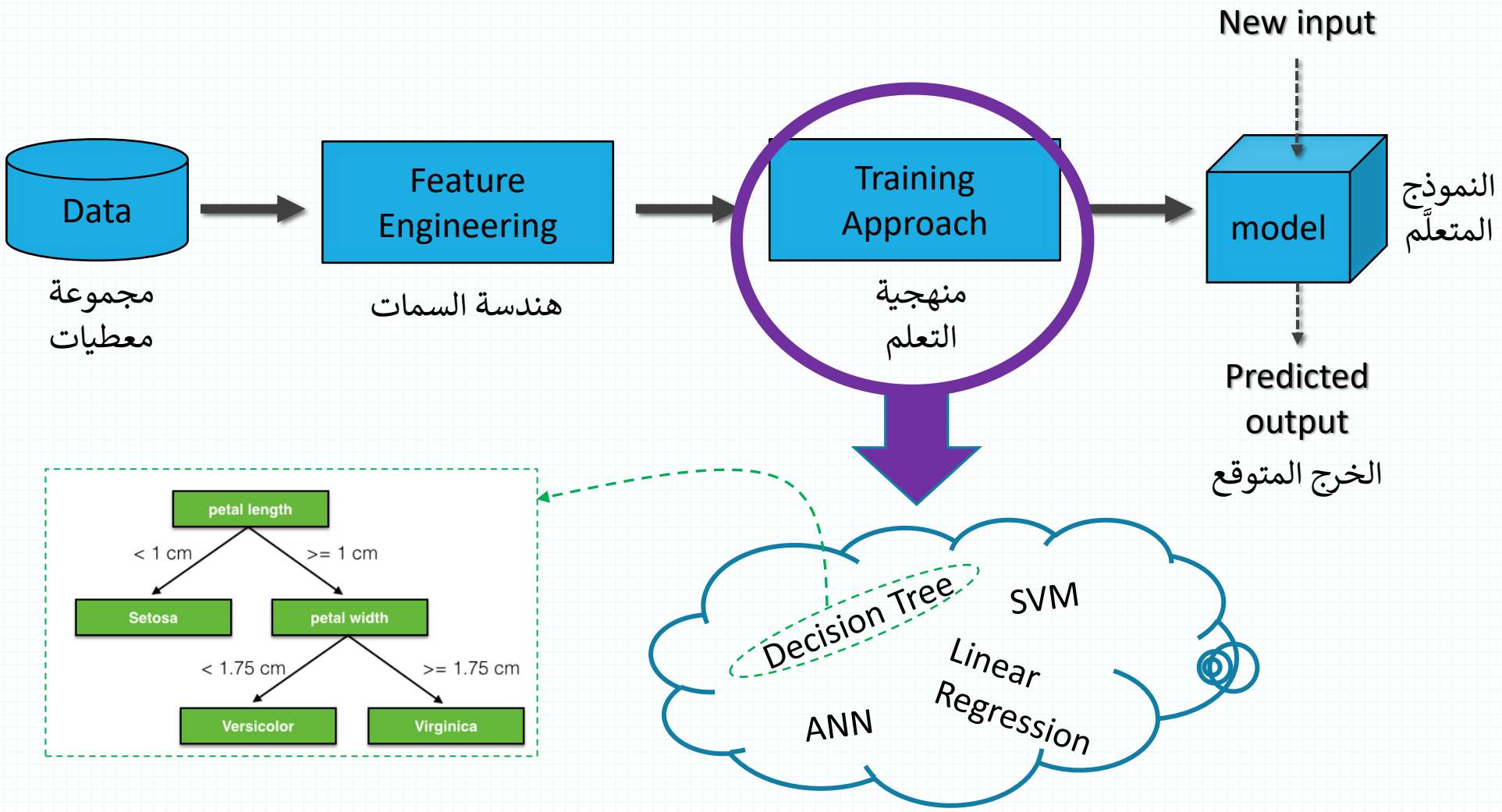
مقرر التعلم التلقائي | السنة الخامسة – هندسة المعلوماتية / الذكاء الصنعي | Week 2

أشجار القرار
Decision Tree

د. رياض سنبل

Access Course Materials

Traditional ML Pipeline

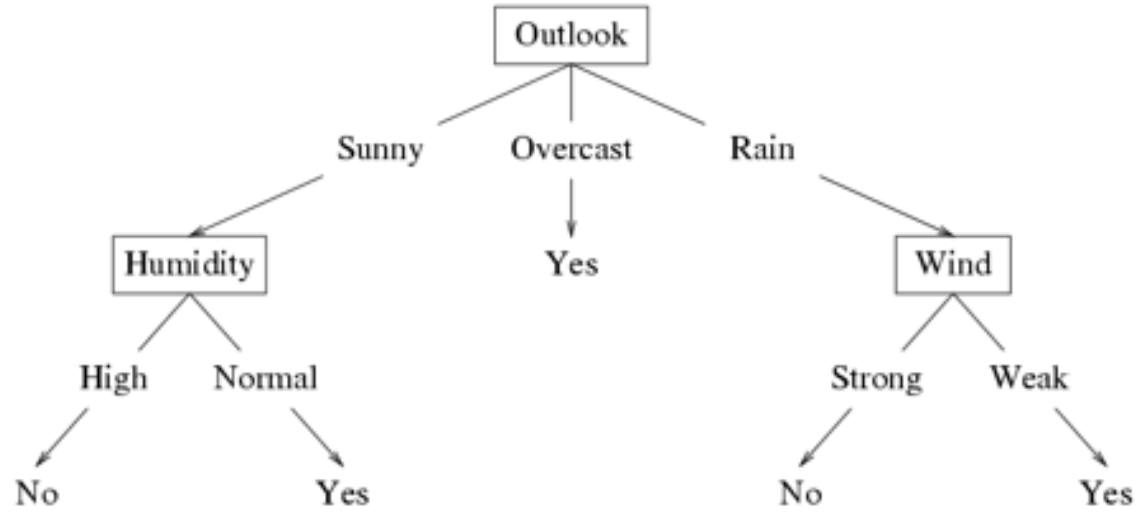


Example: Predict When This Player Will Play Tennis?

Day	Outlook	Temp.	Humidity	Wind	Play Tennis
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	Weak	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Strong	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

Decision Tree Hypothesis Space

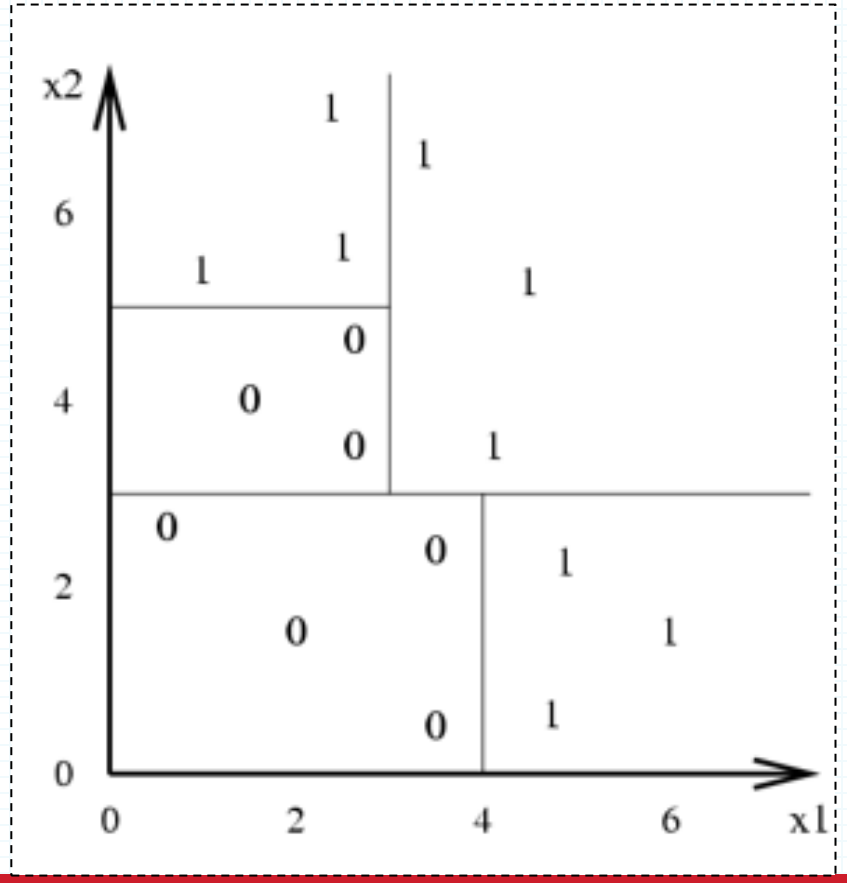
- Internal nodes test the value of particular features x_i and branch according to the results of the test.
- Leaf nodes specify the class.

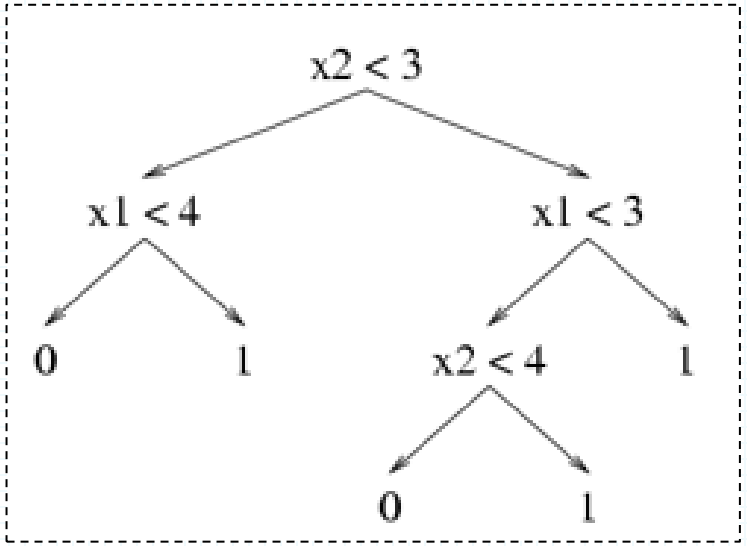


Suppose the features are **Outlook** (x_1), **Temperature** (x_2), **Humidity** (x_3), and **Wind** (x_4). Then the feature vector $\mathbf{x} = (Sunny, Hot, High, Strong)$ will be classified as **No**. The **Temperature** feature is irrelevant.

Decision Tree Decision Boundaries

- Decision trees divide the feature space into axis-parallel rectangles, and label each rectangle with one of the K class.





- Advantages:
  - explicit relationship among features
  - human interpretable model
- Limitations?

Learning Algorithm for Decision Trees

$$S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$$

$$\mathbf{x} = (x_1, \ldots, x_d)$$

$$x_j, y \in \{0,1\}$$

GROWTREE(S)

if ($y = 0$ for all $\langle \mathbf{x}, y \rangle \in S$) **return** new leaf(0)

else if ($y = 1$ for all $\langle \mathbf{x}, y \rangle \in S$) **return** new leaf(1)

else

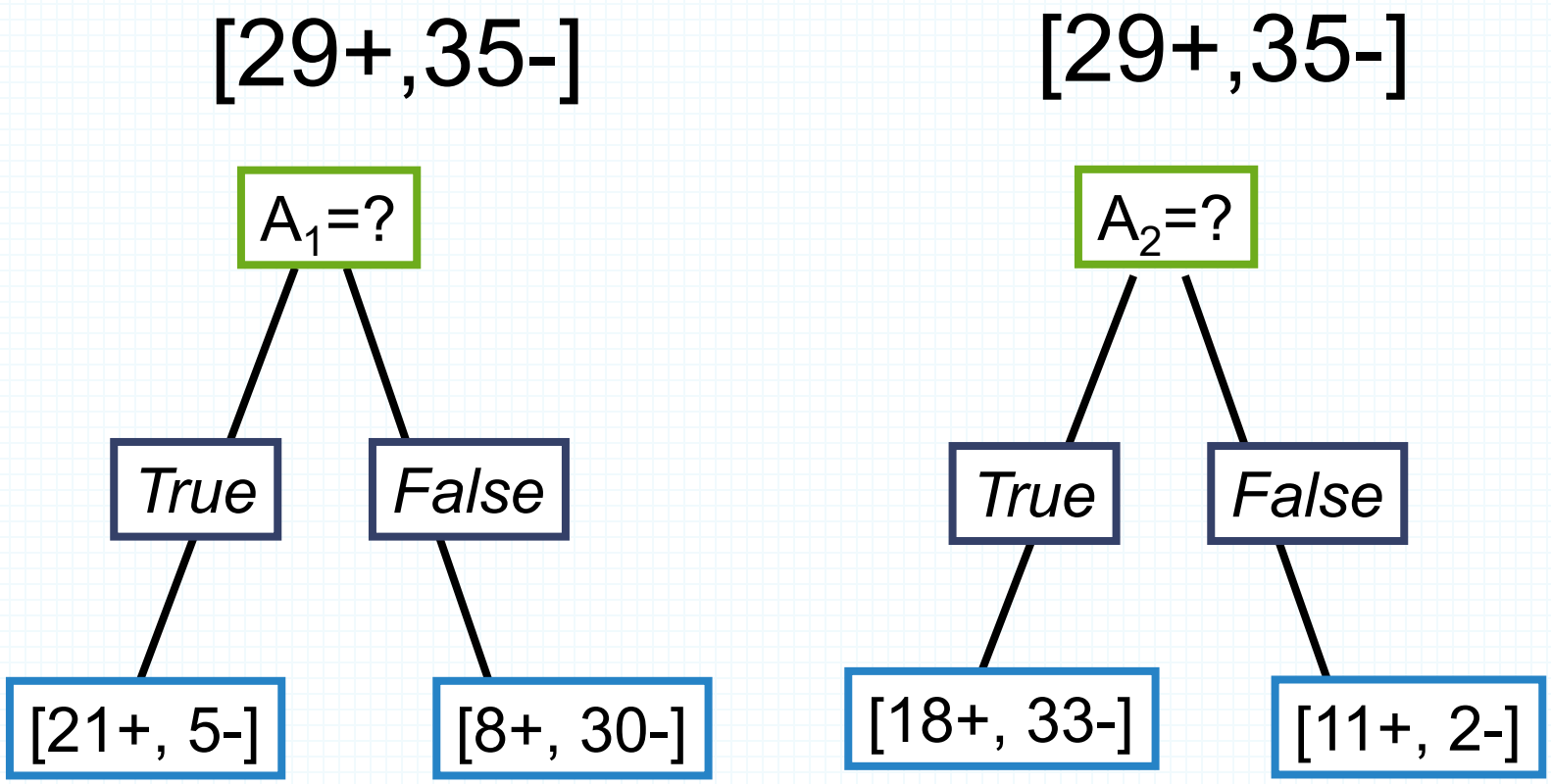
choose best attribute x_j

$S_0 = $ all $\langle \mathbf{x}, y \rangle \in S$ with $x_j = 0$;

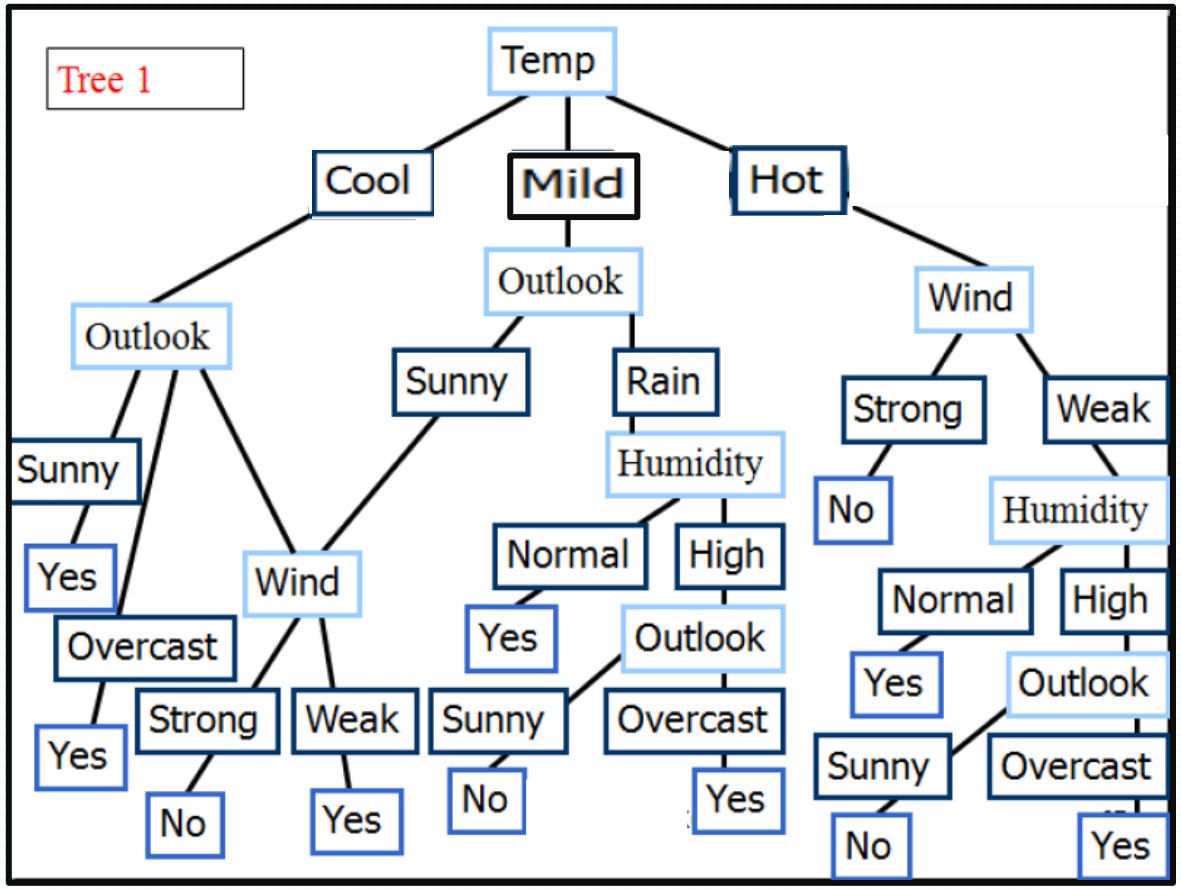
$S_1 = $ all $\langle \mathbf{x}, y \rangle \in S$ with $x_j = 1$;

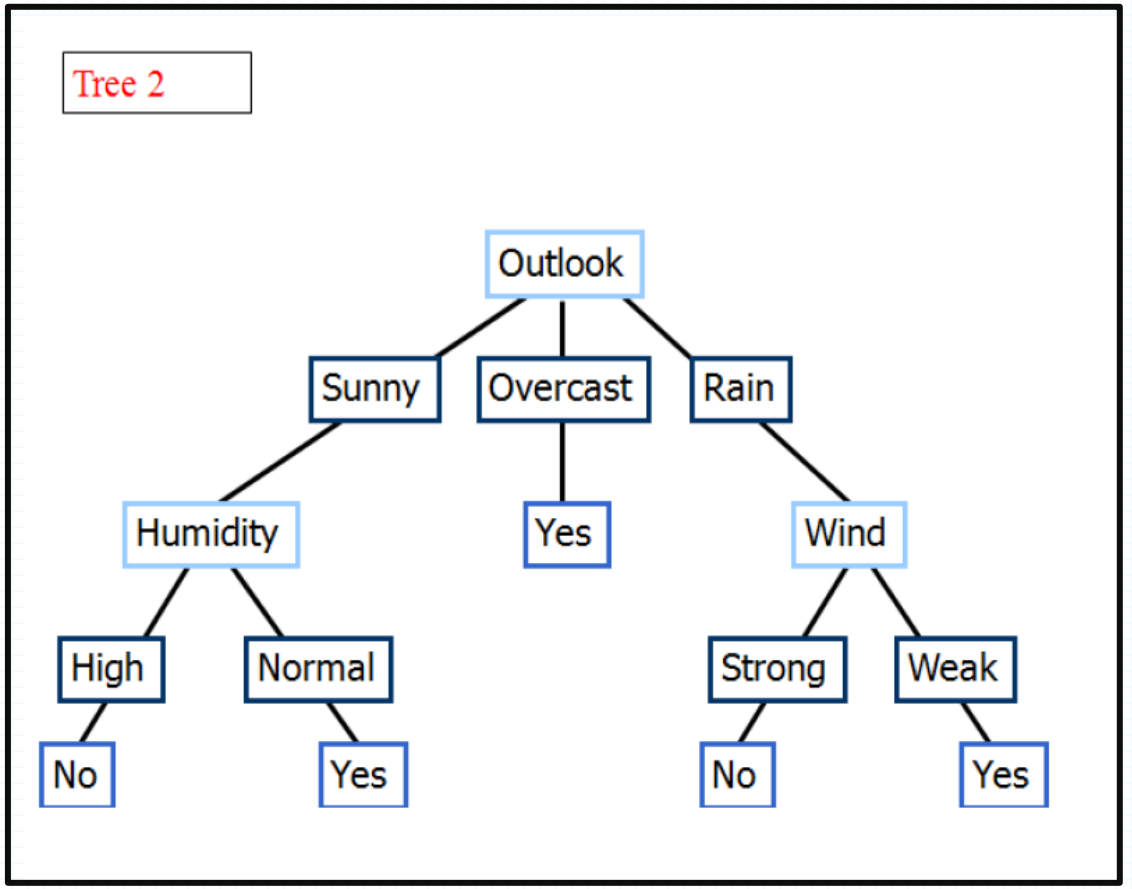
return new node(x_j, GROWTREE(S_0), GROWTREE(S_1))

Which feature is "best"?



Which feature is "best"?





Using Absolute error

ChooseBestfeature(S)

choose **j** to minimize $\mathbf{J_j}$, computed as follows:

$\mathbf{S_0}$ = all $\langle \mathbf{x}, \mathbf{y} \rangle \in \mathbf{S}$ with $\mathbf{x_j = 0}$

$\mathbf{S_1}$ = all $\langle \mathbf{x}, \mathbf{y} \rangle \in \mathbf{S}$ with $\mathbf{x_j = 1}$

$\mathbf{y_0}$ = the most common value of **y** in $\mathbf{S_0}$

$\mathbf{y_1}$ = the most common value of **y** in $\mathbf{S_1}$

$\mathbf{J_0}$ = number of examples $\langle \mathbf{x}, \mathbf{y} \rangle \in \mathbf{S_0}$ with $\mathbf{y \neq y_0}$

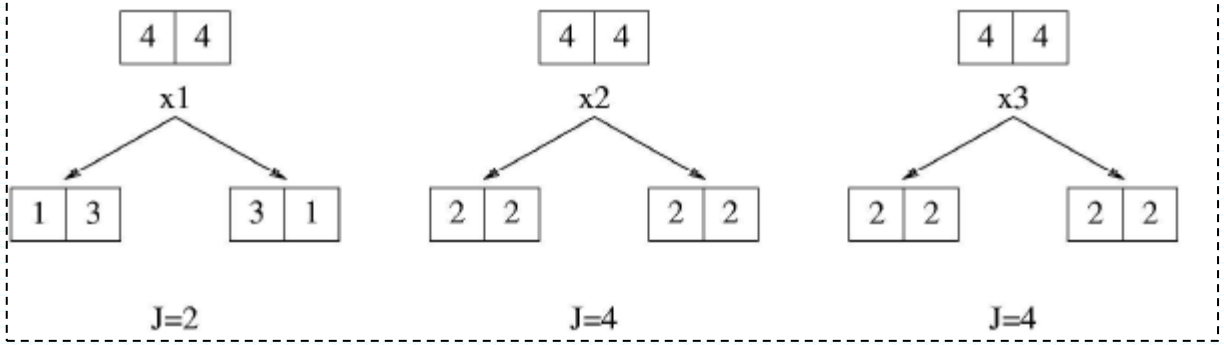
$\mathbf{J_1}$ = number of examples $\langle \mathbf{x}, \mathbf{y} \rangle \in \mathbf{S_1}$ with $\mathbf{y \neq y_1}$

$\mathbf{J_j = J_0 + J_1}$ // total classification errors if we split on feature **j**

return **j**

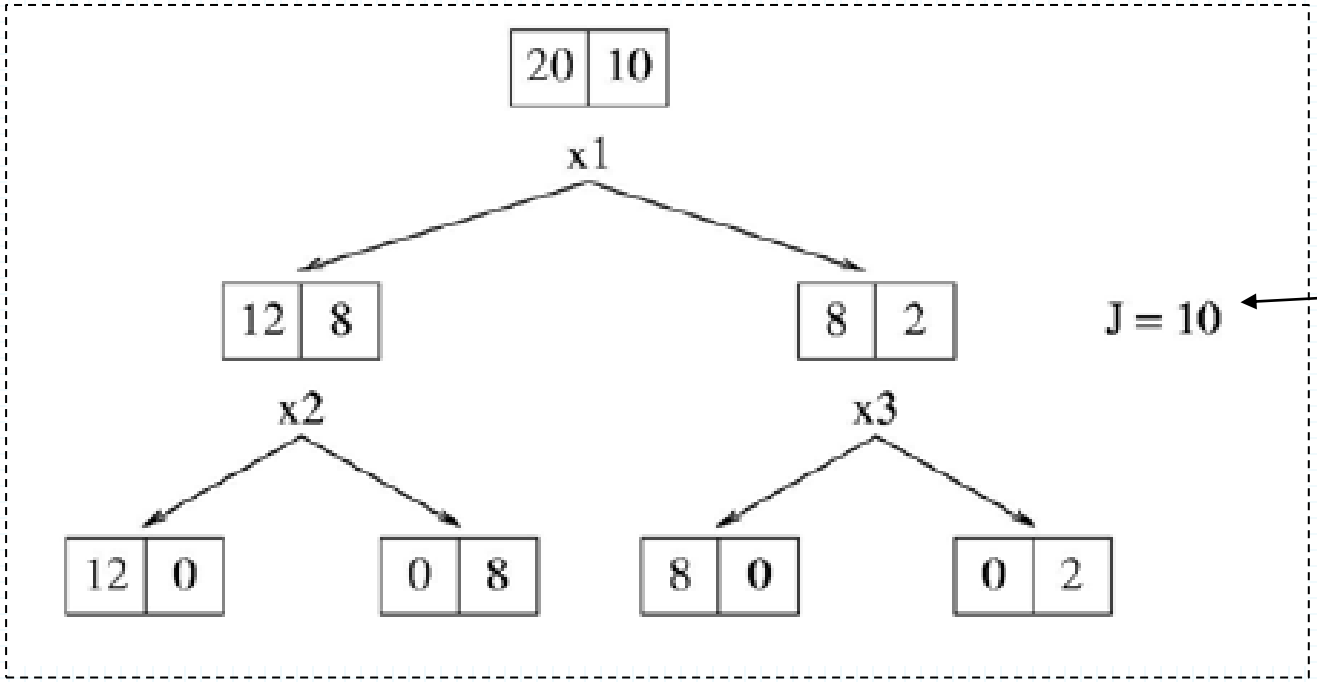
Using Absolute error - Example

x_1	x_2	x_3	y
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0



Using Absolute error - Example

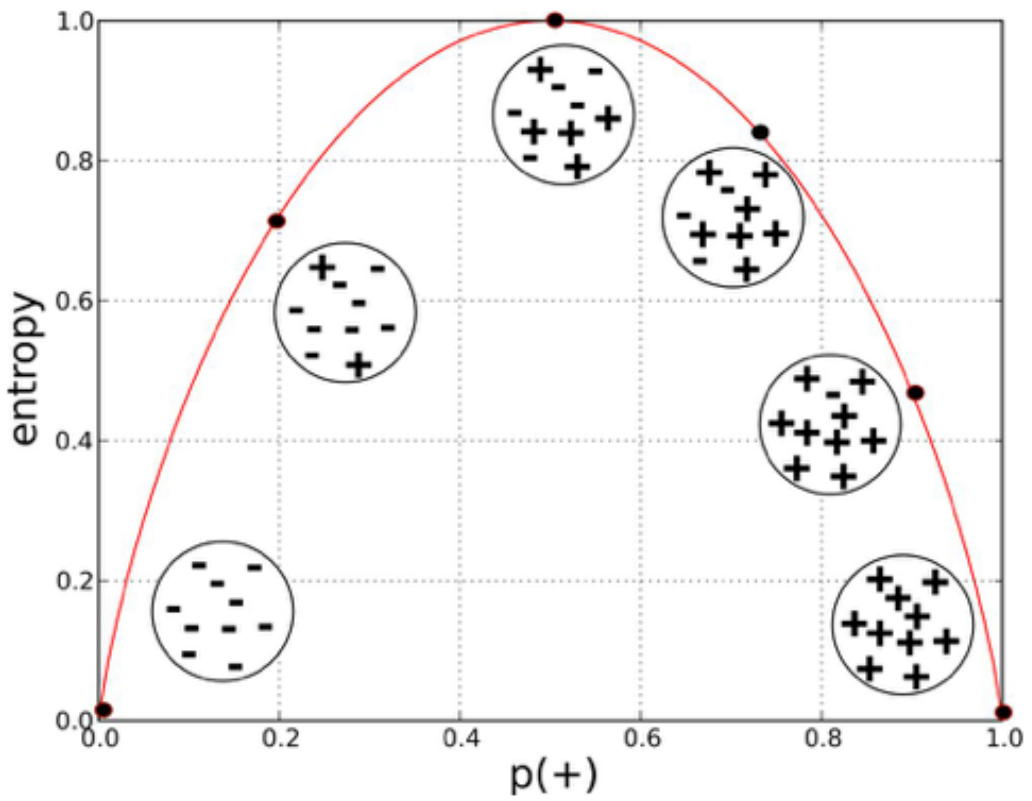
BUT…



Classification error is **not as sensitive to changes in data distribution**, so it might not provide enough signal for deciding where to split next.

Better Criteria?
Entropy!

Entropy



- S is a sample of training examples
- p_+ is the proportion of positive examples
- p_- is the proportion of negative examples
- Entropy measures the impurity of S

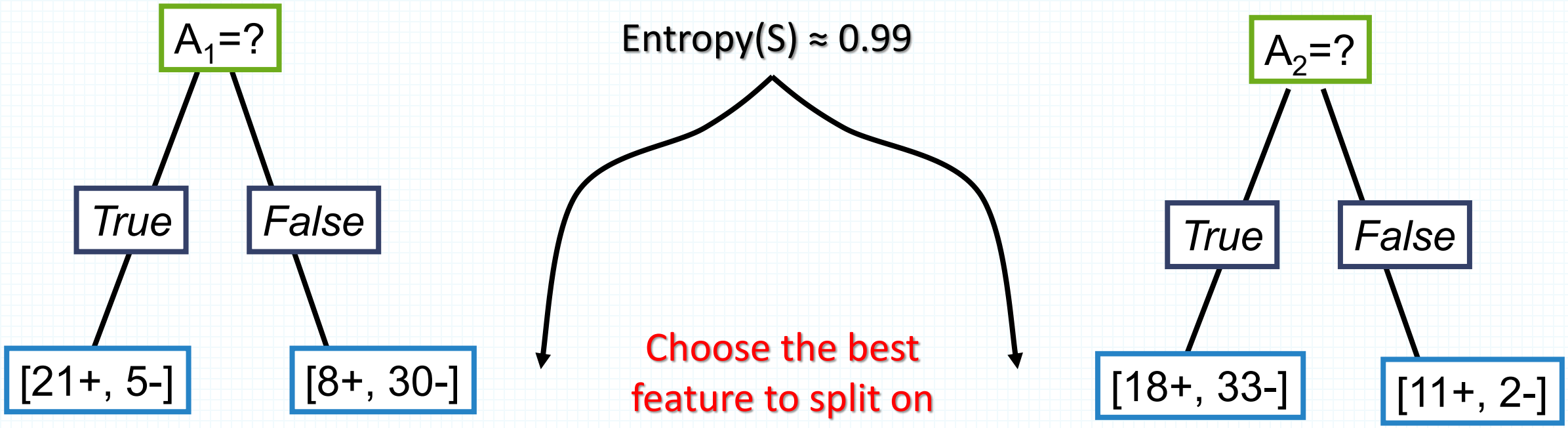
  $Entropy(S) = -p_+ \log_2 p_+ - p_- \log_2 p_-$

Information Gain = Reduction in Entropy

[29+,35-]

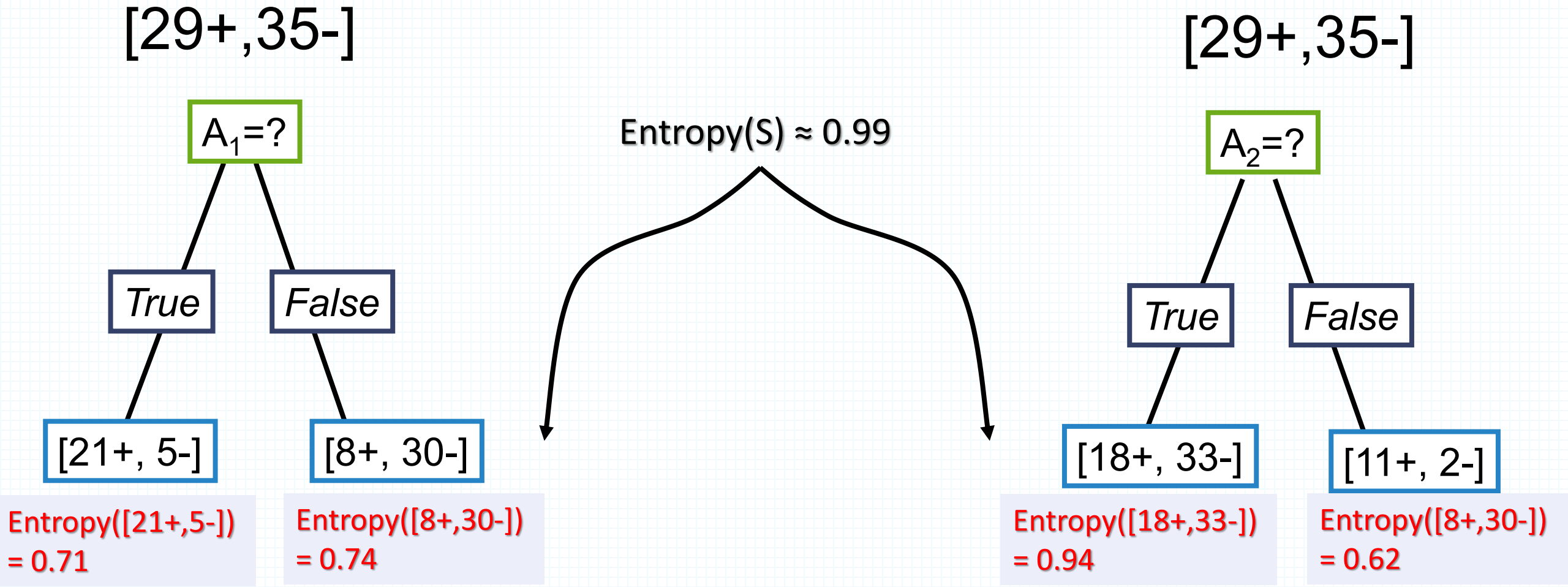
$$\text{Entropy}(S) = -\frac{29}{64}\log_2\left(\frac{29}{64}\right) - \frac{35}{64}\log_2\left(\frac{35}{64}\right)$$

[29+,35-]

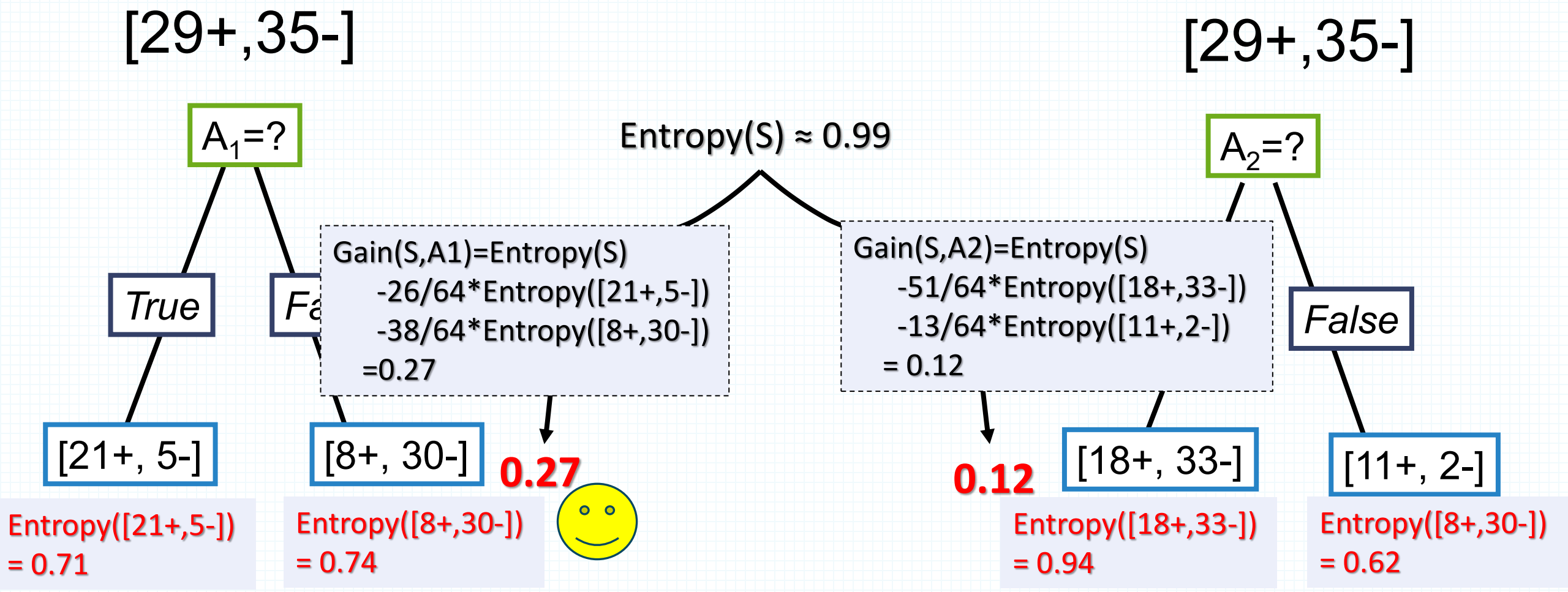


Choose the best feature to split on the one with **highest Information Gain**.

Information Gain = Reduction in Entropy



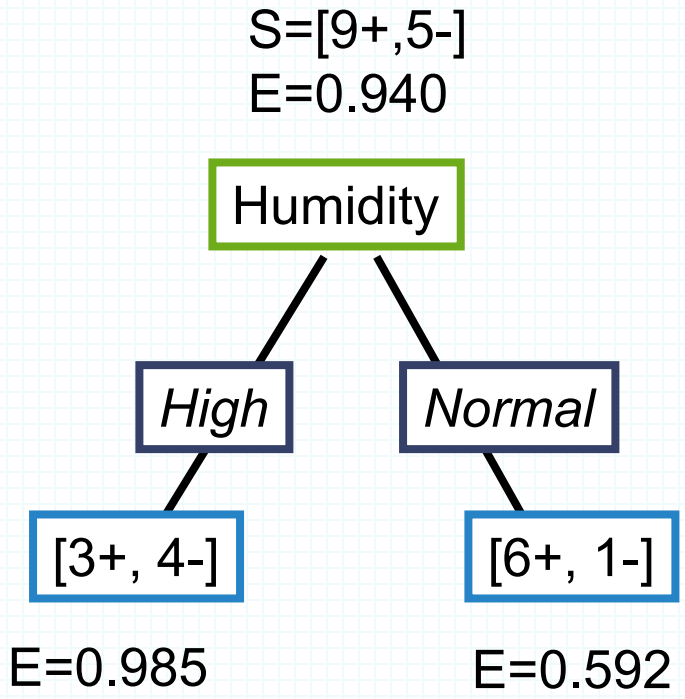
Information Gain = Reduction in Entropy



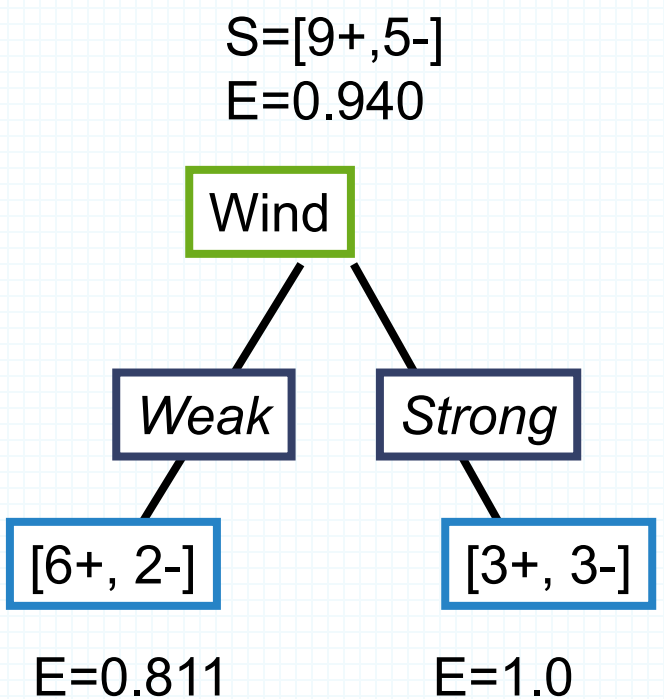
Example

Day	Outlook	Temp.	Humidity	Wind	Play Tennis
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	Weak	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Strong	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

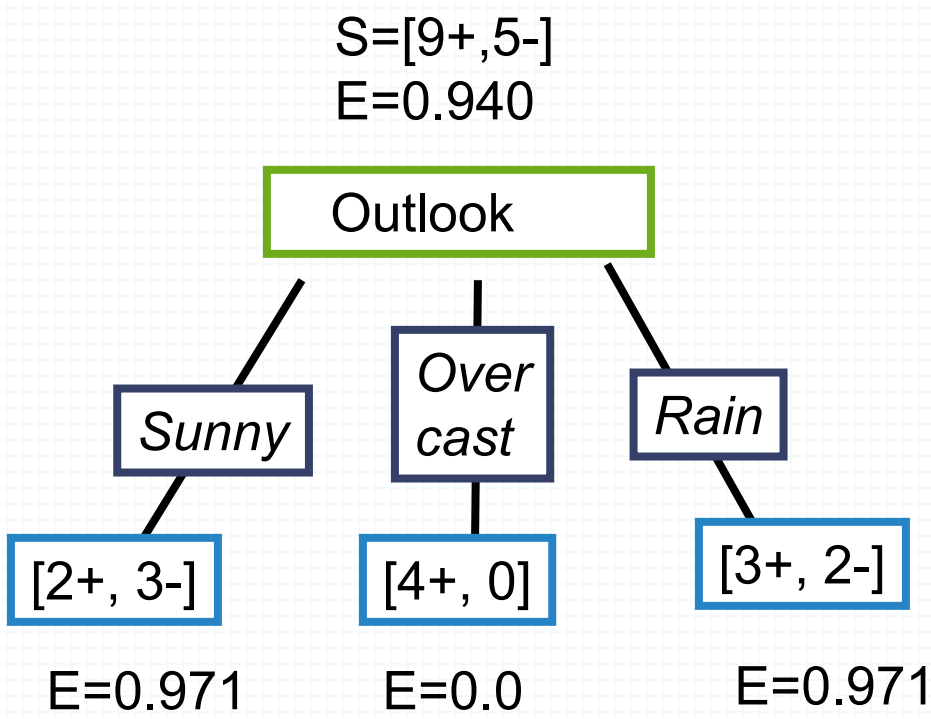
Example: Selecting the Best Feature (ID3 Algorithm)



Gain(S,Humidity)
=0.940-(7/14)*0.985
– (7/14)*0.592
=0.151



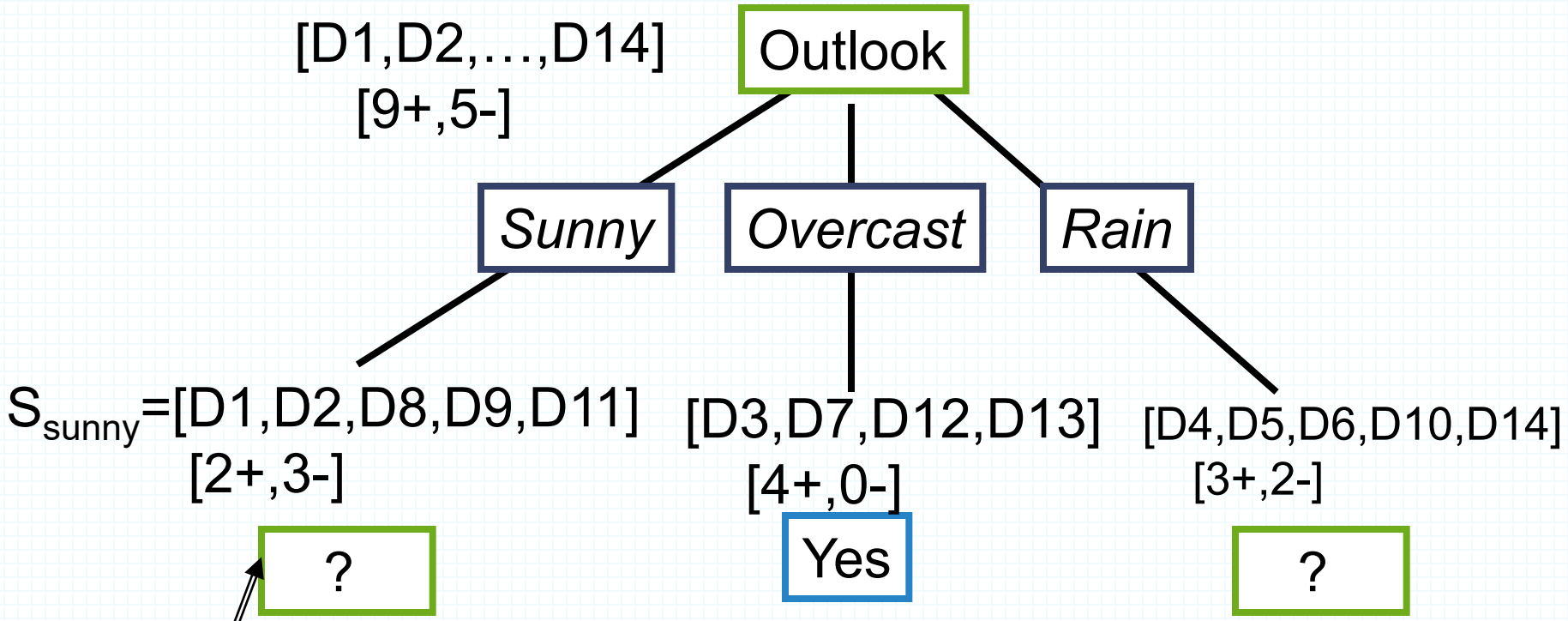
Gain(S,Wind)
=0.940-(8/14)*0.811
– (6/14)*1.0
=0.048



Gain(S,Outlook)
=0.940-(5/14)*0.971
-(4/14)*0.0 – (5/14)*0.0971
=0.247

Example: (ID3 Algorithm)

- Gain(S,Outlook) =0.247
- Gain(S,Humidity) =0.151
- Gain(S,Wind) =0.048
- Gain(S,Temperature) =0.029



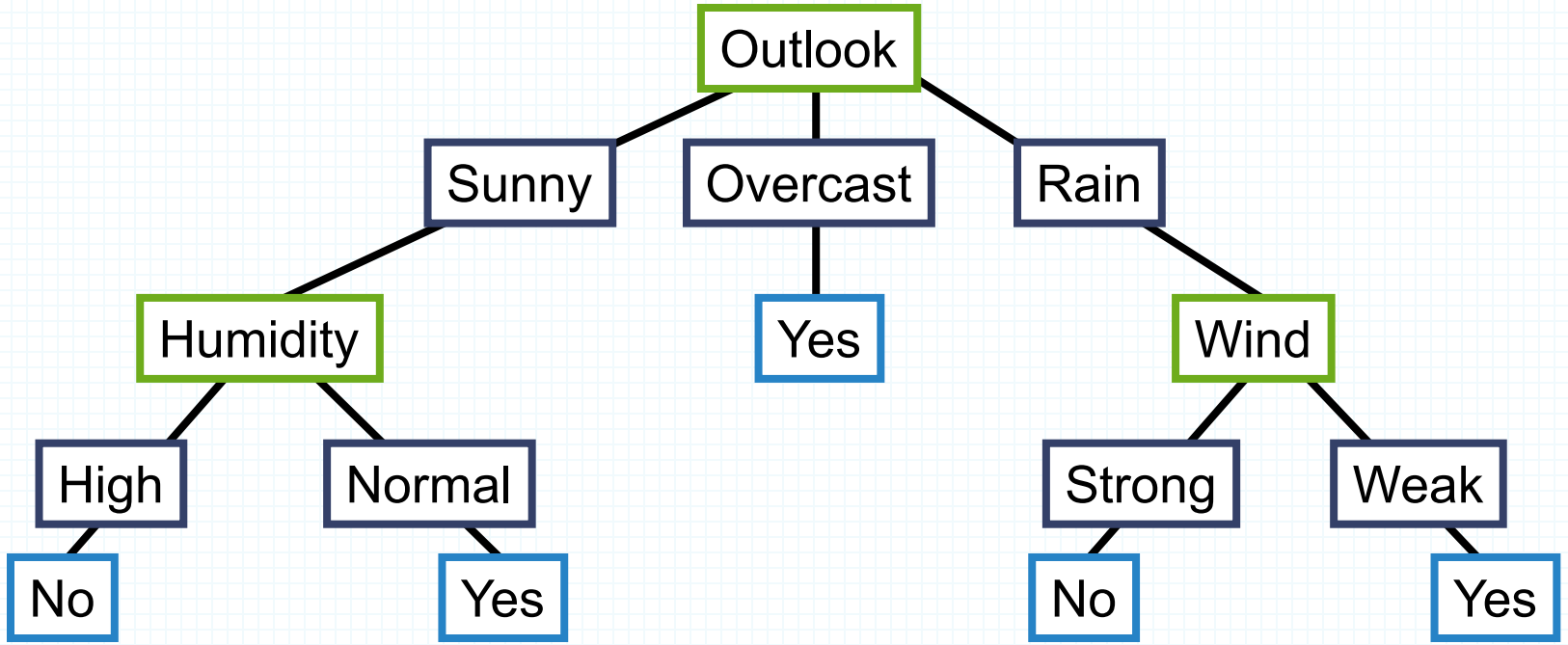
Gain(S_{sunny} , Humidity)=0.970-(3/5)0.0 – 2/5(0.0) = 0.970
Gain(S_{sunny} , Temp.)=0.970-(2/5)0.0 –2/5(1.0)-(1/5)0.0 = 0.570
Gain(S_{sunny} , Wind)=0.970= -(2/5)1.0 – 3/5(0.918) = 0.019

When should we stop

- All samples for a given node belong to the same class.
  - If all instances are "Yes", there's no need to ask more questions.

- There are no remaining features for further partitioning.
  - You ran out of features, but still have both "Yes" and "No" samples → choose the majority class (say, "Yes").
- There are no samples left
  - If a split condition results in an empty group, fill it with the majority class of the parent node.

Converting a Tree to Rules



R_1: If (Outlook=Sunny) $\wedge$ (Humidity=High) Then PlayTennis=No
R_2: If (Outlook=Sunny) $\wedge$ (Humidity=Normal) Then PlayTennis=Yes
R_3: If (Outlook=Overcast) Then PlayTennis=Yes
R_4: If (Outlook=Rain) $\wedge$ (Wind=Strong) Then PlayTennis=No
R_5: If (Outlook=Rain) $\wedge$ (Wind=Weak) Then PlayTennis=Yes

Challenges: (1) Continous features

Convert continous features to disrete intervalls

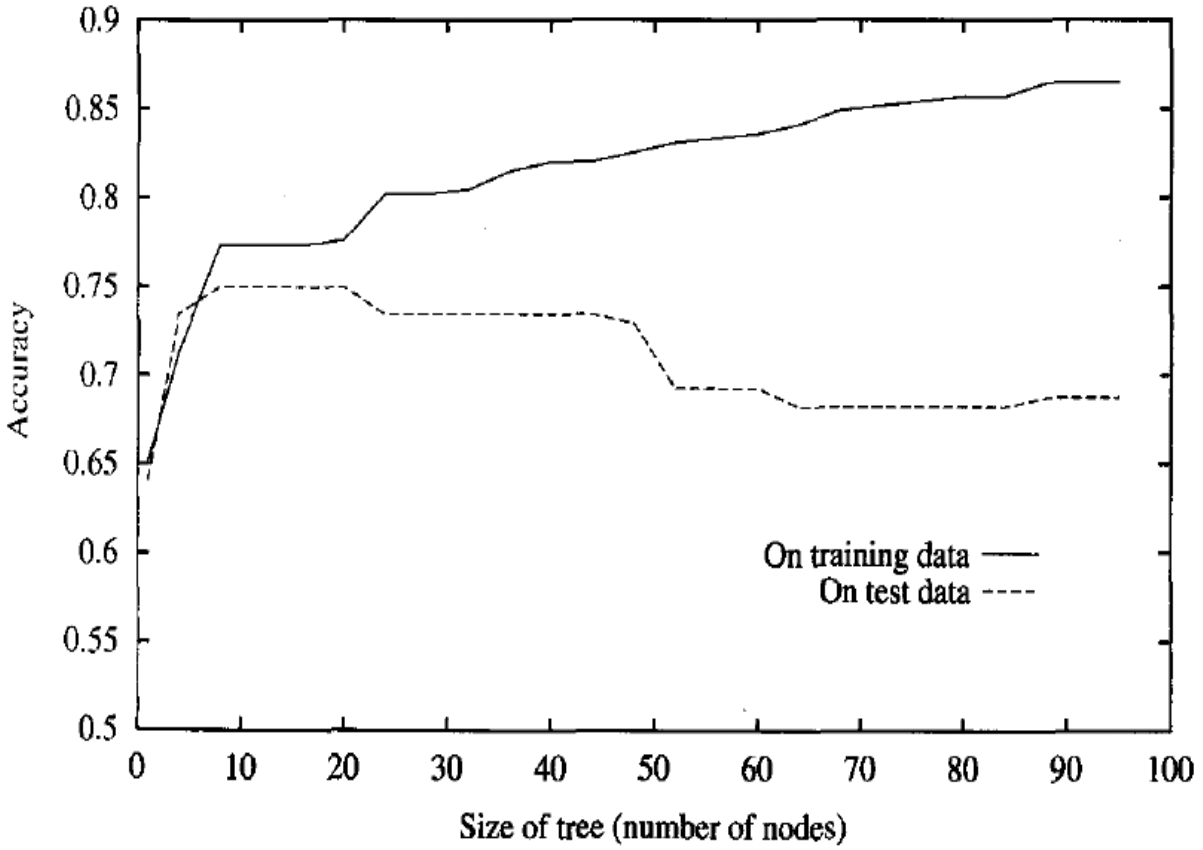
- Temperature=24^{0}C, Temperature =27^{0}C
- (Temperature > 20.0^{0}C) = {true, false}

How to recognise a good threshold?

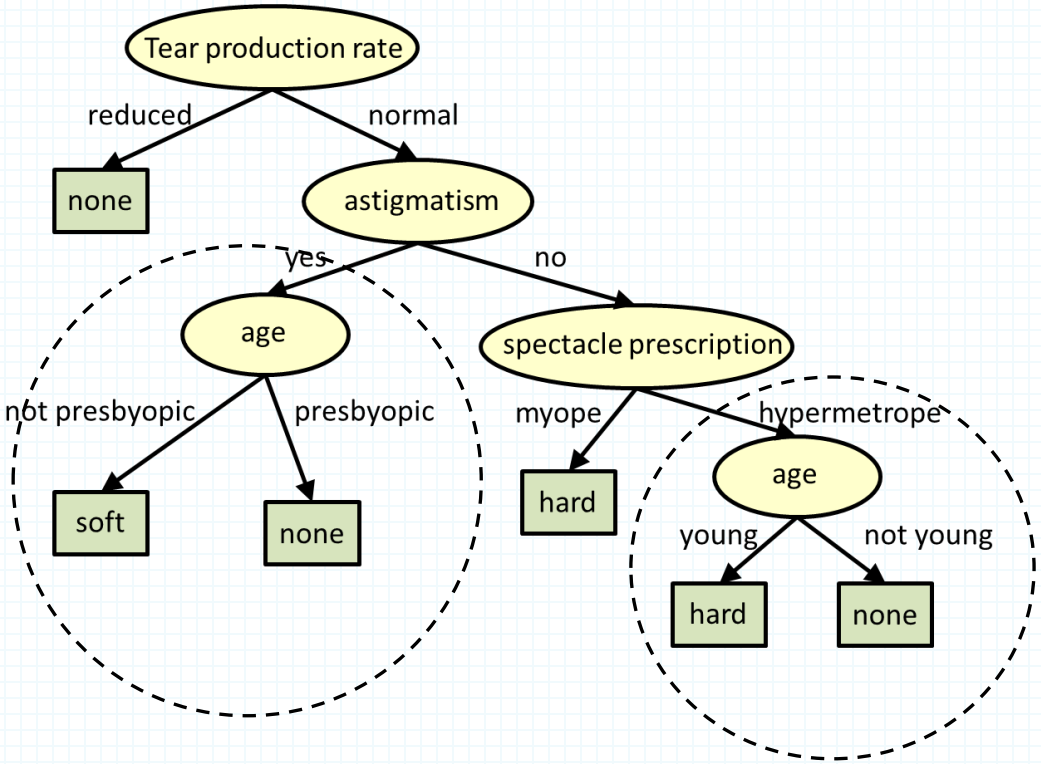
Temperature	15^{0}C	18^{0}C	19^{0}C	22^{0}C	24^{0}C	27^{0}C
Tennis	No	No	Yes	Yes	Yes	No

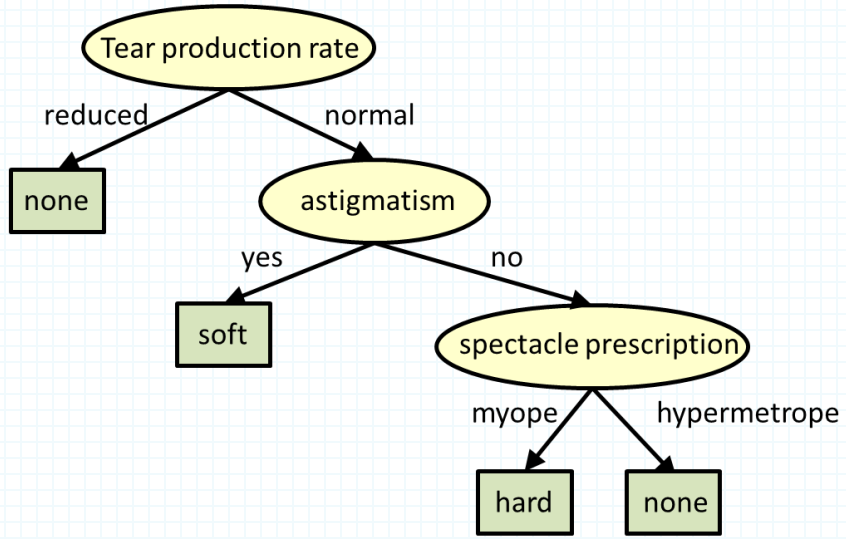
Challenges: (2) Overfitting

- The idea: Stop growing the tree if the improvement is minor
- Pruning Decision Trees involves a set of techniques that can be used to simplify a Decision Tree, and enable it to generalize better (to avoid overtraining).
- Overfitting occurs when the model trains too well and starts learning noise other than the required characteristics



Challenges: (2) Overfitting





Challenges: (2) Overfitting

- Some Possible methods to apply pruning:
  - **Max depth:** controls the maximum depth of the tree.
  - **Min samples split:** controls the minimum number of samples that are required to split an internal node (prevent the tree from creating overfitting by creating branches with very few samples)
  - **Threshold for Class-Related Samples:** Consider setting a threshold for the percentage of samples related to a specific class.
    - Experimentation and evaluation are key to determining the most effective threshold for your particular scenario.

Challenges: (3) Attributes with many values

- Information Gain tends to prefer Attributes with many values.
- It would prefer to split on say the 6-valued feature, over the current 2- or 3-valued attributes.

[10 10 10 10 10 10] [30 30]

- Solution?

$$\text{GainRatio}(S, A) = \frac{\text{Gain}(S, A)}{\text{Split}(S, A)}$$

"Why don't we just divide Information Gain by the number of values the attribute has?"

[10 10 10 10 10 10] [25 25 2 2 3 3] [30 30]

example

Challenges: (3) Attributes with many values

- Solution?

$$\text{GainRatio}(S, A) = \frac{\text{Gain}(S, A)}{\text{Split}(S, A)}$$

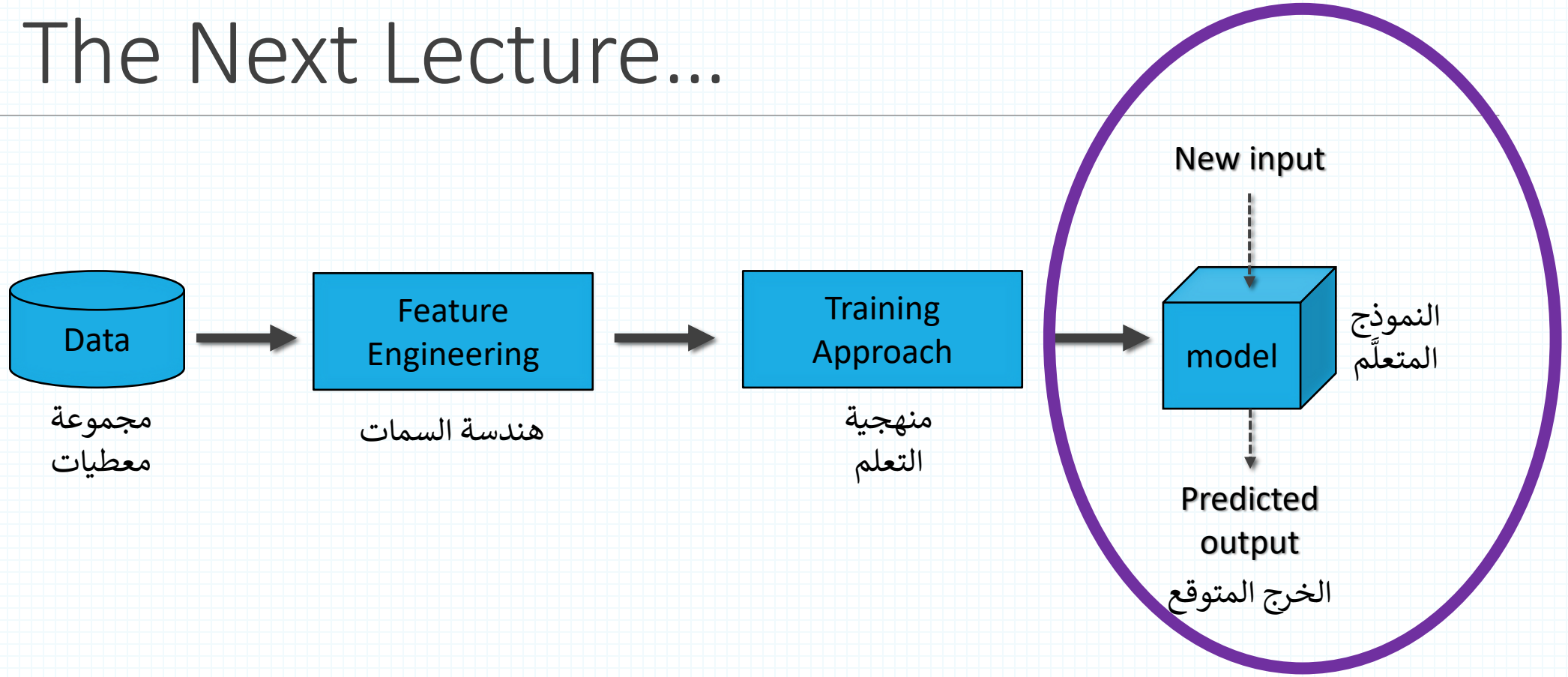
  - Split Information (Split(S, A)) measures **how broadly** and **evenly** the data is split when you use feature **A**.

$$\text{Split}(S, A) = -\sum_{i=1}^{c} \frac{|S_i|}{|S|} \cdot \log_2 \left(\frac{|S_i|}{|S|} \right)$$

Where:

- c: number of distinct values of feature A
- S_i: subset of S where feature A=v_i
- $|S_i|$: number of instances in subset S_i
- |S|: total number of instances

In The Next Lecture…



Estimation Strategies
and
Evaluation Criteria