MACQUARIE University

MACHINE LEARNING

COMP8220

05 – Tree Models

Xuyun Zhang (Department of Computing)

Acknowledgement

MACQUARIE University

- Some slides are from the S1 version designed by Rolf Schwitter (Rolf.Schwitter@mq.edu.au)

Some slides are also based on book of Christopher Bishop "Pattern Recognition and Machine Learning" Springer-Verlag New York (2006) and the slides made by Dr Jia Wu.

Lecture Outline

MACQUARIE University

- Decision Tree Models
  - What is a decision tree?
  - Splitting Heuristics
  - Decision tree regression
- Ensemble Learning
  - Different ensemble mechanisms
  - Random Forest

Lecture Outline

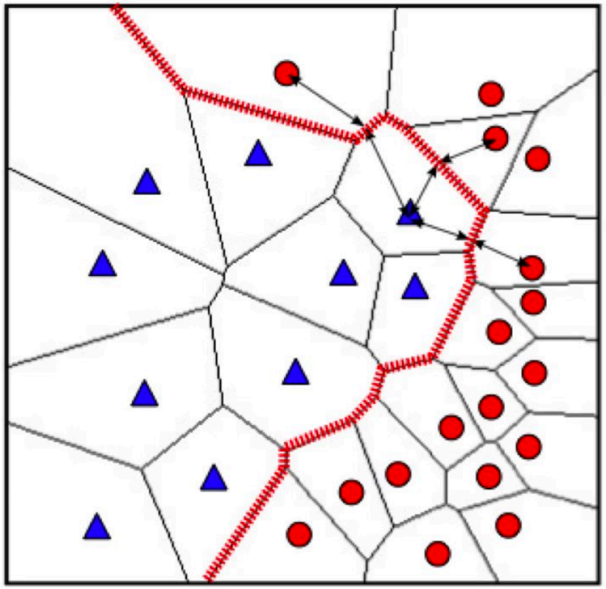
MACQUARIE University

- Decision Tree Models
  - What is a decision tree?
  - Splitting Heuristics
  - Decision tree regression

Revisit 1-NN Classifier

MACQUARIE University

- Feature space partition (Voronoi diagram) for $K = 1$

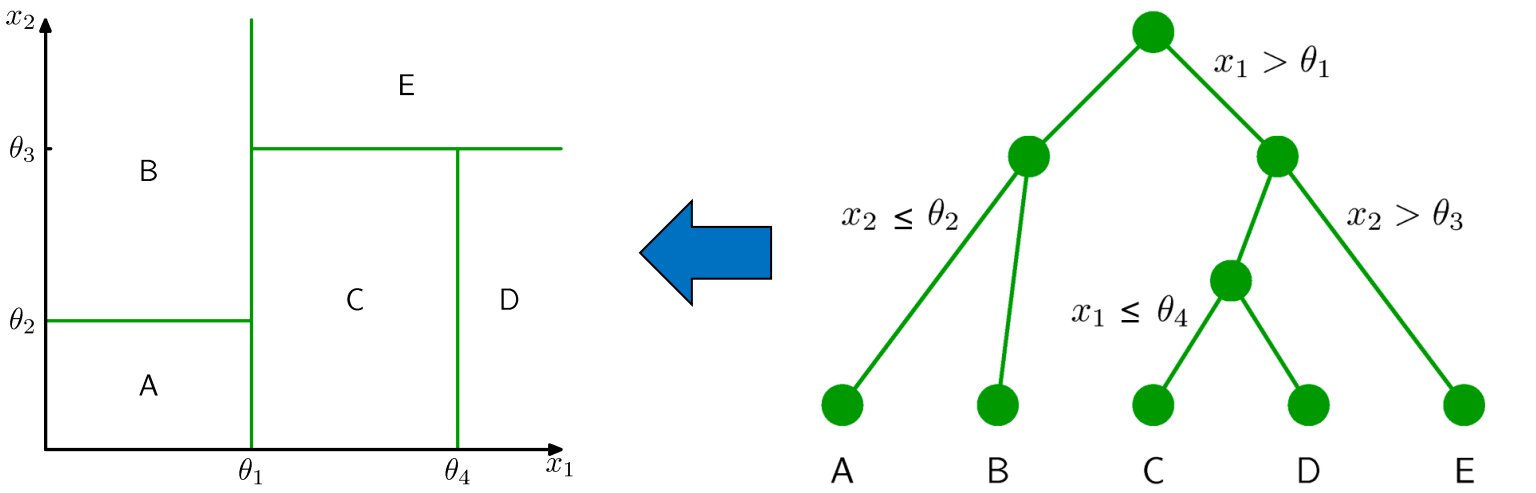


- Question: other ways of partitioning feature space?
  - Instance-based learning with more regular partitions?
  - Can we control the way of generating the partitions?

Decision Trees

MACQUARIE University

- Basic idea: partition feature space along axes to produce decision regions
  - Naturally using tree data structures to achieve recursive top-down partition → decision tree models
  - The same rationale as KNN classification, i.e., similar data instances in a decision region share the same label

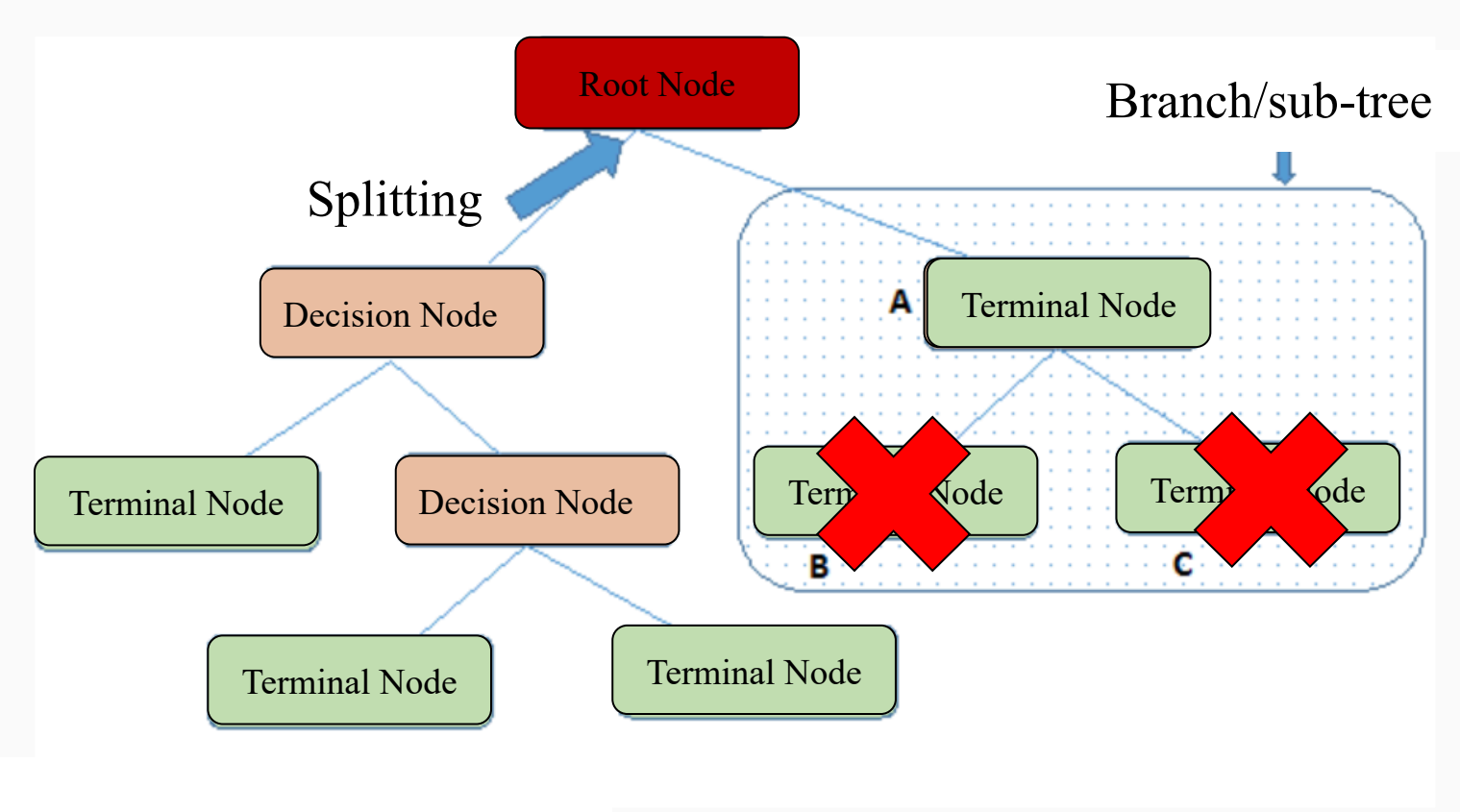


Decision Trees (Cont'd)

MACQUARIE University

- Two of top-10 data mining algorithms (2008 version)
  - C4.5 (No.1)
  - CART (No. 10): Classification and Regression Tree
- High efficiency: logarithmic tree operations
- Non-linear decision boundaries
  - Similar to KNN classification
  - Strong model complexity but prone to overfitting
- High interpretability (vs ANN models)
  - Readily interpretable decision rules
  - E.g., fever ^ dry cough ^ shortness of breath → COVID-19

Basic Tree Terminology



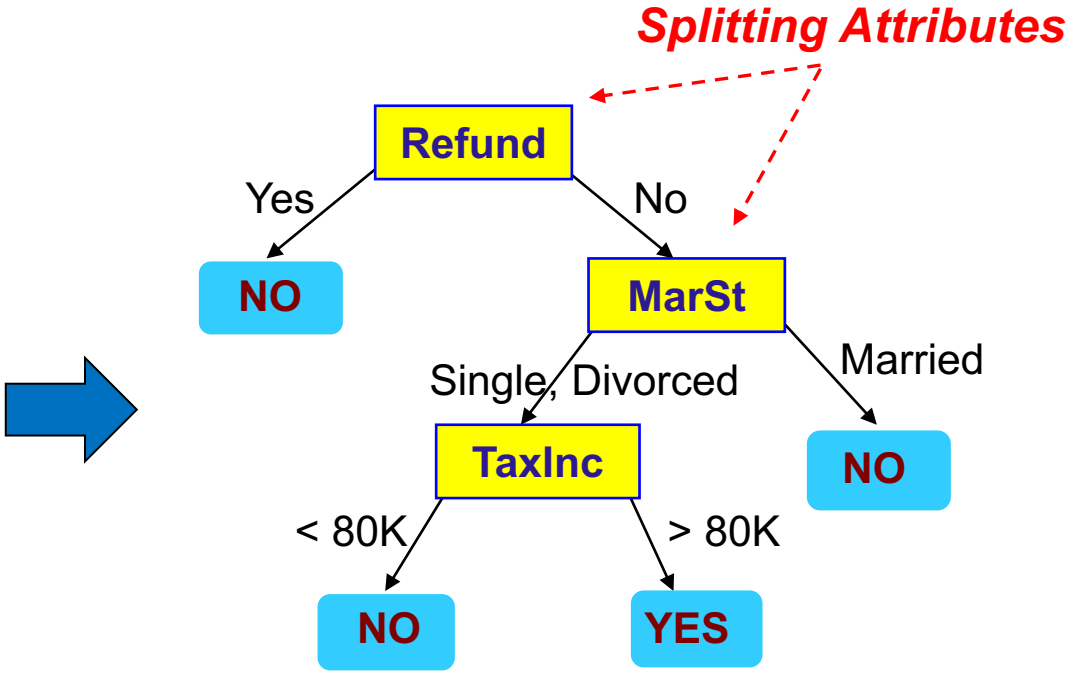
- Pruning

Training

- Training: construct a (decision) tree structure
  - Decision nodes: tracking partitioning (tree structure)
  - Terminal nodes: representing decision regions

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

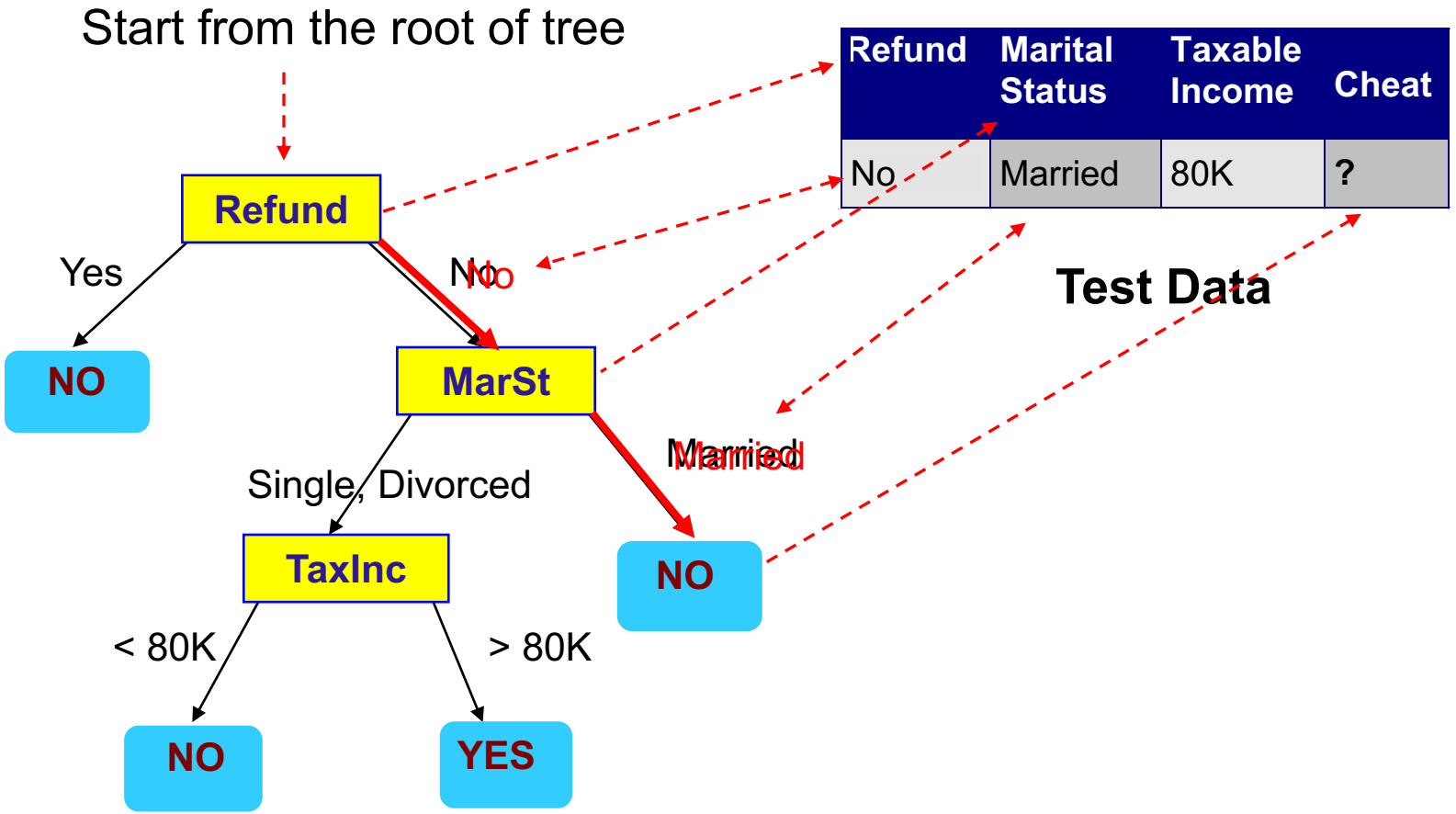
Training Data



Model: Decision Tree

Testing

- Predicting: traverse the built decision tree for a region



How to Partition Data ?

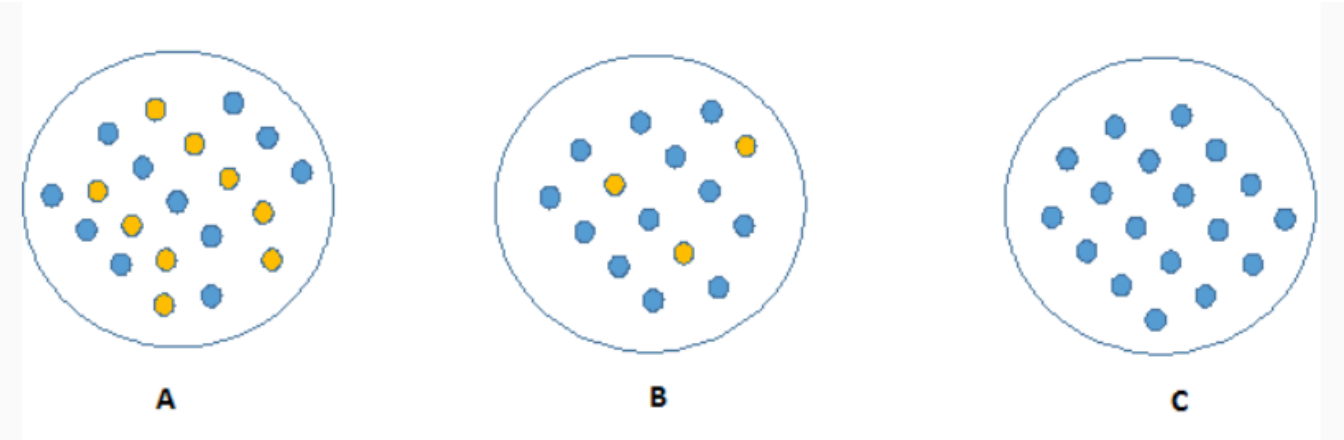
MACQUARIE University

- Two fundamental questions!
- Which axis (dimension/feature) should be chosen?
  - To minimise classification error
  - Computationally infeasible to try all possible tree structures
    - E.g., for 10 binary features, search space size is 2^{10}
  - Greedy optimisation: use a heuristics
    - E.g., information gain
- What values of the chosen attribute for splitting?
  - Depending on data types: categorical or continuous
  - Continuous features (or dimension) need discretisation
  - Binary or multiway tree structure

Splitting Heuristics

MACQUARIE University

- Which partition are more expected after splitting?



- Answer: C. Why?
  - No/few splits are required
  - Voting for class label is easier (similar to KNN classifier)
  - Impurity refers to this quality
    - Entropy (information), Gini impurity, etc.

Entropy

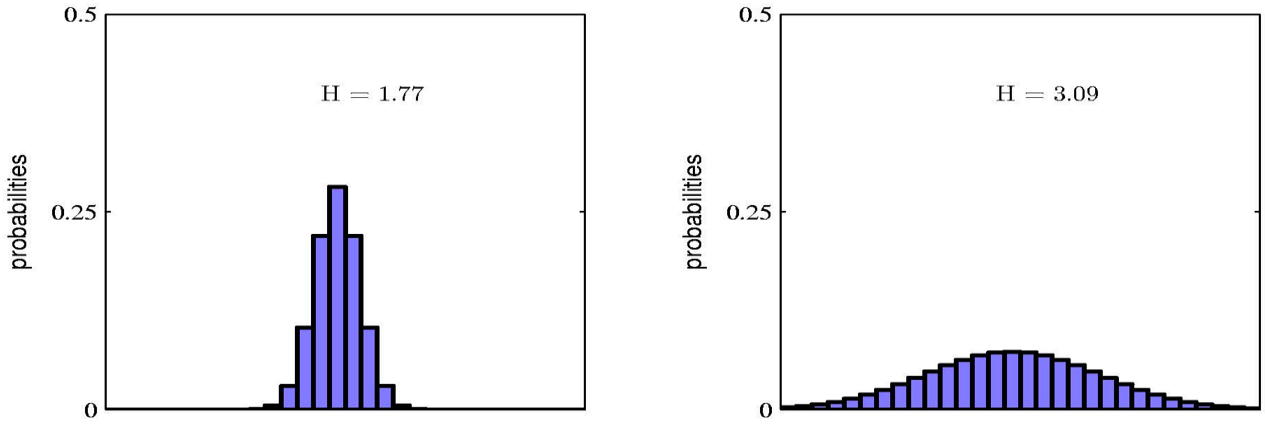
MACQUARIE University

- A measurement of information or uncertainty
- Discrete random variable

$$H[x] = -\sum_x p(x) \log_2 p(x)$$

- Continuous random variable (differential entropy)

$$H[x] = -\int p(x) \ln p(x) dx$$



Information Gain

MACQUARIE University

- Assume a data set D, the distribution of label $\boldsymbol{t}_D$ is

Class	C_1	$\cdots$	C_K	**Total**
Count	c_{p1}	$\cdots$	c_{pK}	N_D

  - Information (entropy) of $\boldsymbol{t}_D$: $Info(\boldsymbol{t}_D) = -\sum_{i=1}^{K} p_i \log_2(p_i)$
  - Probability $p_i = c_{pi}/N_D$
- Partition D by feature ϕ_j, producing $\{D_1, \ldots, D_v\}$
  - $\boldsymbol{t}_D$'s information after partition:

$$Info_{\phi_j}(\boldsymbol{t}_D) = \sum_{i=1}^{V} \frac{N_{D_i}}{N_D} Info(D_i)$$

- Information gain of this partition

$$Gain_D(\phi_j) = Info(\boldsymbol{t}_D) - Info_{\phi_j}(\boldsymbol{t}_D)$$

Gain Ratio

MACQUARIE University

- Info gain issue: prefer attributes with more values
  - Extreme case: prefer unique identifiers e.g., instance IDs
- Solution
  - Normalisation on information gain for each feature
- Split information of a feature ϕ_j (w.r.t. original dataset)

$$SplitInfo_{\mathcal{D}}(\phi_j) = -\sum_{i=1}^{V} \frac{N_{\mathcal{D}_i}}{N_{\mathcal{D}}} \log_2\left(\frac{N_{\mathcal{D}_i}}{N_{\mathcal{D}}}\right)$$

- Gain Ratio: normalised information gain

$$GainRatio_D(\phi_j) = Gain_D(\phi_j) / SplitInfo_{\mathcal{D}}(\phi_j)$$

Gini impurity

MACQUARIE University

- Gini impurity
  - Probability of two samples having different labels when being randomly chosen from a dataset

$$Gini(D) = 1 - \sum_{i=1}^{K} p_i^2 = \sum_{i=1}^{K} p_i(1 - p_i)$$

  - Partition D by feature ϕ_j

$$Gini_D(\phi_j) = \sum_{i=1}^{V} \frac{N_{D_i}}{N_D} Gini(D_i)$$

- The lower the better for splitting
  - Zero Gini impurity implies perfect classification
- Selection of splitting heuristic is data dependent

Example: Data

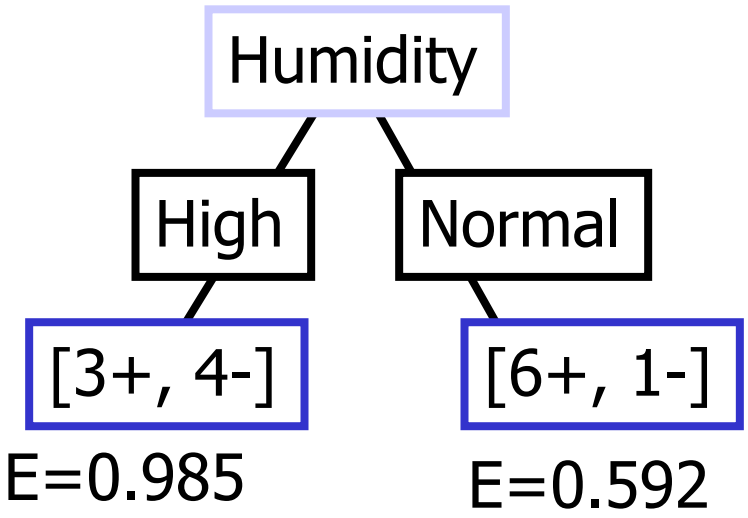
MACQUARIE University

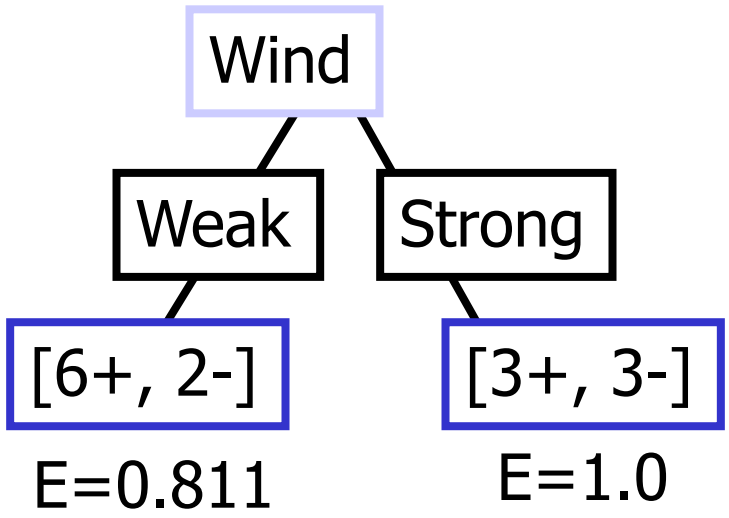
Day	Outlook	Temp.	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Weak	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cold	Normal	Weak	Yes
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: Information Gain

MACQUARIE University

- Information/entropy of label: S=[9+,5-] → E=0.940





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

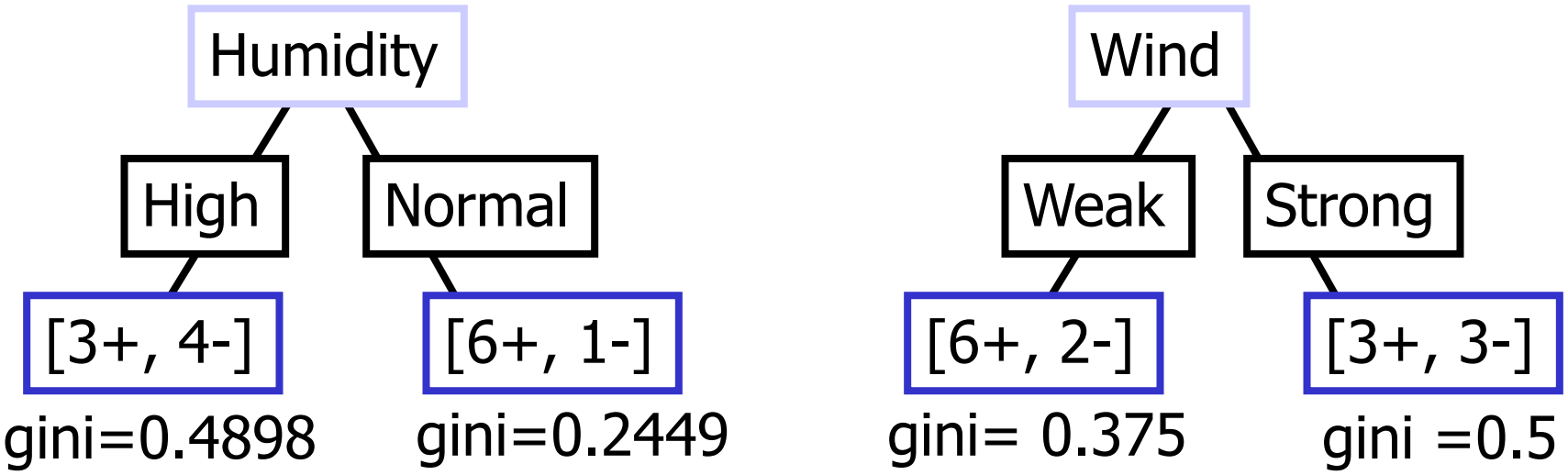
Gain(S,Wind)
=0.940-(8/14)*0.811
– (6/14)*1.0 = 0.048

- Humidity provides greater information gain than Wind

Example: Gini Impurity

MACQUARIE University

- Gini impurity of class label: S=[9+,5-] → gini=0.4590



Gini(S,Humidity)
=(7/14)* 0.4898 +
(7/14)* 0.2449 = 0.3674

Gini(S,Wind)
= (8/14)*0.375
+ (6/14)*0.5 = 0.4286

- Humidity has lower Gini impurity than Wind

When to Stop Tree Growth?

MACQUARIE University

- When to stop the growth of a decision tree?
  - An extreme case: each leaf contains only one instance
  - Overfitting issues (particularly for the extreme case)
- Remedy: pre-pruning by constraining tree structure
  - Minimum number of instances for a node split
  - Minimum number of instances for a terminal node (leaf)
  - Maximum depth of tree (vertical depth)
  - Maximum number of terminal nodes
  - Maximum features to consider for split
- Remedy (empirical): pre-pruning and post-pruning
  - Make use of validation to tune tree structure

Pre-pruning vs Post-pruning

MACQUARIE University

- Pre-pruning
  - Top-down
  - Greedy strategy
    - Stops immediately when no performance improvement
  - Producing small trees
  - Fast
  - Risk of underfitting
- Post-pruning
  - Bottom-up
  - Exhaustive
    - Checking all decision nodes
  - Producing big trees
  - Slow
    - More nodes at lower tree layers)
  - Good trade-off

Continuous Target

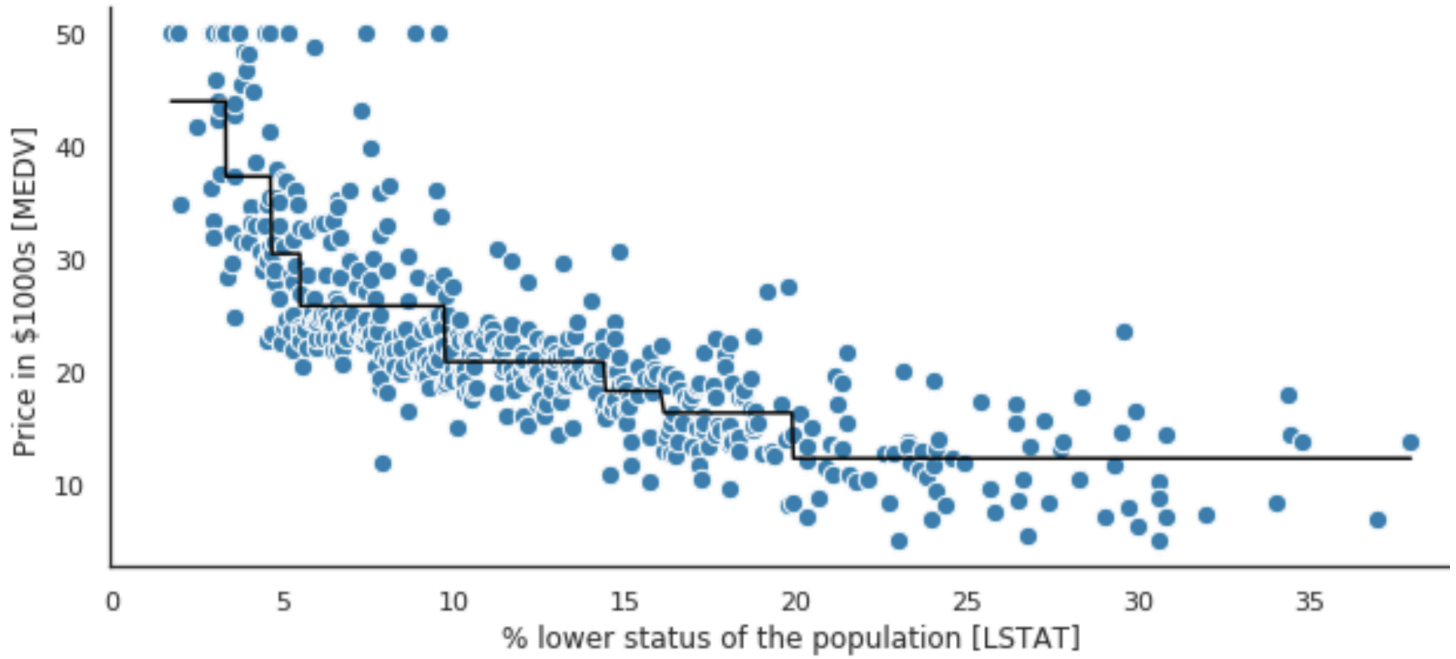
MACQUARIE University

- This is a regression problem!
  - We can follow the same principles as classification
- Basic idea
  - Build a (simple) regression model for a dataset
  - Use a feature to split the dataset into a series of subsets
  - Build regression models for the subsets
  - Choose a feature that can achieve the best performance gain (e.g., mean squared error) from its data partitioning
  - Recursively repeat these steps
- Piece-wise regression model
  - Different space partitions use different regression models

Continuous Target (Cont'd)

MACQUARIE University

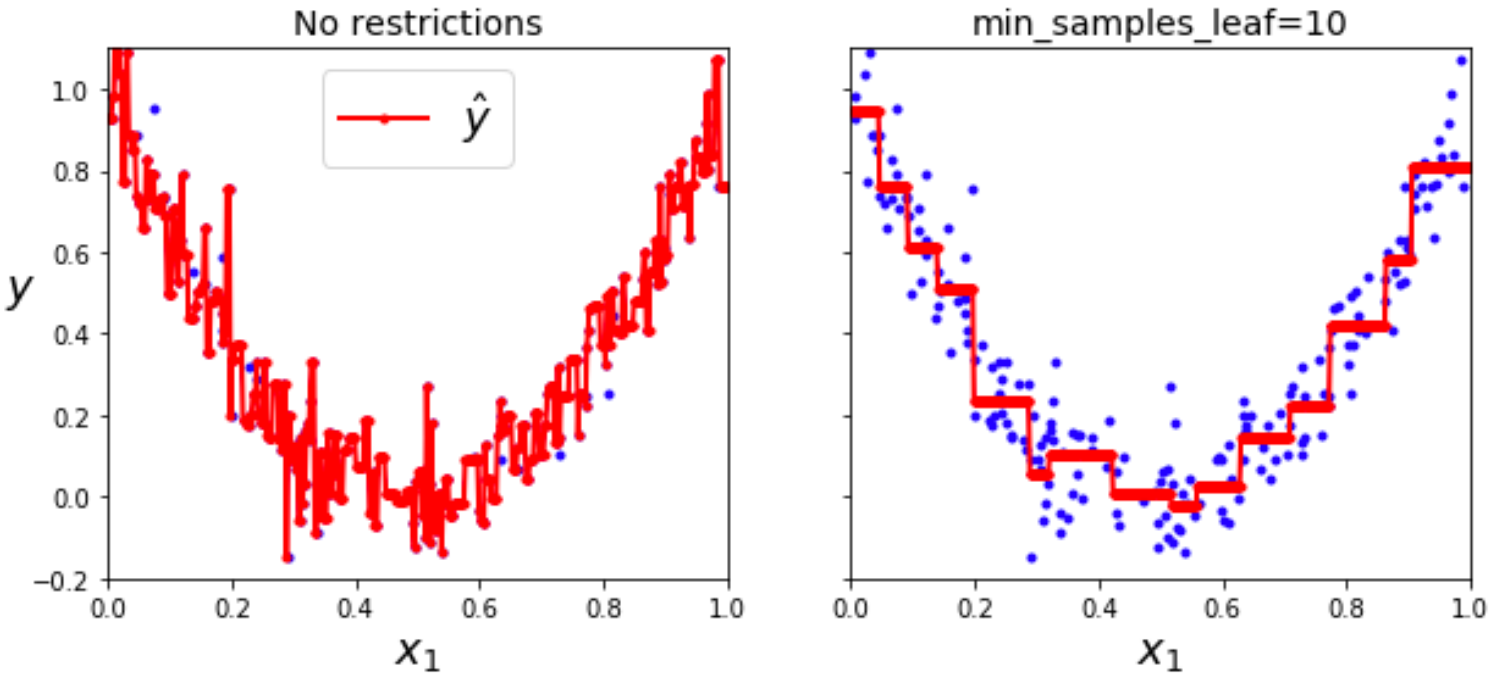
- Example (Boston Housing dataset)
  - Single feature for demo
  - Using MSE
  - Regression model (simple): $y = f(x) = w_0$



Tree Pruning / Regularization

MACQUARIE University

- Pruning/regularization on tree regression models



Decision Tree Algorithm

MACQUARIE University

Generic Decision Tree Construction Algorithm - buildTree($\mathcal{D}$)

Input: $\mathcal{D} = \{\langle \phi(\boldsymbol{x}_1), t_1 \rangle, \cdots, \langle \phi(\boldsymbol{x}_N), t_N \rangle\}$, ϕ is M-dimensional.
Output: A (sub-)tree with root node $Node_{\text{root}}$.

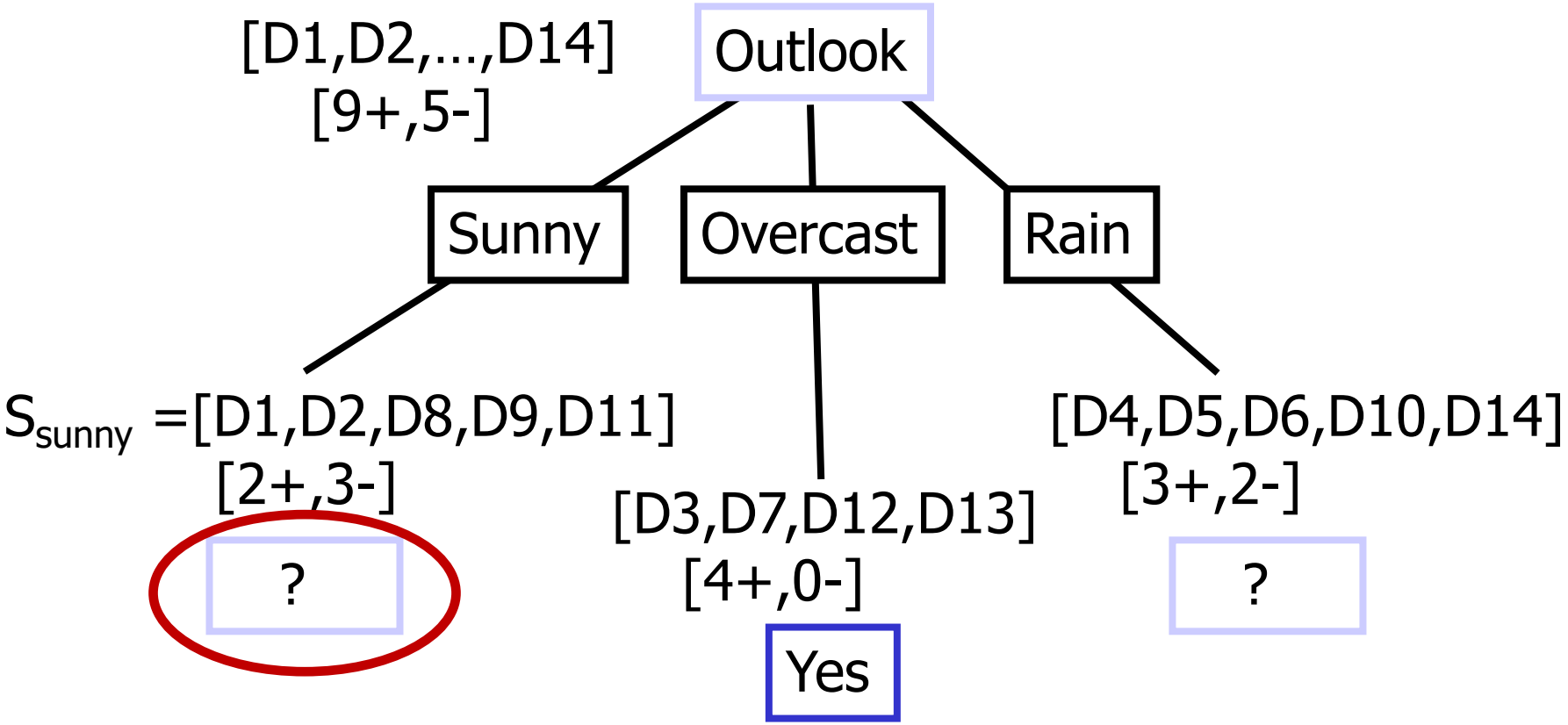
1: **if** $\mathcal{D}$ is "pure" **or** other stopping criteria met **then**
2: **return** $Node_{\text{root}} \leftarrow \text{leafNode}(\mathcal{D})$
3: **for** $i \leftarrow 1 : M$ **do**
4: $h_i \leftarrow$ compute heuristics (e.g., gain ratio) if we split ϕ_i
5: $\langle \phi_{\text{best}}, \Pi_{\text{best}} \rangle \leftarrow \max_{\phi_i}\{h_i\}$, Π_{best} is the best partition for ϕ_i
6: $\{\mathcal{D}_1, \cdots, \mathcal{D}_V\} \leftarrow \mathcal{D}$, i.e., partition $\mathcal{D}$ by Π_{best}
7: **for** $i \leftarrow 1 : V$ **do**
8: $Node_i \leftarrow \text{buildTree}(\mathcal{D}_i)$
9: $Node_{\text{root}} \leftarrow \text{decisionNode}(\phi_{\text{best}}, \Pi_{\text{best}}, \{Node_1, \cdots, Node_V\})$
10: **return** $Node_{\text{root}}$

Typical Algorithms

MACQUARIE University

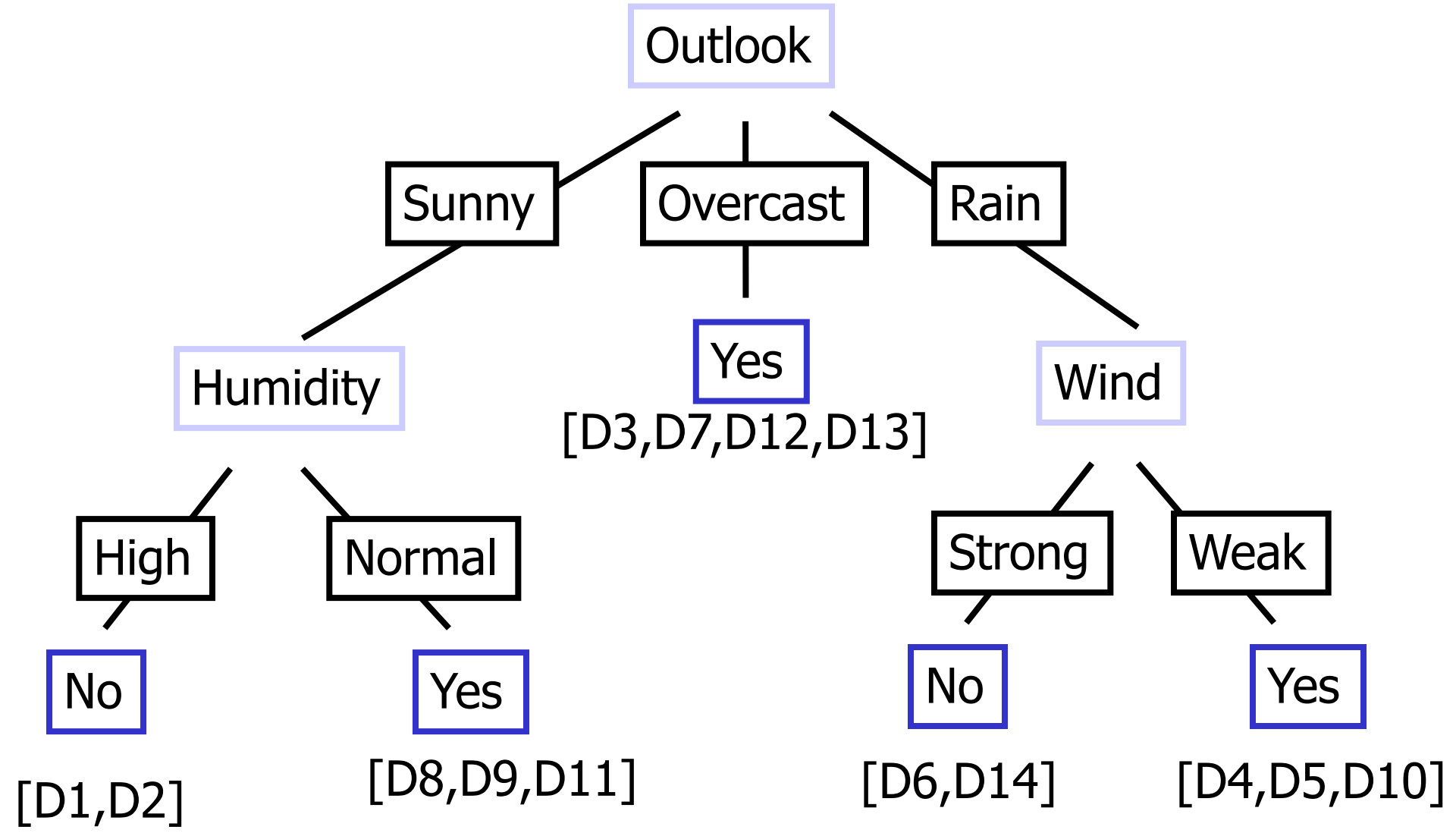
ID3	C4.5	CART
Iterative Dichotomiser 3	Extension of ID3	Classification and Regression Tree
Classification	Classification	Classification Regression
Categorical only	Categorical Numerical	Categorical Numerical
Exhaust attribute values for splitting	Exhaust attribute values for splitting	Binary partition
Interpretable rules	Interpretable rules	No rules
Information gain	Gain ratio	GINI index
No pruning	Pruning	Pruning
...	...	...

Example: Algorithm



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

Example: Algorithm (Cont'd)



Example: Rule Extraction

MACQUARIE University

- Decision rules can be easily extracted from the tree
- R1: If (Outlook=Sunny)^(Humidity=High) Then PlayTennis=No
- R2: If (Outlook=Sunny)^(Humidity=Normal) Then PlayTennis=Yes
- R3: If (Outlook=Overcast) Then PlayTennis=Yes
- R4: If (Outlook=Rain)^(Wind=Strong) Then PlayTennis=No
- R5: If (Outlook=Rain)^(Wind=Weak) Then PlayTennis=Yes

Lecture Outline

MACQUARIE University

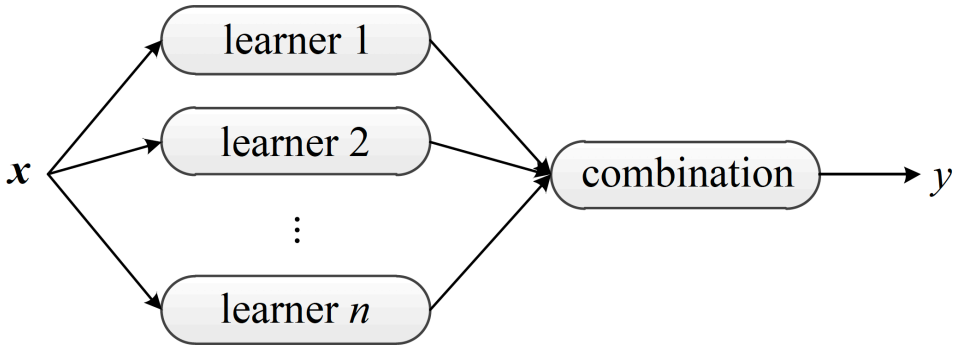
- Decision Tree Models
  - What is a decision tree?
  - Splitting Heuristics
  - Decision tree regression

- Ensemble Learning
  - Different ensemble mechanisms
  - Random Forest

Ensemble Learning

MACQUARIE University

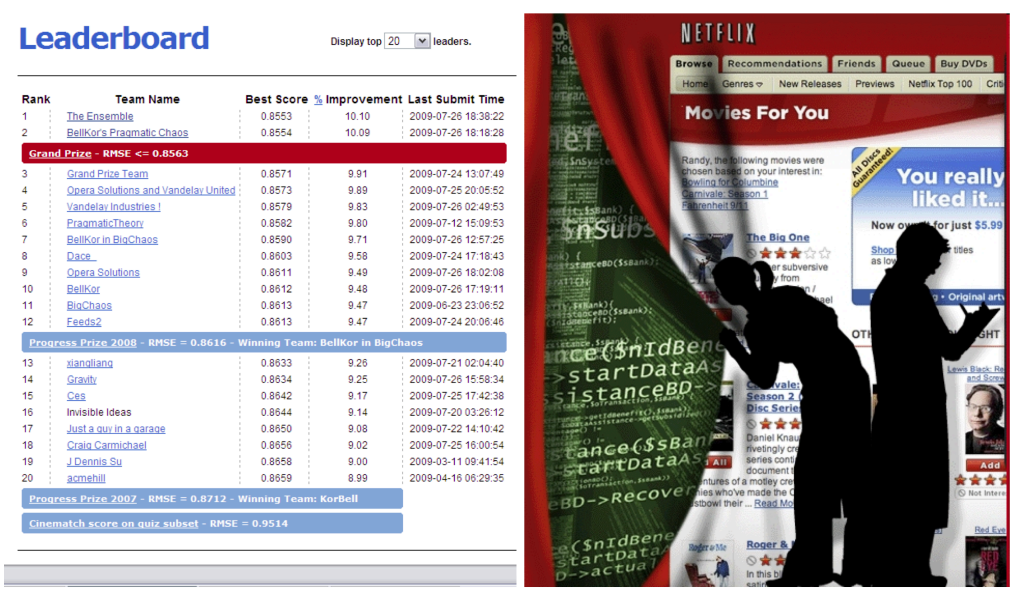
- Question: can we make weak machine learning models useful for real-world applications?
  - Yes! Ensemble learning!
- Ensemble learning: to train multiple learning models to solve the same problem
  - A.k.a. committee-based learning, or multiple classifier system
  - Construct a set of base learners, and combine their results



Ensemble Learning (Cont'd)

- Actually, ensemble learning is very powerful!
- Won KDD-Cup competitions for many times
  - A very famous data mining competition (since 1997)
  - All the first-place and second-place winners used ensemble methods (2009-2011)

Machine learning competition with a $1 million prize



Leaderboard

Display top 20 leaders.

Rank	Team Name	Best Score	% Improvement	Last Submit Time
1	The Ensemble	0.8553	10.10	2009-07-26 18:38:22
2	BellKor's Pragmatic Chaos	0.8554	10.09	2009-07-26 18:18:28
	Grand Prize - RMSE <= 0.8563			
3	Grand Prize Team	0.8571	9.91	2009-07-24 13:07:49
4	Opera Solutions and Vandelay United	0.8573	9.89	2009-07-25 20:05:52
5	Vandelay Industries !	0.8579	9.83	2009-07-26 02:49:53
6	PragmaticTheory	0.8582	9.80	2009-07-12 15:09:53
7	BellKor in BigChaos	0.8590	9.71	2009-07-26 12:57:25
8	Dace	0.8603	9.58	2009-07-24 17:18:43
9	Opera Solutions	0.8611	9.49	2009-07-26 18:02:08
10	BellKor	0.8612	9.48	2009-07-26 17:19:11
11	BigChaos	0.8613	9.47	2009-06-23 23:06:52
12	Feeds2	0.8613	9.47	2009-07-24 20:06:46
	Progress Prize 2008 - RMSE = 0.8616 - Winning Team: BellKor in BigChaos			
13	xiangliang	0.8633	9.26	2009-07-21 02:04:40
14	Gravity	0.8634	9.25	2009-07-26 15:58:34
15	Ces	0.8642	9.17	2009-07-25 17:42:38
16	Invisible Ideas	0.8644	9.14	2009-07-20 03:26:12
17	Just a guy in a garage	0.8650	9.08	2009-07-22 14:10:42
18	Craig Carmichael	0.8656	9.02	2009-07-25 16:00:54
19	J Dennis Su	0.8658	9.00	2009-03-11 09:41:54
20	acmehill	0.8659	8.99	2009-04-16 06:29:35
	Progress Prize 2007 - RMSE = 0.8712 - Winning Team: KorBell			
	Cinematch score on quiz subset - RMSE = 0.9514			

Ensemble Learning Concepts

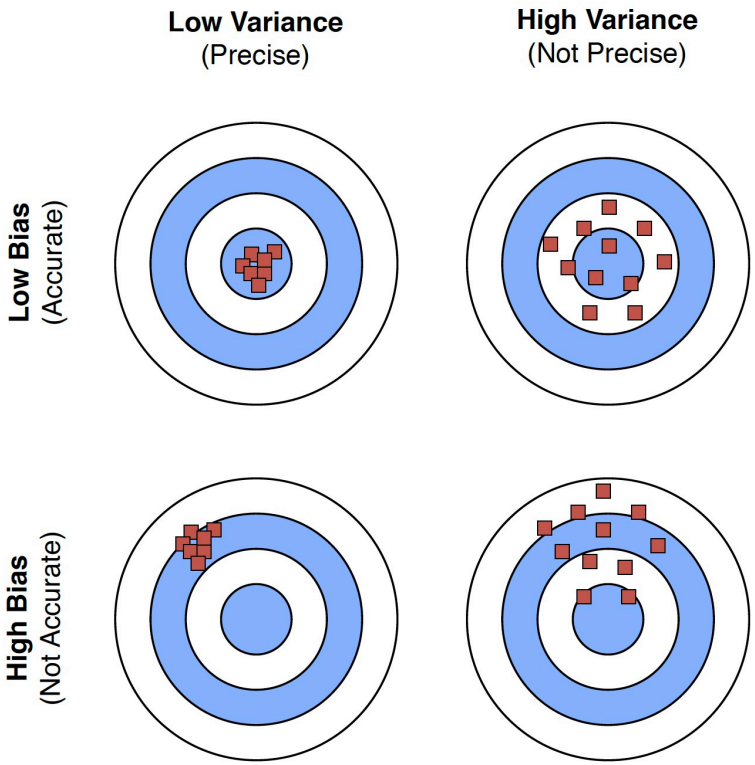
MACQUARIE University

- Homogeneous ensembles
  - Base learners are generated by the same learning algorithm
- Heterogeneous ensembles
  - Multiple learning algorithms are used for base learners
  - E.g., KNN classifier + Perceptron
- Weak learner
  - The ML models just slightly better than random guess
  - Base learners are also referred to as weak learners
  - E.g. decision stump (one-level decision tree)
- Strong learner
  - Usually makes very accurate prediction, e.g., SVM

Why It Works?

MACQUARIE University

- Prediction error decomposition
  - Bias: erroneous assumption of the model
  - Variance: error from fluctuation of training data
  - Irreducible noise
- Two ways to improve prediction performance
  - Reducing Bias
  - Reducing Variance

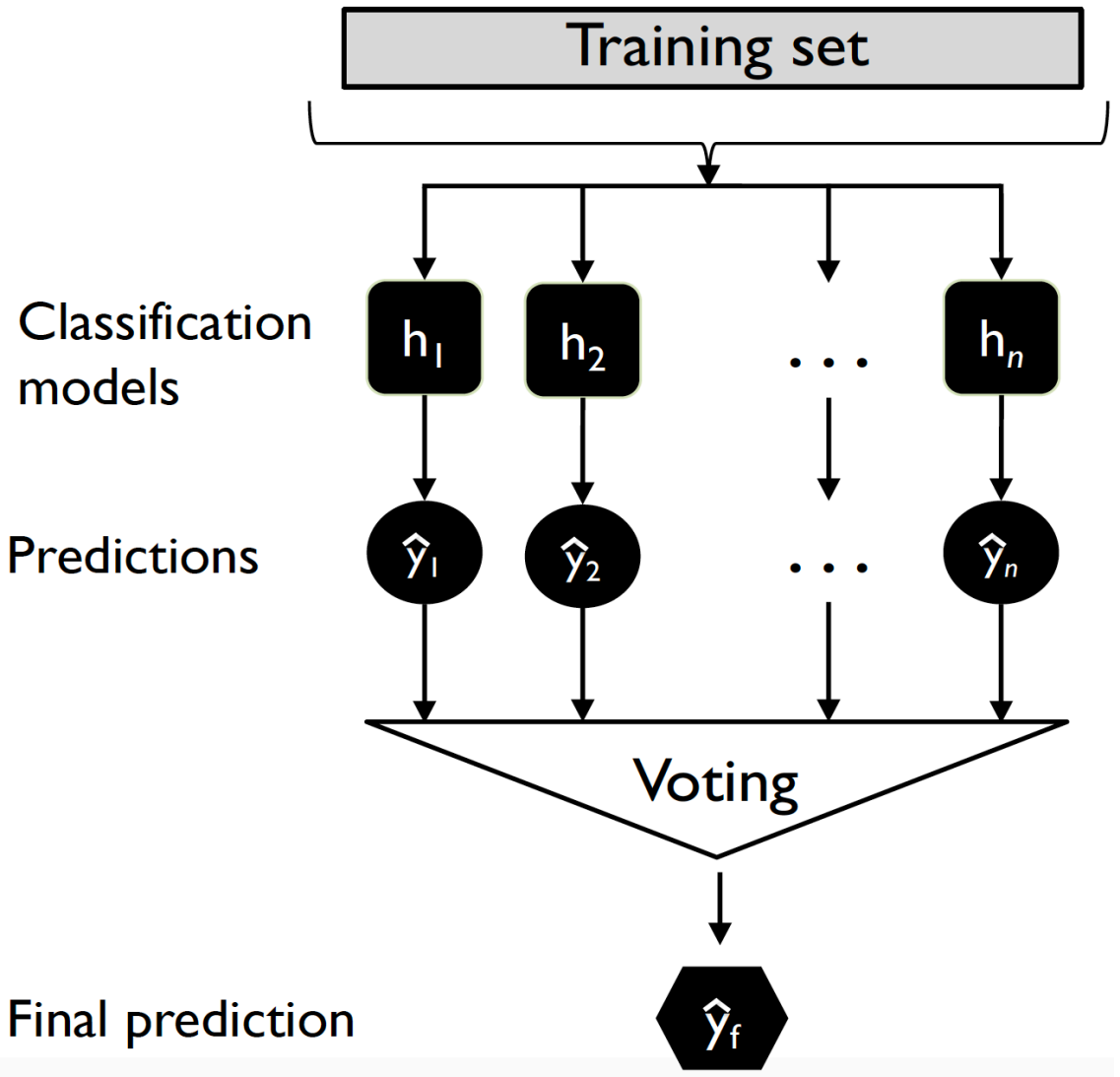


Ensemble Strategies

MACQUARIE University

- Mixture of experts
  - Both homogeneous and heterogeneous base learners
  - (Weighted) averaging and voting
    - *Boosting and bagging can be regarded as special cases*
  - Stacking (meta-learning)
- Boosting
  - Sequential homogeneous base learners
  - E.g., AdaBoost
- Bagging
  - Parallel homogeneous base learners
  - E.g., Radom Forest

Voting

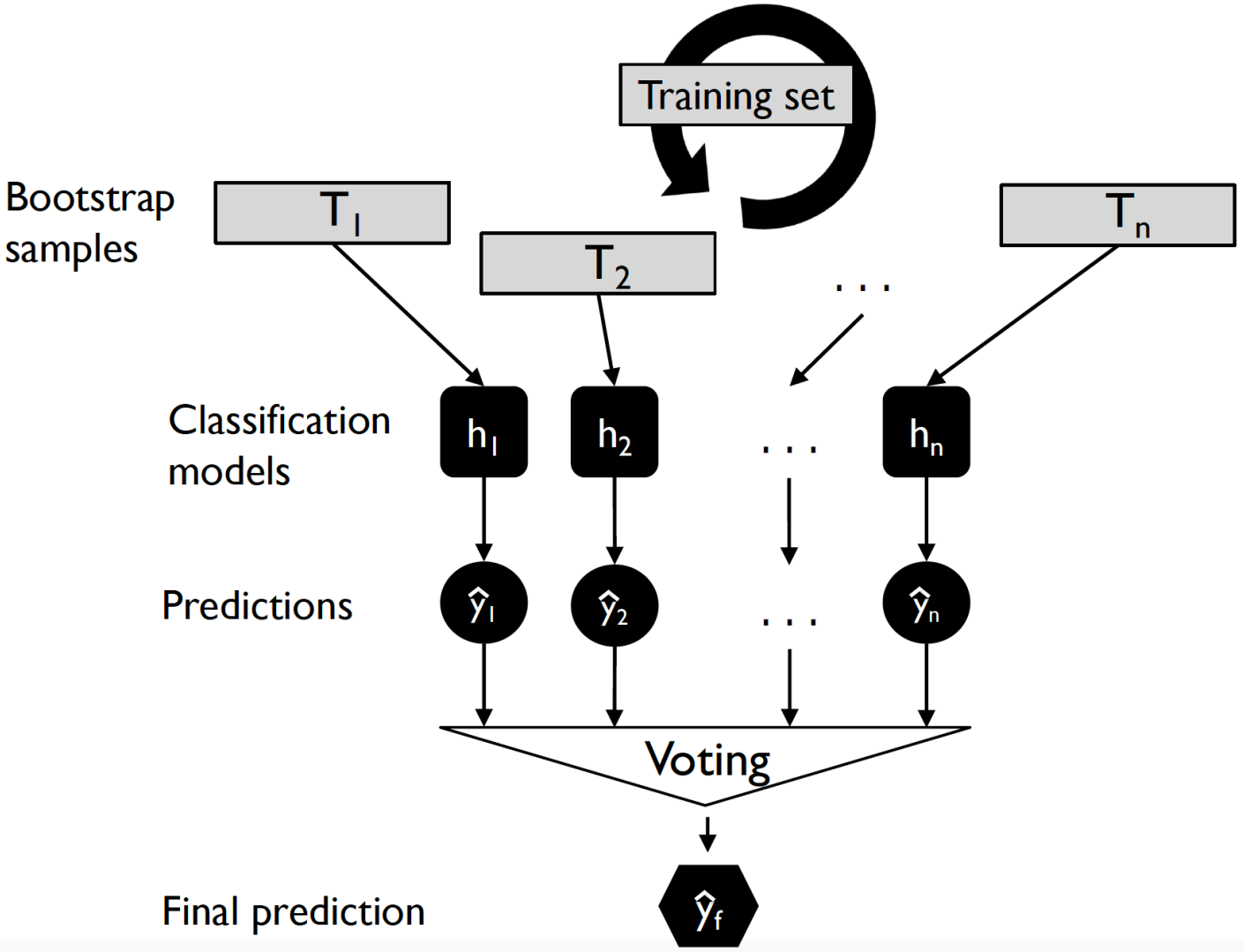

Training set
Classification models
h_1
h_2
h_n
Predictions
$\hat{y}_1$
$\hat{y}_2$
$\hat{y}_n$
Voting
Final prediction
$\hat{y}_f$

Voting/Averaging

MACQUARIE University

- Different types of voting for classification
  - Plural voting: mode, i.e., the class receives the most votes
  - Majority voting: votes should be more than half
    - Reject the prediction, otherwise
  - Weighted voting
    - Class prediction can be probabilistic (soft)
- Different types of averaging for regression
  - Averaging
  - Weighted averaging
    - All ensemble can be regarded as weighted averaging
    - Weights can be learned from data
  - Other aggregation options: max, median, etc.

Bagging


Training set
Bootstrap samples
T_1
T_2
T_n
Classification models
h_1
h_2
h_n
Predictions
$\hat{y}_1$
$\hat{y}_2$
$\hat{y}_n$
Voting
Final prediction
$\hat{y}_f$

Bootstrap Sampling

- Repeatedly draw n samples for a data set
- Around 36.8% data are not be sampled for training
  - These data can be used for validation

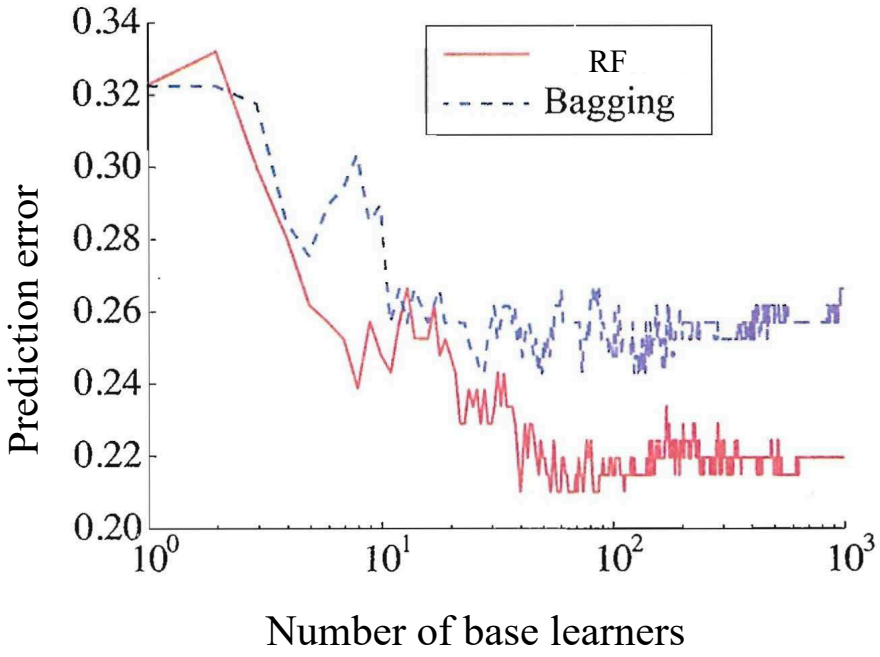
Random Forest

MACQUARIE University

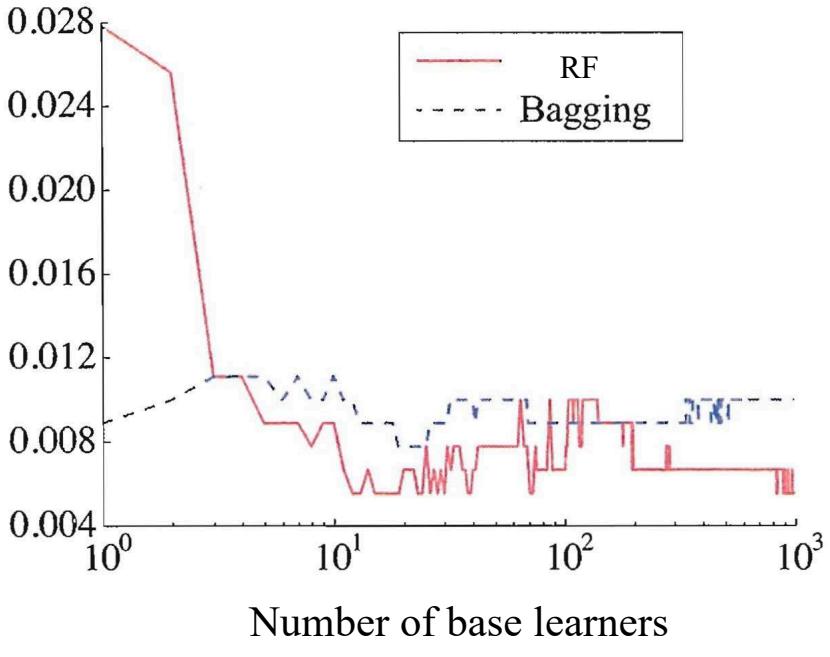
- In terms of bias-variance decomposition, bagging aims to reduce variance
  - Can work well with base learners that are sensitive to the fluctuation in data, e.g., decision tree (without pruning)
- Random forest (RF)
  - Application of bagging on decision tree models
  - Besides, feature bagging is also used: the splitting feature at every split is chosen from a random subset of k features
    - $k = 1$: randomly select a feature for splitting
    - $k = d$: the same as the original decision tree algorithms
    - "Rule-of-thumb" value: $k = \log_2 d$
  - The performance is surprisingly good

Random Forest (Cont'd)

MACQUARIE University



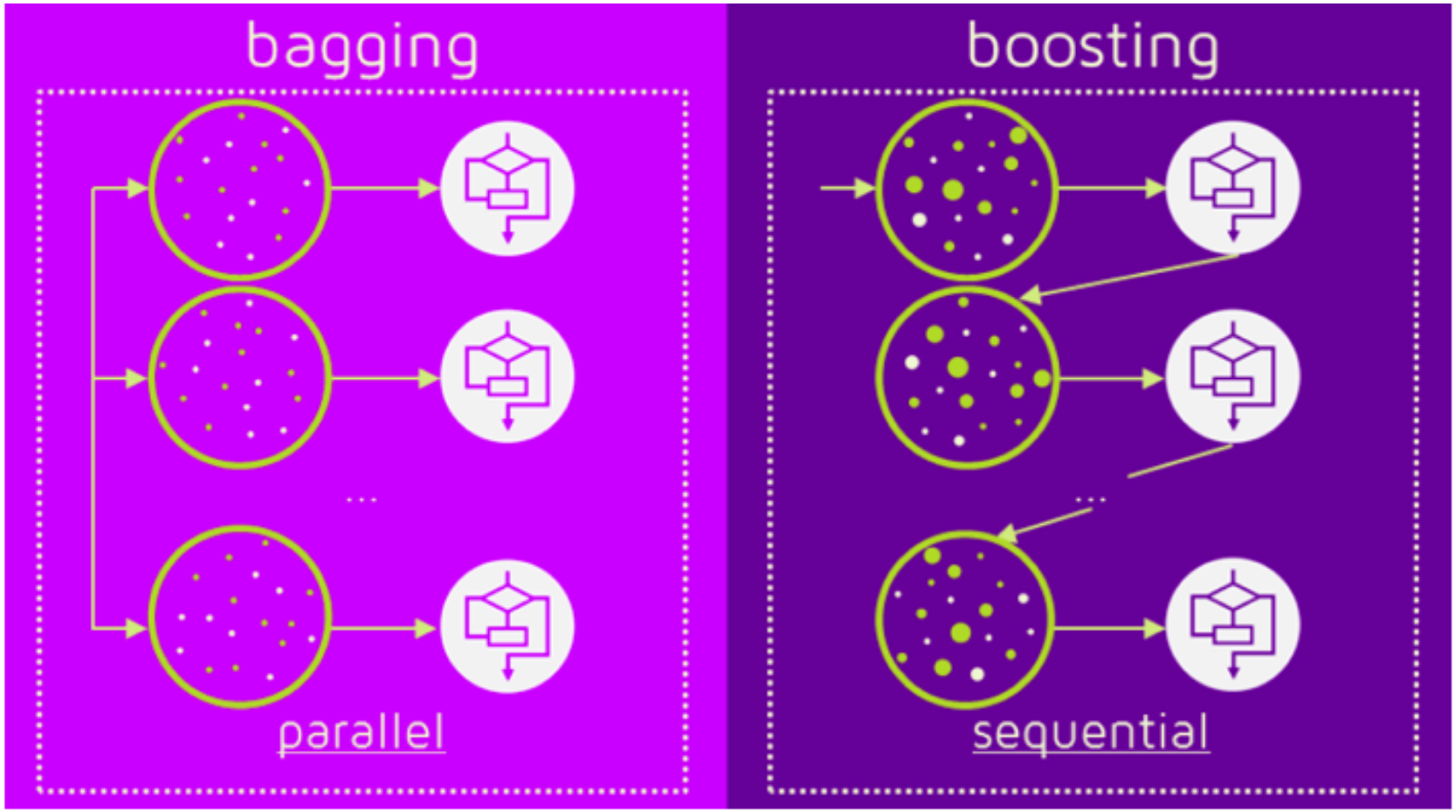
Glass dataset



Auto-mpg

Boosting

MACQUARIE University



Boosting (Cont'd)

MACQUARIE University

- Boosting is to boost weak learners to strong learners
  - Reducing the bias
- Iterative process
  - Data instances are re-weighted in each iteration
    - To focus more on "hard" (misclassified) data instances
  - What if a base learner can't learn from weighted instances?
    - Re-sampling according to the weight distribution
- Combine base learners generated in each iteration
- Two key questions:
  - How to perform re-weighting for training data?
  - How to determine weights to combine base learners?

AdaBoost

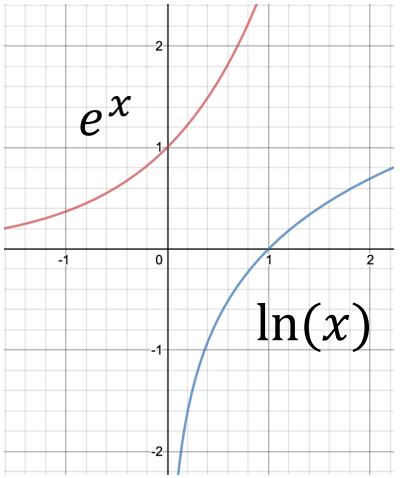
MACQUARIE University

- Two broad categories of boosting
  - Adaptive boosting (AdaBoost)
    - Subsequent weak learners are tweaked in favor of those instances misclassified by previous classifiers
    - Use the exponential error function: $E(y(x)) = e^{-y(x)t(x)}$
  - Gradient boosting
    - A more generic algorithm to find approximate solutions to additive modelling problems
    - Functional gradient descent for more error functions
- AdaBoost
  - Yoav Freund and Robert Schapire won the 2003 Gödel Prize for their work on AdaBoost (1997)

AdaBoost (Cont'd)

MACQUARIE University

- Algorithm steps
  - Initialization: round (T), sample weight distribution ($\boldsymbol{w}^{(0)}$)
  - Repeat the following steps for $1 \leq t \leq T$
    - Normalize sample weights: $\boldsymbol{w}_i^{(t)} = \boldsymbol{w}_i^{(t)} / \sum_j \boldsymbol{w}_j^{(t)}$
    - Fit the base learner with data: $h^{(t)}$
    - Compute error: $\epsilon^{(t)} = \sum_i \boldsymbol{w}_i^{(t)} \mathbf{1}(h^{(t)}(\boldsymbol{x}_i) \neq y_i)$
    - If $\epsilon^{(t)} > 0.5$, then stop/skip
    - Set base learner weight: $\alpha^{(t)} = \frac{1}{2} \ln\left(\frac{1-\epsilon^{(t)}}{\epsilon^{(t)}}\right)$

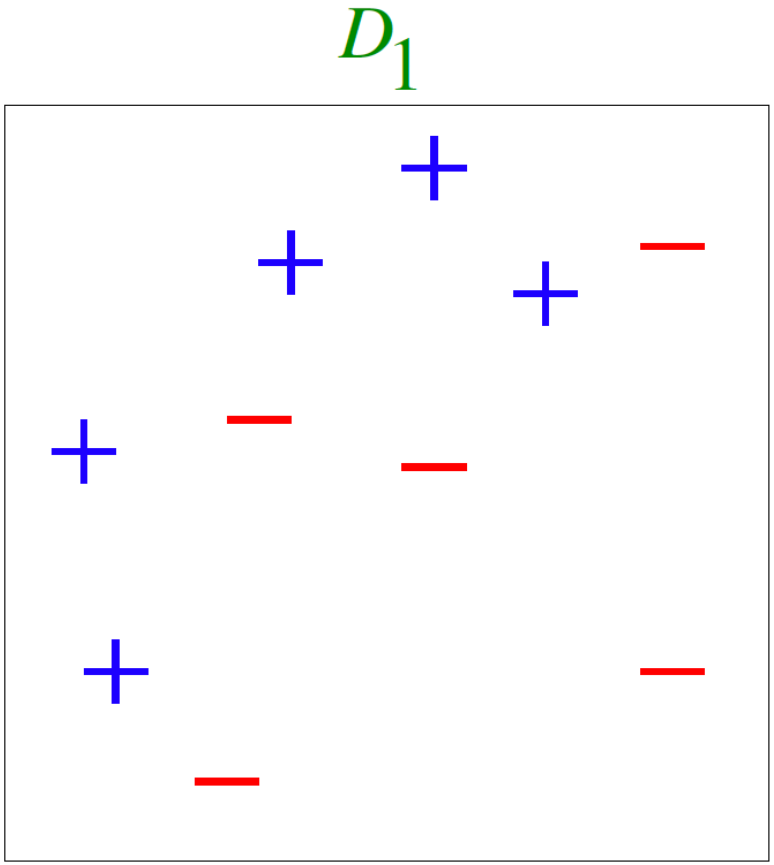


    - Update sample weight: $\boldsymbol{w}_i^{(t+1)} = \boldsymbol{w}_i^{(t)} \times \begin{cases} e^{-\alpha^{(t)}}, \text{if } h^{(t)}(\boldsymbol{x}_i) = y_i \\ e^{\alpha^{(t)}}, \text{ if } h^{(t)}(\boldsymbol{x}_i) \neq y_i \end{cases}$
  - Ensemble model: $y(\boldsymbol{x}) = \text{sgn}(\sum_{t=1}^{T} \alpha^{(t)} h^{(t)}(\boldsymbol{x}))$

AdaBoost Example

MACQUARIE University

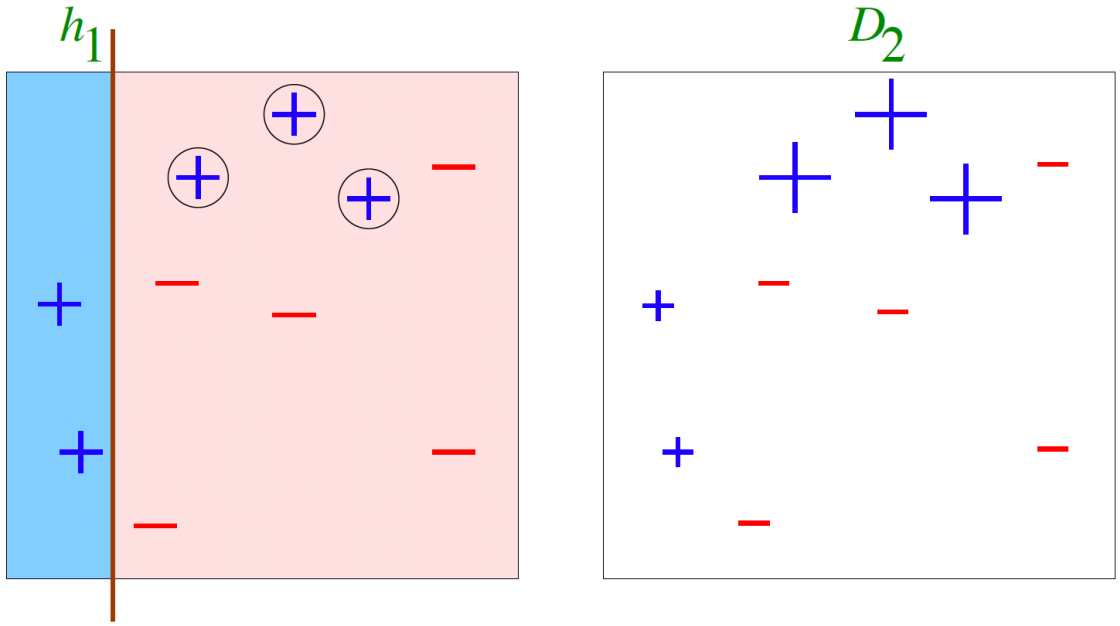
- Binary classification on 2-dimensional data (5+, 5-)



AdaBoost Example (Cont'd)

MACQUARIE University

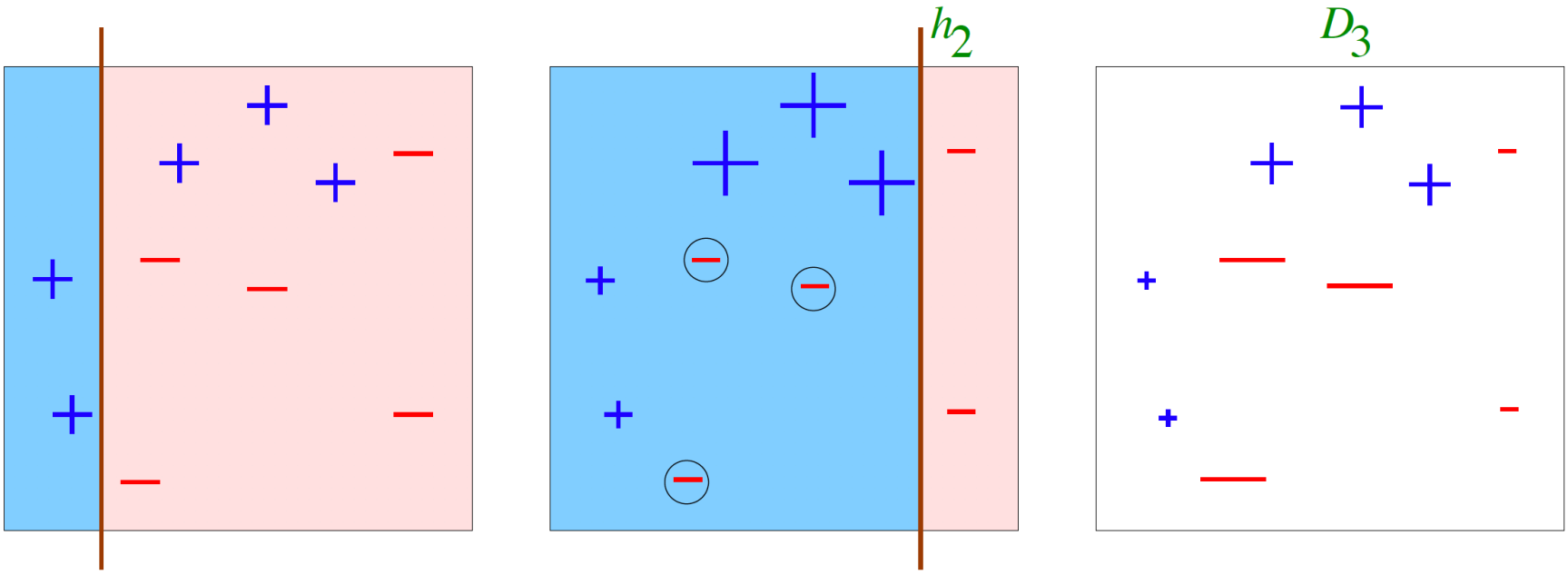
- Base learner: decision stump (one-level decision tree)
  - Round 1



AdaBoost Example (Cont'd)

MACQUARIE University

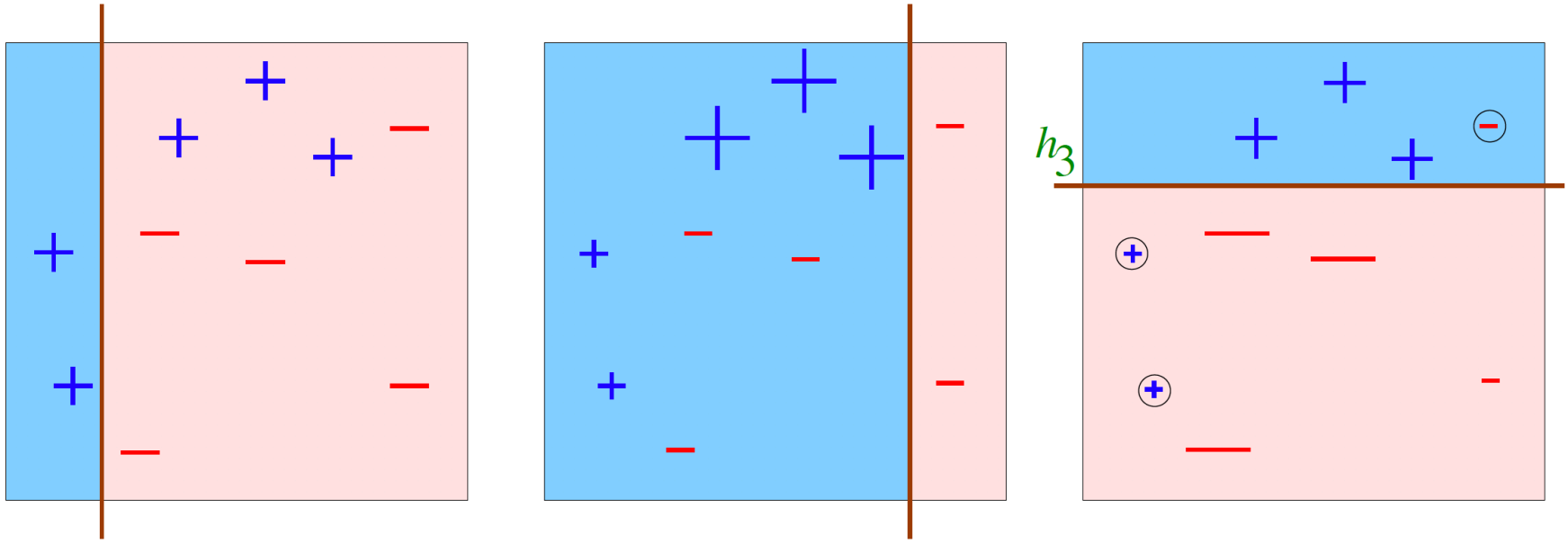
- Base learner: decision stump (one-level decision tree)
  - Round 2



AdaBoost Example (Cont'd)

MACQUARIE University

- Base learner: decision stump (one-level decision tree)
  - Round 3



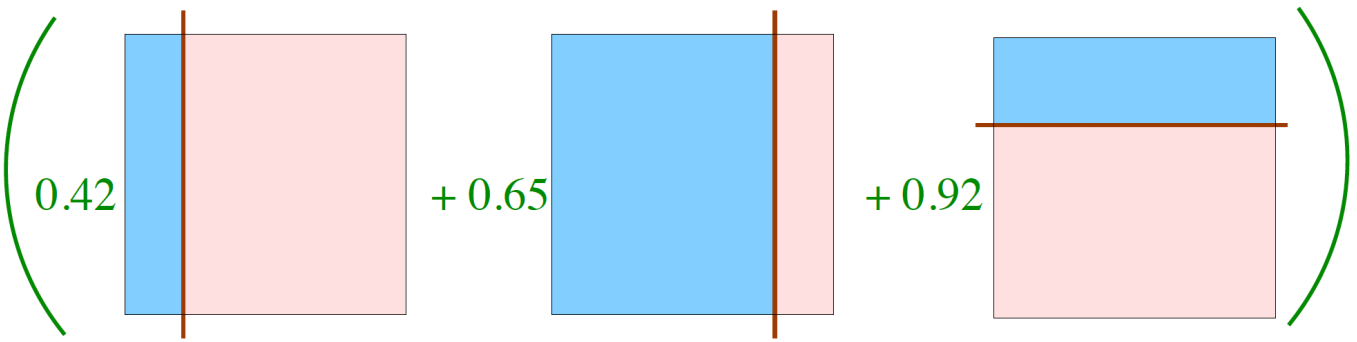
$\varepsilon_3 = 0.14$

$\alpha_3 = 0.92$

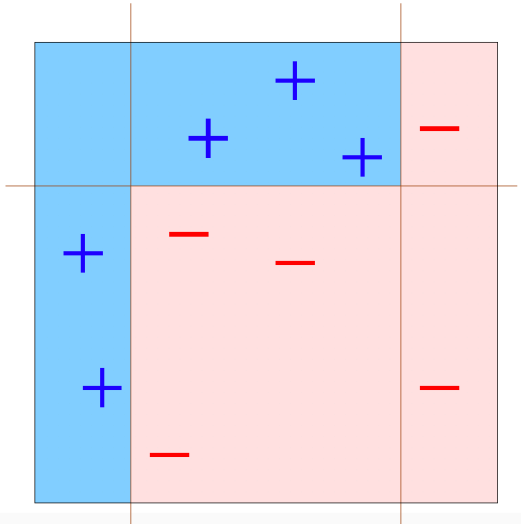
AdaBoost Example (Cont'd)

MACQUARIE University

$H_{\text{final}} = \text{sign}\left(0.42 \; + 0.65 \; + 0.92 \right)$



$=$



Stacking

- Naïve stacking (bottom)
- Stacking with CV (right)
  - To mitigate overfitting