Department of COMPUTER SCIENCE

WARWICK
THE UNIVERSITY OF WARWICK

Machine Learning
CS342

Lecture 6: Decision Tree Learning & Classification Metrics

Dr. Theo Damoulas
T.Damoulas@warwick.ac.uk

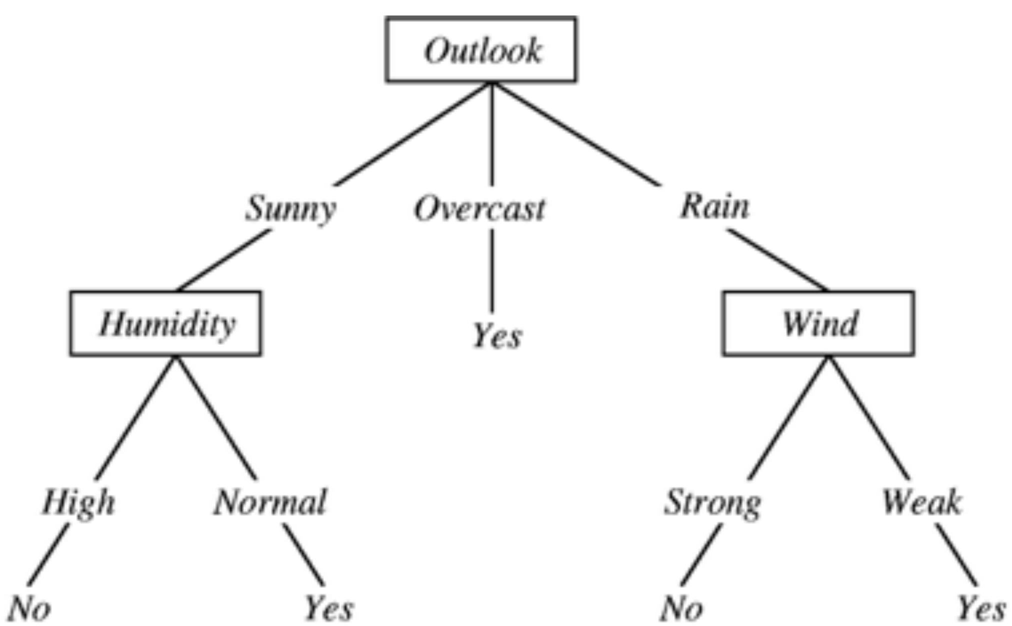
Office hours: Mon & Fri 10-11am @ CS 307

Recap: Decision Tree Learning & ID3

Task: *Classify Saturday mornings as suitable or not for playing tennis*

Nominal data:

Discrete-valued attributes

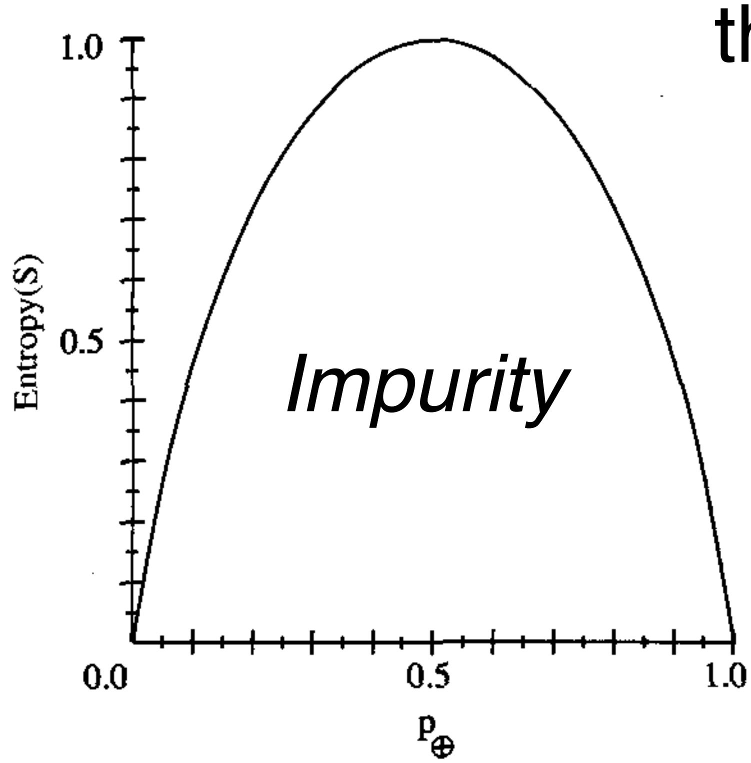


Play Tennis?	Weather Outlook	Humidity	Wind	Temperature
Yes	Sunny	Normal	Weak	Medium
Yes	Overcast	High	Strong	Medium
No	Sunny	High	Weak	Medium
Yes	Overcast	Normal	Weak	Medium
No	Rain	High	Strong	Medium
No	Rain	Normal	Strong	Medium
Yes	Rain	High	Weak	Medium
No	Sunny	High	Weak	Medium
…	…	…	…	…

Recursive greedy top-down algorithm so “best” attribute at top

Recap: Entropy

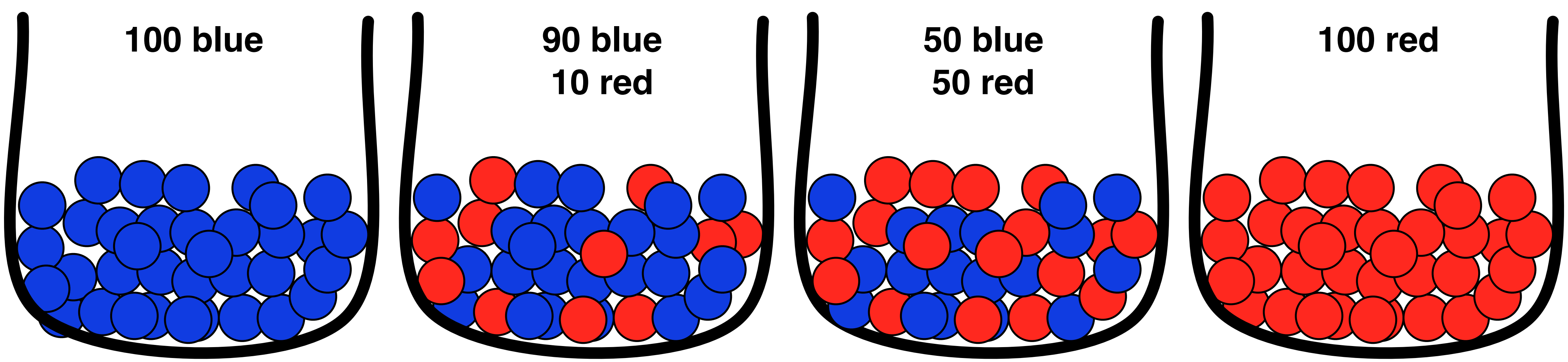
Entropy as a measure of “homogeneity” of a set S of examples that can take one of C values



$$\text{Entropy}(S) = -\sum_{i=1}^{C} p_i \log_2 p_i$$

where p_i frequency of items with type i in set S

Entropy = 0 | Entropy = 0.47 | Entropy = 1 | Entropy = 0

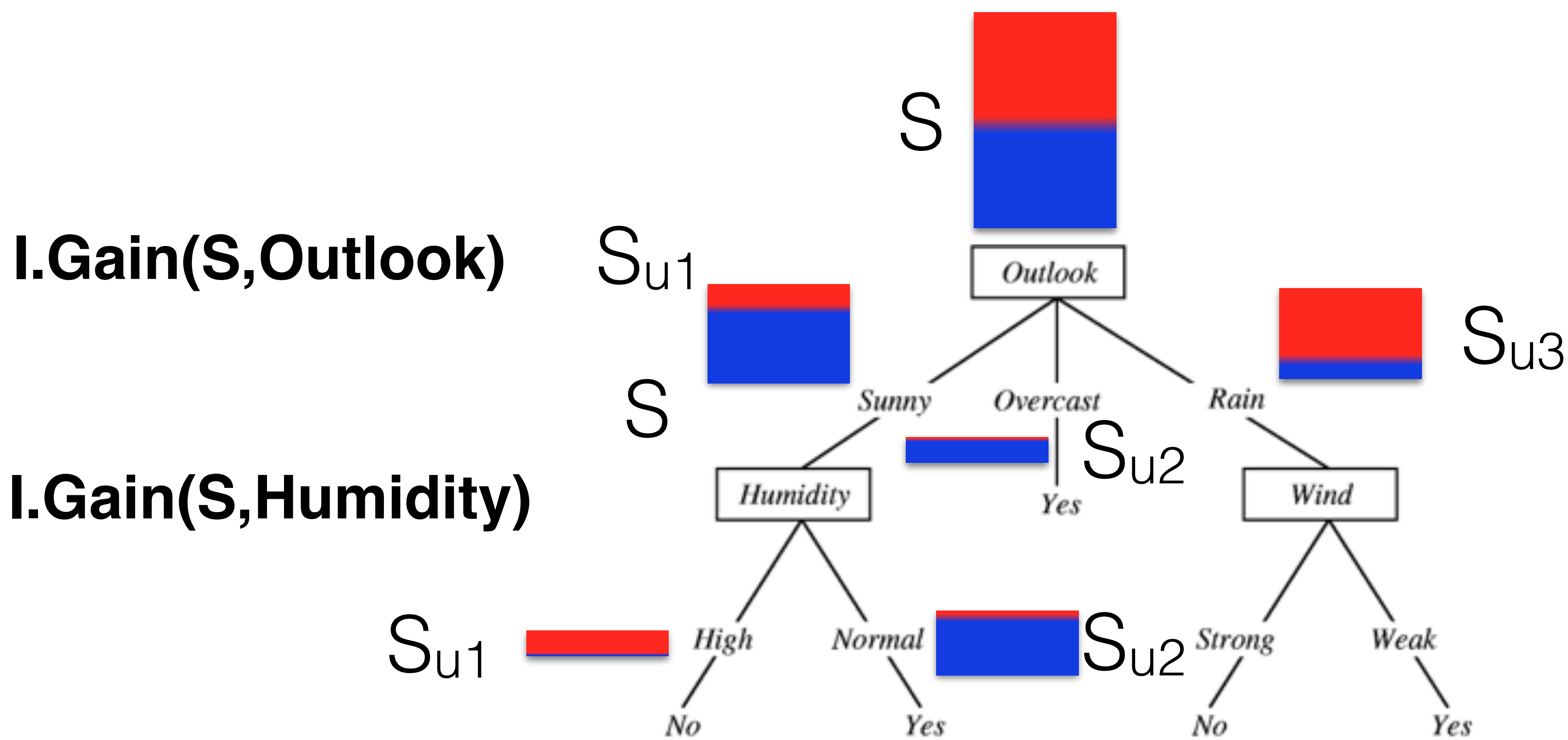


Recap: from Entropy to Information Gain

Information Gain = Relative Entropy = Kullback-Leibler (KL) Divergence

$$\text{I.Gain}(S, A) = \text{Entropy}(S) - \sum_i \frac{|S_{u_i}|}{|S|} \text{Entropy}(S_{u_i})$$

where A is the attribute, u_i is the possible values of A, S_{u_i} is the subset of S where $A = u_i$



Recap: ID3 algorithm

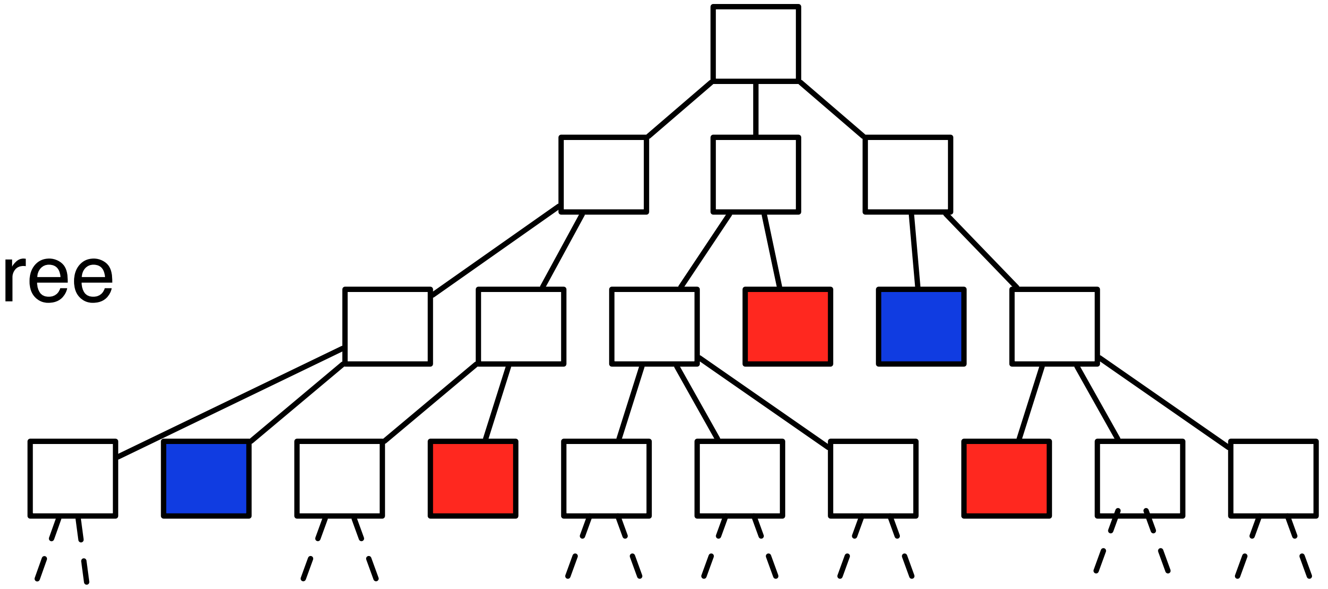
One of the first DT algorithms focused on ***nominal attributes*** and classification

```
ID3(Observations, Targets, Attributes)
• Create a Root node for the tree
• If all observations are class +1, return single-node tree Root with
  label +1
• If all observations are class -1, return single-node tree Root with
  label -1
• If Attributes is empty, return the single-node tree Root with label
  the most common value in Targets
• Else begin:
        • A ⟵ best attribute from Attributes (highest Information Gain)
        • The decision attribute for Root ⟵ A
        • For each possible vale u_i of A:
                • Add a new tree branch below Root for A = u_i
                • S_u_i ⟵ Subset of Observations with A = u_i
                • If S_u_i is empty:
                  • Add leaf node with label the most common value in
                    Targets
                • Else add below branch
                  ID3(S_u_i, Targets, Attributes - {A}) )
```

Recap: Complexity & Overfitting in DTs

So how do we vary the complexity of our DTs?

Model Complexity: Depth of the tree



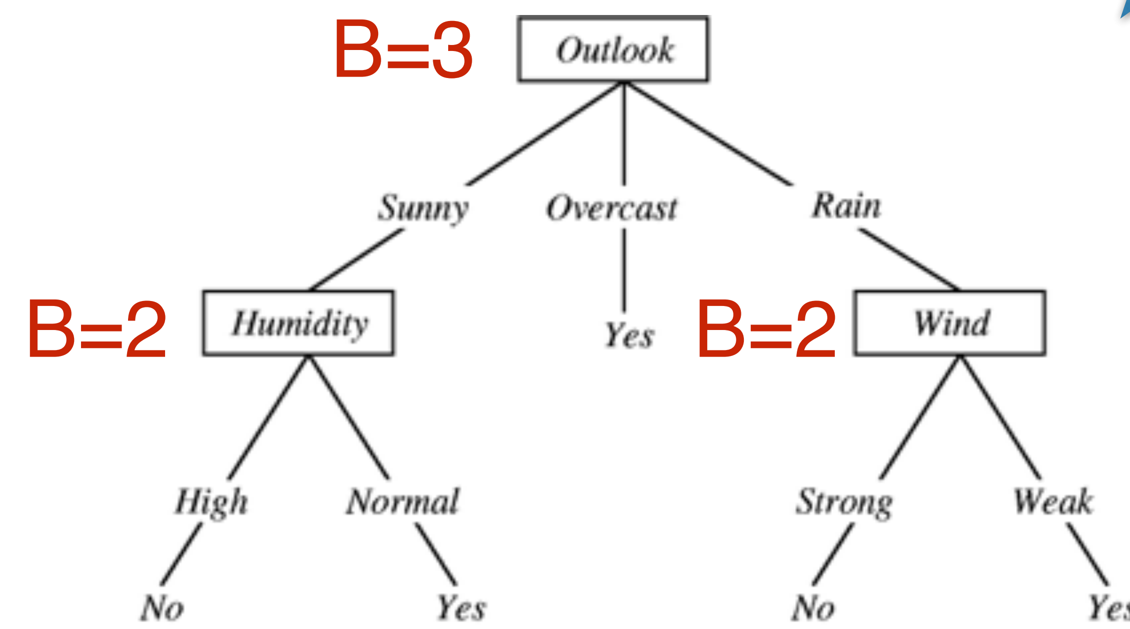
Two general approaches to avoid overfitting in trees:

1) ***Stop growing*** the tree earlier

3) Allow the tree to overfit the training data and then ***post-prune*** it

Decision Tree Learning: C4.5

Duda et. al. book. Chapter 8, up to 8.5
See module website



Next generation/evolution of ID3

Both are TDIDT: Top-Down Induction of Decision Trees
Improvements from ID3 to C4.5:

What entropy-based measure is ID3 using to choose attributes?

$$\text{I.Gain}(S, A) = \text{Entropy}(S) - \sum_i \frac{|S_{u_i}|}{|S|} \text{Entropy}(S_{u_i})$$

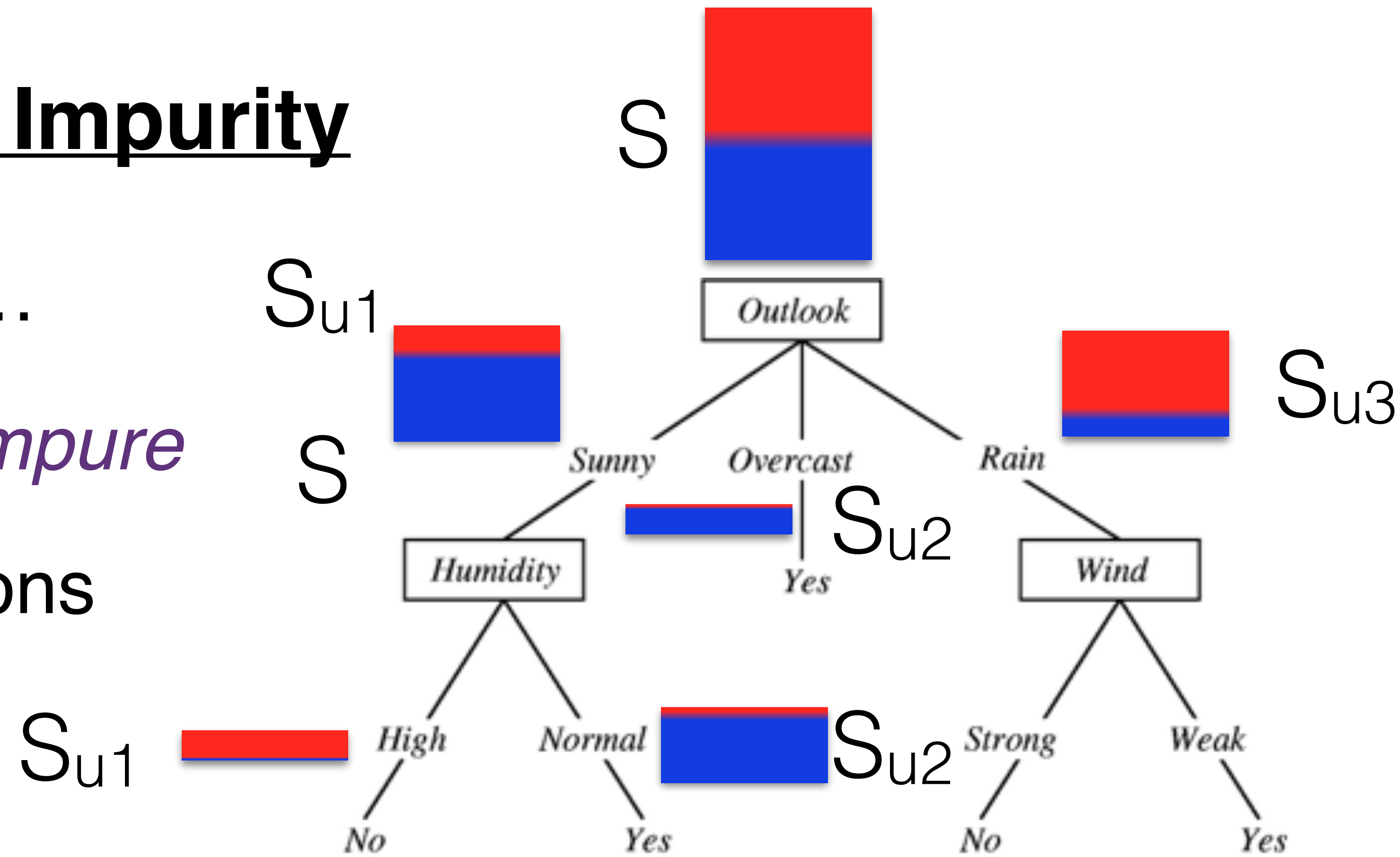
Problem:
Information Gain is biased to prefer highest branching factor (B) nodes!

Decision Tree Learning: Impurity

Some naming conventions…

The leaf subsets Sui can be *impure*

Contain samples/observations from many classes



Entropy is ***one such impurity measure***: *Entropy impurity*

$$i(N) = \text{Entropy}(\text{Set } S \text{ at node } N) = -\sum_{i=1}^{C} p_i \log_2 p_i$$

Decision Tree Learning (C4.5): Branching factor & Entropy

Information Gain is then measuring the *drop in impurity of a split s*

$$\Delta i(s) = \mathrm{I.Gain}(S, A) = \mathrm{Entropy}(S) - \sum_i \frac{|S_{u_i}|}{|S|}\mathrm{Entropy}(S_{u_i})$$

ID3 Problem: I.Gain is biased towards higher branching factor attributes..
Lets "hack it" and normalise it to avoid this

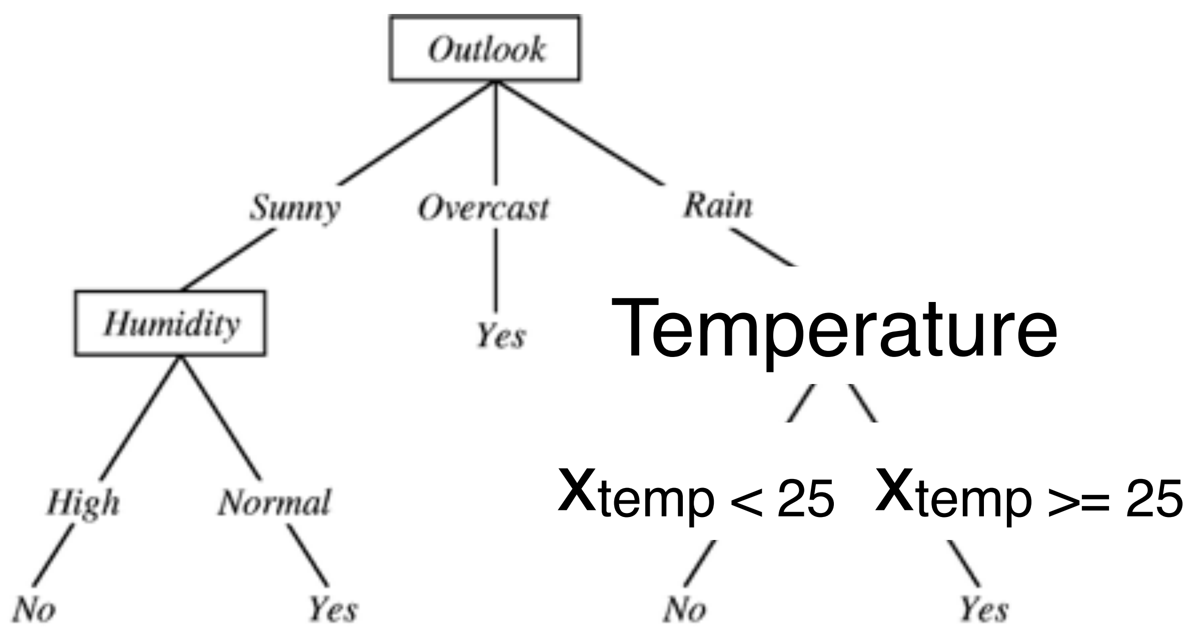
1) [C4.5 Improvement (resolve branching bias)]: **Gain Ratio Impurity**

$$\Delta_{i_B(s)} = \frac{\Delta_i(s)}{-\sum_{k=1}^{B} P_k \log_2 P_k}$$

P_k is the fraction of data on branch k so we normalise Information Gain by the Entropy of the data split.

Decision Tree Learning (C4.5): Continuous Attributes

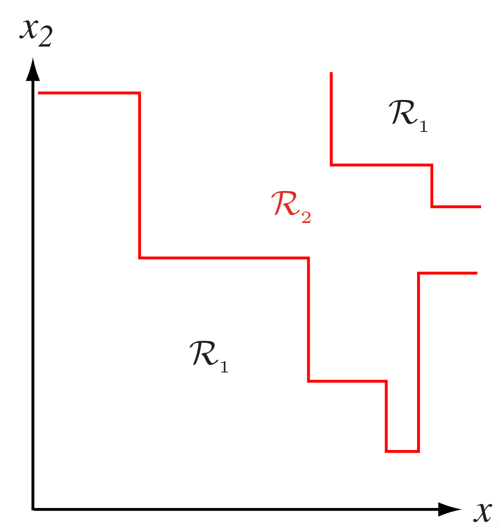
2) [C4.5 Improvement (deal with continuous values)]: Thresholding

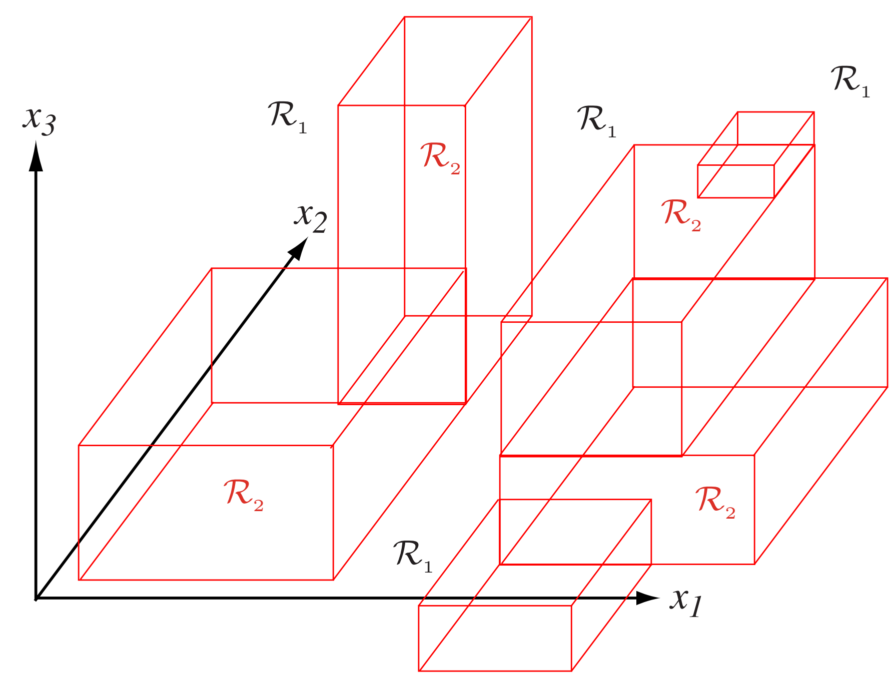


Threshold the continuous attribute and create the split test on the threshold

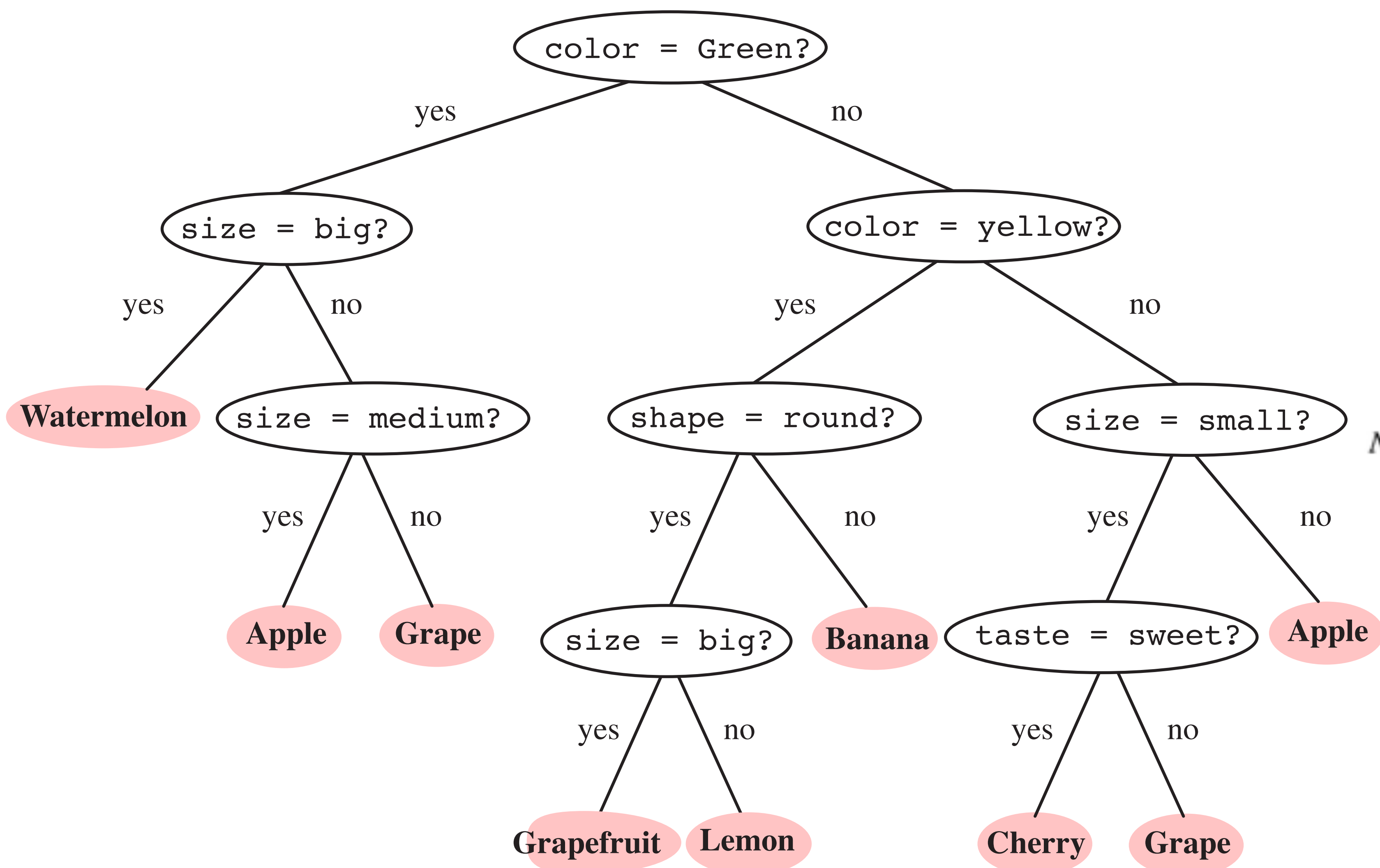
axis-parallel decision boundaries

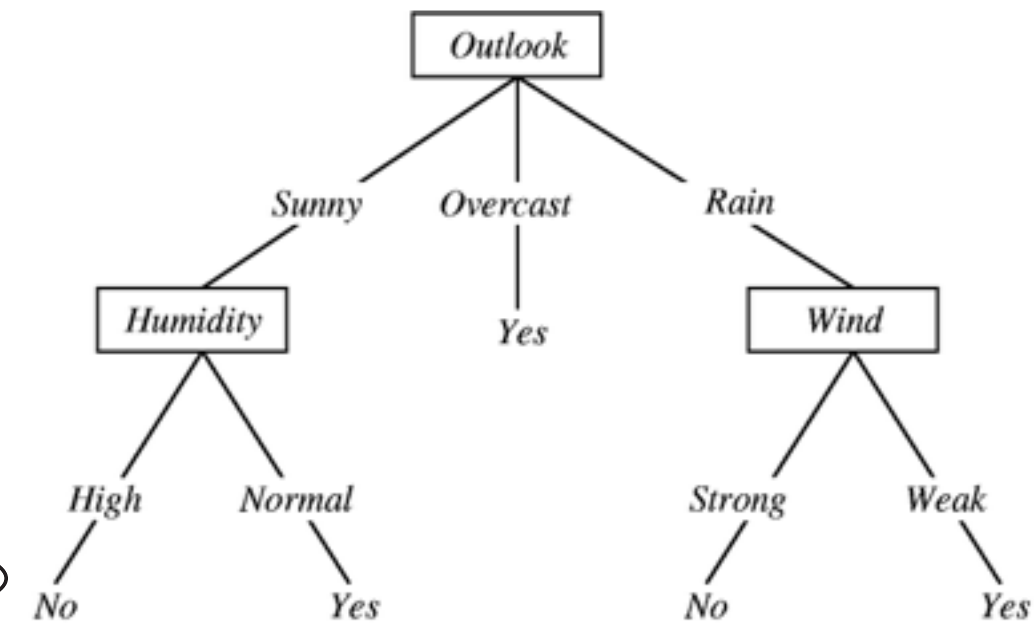
So how would the decision boundary look for binary classification (two classes)?





Decision Tree Learning (CART): Transformation to Binary Tree





Every tree can be represented using just binary decisions
Binary Tree!

Decision Tree Learning: CART

CART = Classification and Regression Trees

Differences between C4.5 and CART:

1) Different measure of impurity: from Entropy Impurity to **Gini Impurity**
2) Always Binary tree
3) Can handle missing values nicely (Surrogate splits)

Gini/Variance Impurity:

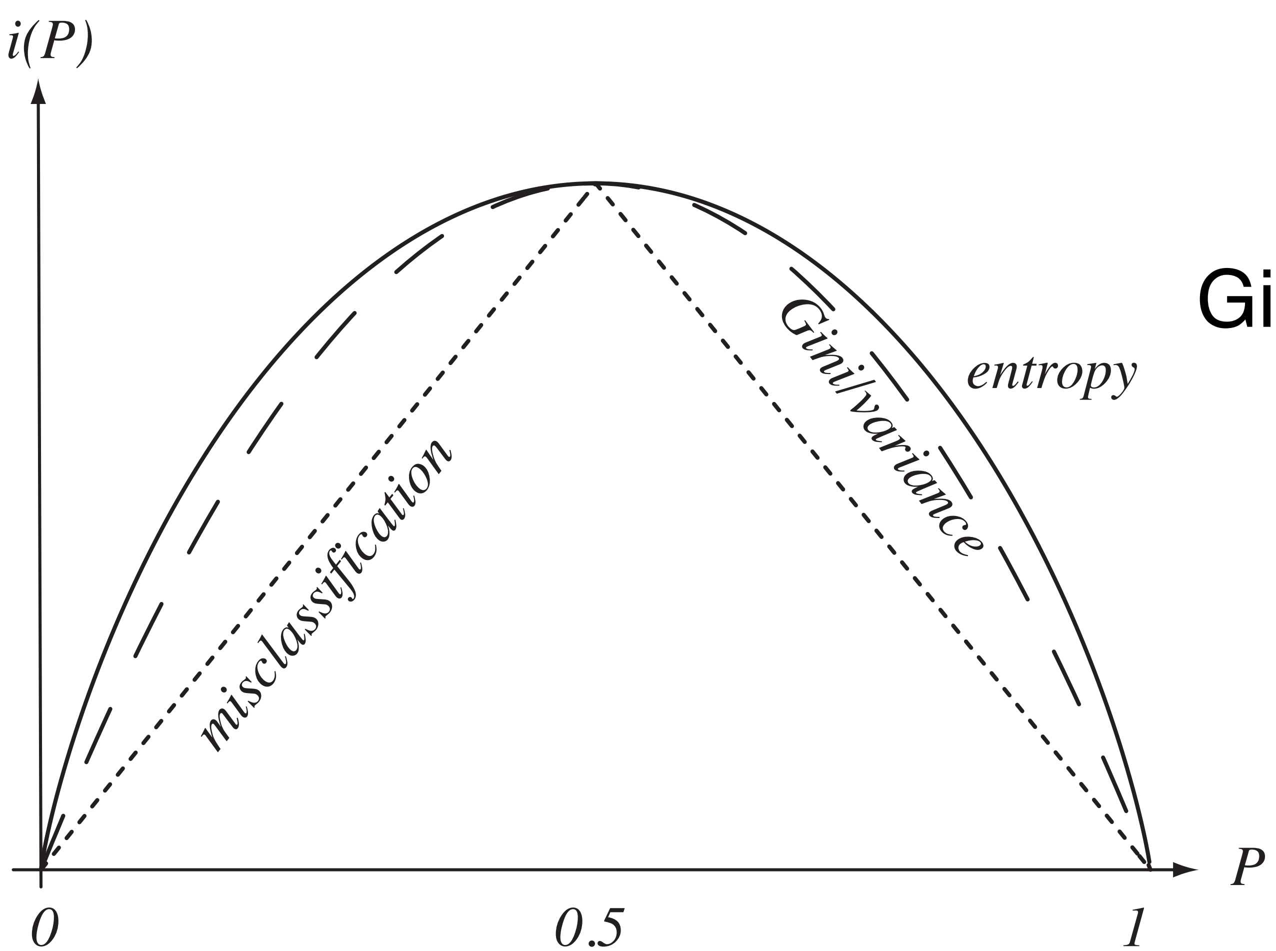
$$i(N) = p_1 p_2 \text{ where } p_i \text{ fraction of observations with class i}$$

Generalisation to multiclass $$i(N) = \sum_{i \neq j} p_i p_j$$

Decision Tree Learning: CART

Lets visualise the two “impurity” measures

$$i(N) = \text{Entropy}(\text{Set } S \text{ at node } N) = -\sum_{i=1}^{C} p_i \log_2 p_i$$



Gini impurity $\quad i(N) = \sum_{i \neq j} p_i p_j$

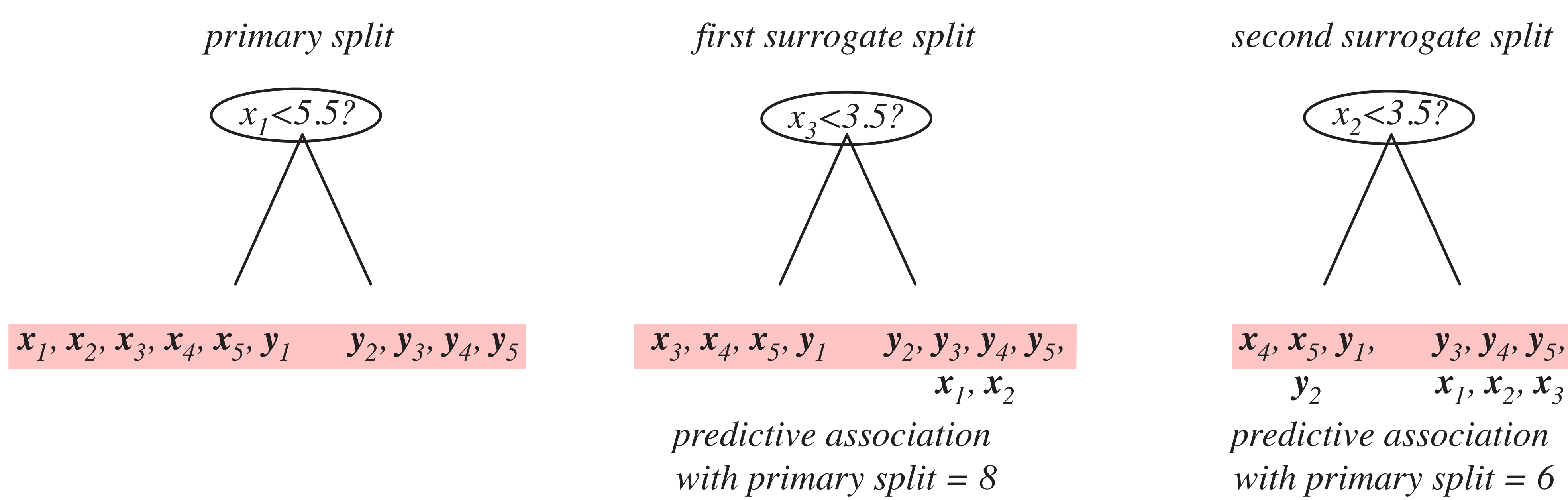
Handling Missing values with CART

Surrogate tests for Missing values

Main idea: *for every non-leaf node create additional (surrogate) rules based on other attributes that mimic primary split behaviour*

Obs. of 1st Class $\overset{\mathbf{x}_1}{\begin{pmatrix}0\\7\\8\end{pmatrix}}, \overset{\mathbf{x}_2}{\begin{pmatrix}1\\8\\9\end{pmatrix}}, \overset{\mathbf{x}_3}{\begin{pmatrix}2\\9\\0\end{pmatrix}}, \overset{\mathbf{x}_4}{\begin{pmatrix}4\\1\\1\end{pmatrix}}, \overset{\mathbf{x}_5}{\begin{pmatrix}5\\2\\2\end{pmatrix}}$

Obs. of 2nd Class $\overset{\mathbf{y}_1}{\begin{pmatrix}3\\3\\3\end{pmatrix}}, \overset{\mathbf{y}_2}{\begin{pmatrix}6\\0\\4\end{pmatrix}}, \overset{\mathbf{y}_3}{\begin{pmatrix}7\\4\\5\end{pmatrix}}, \overset{\mathbf{y}_4}{\begin{pmatrix}8\\5\\6\end{pmatrix}}, \overset{\mathbf{y}_5}{\begin{pmatrix}9\\6\\7\end{pmatrix}}.$



Avoiding Overfitting: Pre-pruning or Post-pruning

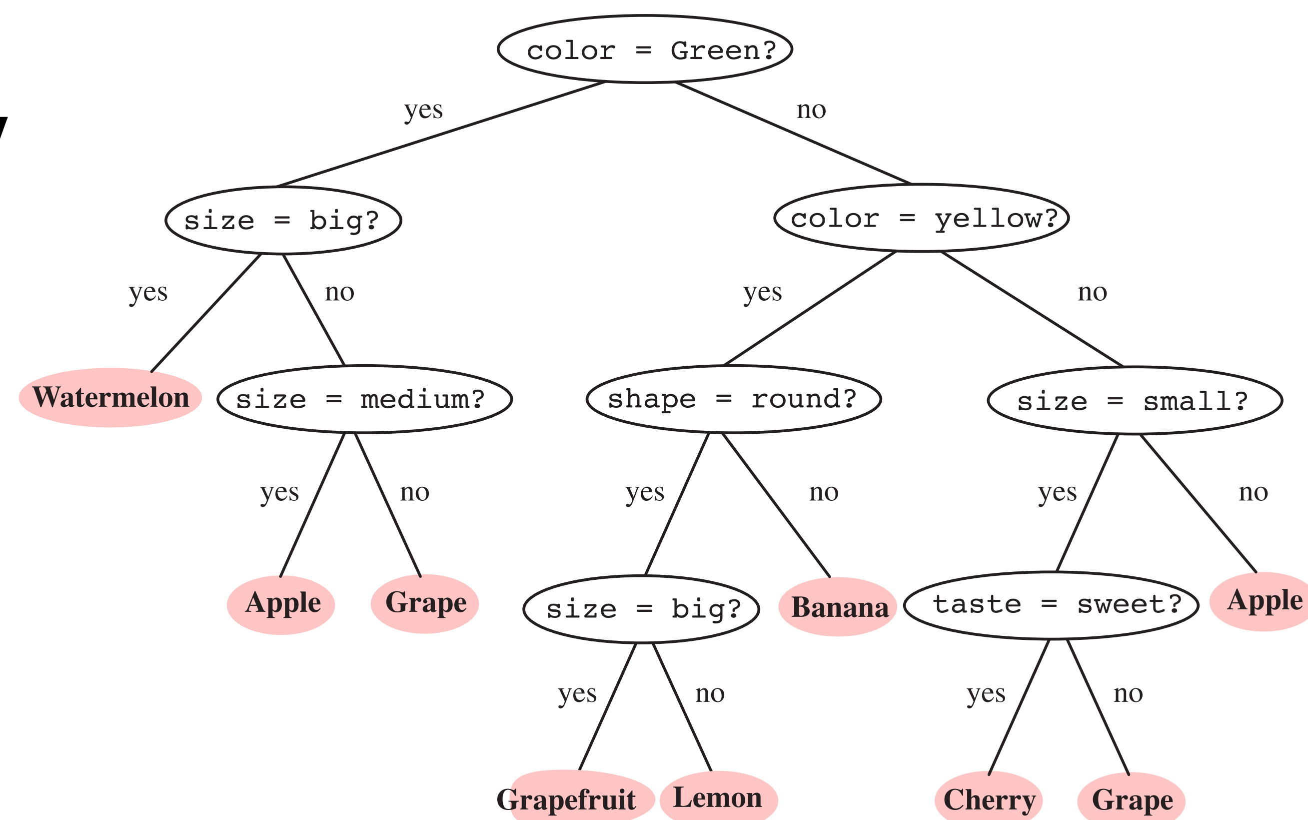
Pre-pruning: 1) Stop splitting when ***Cross-Validation error*** minimised
2) Stop splitting when there is no ***statistical significant*** impurity reduction

Post-pruning: 1) ***Merging***: Grow full tree and try to merge pairs of neighbouring leaf nodes (impurity will increase) back to ancestor

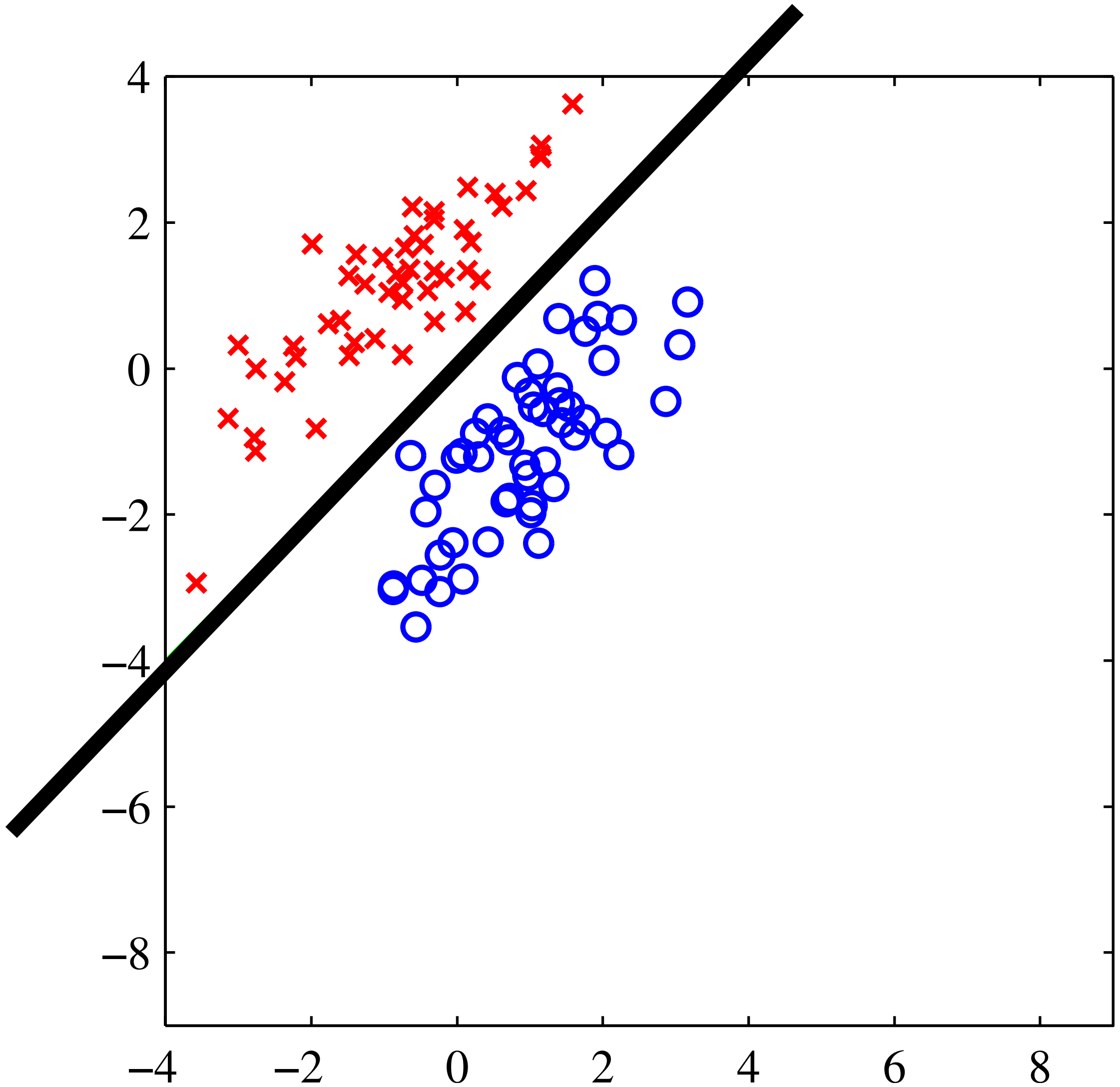
Increase in impurity (Cost) vs Model Complexity trade-off

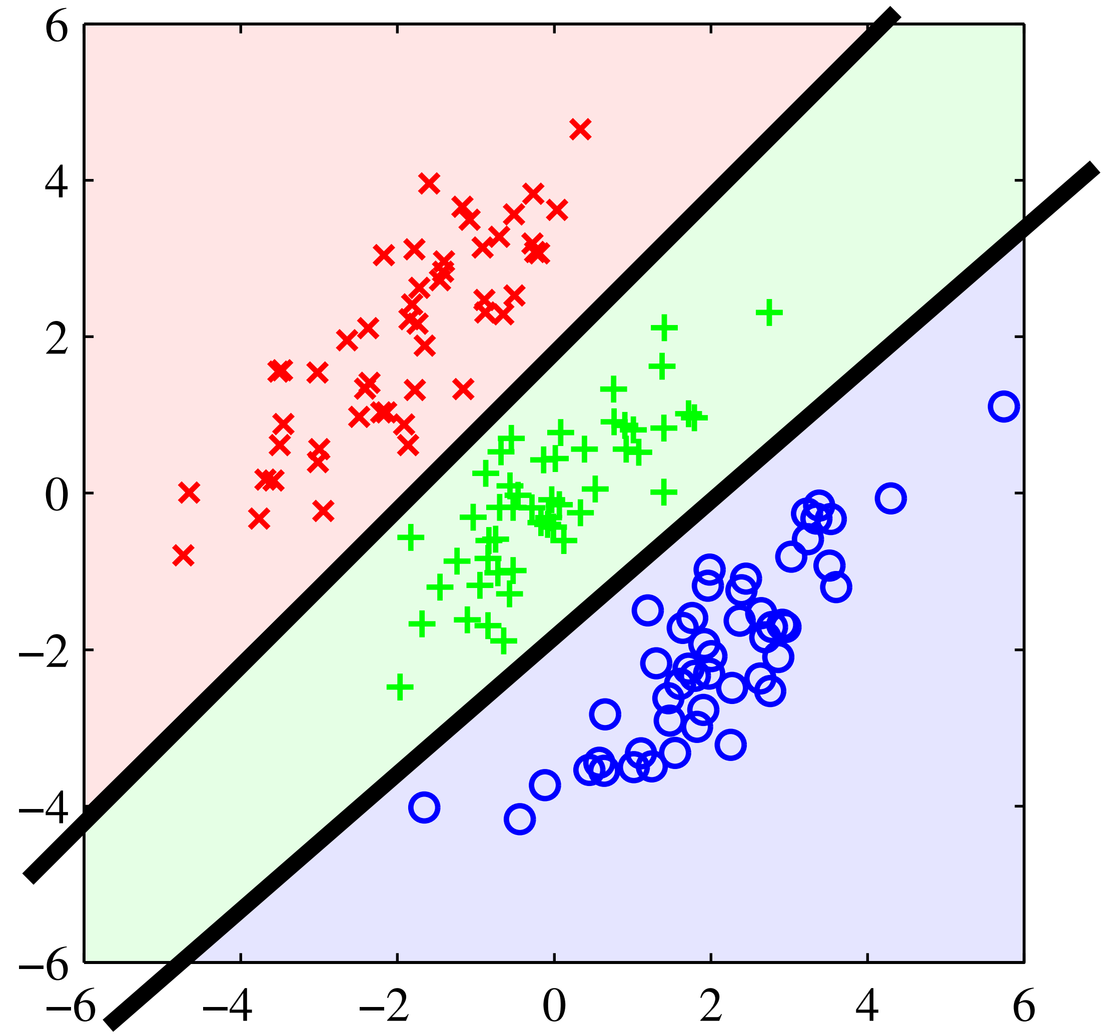
Cost-complexity pruning

Inverse of splitting



Classification: Binary and Multi-class (Linear DBs)



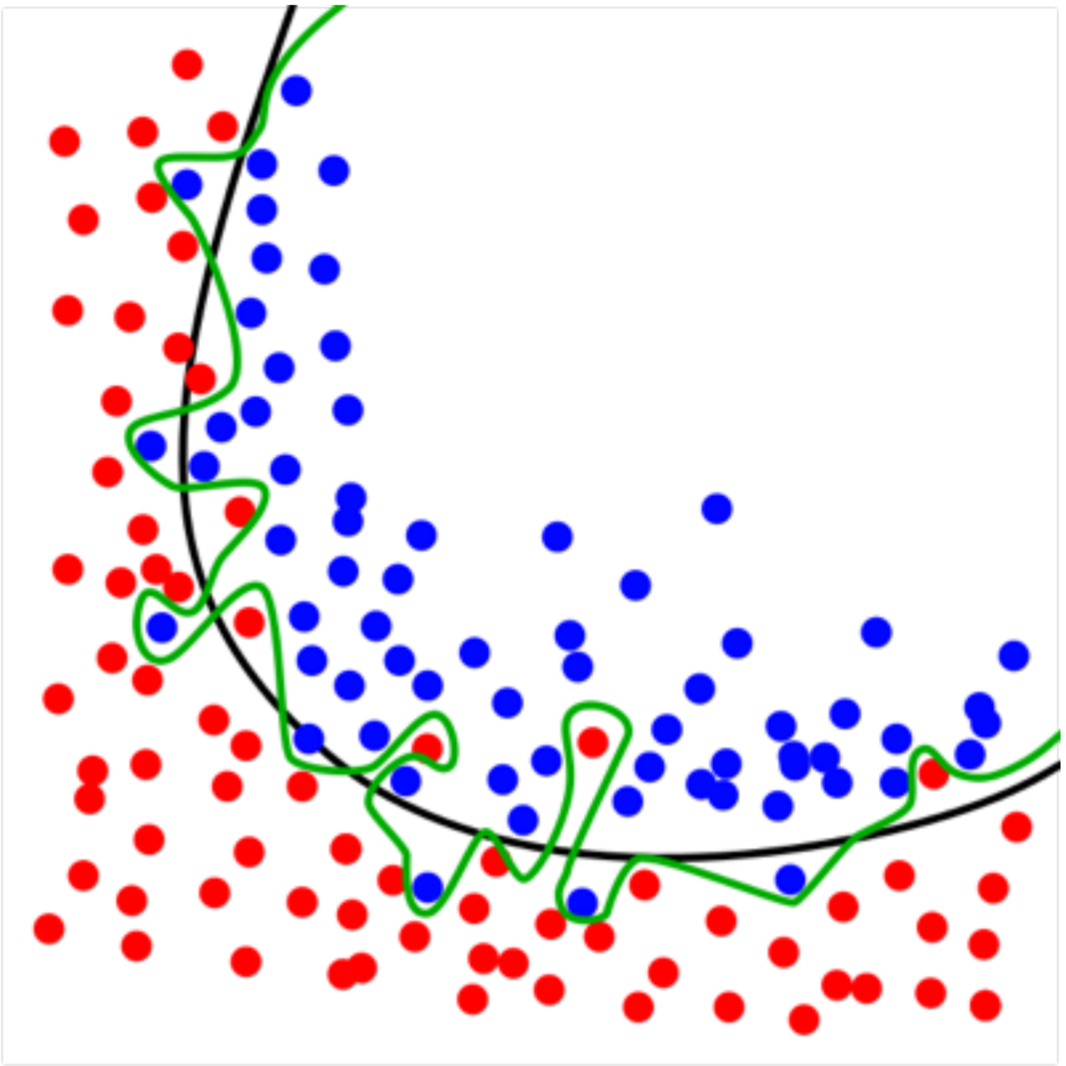


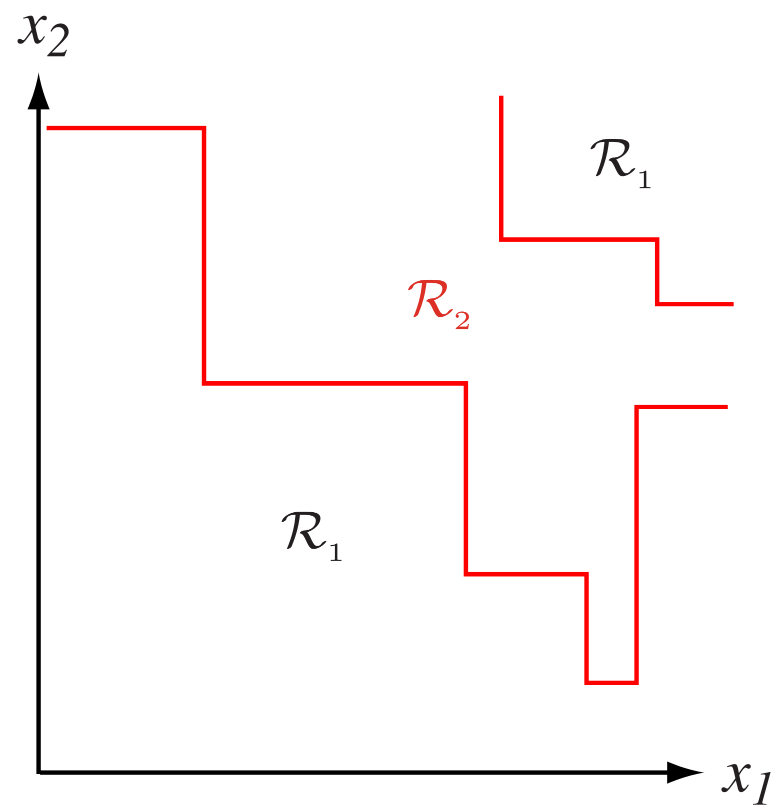
$t_n \in \{-1, 1\}$

$t_n \in \{1, 2, \ldots, C\}$

Sometimes binary classification also as {0,1} or {“positive”, “negative”}

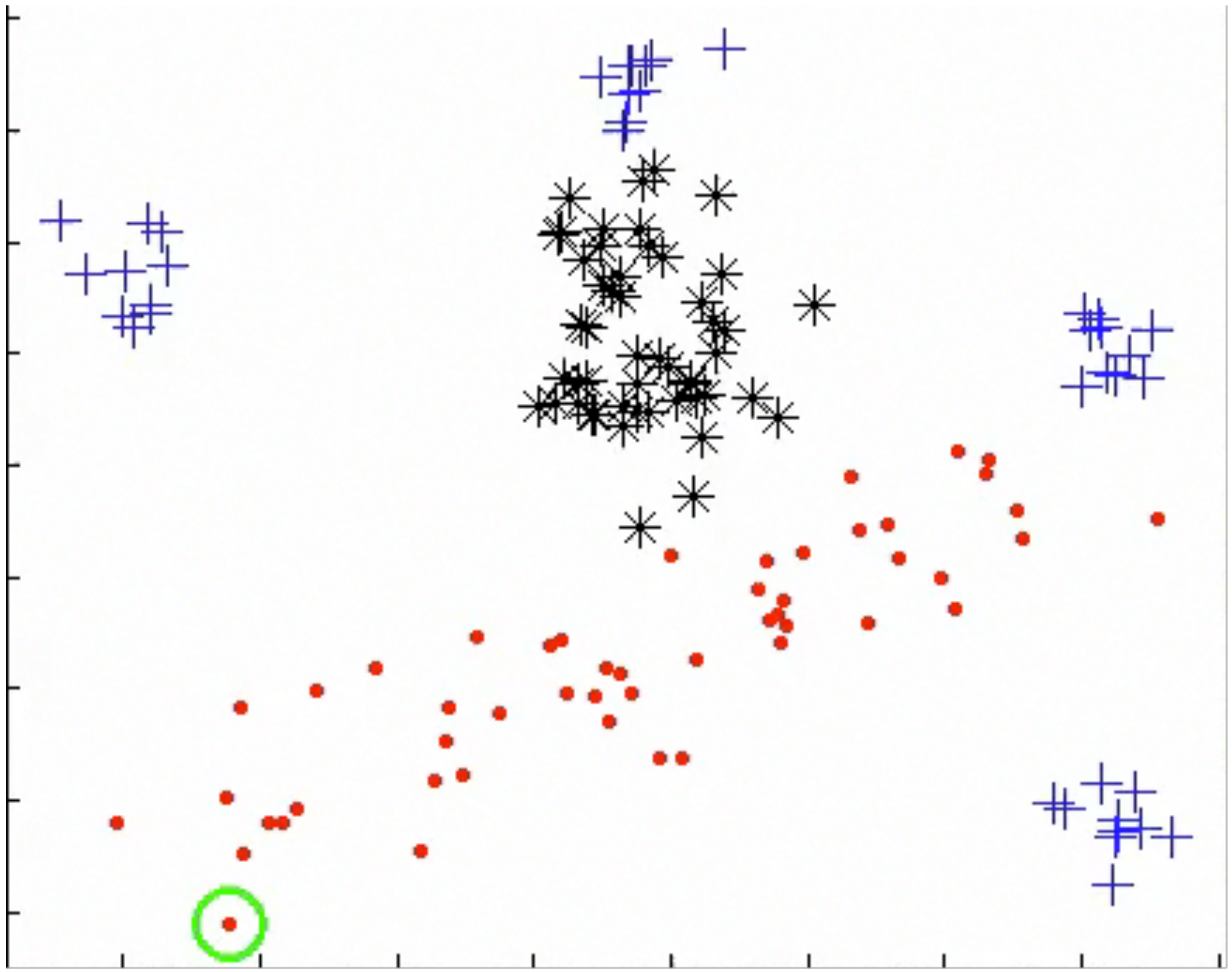
Classification: Binary and Multi-class (Non-Linear DBs)





$$t_n \in \{-1, 1\}$$

Classification: Binary and Multi-class (Multinomial)



Video time…

Classification: Performance Metrics (Binary classification)

e.g. R&G. book. Chapter 5: 5.4-5.5

How would you summarise your performance?

Raw Classification Accuracy
a.k.a. **0-1 Loss**

For every observation:
- Pay 1 if misclassified
- Else Pay 0

Average across all observations

Predicted Target	True Target
1	1
-1	1
1	1
-1	-1

Classification: Performance Metrics

e.g. R&G. book. Chapter 5: 5.4-5.5

Confusion Matrix (binary classification)

	+	-
Predicted +	True Positives	False Positives
Predicted -	False Negatives	True Negatives

Total = 400 observations

	+	-
Predicted +	54	26
Predicted -	12	308

Class + predictions not looking great! 1-0 Loss wouldn't tell us that.
These errors (FP/FN) might be crucial in our application

Confusion Matrix for multi-class?

Classification: Performance Metrics

Confusion Matrix (multiclass classification)

Any problems??

	Class 1	Class 2	Class 3	Class 4
Predicted Class 1	124	13	7	431
Predicted Class 2	12	151	0	2
Predicted Class 3	3	1	1876	230
Predicted Class 4	102	8	15	300

Classification: Performance Metrics

Accuracy, Precision, Recall, F1-score, Sensitivity, Specificity, ROC, AUC

All simple functions of TP-TN-FP-FN

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

$$\text{F1} = 2\frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

$$\text{Accuracy} = \frac{TP + TN}{TP + FN + FP + TN}$$

Specificity - Sensitivity and ROC curve

$$\text{Specificity} = \frac{TN}{TN + FP}$$

$$\text{Sensitivity} = \text{Recall} = \frac{TP}{TP + FN}$$

Both values lie between 0 and 1
Say +ve is diseased people and -ve healthy

Specificity is proportion of healthy people (TN+FP) correctly classified as healthy (True negative rate)

Sensitivity is proportion of diseased people (TP+FN) correctly classified as diseased (a.k.a True positive rate)

Classification: Performance Metrics: The ROC curve

Receiver Operating Characteristic (ROC) Curve

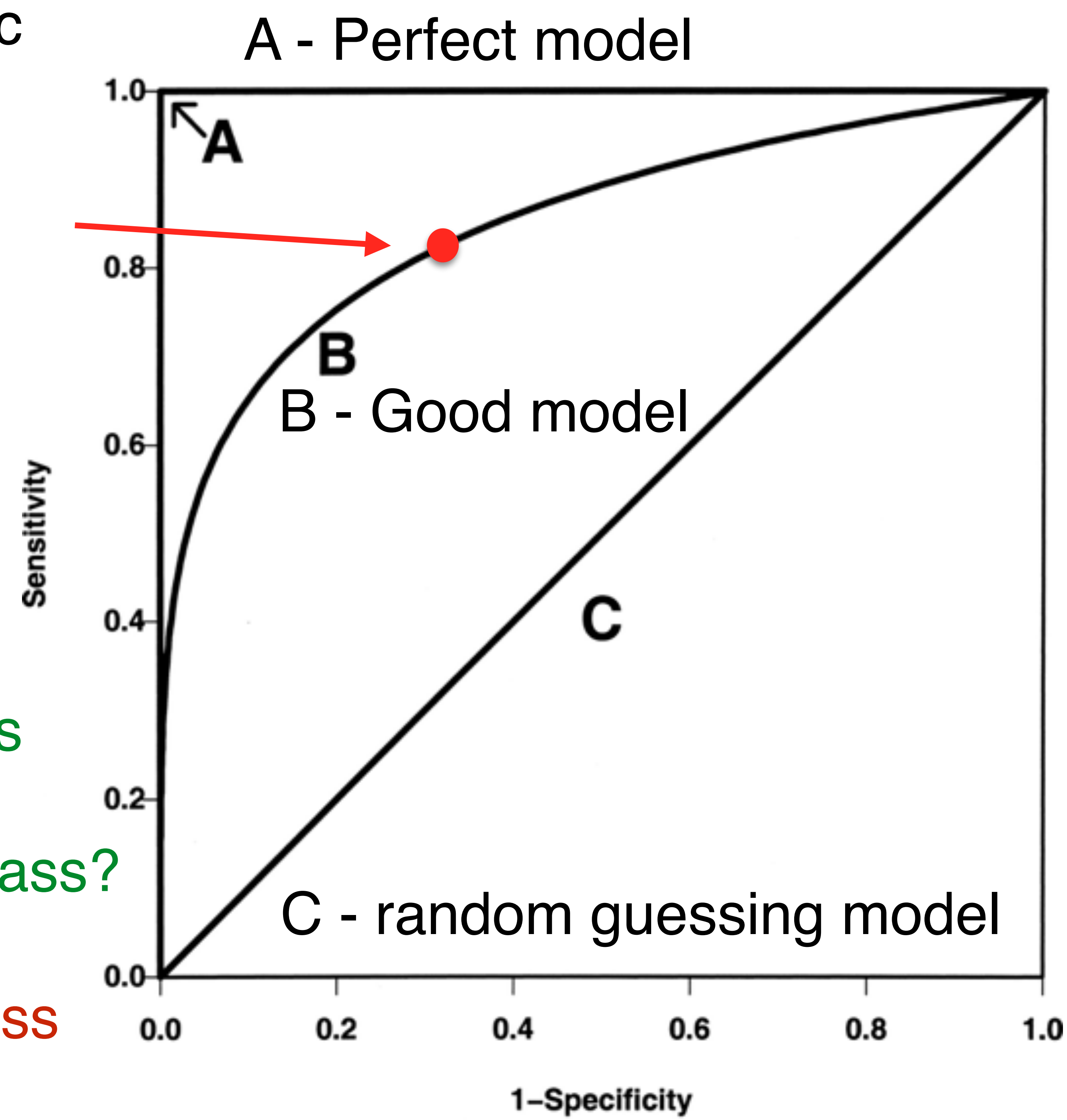
Every model has a ROC curve. Every point is a decision setting of that model

Vary decision threshold (e.g. +1 if p(class=+1) > 0.5) to create full curve

What threshold for k-NN & DTs? They are non-probabilistic models

What can use as "probability" of class?

Think how each model assigns class



Classification: Performance Metrics: ROC and AUC

Use the Area under the curve (AUC) as a performance metric

AUC is using the whole ROC curve across decision thresholds

Takes class imbalance into account

Does not easily/nicely generalise to multiclass setting