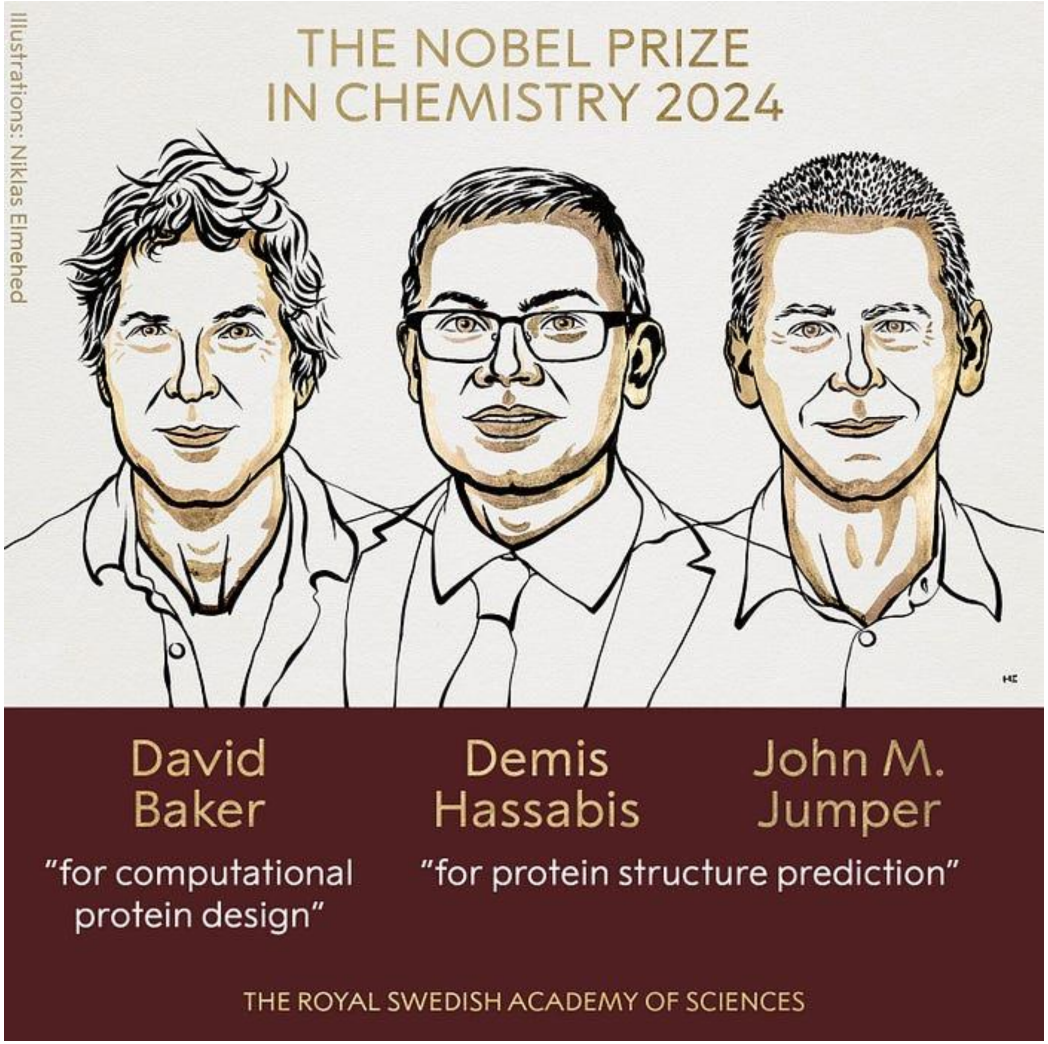
DATA SCIENCE

COMP2200/6200

11 – Decision Tree

Xuyun Zhang (School of Computing)

Another Big News!



The Conversation: https://theconversation.com/i-was-a-beta-tester-for-the-nobel-prize-winning-alphafold-ai-its-going-to-revolutionise-health-research-241084

Session 2 LEU Survey

MACQUARIE University

Provide feedback on teaching and course units

Image source: https://teche.mq.edu.au/2022/09/myth-busting-the-reporting-of-learner-experience-of-unit-surveys/

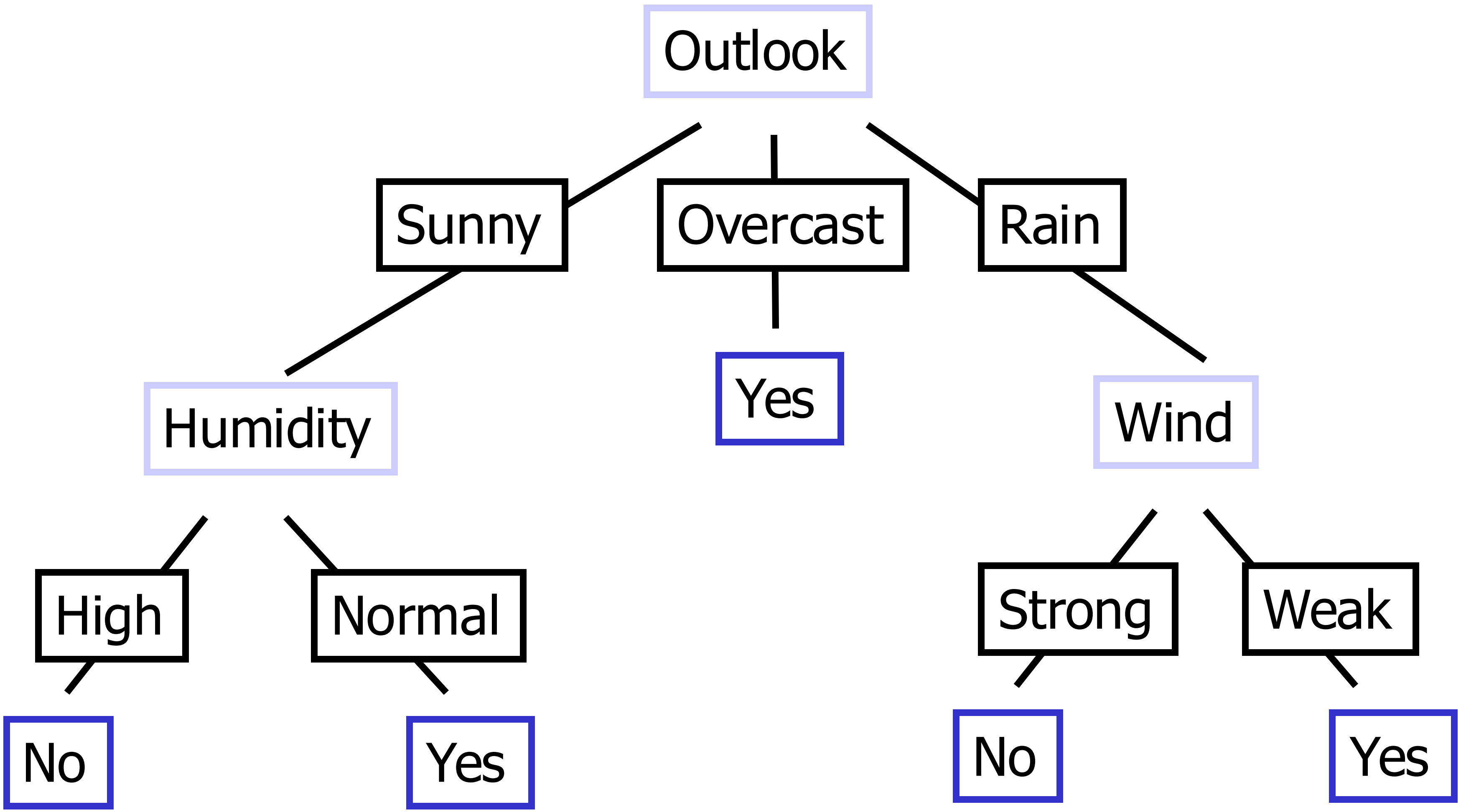
Lecture Outline

MACQUARIE University

- Decision Tree
  - Tree Structure
  - Splitting Heuristics
- Practical
  - DecisionTreeClassifier
  - DecisionTreeRegressor

Decision Making Example

- Example: play tennis based on the weather

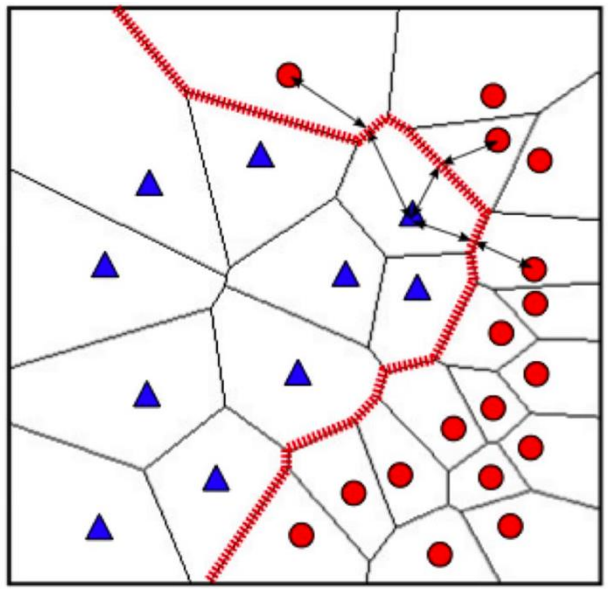


Example: Rule Extraction

- 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

Revisit 1-NN Classifier

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

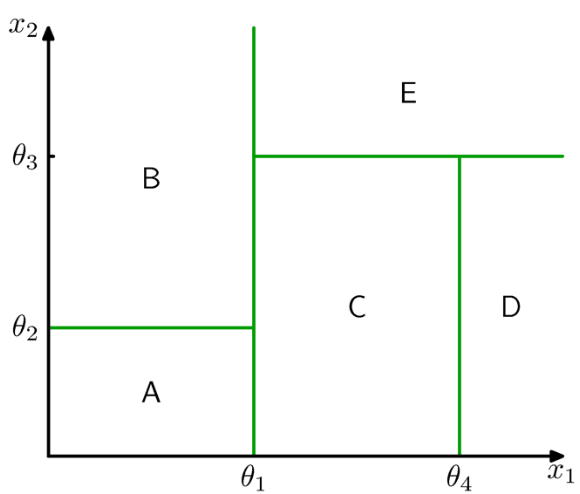


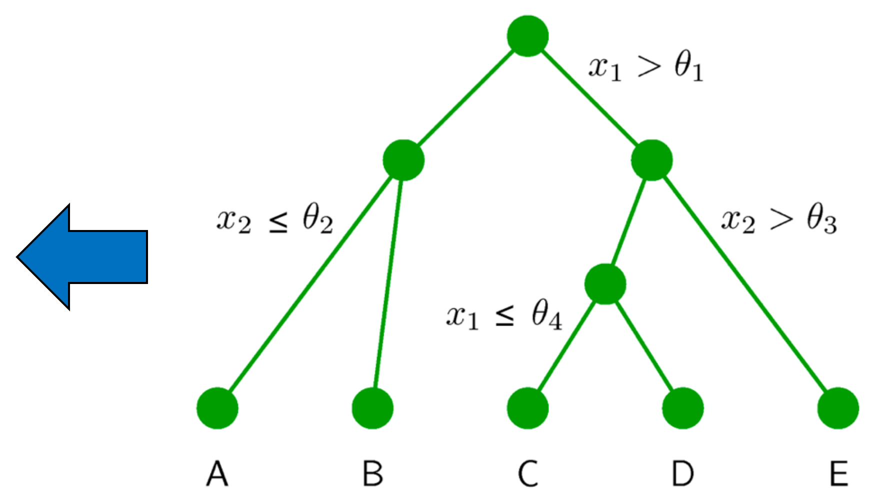
❖ Question: other ways of partitioning feature space?

- Instance-based learning with more regular partitions?
- Can we control the way of generating the partitions?

Decision Trees

- 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

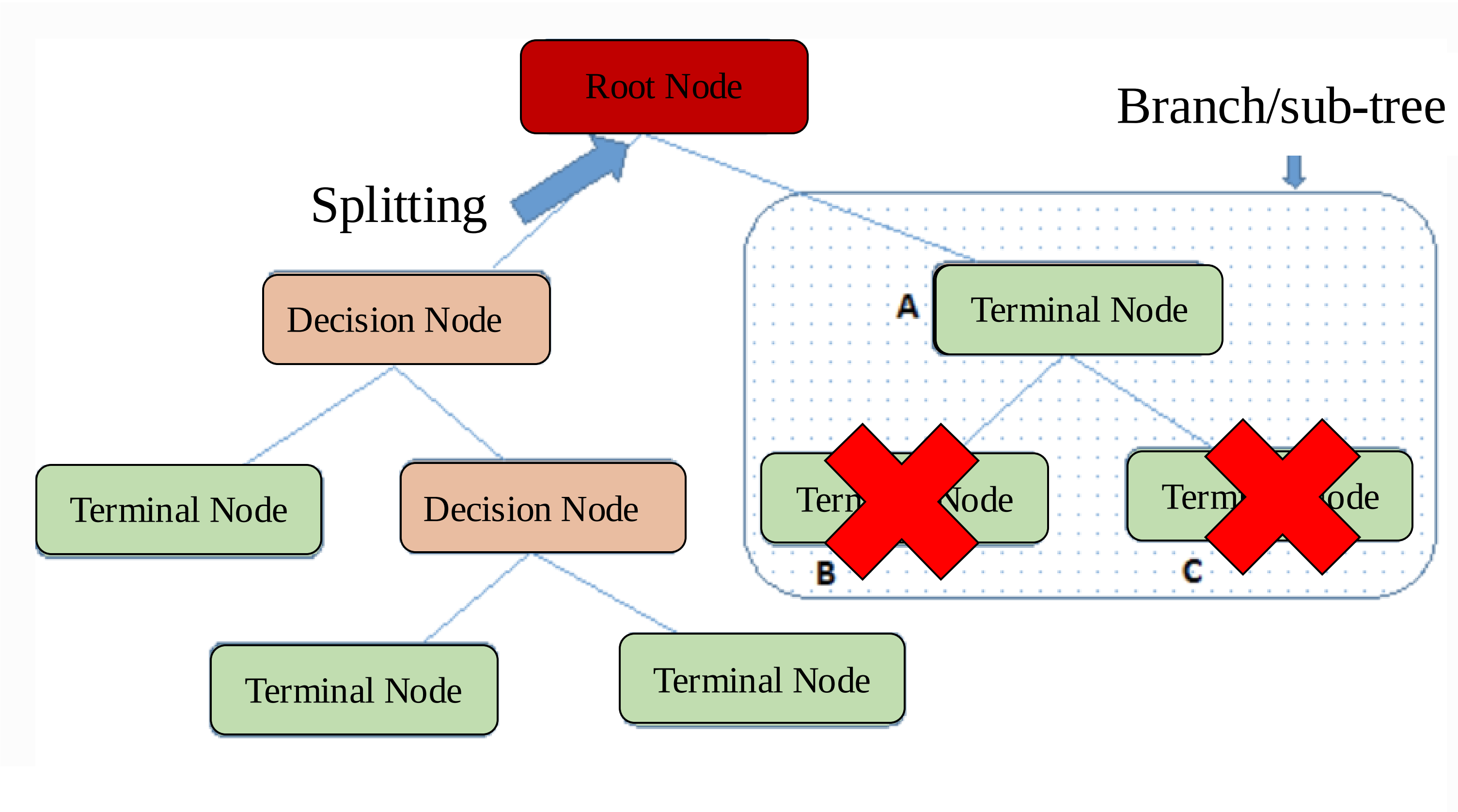




Decision Trees (Cont'd)

- 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



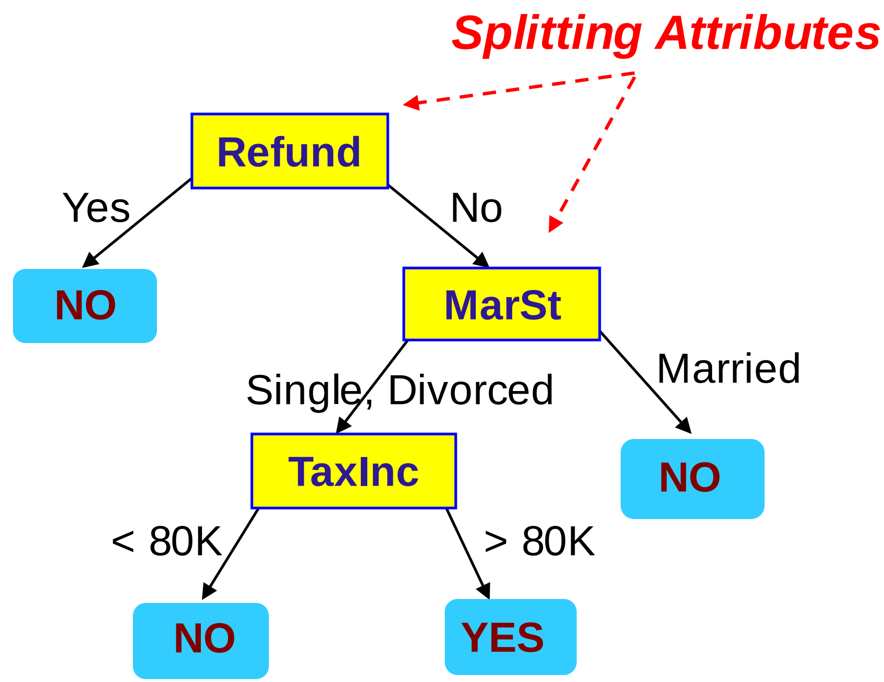
- Pruning

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

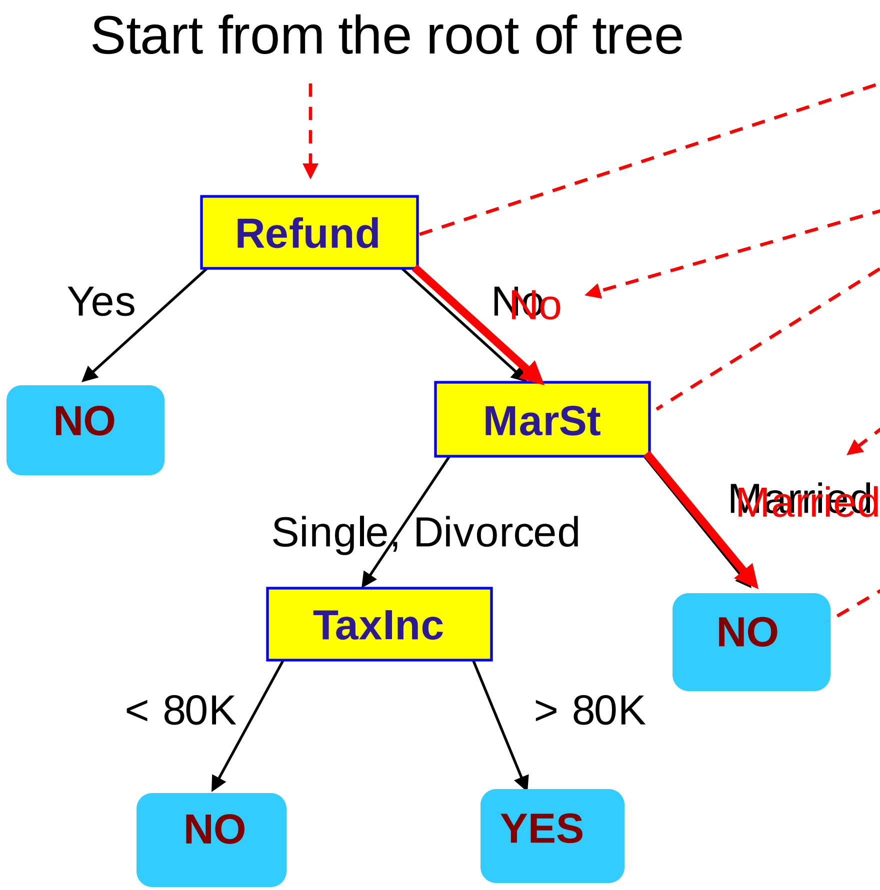
Training Data



Model: Decision Tree

Testing

- Predicting: traverse the built decision tree for a region



Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

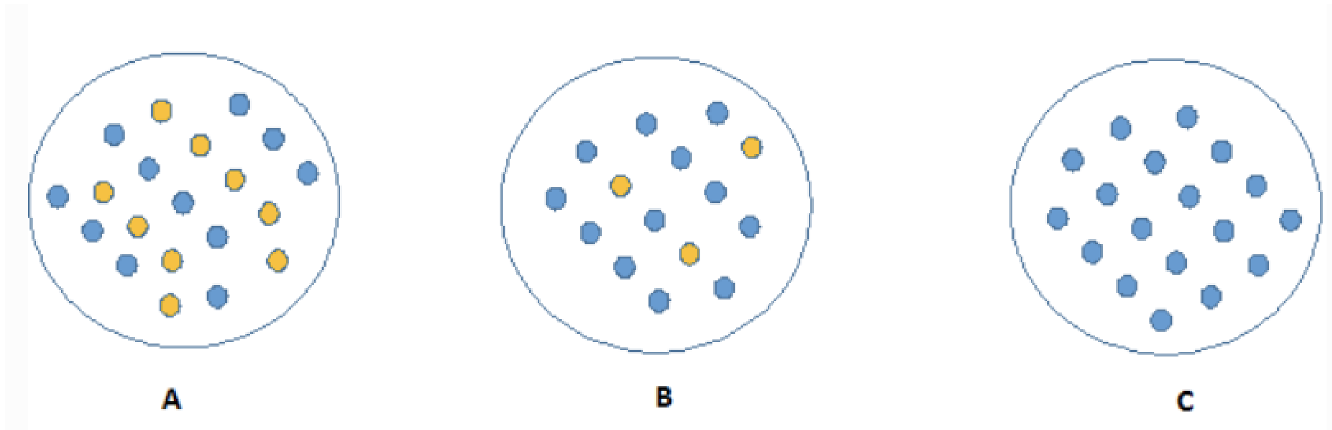
Test Data

How to Partition Data ?

- 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

- 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.

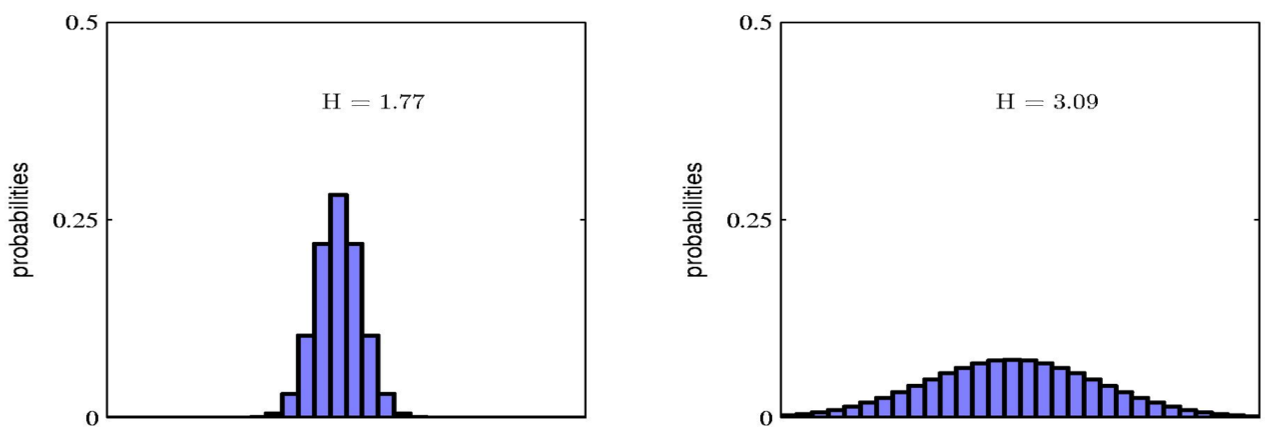
Entropy

- 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

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

Class	$\mathcal{C}_1$	$\cdots$	$\mathcal{C}_K$	**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

- 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

$$GainRatio_{\mathcal{D}}(\phi_j) = Gain_{\mathcal{D}}(\phi_j)/SplitInfo_{\mathcal{D}}(\phi_j)$$

Gini impurity

- 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

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

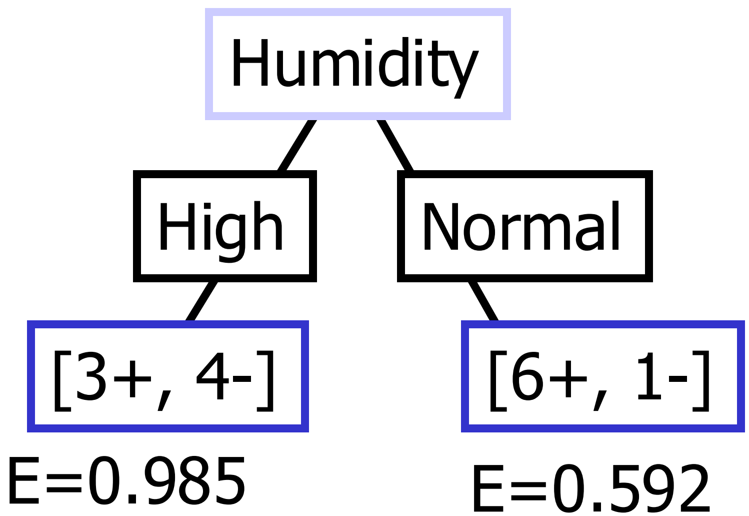
- The lower the better for splitting
  - Zero Gini impurity implies perfect classification
- Selection of splitting heuristic is data dependent

Example: Data

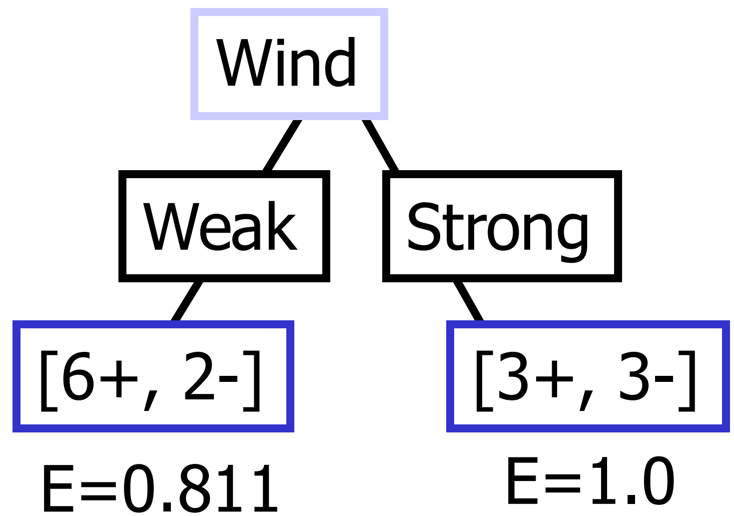
Day	Outlook	Temp.	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Weak	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cold	Normal	Weak	Yes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

- 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

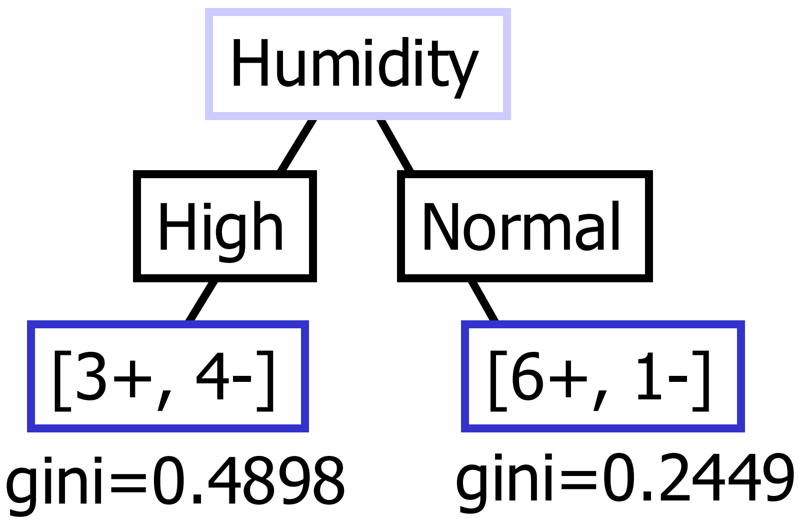


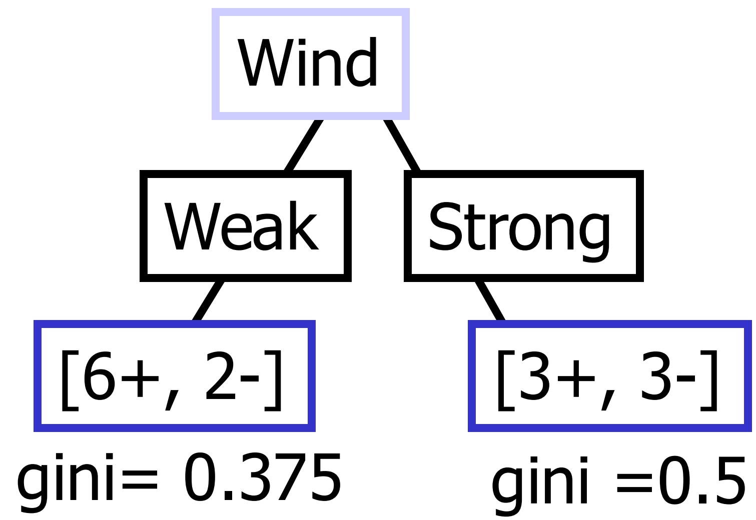
Gain(S,Wind)
=0.940-(8/14)*0.811
– (6/14)*1.0 = 0.048

- Humidity provides greater information gain than Wind

Example: Gini Impurity

- 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

Continuous Features

- How to calculate impurity for continuous features?
  - Discretization
- Binary discretization
  - Sort feature values, $v_1, \dots, v_i, v_{i+1}, \dots, v_n$, where $v_i \leq v_{i+1}$
  - A feature is split into two parts by a threshold $t_i = \frac{v_i + v_{i+1}}{2}$
  - Calculate the impurity for the two discrete values
  - There are $n-1$ possible splitting thresholds to choose
- A continuous feature can be further chosen to be split for many times on demand
  - Different from a discrete feature

Continuous Features (Cont'd)

- Example, given a dataset D

Instance	1	2	3	4	5
Temperature	18°C	19°C	22°C	24°C	27°C
playTennis	No	Yes	Yes	Yes	No

  - Threshold $t = 20$°C
  - $\{\text{Temp.} < t\} = \{1, 2\}$ and $\{\text{Temp.} \geq t\} = \{3, 4, 5\}$
  - $Info(D) = -\left(\frac{2}{5}\log_2\frac{2}{5} + \frac{3}{5}\log_2\frac{3}{5}\right)$
  - $Info\left(D_{Temp.<t}\right) = -(\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2})$
  - $Info\left(D_{Temp.\geq t}\right) = -(\frac{2}{3}\log_2\frac{2}{3} + \frac{1}{3}\log_2\frac{1}{3})$
  - $Info_D(t) = \frac{2}{5}Info\left(D_{Temp.<t}\right) + \frac{3}{5}Info\left(D_{Temp.\geq t}\right)$

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

- 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

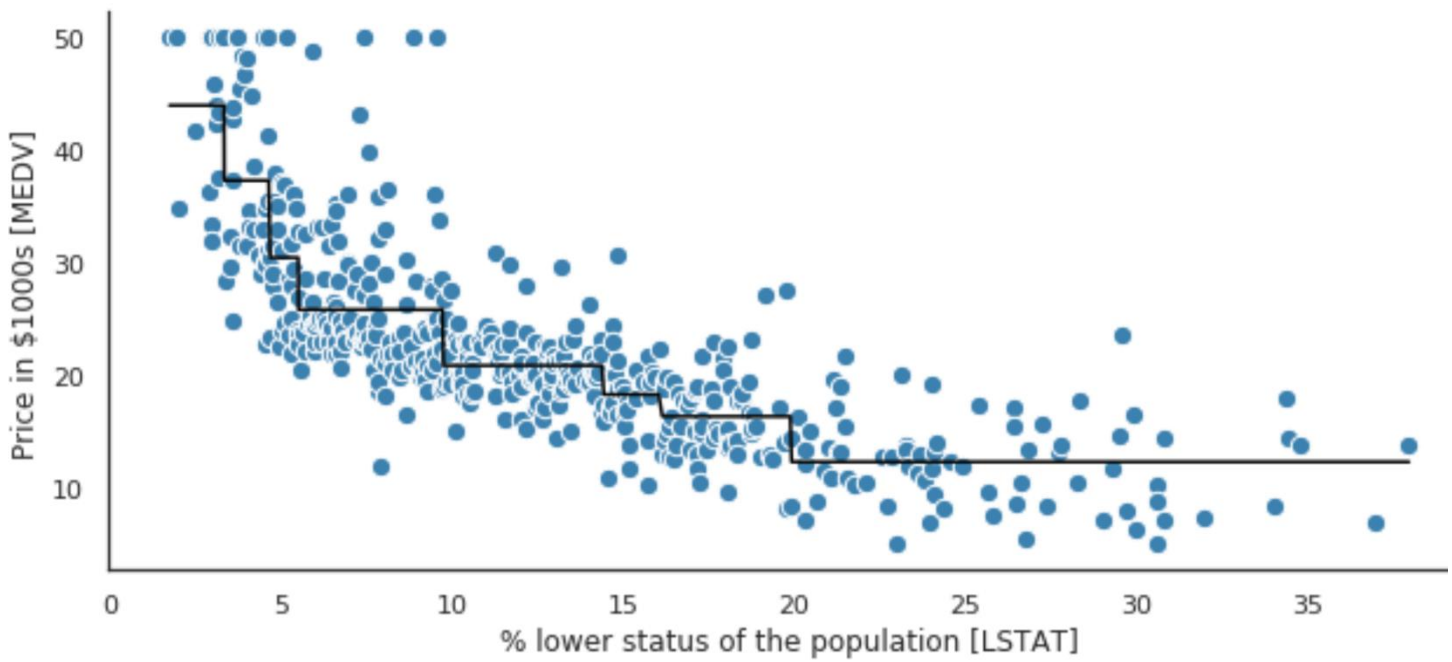
Continuous Target

- 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): $y = f(x) = w_0$



Decision Tree Algorithm

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

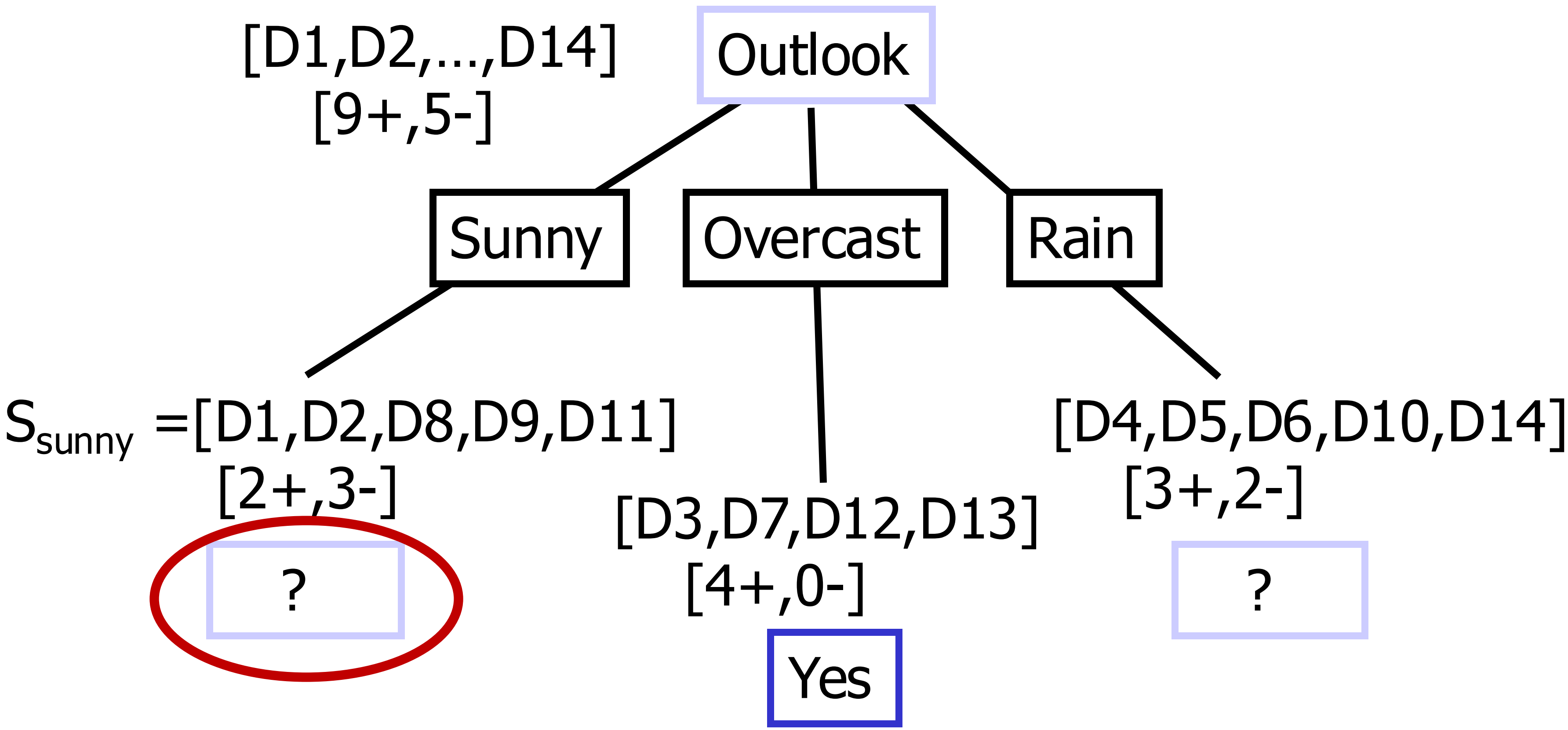
Input: $\mathcal{D} = \{\langle \phi(\boldsymbol{x}_1), t_1 \rangle, \cdots, \langle \phi(\boldsymbol{x}_N), t_N \rangle\}$, ϕ is M-dimensional.
Output: A (sub-)tree with root node $Node_{\text{root}}$.

```
if D is "pure" or other stopping criteria met then
    return Node_root ← leafNode(D)
for i←1 : M do
    h_i ← compute heuristics (e.g., gain ratio) if we split φ_i
⟨φ_best, Π_best⟩←max_{φ_i}{h_i}, Π_best is the best partition for φ_i
{D_1, ··· , D_V} ← D, i.e., partition D by Π_best
for i←1 : V do
    Node_i←buildTree(D_i)
Node_root ← decisionNode(φ_best, Π_best, {Node_1, ··· , Node_V})
return Node_root
```

Typical Algorithms

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

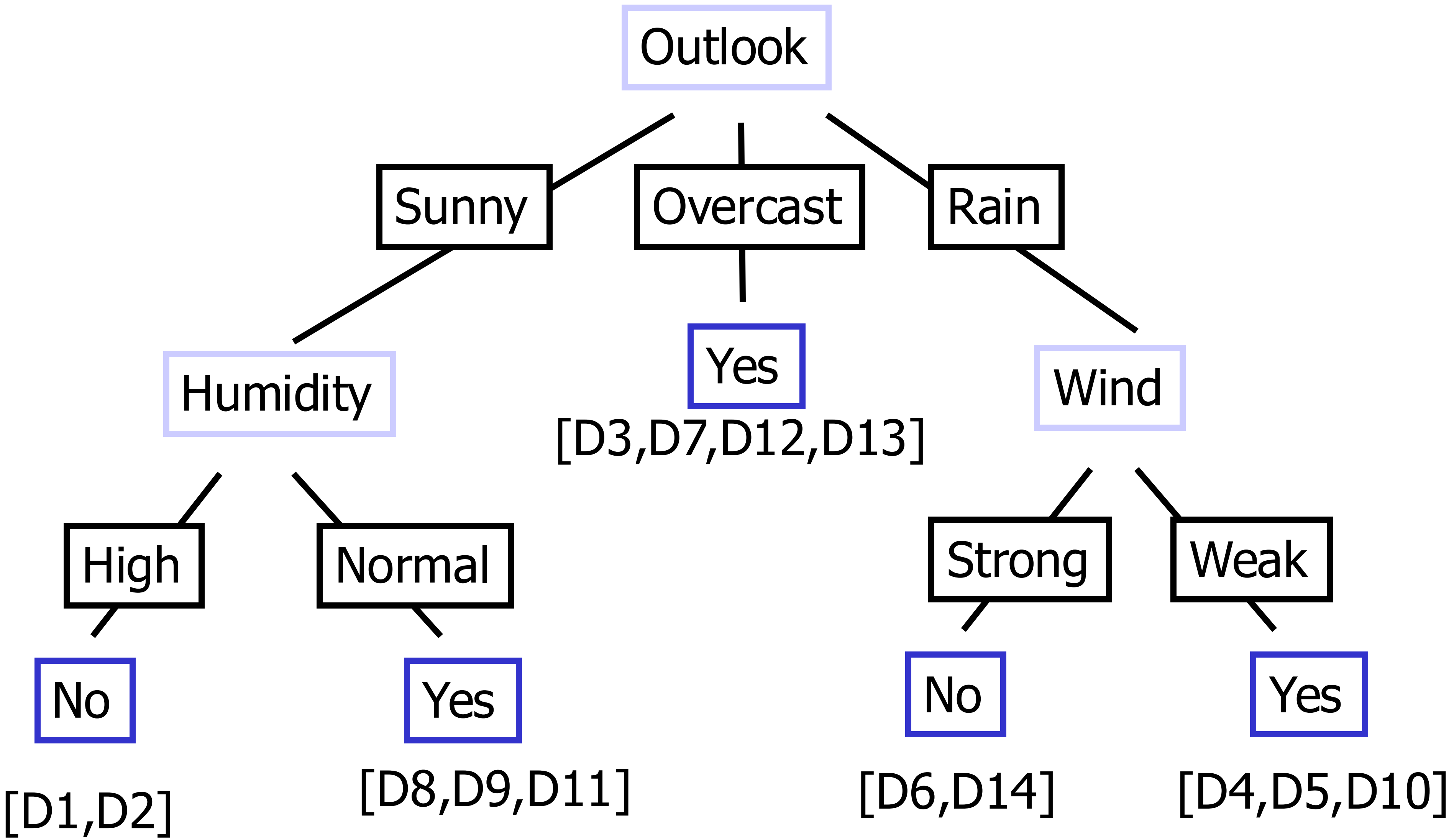


$\text{Gain}(S_{sunny}, \text{Humidity}) = 0.970 - (3/5)0.0 - 2/5(0.0) = 0.970$

$\text{Gain}(S_{sunny}, \text{Temp.}) = 0.970 - (2/5)0.0 - 2/5(1.0) - (1/5)0.0 = 0.570$

$\text{Gain}(S_{sunny}, \text{Wind}) = 0.970 = -(2/5)1.0 - 3/5(0.918) = 0.019$

Example: Algorithm (Cont'd)



Lecture Outline

MACQUARIE University

- Decision Tree
  - Tree Structure
  - Splitting Heuristics
- Practical
  - DecisionTreeClassifier
  - DecisionTreeRegressor

DecisionTreeClassifier

MACQUARIE University

- *class* sklearn.tree.**DecisionTreeClassifier**(*, *criterion='gini'*, *splitter='best'*, *max_depth=None*, *min_samples_split=2*, *min_samples_leaf=1*, *min_weight_fraction_leaf=0.0*, *max_features=None*, *random_state=None*, *max_leaf_nodes=None*, *min_impurity_decrease=0.0*, *class_weight=None*, *ccp_alpha=0.0*, *monotonic_cst=None*)
- https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html#sklearn.tree.DecisionTreeClassifier
- https://scikit-learn.org/stable/modules/tree.html#tree

DecisionTreeClassifier (Cont'd)

- **criterion**{*"gini", "entropy", "log_loss"*}, *default="gini"*
  - The function to measure the quality of a split.
  - "gini" for the Gini impurity
  - "log_loss" and "entropy" both for the Shannon information gain
- **max_depth**, *default=None*
  - The maximum depth of the tree.
  - If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.

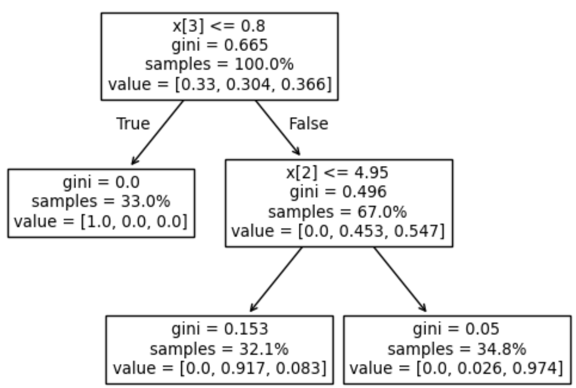
DecisionTreeClassifier (Cont'd)

MACQUARIE University

- **min_samples_split**, *default=2*
  - The minimum number of samples required to split an internal node.
- **min_samples_leaf**, *default=1*
  - The minimum number of samples required at a leaf node.
  - A split point at any depth will only be considered if it leaves at least min_samples_leaf training samples in each of the left and right branches.
  - This may have the effect of smoothing the model, especially in regression.
- **max_leaf_nodes**, *default=None*
  - Grow a tree with max_leaf_nodes in best-first fashion.

DecisionTreeClassifier Attributes

- feature_importances_
  - Feature importances: The importance of a feature is computed as the (normalized) total reduction of the criterion brought by that feature, aka, Gini importance.
- **tree_** *Tree instance*
  - The underlying Tree object.



DecisionTreeRegressor

- *class* sklearn.tree.**DecisionTreeRegressor**(**, criterion='squared_error', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_decrease=0.0, ccp_alpha=0.0, monotonic_cst=None*)
- https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html#sklearn.tree.DecisionTreeRegressor
- https://scikit-learn.org/stable/modules/tree.html#tree

DecisionTreeRegressor (Cont'd)

- **criterion**{*"squared_error", "friedman_mse", "absolute_error", "poisson"*}, *default="squared_error"*
  - The function to measure the quality of a split.
  - "squared_error" for the mean squared error
    - equal to variance reduction as feature selection criterion and minimizes the L2 loss using the mean of each terminal node
  - "friedman_mse", which uses mean squared error with Friedman's improvement score for potential splits.
  - "absolute_error" for the mean absolute error, minimizing the L1 loss using the median of each terminal node
  - "poisson" uses reduction in the half mean Poisson deviance to find splits.

Demos

- Examples in Week 13's Practical tasks

Summary

- Space partition and decision trees
- Root node, decision node, and terminal node
- Impurity
  - Entropy and Gini index
- Information gain and gain ratio
- Discretization for continuous feature
- Tree model complexity, pre-/post-pruning
- Regression tree model for continuous targets
- ID3, C4.5, and CART algorithms