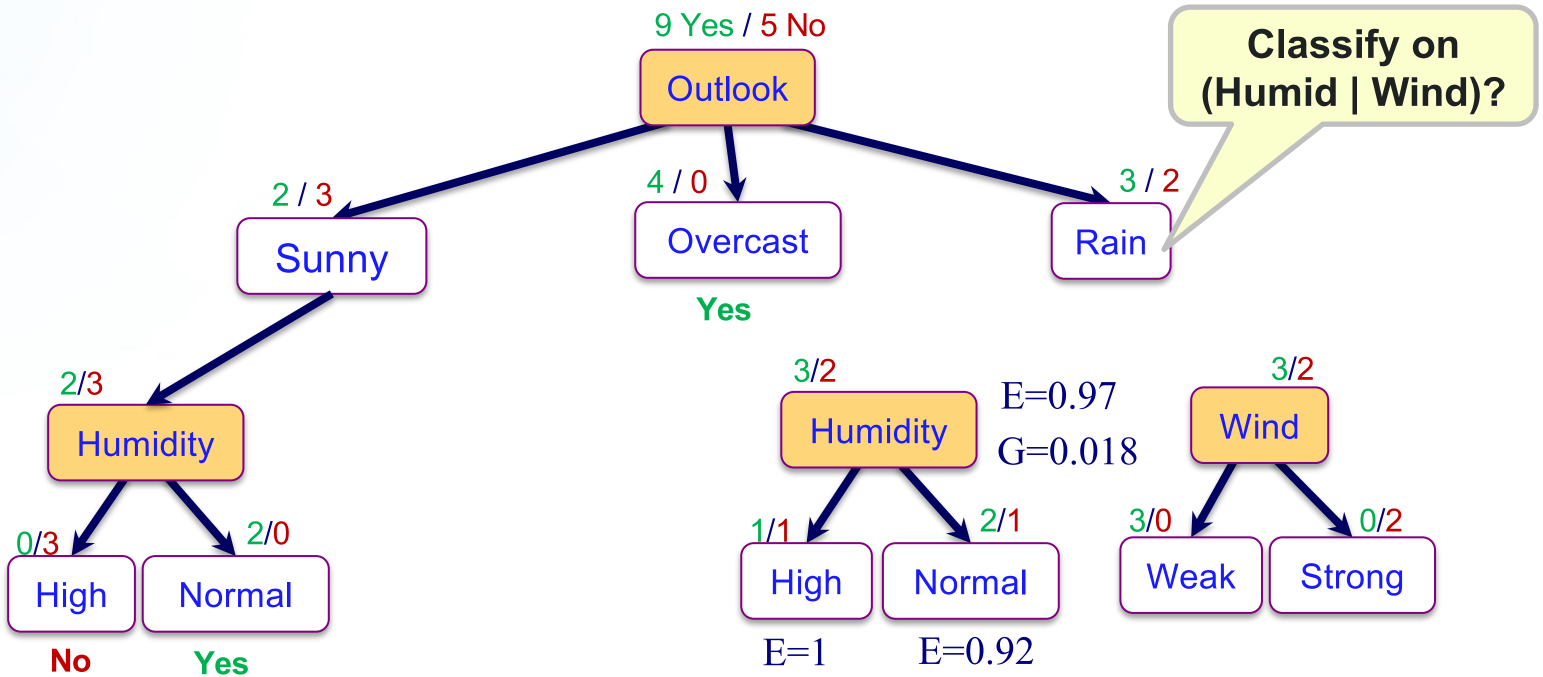
$$\mathit{Gain}(S, \text{Wind}) = 0.97 \quad - (3/5) \cdot 0.97 \;\; - (2/5) \cdot 1 = -0.012$$

➔ Outlook has the highest gain (0.97)

Full Example (Entropy & Information Gain)



Full Example (Entropy & Information Gain)



$$Entropy(\text{Humidity}) = -3/5 \cdot \log_2(3/5) - 2/5 \cdot \log_2(2/5) = 0.97$$

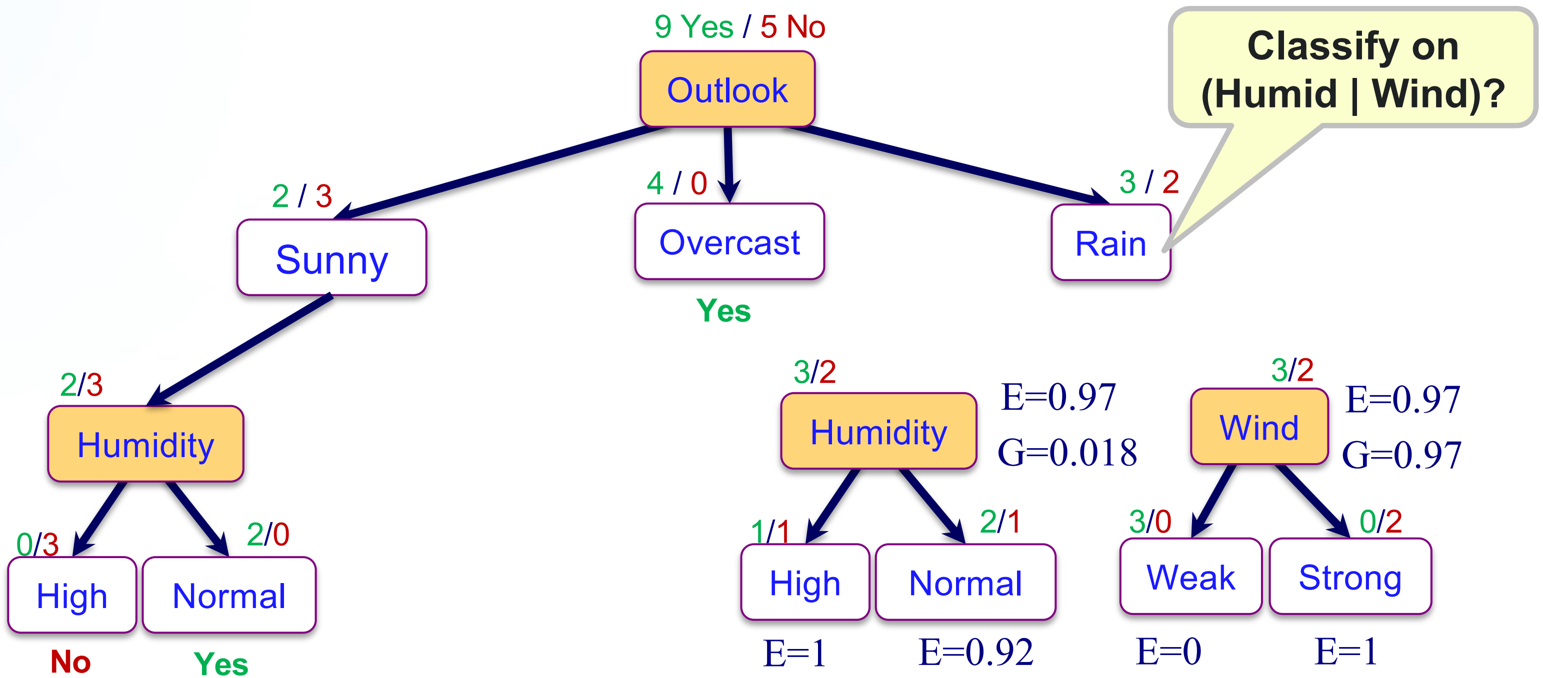
$$Entropy(\text{High}) = -1/2 \cdot \log_2(1/2) - 1/2 \cdot \log_2(1/2) = 1$$

$$Entropy(\text{Normal}) = -2/3 \cdot \log_2(2/3) - 1/3 \cdot \log_2(1/3) = 0.92$$

$$Gain(\text{S, Humidity}) = 0.97 \quad - (2/5) \cdot 1 \; - (3/5) \cdot 0.92 = 0.018$$

Day	Outlook	Humid	Wind
D4	Rain	High	Weak
D5	Rain	Normal	Weak
D6	Rain	Normal	Strong
D10	Rain	Normal	Weak
D14	Rain	High	Strong

Full Example (Entropy & Information Gain)



$$Entropy(\text{Wind}) = -3/5 \cdot \log_2(3/5) - 2/5 \cdot \log_2(2/5) = 0.97$$

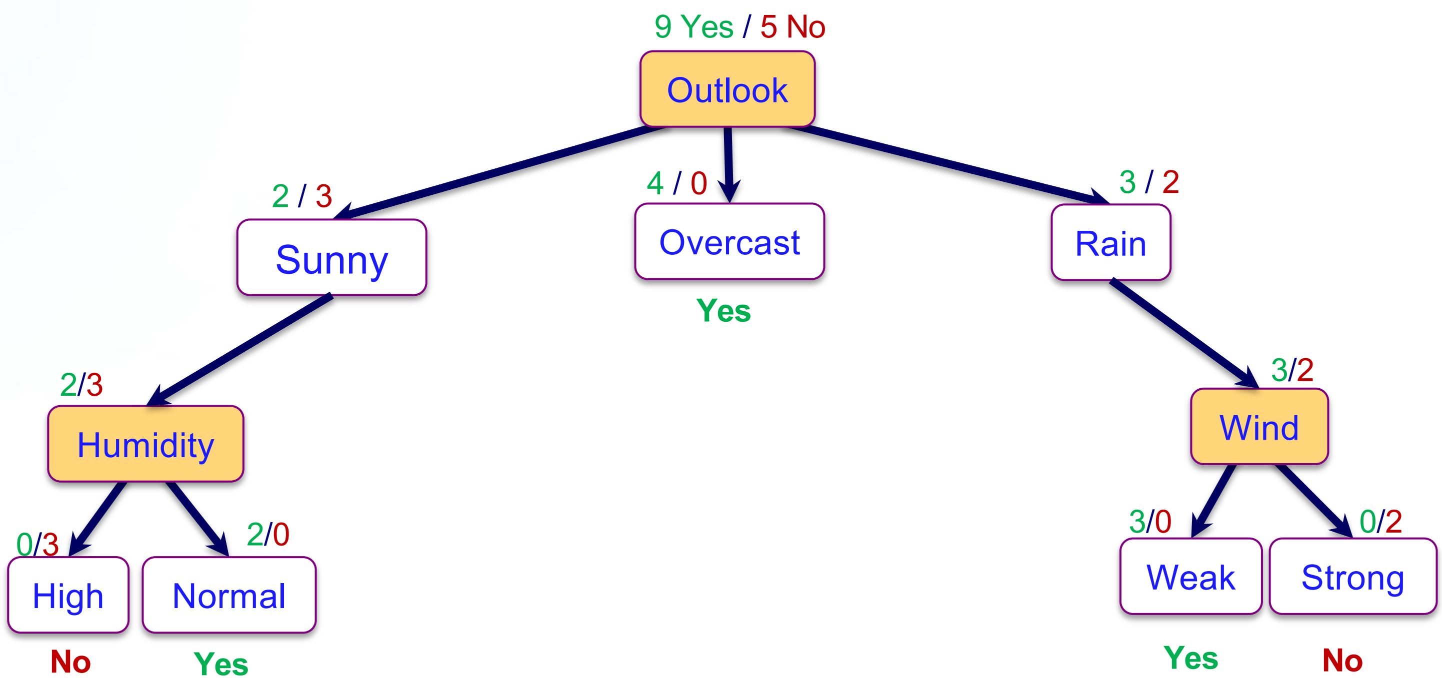
$$Entropy(\text{Weak}) = -3/3 \cdot \log_2(3/3) - 0/3 \cdot \log_2(0/3) = 0$$

$$Entropy(\text{Strong}) = -0/2 \cdot \log_2(0/2) - 2/2 \cdot \log_2(2/2) = 0$$

$$Gain(\text{S, Wind}) = 0.97 \quad - (3/5) \cdot 0 \ - (2/5) \cdot 0 = 0.97$$

Day	Outlook	Humid	Wind
D4	Rain	High	Weak
D5	Rain	Normal	Weak
D6	Rain	Normal	Strong
D10	Rain	Normal	Weak
D14	Rain	High	Strong

Full Example (Entropy & Information Gain)



Decision Rule:

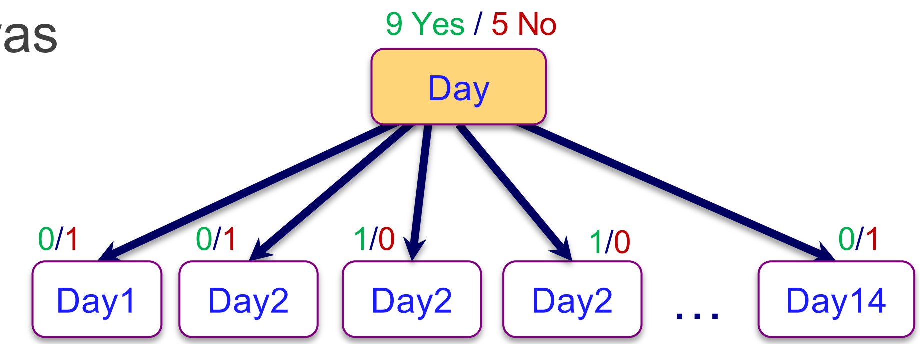
Yes ⇔ (Outlook=Overcast) ∨
(Outlook=Sunny ∧ Humidity=Normal) ∨
(Outlook=Rain ∧ Wind=Weak)

Problems with Information Gain

Based on [1]

What happen if "Day" was used for splitting?

➔ cannot classify new data (Day15?!)



All subsets perfectly pure → optimal split

Information gain tends to favor attributes with lots of values.

We may use the notion of **Gain Ratio**

= Gain(S,A)/ SplitEntropy(S,A).

Machine Learning

Decision Tree Classification

In this lecture:

- **Part 1:** Motivation Example
- **Part 2:** ID3 Algorithm
- **Part 3:** Entropy and Information Gain
- **Part 4: Overfitting and Pruning**
- **Part 5:** Classifying Continuous/Numerical Values
- **Part 6:** Pros and Cons of Decision Trees
- **Part 7:** Using R and Python to learn Decision Trees

Overfitting in Decision Trees

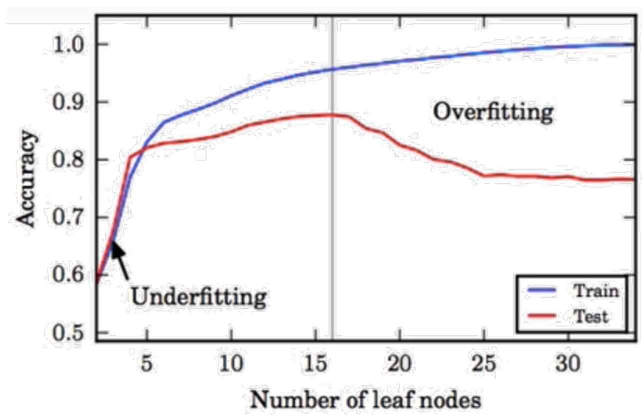
- Can always classify training examples perfectly
  – keep splitting until each node contains 1 example
  – singleton = pure

- The more we split the the higher accuracy, but also the bigger the tree, the more specific decision tree.

- As a result: The decision tree will be too specific and accurate for the training data, but becomes less accurate for new data. Thus, the tree now not be able to classify data that didn't see before.

- In other words: the algorithm becomes too specific to the data we use to train it, and cannot generalize well to new data.

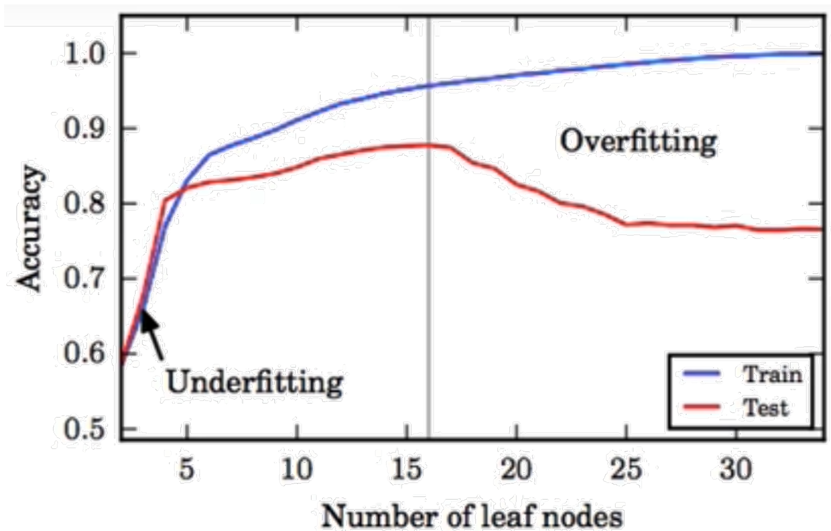
- This is called **overfitting**



Overfitting in Decision Trees

- Overfitting occurs when trying to model the training data perfectly.
- Overfitting means poor generalization.
- The test performance tells us how well our model generalizes, not the training performance

➔ Use Validation Test



Avoid Overfitting – Pruning

Try not to grow a tree that is too large to avoid overfitting.

When to stop growing the tree?

Possible stopping (**pre-pruning**) criteria:

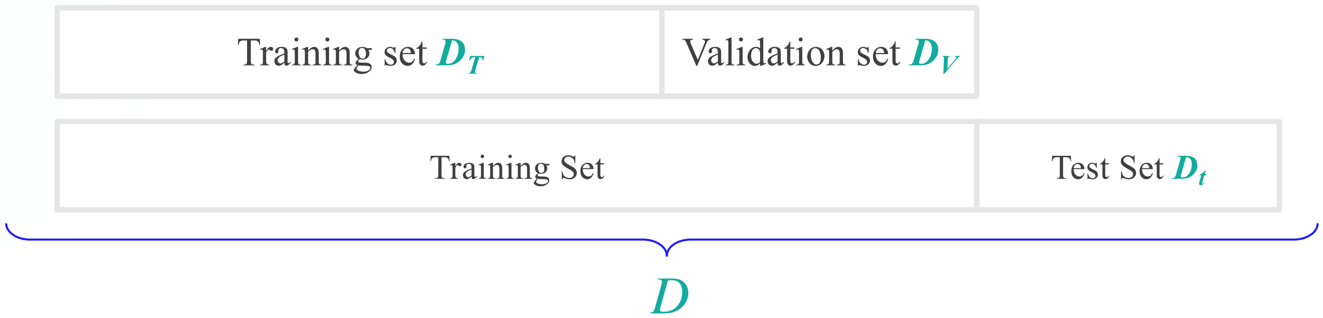
- Maximum depth reached
- Number of samples in each branch below certain threshold
- Benefit of splitting is below certain threshold.

Or we can grow a tree maximally then **post-prune** it.

this require a validation test

Avoid Overfitting – post-pruning

Creating the validation set



Be sure to have separate training, validation, and test sets

- Training set D_T, to build the tree
- Validation set D_v to prune the tree
- Test set D_t to evaluate the final model.

Splitting of (⅔ training ,⅓ testing) are common and best practice.

Testing data for later evaluation should not be used for pruning or you will not get honest estimate of the model’s performance

Avoid Overfitting – post-pruning

Prune the branches that will not do well on the future data

- Use validation set to get an error estimate E_T,
- For each node n in the tree (pretend that all of its descendant nodes are pruned) then calculate the error $E_{T'}$ - as if these nodes were deleted.
- Prune tree at the node that yields the highest error reduction.
- Repeat until further pruning is harmful.

Machine Learning

Decision Tree Classification

In this lecture:

- Part 1: Motivation Example
- Part 2: ID3 Algorithm
- Part 3: Entropy and Information Gain
- Part 4: Overfitting and Pruning
- Part 5: **Classifying Continuous/Numerical Values**
- Part 6: Pros and Cons of Decision Trees
- Part 7: Using R and Python to learn Decision Trees

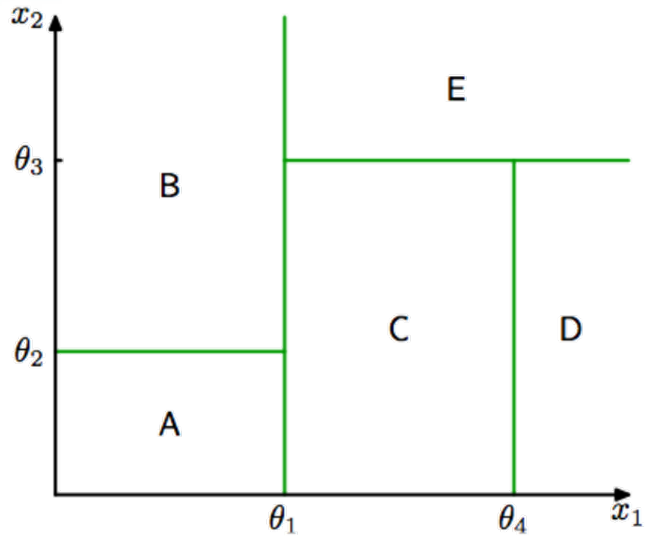
Continuous Attributes

Based on [1]

Dealing with continuous-valued attributes:

create a split: (Temperature > 72.3) = True, False

Threshold can be optimized (WF 6.1)



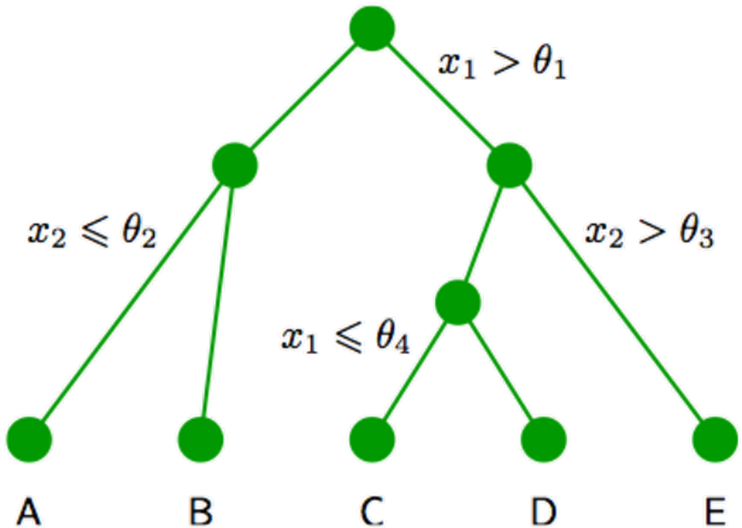


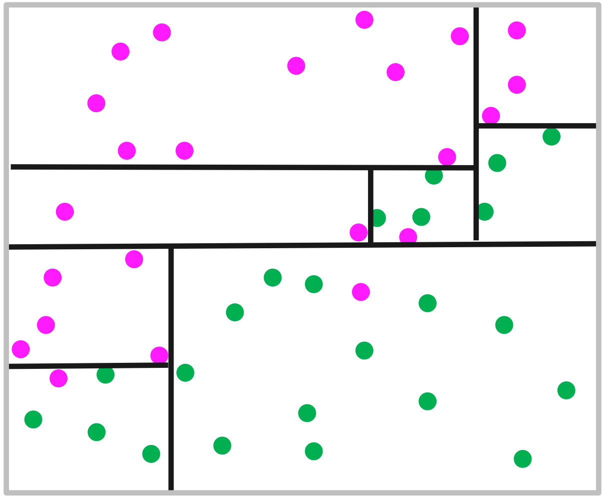
Figure credit: Chris Bishop, PRML

Continuous Attributes

Based on [1]

Classifying Numerical values using Decision Trees
This is like solving "regression problems" using Decision Trees:

- Regression algorithms can draw a boundary line between the data.
- Decision Trees are able to only make axis-aligned splits of data. (only vertical and horizontal lines)
- Decision Trees introduces a threshold for each axis individually.
- But if keep introducing axis-aligned splits (the tree becomes bigger) and we end up overfitting.



Machine Learning

Decision Tree Classification

In this lecture:

- Part 1: Motivation Example
- Part 2: ID3 Algorithm
- Part 3: Entropy and Information Gain
- Part 4: Overfitting and Pruning
- Part 5: Classifying Continuous/Numerical Values
- Part 6: **Pros and Cons of Decision Trees**
- Part 7: Using R and Python to learn Decision Trees

What is good about Decision Trees

- Interpretable: humans can understand decisions
- Easily handles irrelevant attributes (G=0)
- Very compact: #nodes << D after pruning
- Very fast at testing time: O(Depth)

Limitations for Decision Trees

- Greedy (may not find best tree).
- Instances are represented by attribute/value pairs(e., Outlook: sunny, Wind: strong), but what if we have discrete input values.
- The target function has discrete output values (e.g., Yes, No), thus we cannot have continues number output values.
- The training data may contain errors, or missing attributes
- Uncertainty in the data (e.g., suppose we have two exact days/features, one with "yes" and one with "no". ➔ no classifier can help in such totally Uncertain data.

Machine Learning

Decision Tree Classification

In this lecture:

- ❑ **Part 1:** Motivation Example
- ❑ **Part 2:** ID3 Algorithm
- ❑ **Part 3:** Entropy and Information Gain
- ❑ **Part 4:** Overfitting and Pruning
- ❑ **Part 5:** Classifying Continuous/Numerical Values
- ❑ **Part 6:** Pros and Cons of Decision Trees

Part 7: **Using R and Python to learn Decision Trees**

Using R to learn Decision Trees

Input.csv

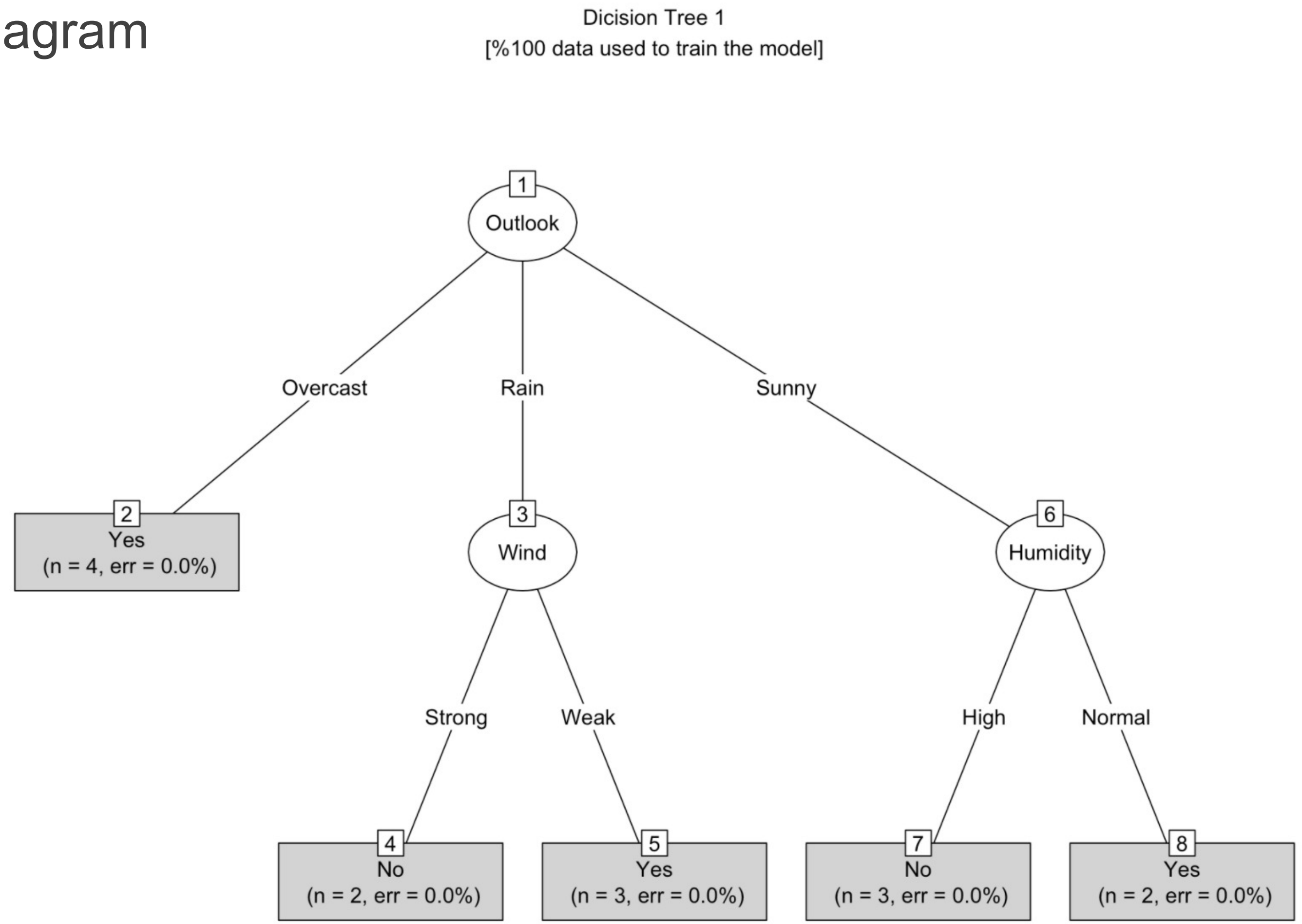
```
Day,Outlook,Humidity,Wind,Play
D1,Sunny,High,Weak,No
D2,Sunny,High,Strong,No
D3,Overcast,High,Weak,Yes
D4,Rain,High,Weak,Yes
D5,Rain,Normal,Weak,Yes
D6,Rain,Normal,Strong,No
D7,Overcast,Normal,Strong,Yes
D8,Sunny,High,Weak,No
D9,Sunny,Normal,Weak,Yes
D10,Rain,Normal,Weak,Yes
D11,Sunny,Normal,Strong,Yes
D12,Overcast,High,Strong,Yes
D13,Overcast,Normal,Weak,Yes
D14,Rain,High,Strong,No
```

DT_example.R

```
require(C50) # the package that has the C5.0 decision tree
require(gmodels)  # a package used draw diagrams and graphs

print("Choose the data file when prompted")
dataset = read.table(file.choose(), header = T, sep=",")
# to exclude the DayNo column (column #1)
dataset = dataset[,-1]

# apply the decision tree algorithm to the training data feature columns,
and class column (output), and generate a DT Model.
model = C5.0(dataset[, -4], dataset[, 4])

# we plot the diagram of the generated decision tree
plot(model, type="s", main="Decision Tree 1\n[%100 data used to train the
model]")
```

Using R to learn Decision Trees

Output Diagram



Using Python to learn Decision Trees

by Jnan Morrar

Colab Link https://colab.research.google.com/drive/1Og4gi5ifdqVLdv5QEY9GoSvo-A3eLCdy?usp=sharing

```
# Mount Drive
from google.colab import drive
drive.mount("/content/drive")
```

```
[ ] from sklearn import tree #For our Decision Tree
    import pandas as pd # For our DataFrame
    import pydotplus # To create our Decision Tree Graph
    from IPython.display import Image  # To Display a image of our graph
```

```
# Read the input file.csv
data_inputs = pd.read_csv("/content/drive/My Drive/Ai Projects/Data/DataSheet.csv")
data_inputs.head() # print the first five values
```

	Day	Outlook	Humidity	Wind	Play
0	D1	Sunny	High	Weak	No
1	D2	Sunny	High	Strong	No
2	D3	Overcast	High	Weak	Yes
3	D4	Rain	High	Weak	Yes

Using Python to learn Decision Trees

by Jnan Morrar

Colab Link https://colab.research.google.com/drive/1Og4gi5ifdqVLdv5QEY9GoSvo-A3eLCdy?usp=sharing

Remove the day column from the input file

```
data_inputs = data_inputs.drop(["Day"] , axis = 1)
data_inputs.head()
```

	Outlook	Humidity	Wind	Play
0	Sunny	High	Weak	No
1	Sunny	High	Strong	No
2	Overcast	High	Weak	Yes
3	Rain	High	Weak	Yes
4	Rain	Normal	Weak	Yes

convert the categorical variables (Outlook, Humidity, Wind) into dummy/indicator variables or (binary variables) essentially 1's and 0's

```
[ ] dummy_variables = pd.get_dummies(data_inputs[ ['Outlook', 'Humidity', 'Wind'] ])
    dummy_variables.head()
```

	Outlook_Overcast	Outlook_Rain	Outlook_Sunny	Humidity_High	Humidity_Normal	Wind_Strong	Wind_Weak	Wind_strong
0	0	0	1	1	0	0	1	0
1	0	0	1	1	0	1	0	0
2	1	0	0	1	0	0	1	0
3	0	1	0	1	0	0	1	0

Using Python to learn Decision Trees

by Jnan Morrar

Colab Link https://colab.research.google.com/drive/1Og4gi5ifdqVLdv5QEY9GoSvo-A3eLCdy?usp=sharing

Remove the day column from the input file

```
data_inputs = data_inputs.drop(["Day"] , axis = 1)
data_inputs.head()
```

	Outlook	Humidity	Wind	Play
0	Sunny	High	Weak	No
1	Sunny	High	Strong	No
2	Overcast	High	Weak	Yes
3	Rain	High	Weak	Yes
4	Rain	Normal	Weak	Yes

convert the categorical variables (Outlook, Humidity, Wind) into dummy/indicator variables or (binary variables) essentially 1's and 0's

```
[ ] dummy_variables = pd.get_dummies(data_inputs[ ['Outlook', 'Humidity', 'Wind'] ])
dummy_variables.head()
```

	Outlook_Overcast	Outlook_Rain	Outlook_Sunny	Humidity_High	Humidity_Normal	Wind_Strong	Wind_Weak	Wind_strong
0	0	0	1	1	0	0	1	0
1	0	0	1	1	0	1	0	0
2	1	0	0	1	0	0	1	0
3	0	1	0	1	0	0	1	0

Using Python to learn Decision Trees

by Jnan Morrar

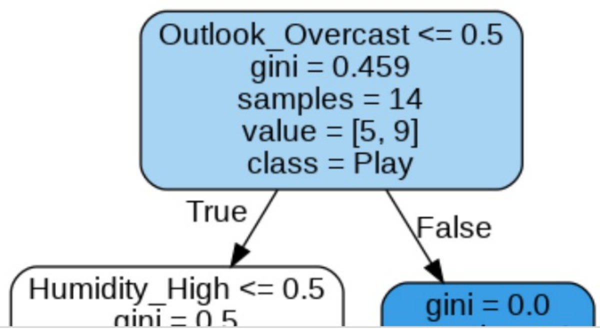
Colab Link https://colab.research.google.com/drive/1Og4gi5ifdqVLdv5QEY9GoSvo-A3eLCdy?usp=sharing

The decision tree classifier & Training the Decision Tree

```
[ ] clf = tree.DecisionTreeClassifier()
    clf_train = clf.fit(dummy_variables, data_inputs['Play']) # Training the Decision Tree
```

Double-click (or enter) to edit

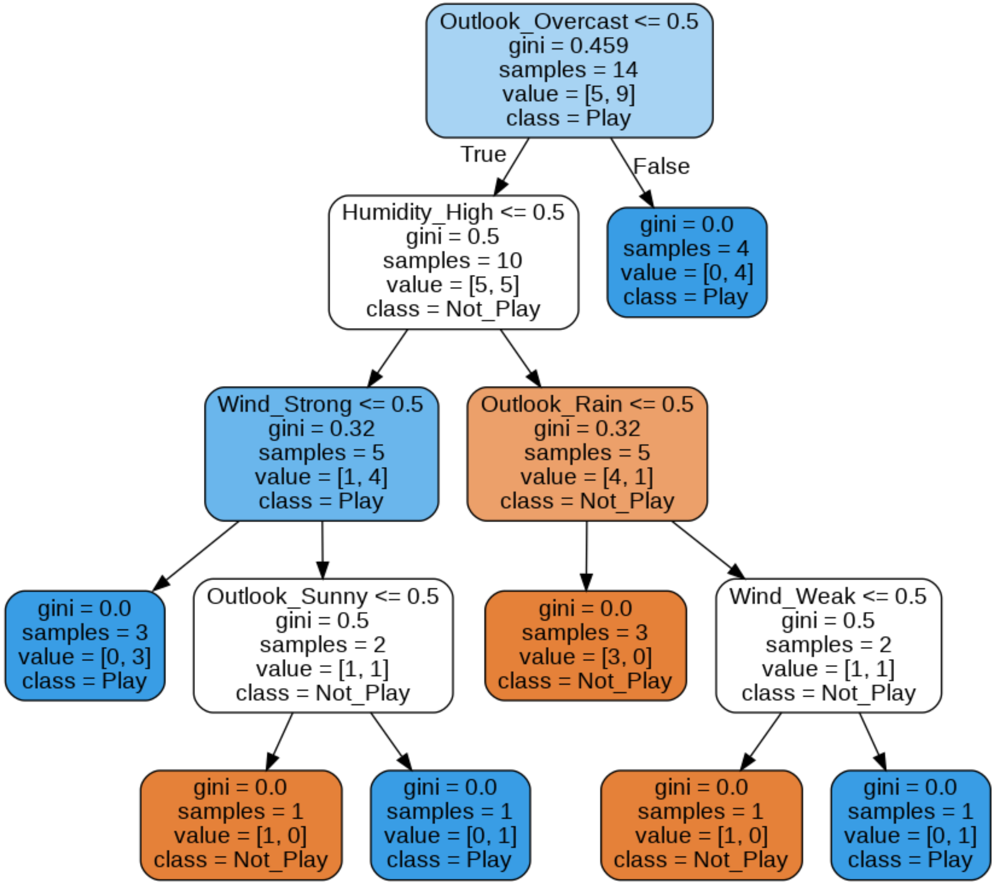
```
#Create Dot Data
dot_data = tree.export_graphviz(clf_train, out_file=None, feature_names=list(dummy_variables.columns.values),
                                class_names=['Not_Play', 'Play'], rounded=True, filled=True)
#Create Graph from DOT data
graph = pydotplus.graph_from_dot_data(dot_data)
# Show graph
Image(graph.create_png())
```



Using Python to learn Decision Trees

by Jnan Morrar

Colab Link https://colab.research.google.com/drive/1Og4gi5ifdqVLdv5QEY9GoSvo-A3eLCdy?usp=sharing



References

[1] Victor Lavrenko, Charles Su;on: IAML: Decision Trees Lecture Notes 2011

[2] Francisco Lacobelli: Lecture Notes on Decision Trees, 2016

[3] Sami Ghawi, Mustafa Jarrar: Lecture Notes on Introduction to Machine Learning, Birzeit University, 2018

[4] Mustafa Jarrar: Lecture Notes on Decision Trees Machine Learning, Birzeit University, 2018

[5] Mustafa Jarrar: Lecture Notes on Linear Regression Machine Learning, Birzeit University, 2018