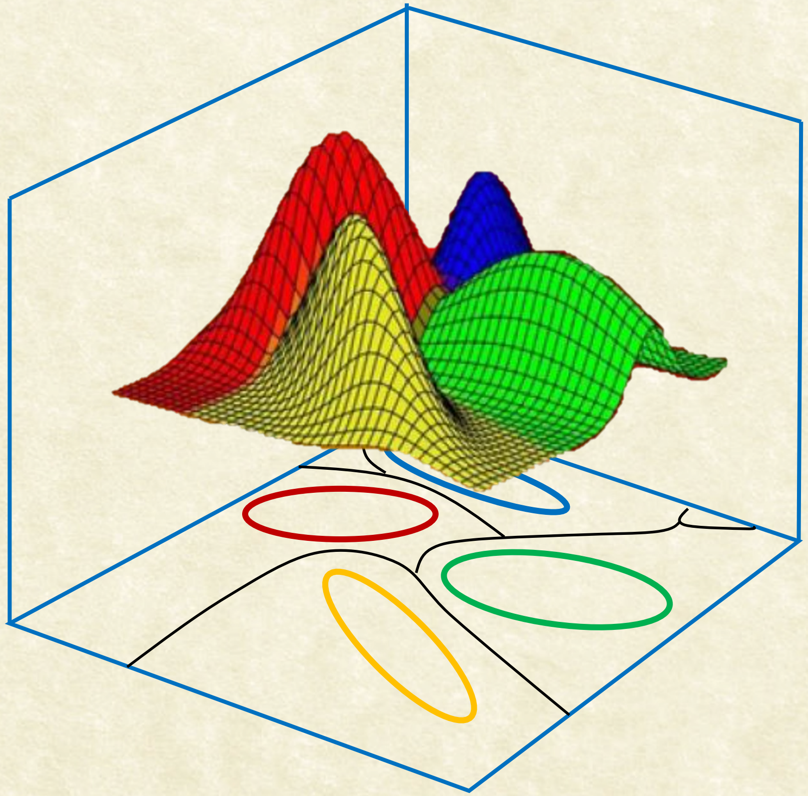
CS7.403: Statistical Methods in AI

Monsoon 2022: Decision Trees



Anoop M. Namboodiri

Biometrics and Secure ID Lab, CVIT,

IIIT Hyderabad

Decision Trees: A Recap

Let’s play a game: Guess the Animal

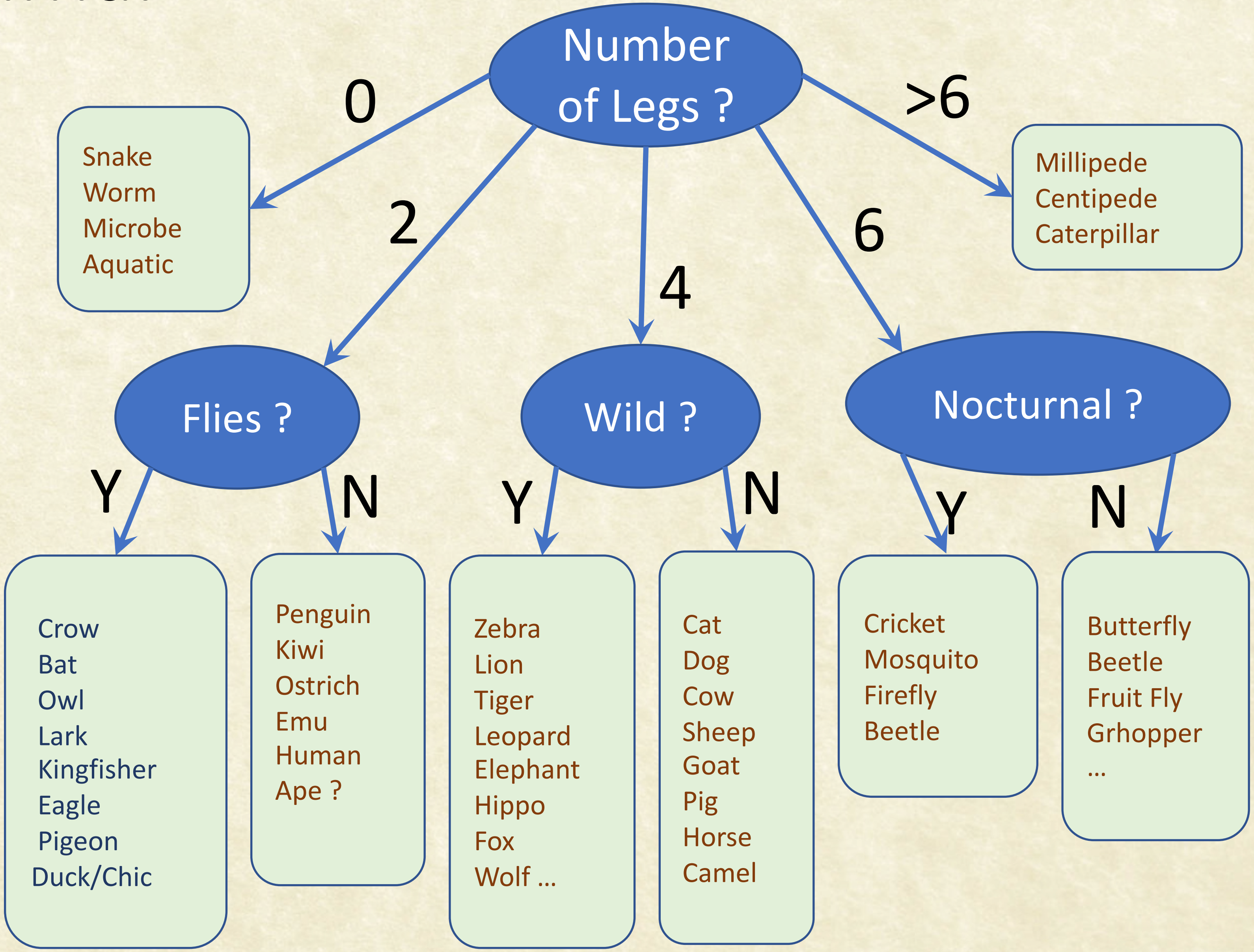
- I have an animal picture. [not just these]
- You can ask a set of questions. Can you guess the animal based on my answers?
- What question to ask?
  - Number of Legs
  - Ability to fly
  - Wild / Domesticated
  - Nocturnal or not
  - Land/Aquatic/Amphibian
  - Has fur/feather or not
  - Farm Animal or not
  - Vertebrate or not




Guess the Animal

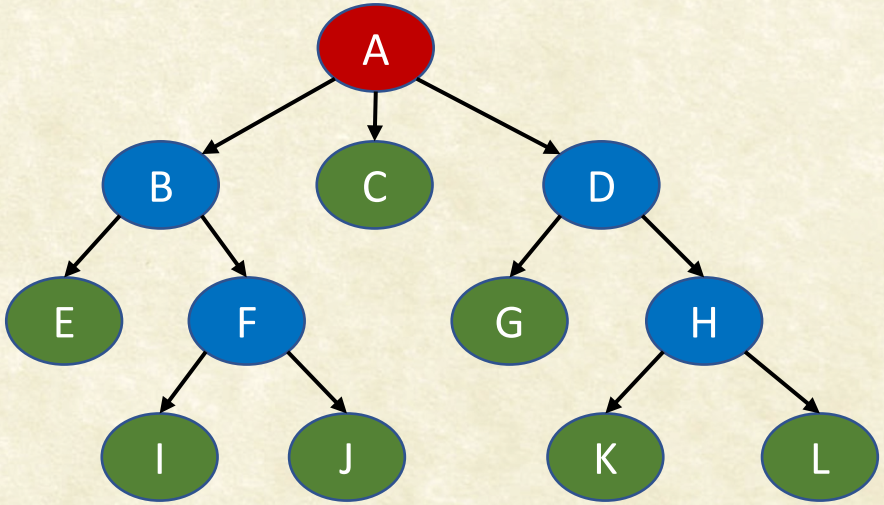
<u>Questions</u>

- How many legs?
- Does it fly?
- Is it a wild animal?
- Is it nocturnal?
- Fur/Feather?
- Farm Animal
- Is it a vertebrate?
- Is it a land animal?



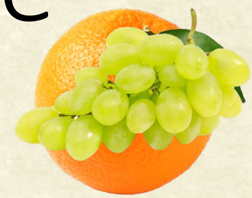
Tree: Terminologies

- **Node**: Each element of a set that forms the tree [vertices]
- **Edge**: A line between two connected nodes
- **Root**: A special node with no incoming edges
- **Leaf**: A node with no outgoing edges
- **Internal Node**: A node that is not a root or leaf
- **Depth**: Distance from a node to the root
- **Height**: Max distance to a leaf

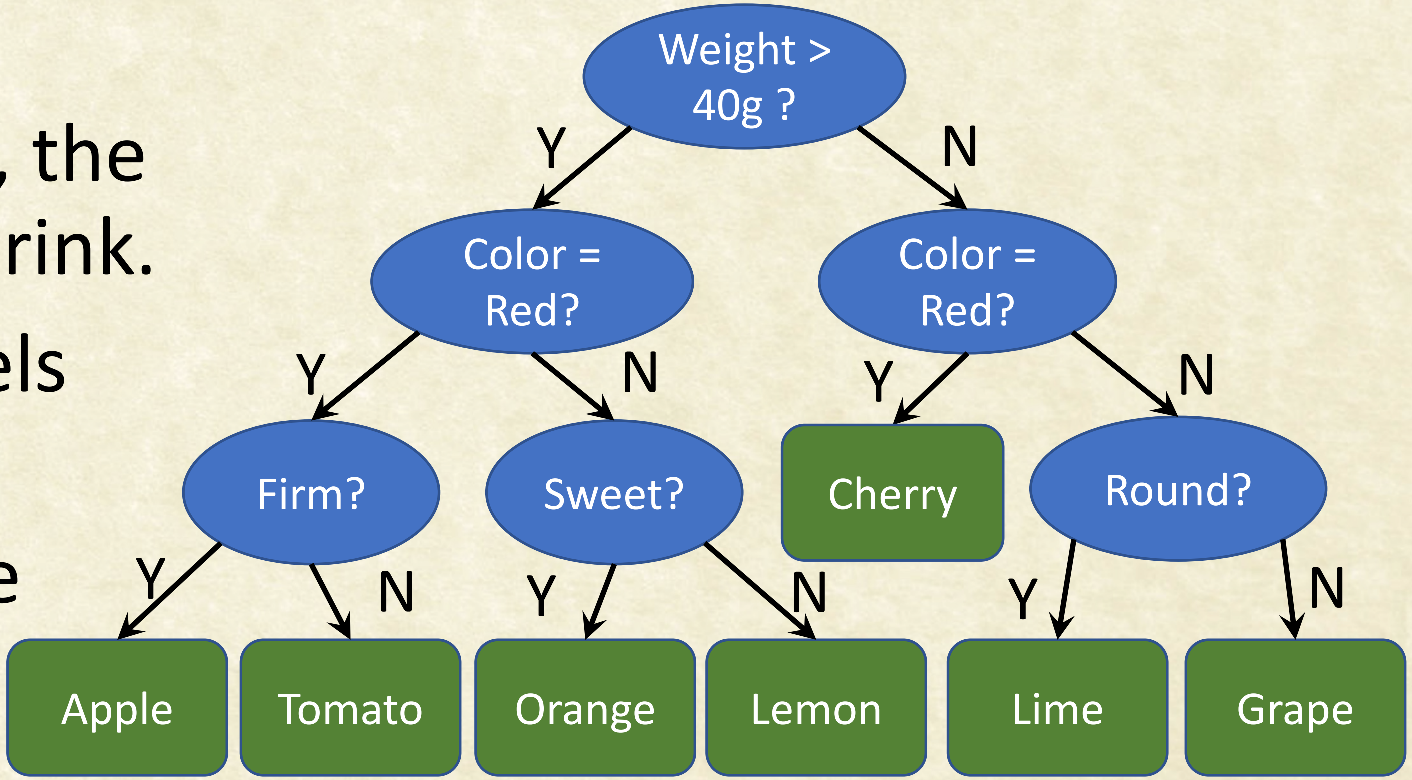


- **Path**: A sequence of nodes with edges between consecutive pairs
- **Parent, Child**: Nodes with edge from parent to child.
- **Ancestor, Descendant**: Path from ancestor to descendant.
- **Siblings**: Nodes with same parent
- **Degree**: Number of edges from a node (in and out) [branching factor; binary tree]

Classification using a Decision Tree

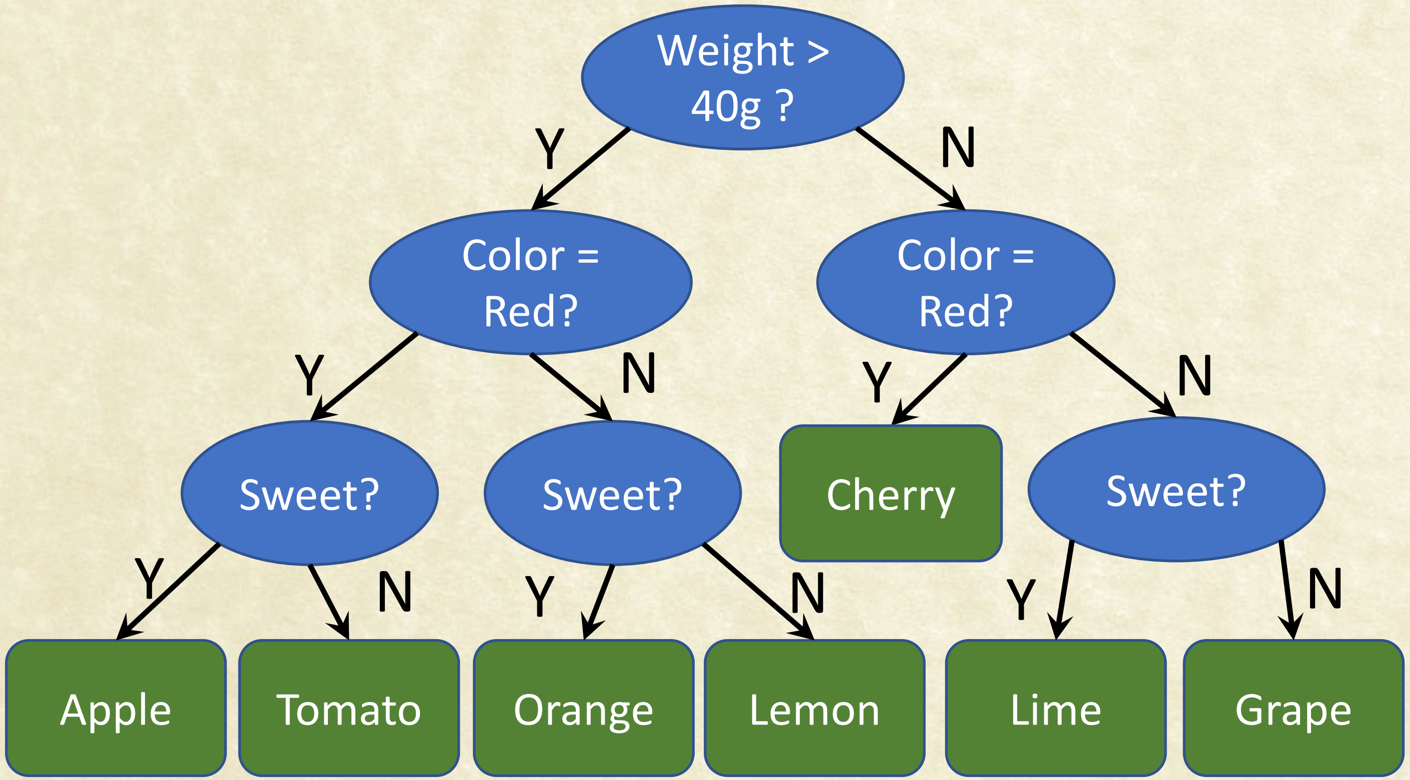


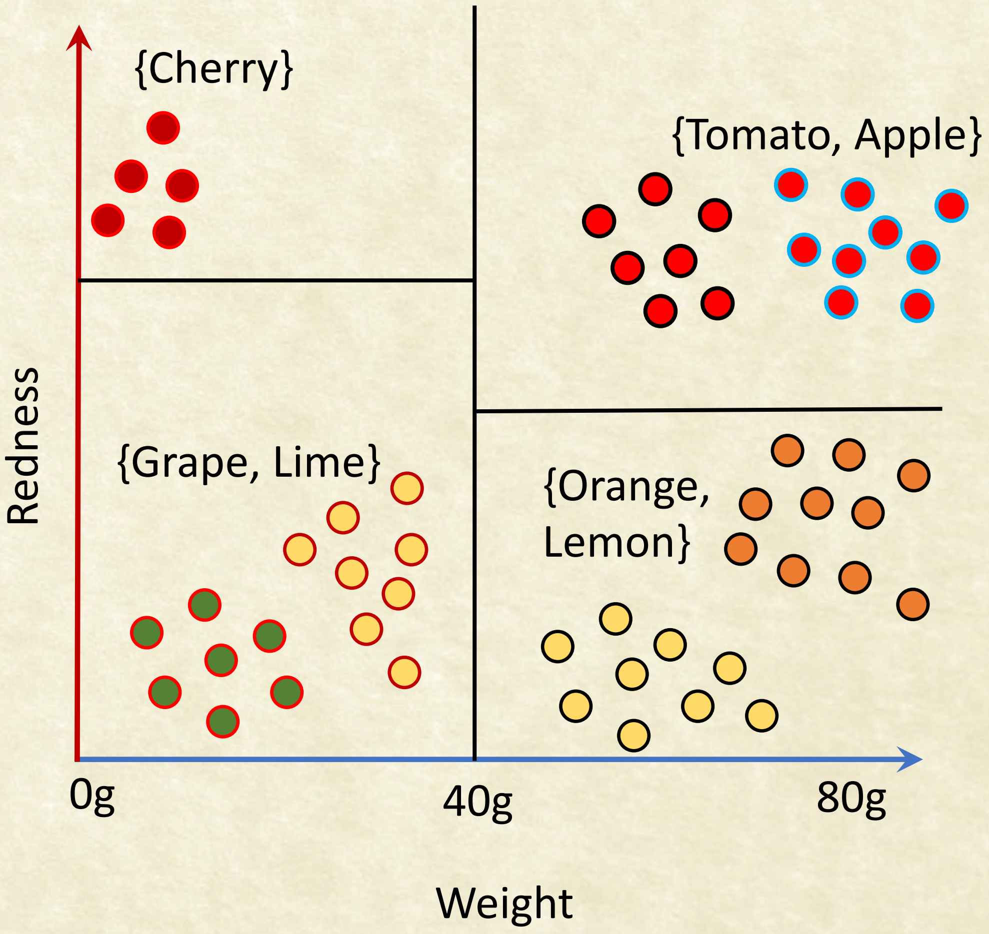
- Each Node (except leaves) has a question / decision.
- Depending on the answer, the set of possible answers shrink.
- Leaf nodes have class labels
- Given a test sample, we traverse the path from the root to a leaf based on decisions at each node.



Decision Boundaries

- Boundaries are Axis-Parallel
- What if we check a combination of features?



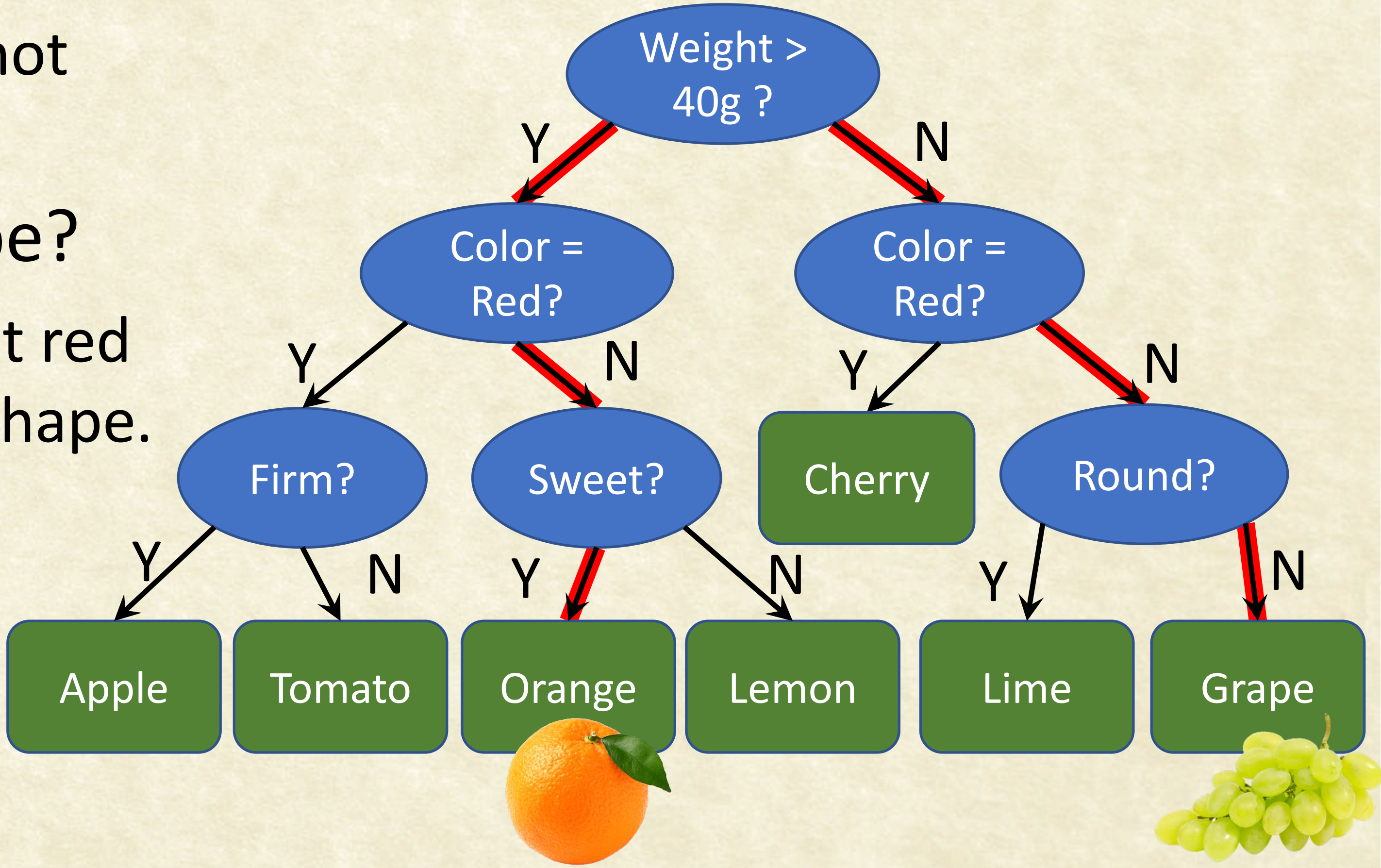


Features for Classification

- Decision trees can handle
  - Numerical Data (Price in Rs, Height in cms)
    - Divide into two based on a threshold on its value
  - Categorical Data ([sedan, SUV, coupe, hatchback], [red, white, green])
    - Divide into k groups based on value
  - Ordinal data ([bad, fair, good, excellent])
    - Divide into two at any point of the values
- Approaches
  - Use features as above
  - Convert all features to categorical

Interpreting a Classification

- Why did you call it an Orange?
  - It weighs more than 40g, is not red in color and is sweet.
- Why did you call this a Grape?
  - It weighs less than 40g, is not red in color and is not round in shape.
- Each classification decision can be explained in plain text (if features are meaningful)
  - Useful in many industries



- Medical diagnosis
- Credit risk analysis

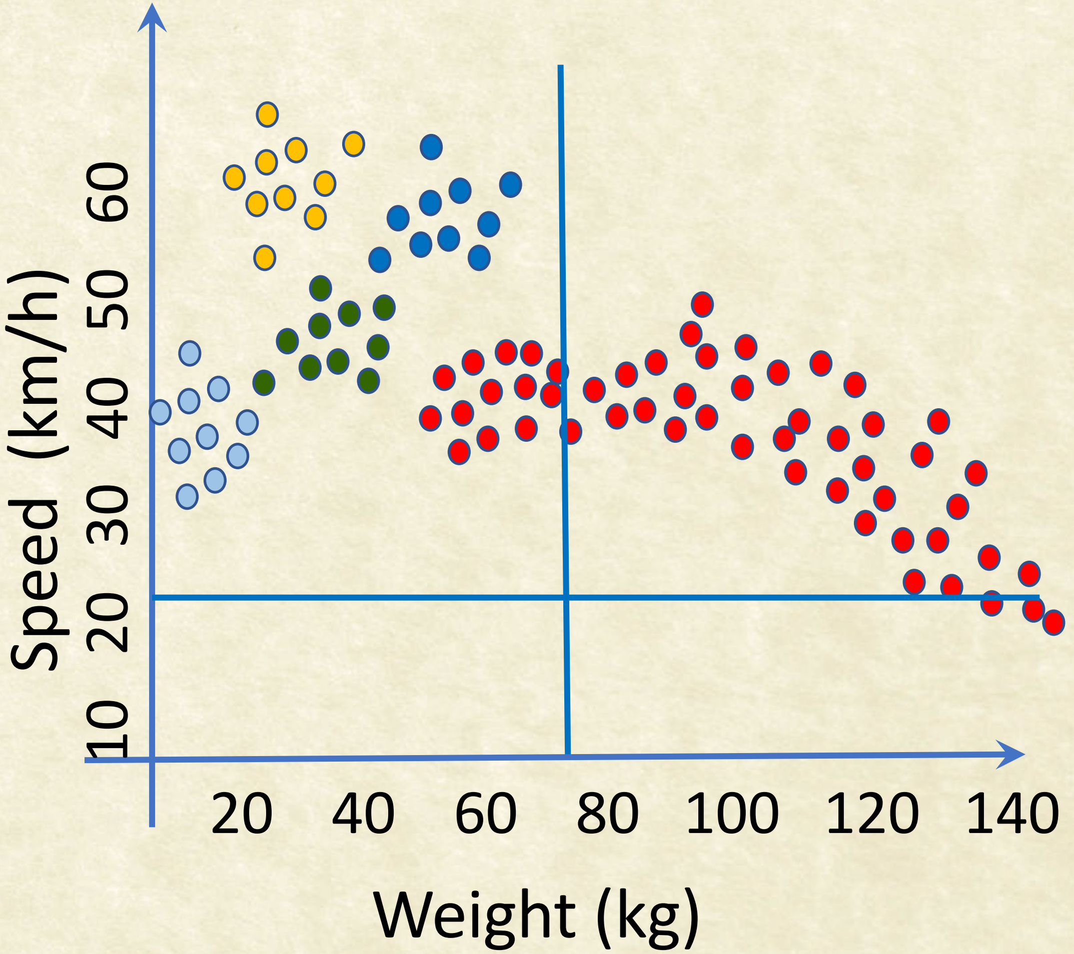
Questions?

Building a Decision Tree

What is a good Question?

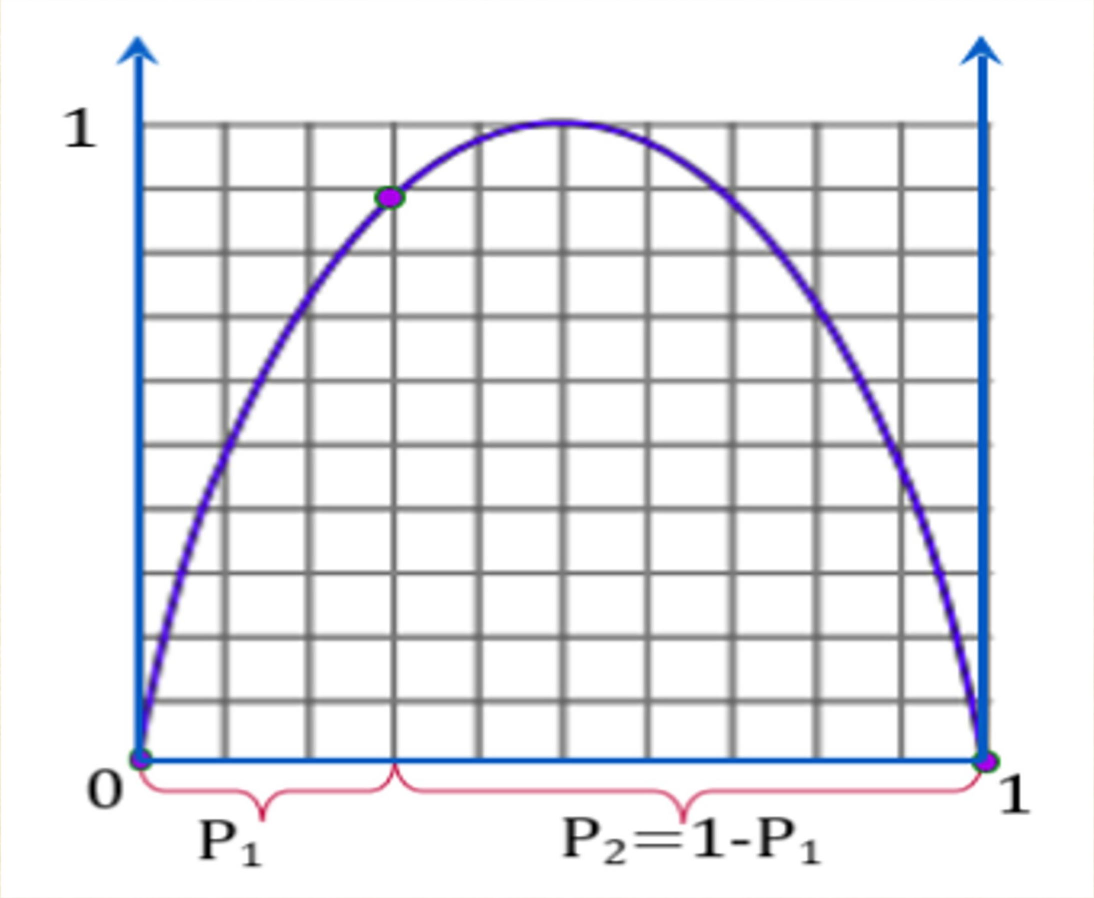
What is a Good Question?

- Consider the question of animal classification
- Which question if answered will reduce the possible number of animals the most?
- More precisely, which question (*feature+threshold*) will reduce our uncertainty the most
- Mathematically, reduce the Entropy the most



What is Entropy?

- Measure of **Uncertainty**
- Mathematically:
  - $\mathrm{H}(x) = -\sum_i P_i \, log_2 P_i$
- If the set contains just two classes:
  - $\mathrm{H} = -P_1 \, log_2 P_1 - P_2 \, log_2 P_2$
- Measure of Impurity
  - Not the only one



Solution: Maximize Information Gain

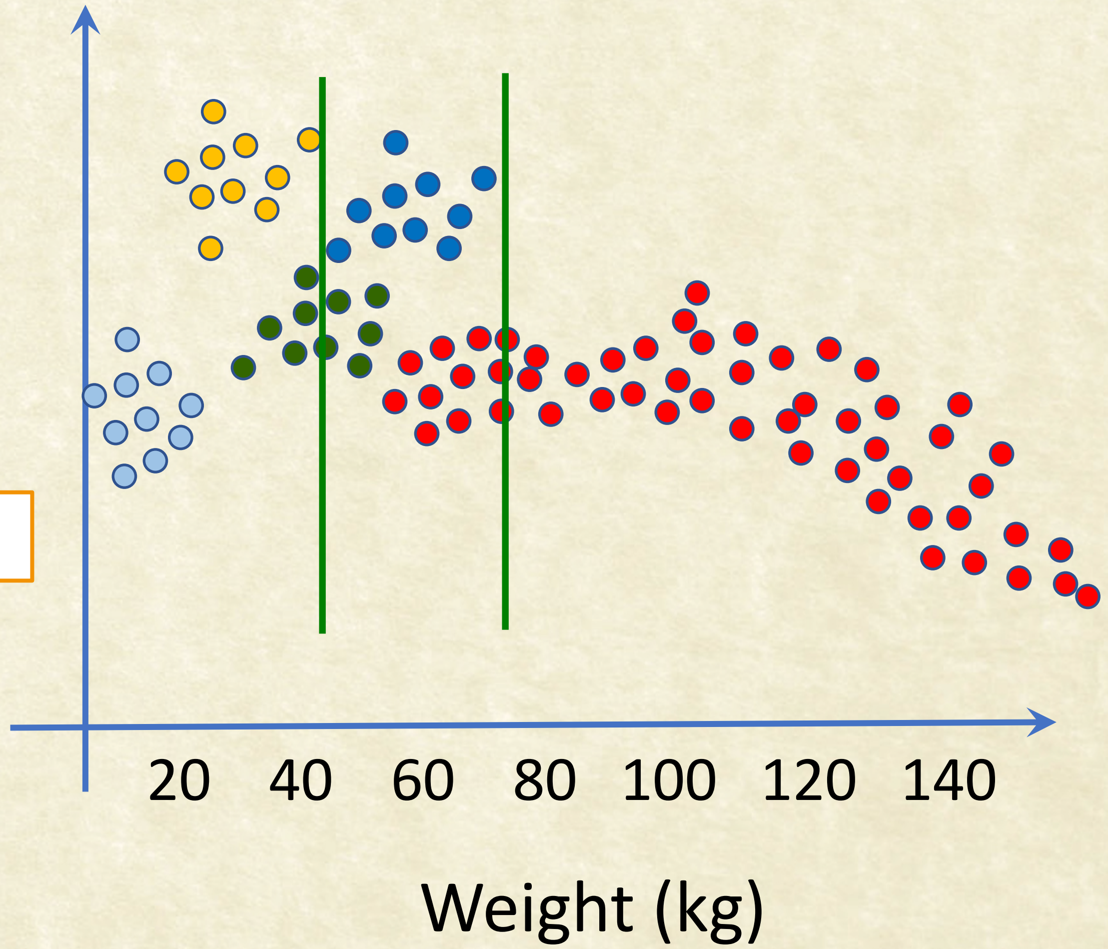
Initial Entropy: $5\times(-0.2\ log_2\,0.2) = 2.32$

$$H_1 = \frac{1}{2}\begin{pmatrix}(-0.4\ log_2 0.4) + (-0.4\ log_2 0.4) \\ + (-0.2\ log_2 0.2)\end{pmatrix} + \frac{1}{2}\begin{pmatrix}(-0.4\ log_2 0.4) + (-0.4\ log_2 0.4) \\ + (-0.2\ log_2 0.2)\end{pmatrix} = 1.52$$

Information Gain = 0.8

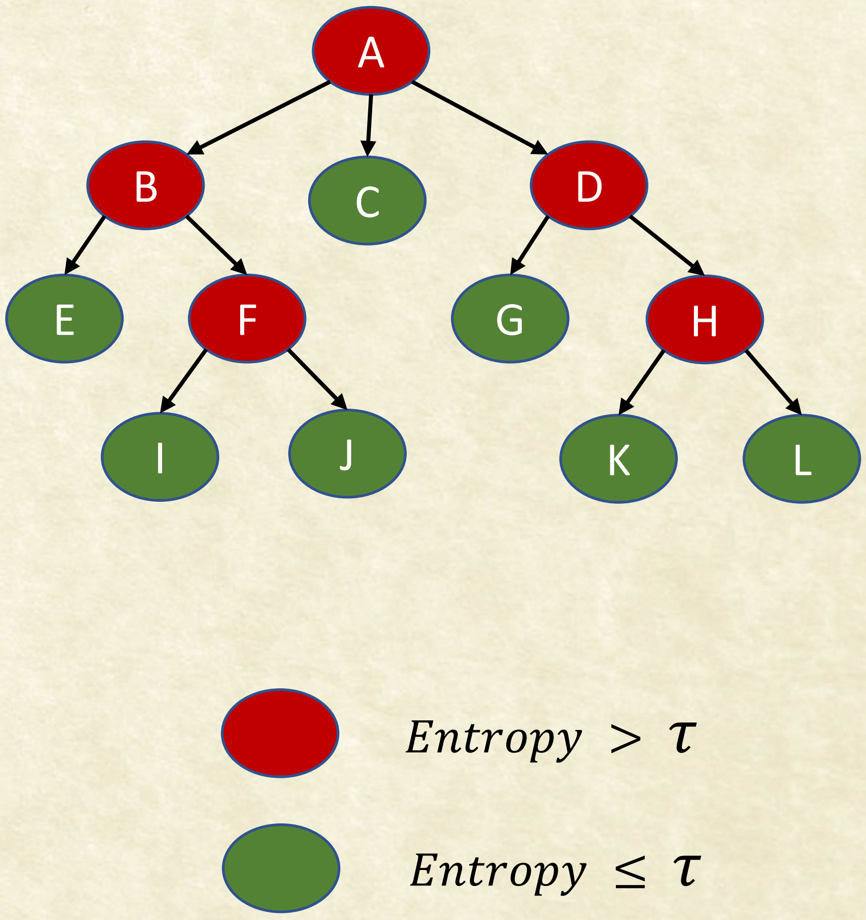
$$H_2 = \frac{1}{6}(-1.0\ \ log_2 1.0) + \frac{5}{6}\begin{pmatrix}(-0.22\ log_2 0.22) + (-0.22\ log_2 0.22) + \\ (-0.22\ log_2 0.22) + (-0.22\ log_2 0.22) + \\ (-0.12\ log_2 0.12)\end{pmatrix} = 1.91$$

Information Gain = 0.41



Building a Decision Tree

1. Find the best feature (and threshold) to split the training data
   - Use an objective metric like Entropy to decide
2. Partition the training data as per the selected feature and threshold
3. For each partition, if the entropy is low, stop.
   - else, Repeat the first two steps for that partition



Summary of Decision Trees

- Efficient, Compact and Effective
- Interpretable as a set of rules
- Ability to handle categorical and numerical features
- Can indicate most useful features
- Can also do regression
- Computationally expensive to train
- Does not handle non-rectangular regions well
- Not suitable for continuous variable regression
- Tends to overfit

Questions?

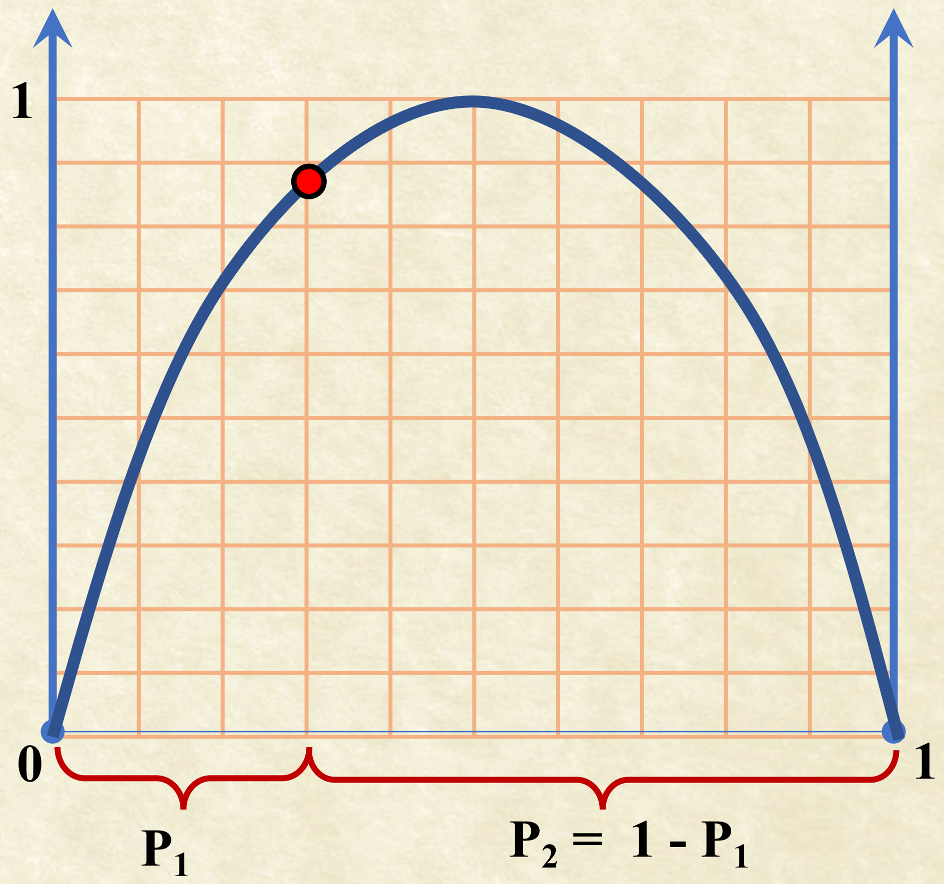
Impurity Metrics

You Optimize what you Measure

Ref: DHS, Chap 8.

Entropy as Impurity

- Measure of **Uncertainty**
- Mathematically, Entropy:
  - $i_H(N) = -\sum_i P_i \, log_2 P_i$
  - P_i is the fraction of samples that belong to class ω_i at node N.
- If we have just two classes, the impurity becomes:
  - $i_H(\mathrm{N}) = -P_1 \, log_2 P_1 - P_2 \, log_2 P_2$



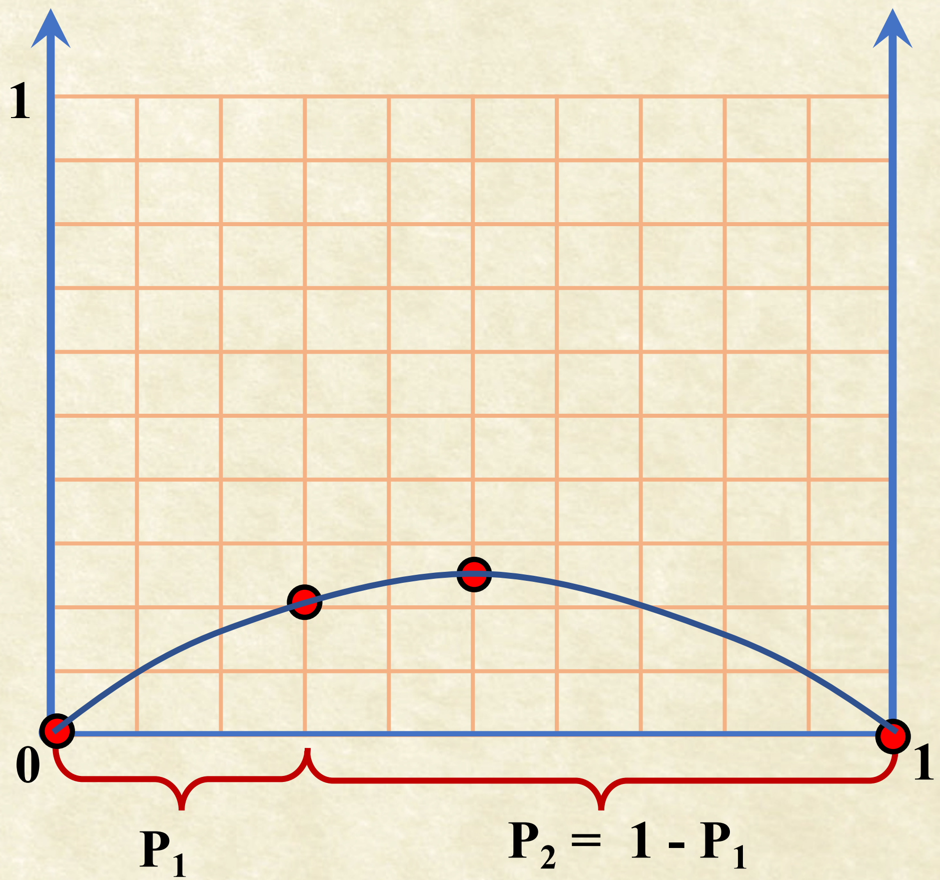
Variance Impurity

- Consider the case of only two-class
  - We need zero impurity when only one class is present.
- A possible metric:

$$i_\sigma(\mathrm{N}) = P(\omega_1)P(\omega_2)$$

  - Zero when either of $P(\omega_1)$ or $P(\omega_2)$ is zero.
  - Maximum when $P(\omega_1) = P(\omega_2)$.
  - Called variance impurity
    - Variance of the distribution with $\omega_1 = 1$ and $\omega_2 = 0$

Proof



Gini Impurity

- Extending variance impurity to 3 classes: $P(\omega_1)P(\omega_2)P(\omega_3)$?

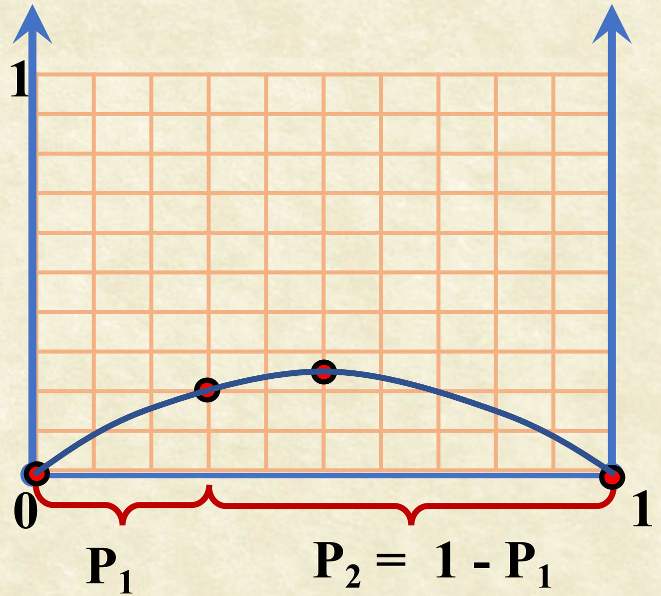
$$i(\mathrm{N}) = P(\omega_1)P(\omega_2) + P(\omega_2)P(\omega_3) + P(\omega_1)P(\omega_3)$$

- Generalizing to c classes?

$$i_G(\mathrm{N}) = \sum_{i \neq j} P(\omega_i)P(\omega_j)$$

Proof

- Called Gini Impurity



Misclassification Impurity

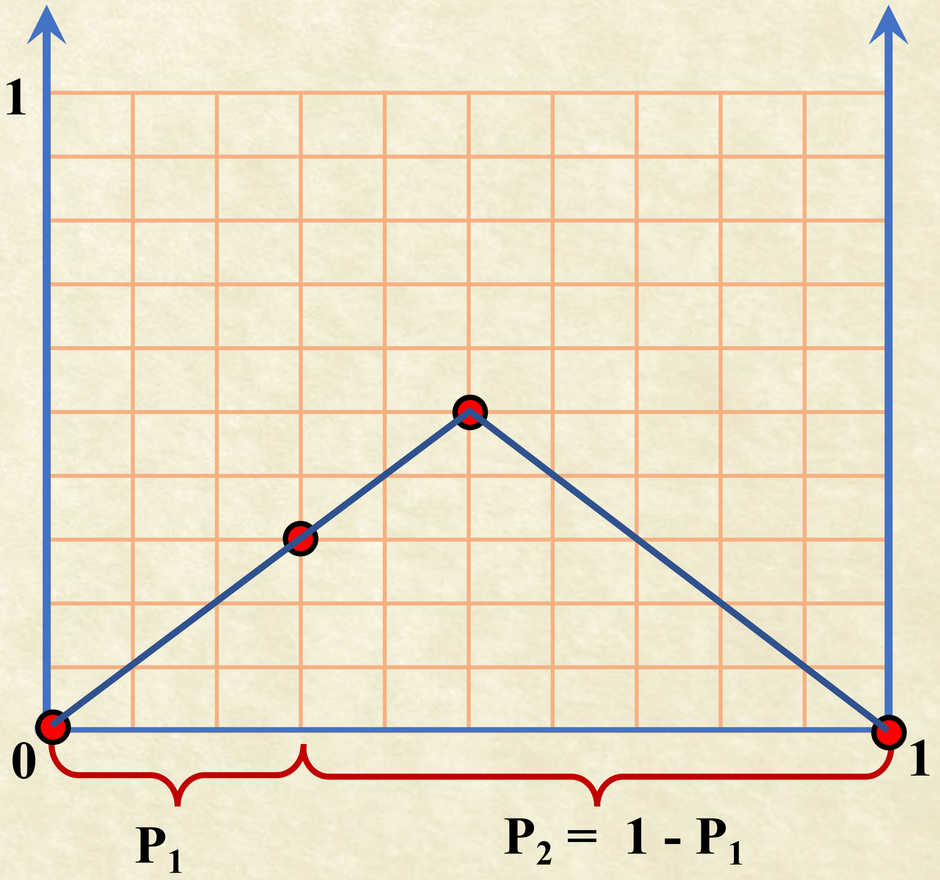
- What is the expected misclassification (error) rate at node N if we were to consider it as a leaf and assign a label?
  - Which label to assign?

$$label = \operatorname*{argmax}_k P(\omega_k)$$

- Misclassification impurity:

$$i_e(\mathrm{N}) = 1 - \max_j P(\omega_j)$$

- Strongly peaked
- Discontinuous derivative



Metric for Deciding the Split

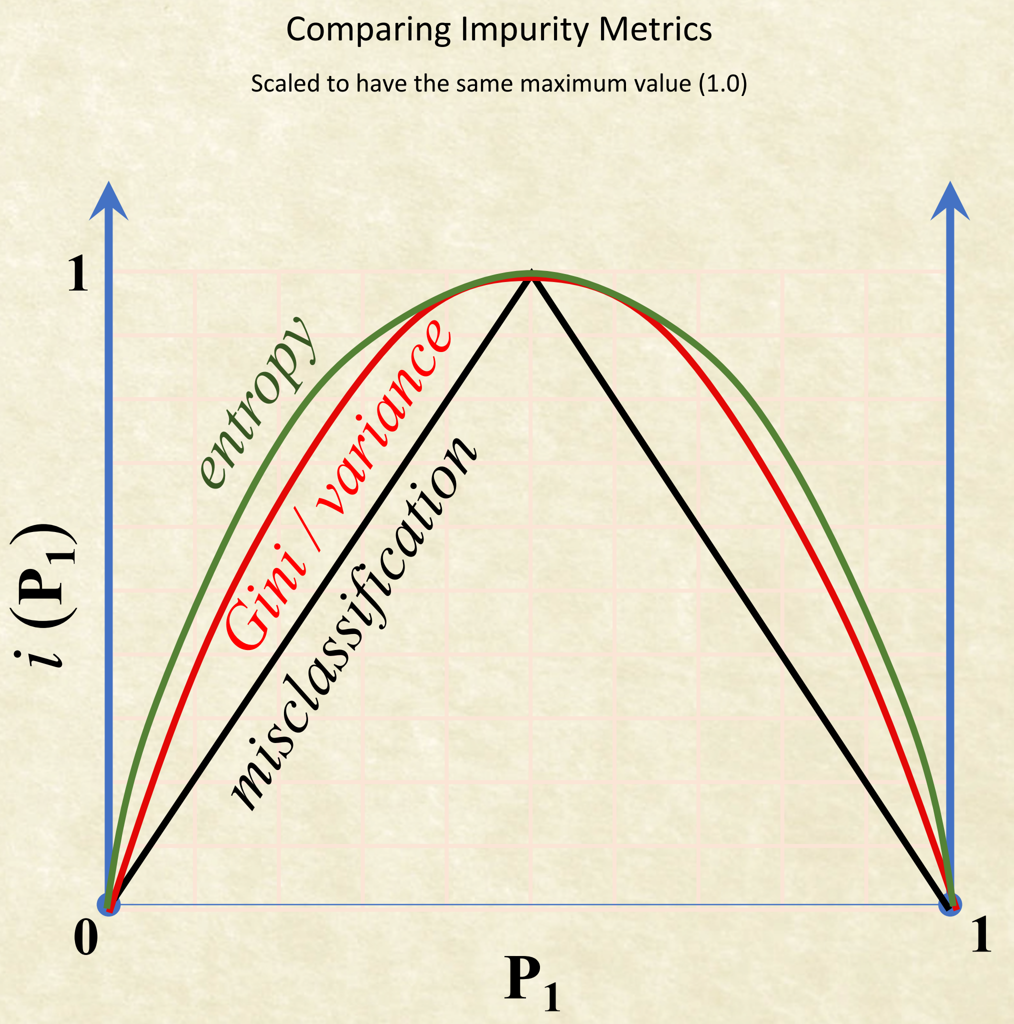
- For the best split, we want the impurity to reduce as much as possible.

$$\Delta i(T,N) = i(N) - P_L i(N_L) - P_R i(N_R),$$

where $\Delta i(T,N)$ is the change in impurity at node N when the test T is used for splitting; $P_{L/R}$ is the fraction of samples at N that moves to $N_{L/R}$ when using test T.

- Generic Case:

$$\Delta i(T,N) = i(N) - \sum_k P_k i(N_k)$$



Refining Impurity Reduction Scores

- A high branching factor → greater impurity reduction
  - Need to scale impurity reduction based on branching factor, B
- Prefer balanced splits
  - Highly imbalanced splits may result in overfitting, increase model size, and reduce efficiency during classification
- Scaled criterion: Gain Ratio Impurity

$$\Delta i_B(N) = \frac{\Delta i(N)}{-\sum_k P_k log(P_k)}$$

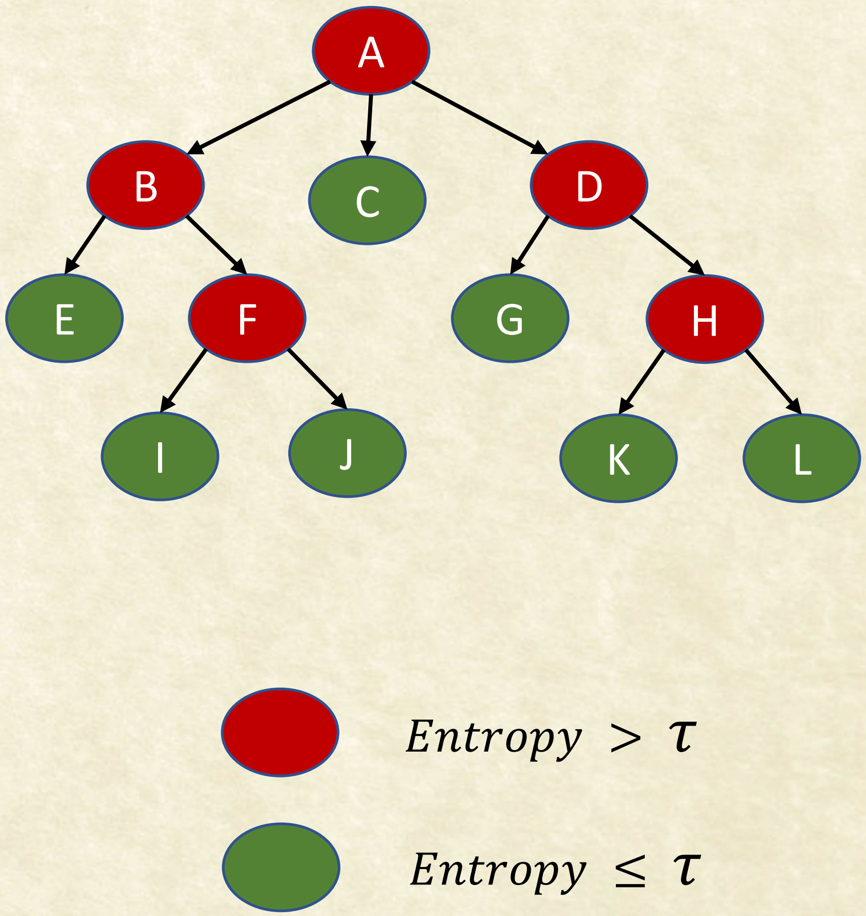
Questions?

Decision Tree Algorithms (CART, ID3, C4.5)

It is your choices that defines you

Recap: Building a Decision Tree

1. Find the best feature (and threshold) to split the training data
    - Use an objective metric like Entropy to decide
2. Partition the training data as per the selected feature and threshold
3. For each partition, if the entropy is low, stop.
    - else, Repeat the first two steps for that partition



Six Questions while DT Training

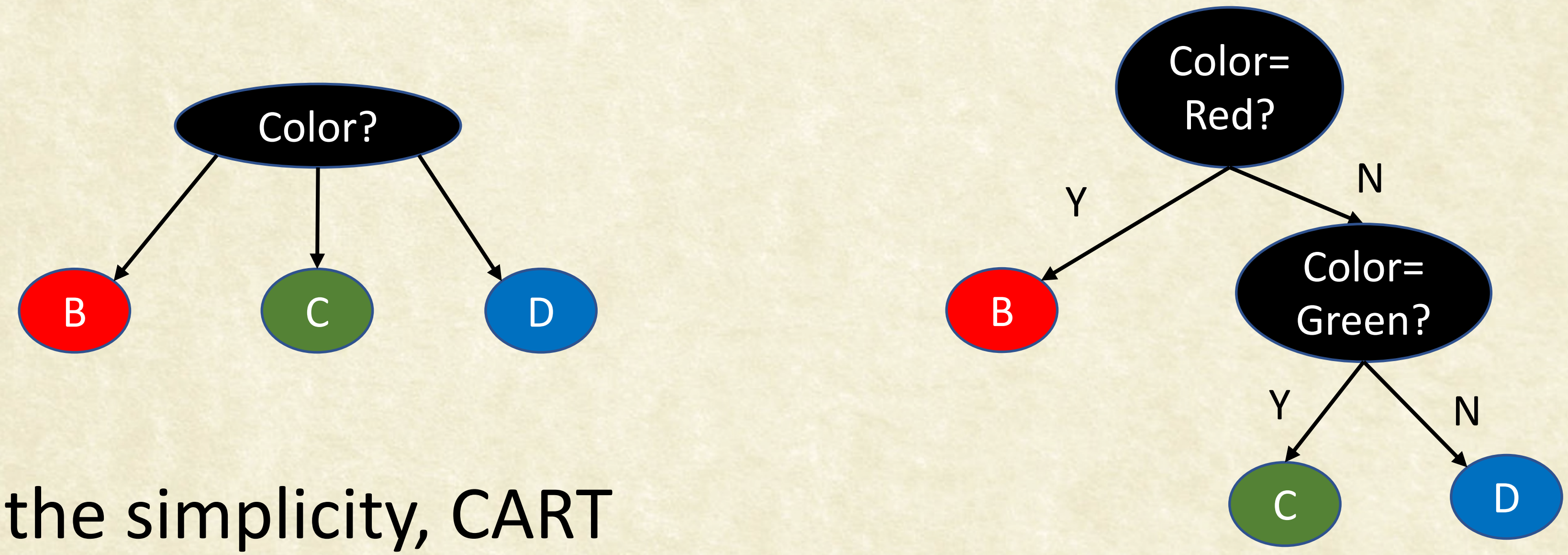
1. Should a node be split into two or multiple subsets?
2. Which property should be tested at a node?
3. When should a node be declared a leaf?
4. If a tree becomes large, how can it be simplified / pruned?
5. If a leaf node is impure, how to assign a category label?
6. How should missing data be handled?

- Let us answer each of these in the context of CART algorithm

CART Algorithm Steps

1. Branching Factor or Branching Ratio (B):

- Each decision or outcome is a split
- A p-way split can be expressed as multiple binary splits



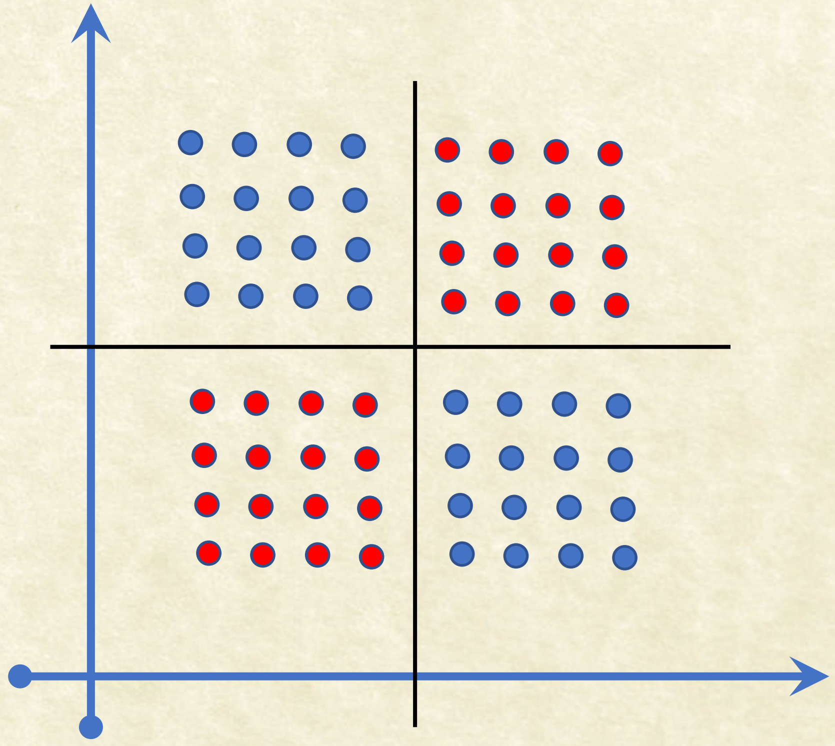
- Due to the simplicity, CART focuses on binary splits

CART Algorithm Steps

2. Test at each node.
   - Best Gain Ratio Impurity criterion
   - Cost Function: Gini (classification), MSE (regression)
3. Deciding a node to be a leaf
   - Use cross-validation
   - Threhold: $\Delta i(N) < \beta$
   - Threshold on number of samples (10 samples or 5%)
   - Use a global criterion: $\alpha.\,size + \sum_{leaf\ N} i(N)$

CART Algorithm Steps

4. Pruning a Tree
   - Horizon Effect: Develop further and prune
   - Remove children if results in minimum increase in impurity
   - Increases classification speed; Accuracy
5. Assigning Labels to Leaves
   - Assign label of most frequent sample
   - Use weighted voting if priors are given
6. Handling Missing Data
   - Use sample with available feature values only at a node while computing impurity



Other Algorithms

- ID3
  - Nominal (unordered) features only. Real values are binned.
  - Each split has branching factor B_j
  - Number of levels = Number of attributes
  - Learn until all leafs are pure. Can be pruned.
- C4.5
  - Real Valued: CART, Nominal: ID3 (multiway splits)
  - Gain ratio impurity
  - Pruning based on statistical significance of splits
  - Follow all paths for missing features; Weighted probability to combine

Questions?

Notes on Learning

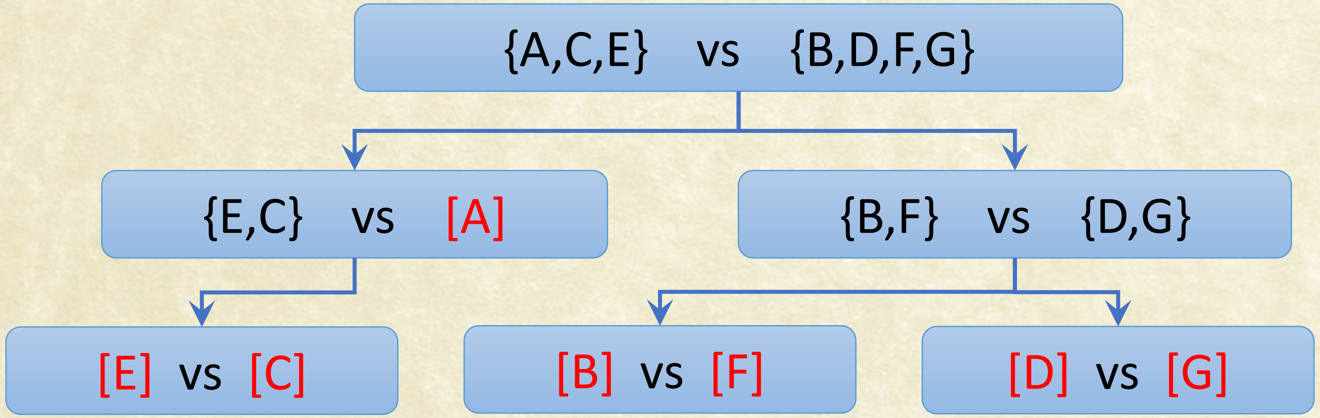
A few interesting points

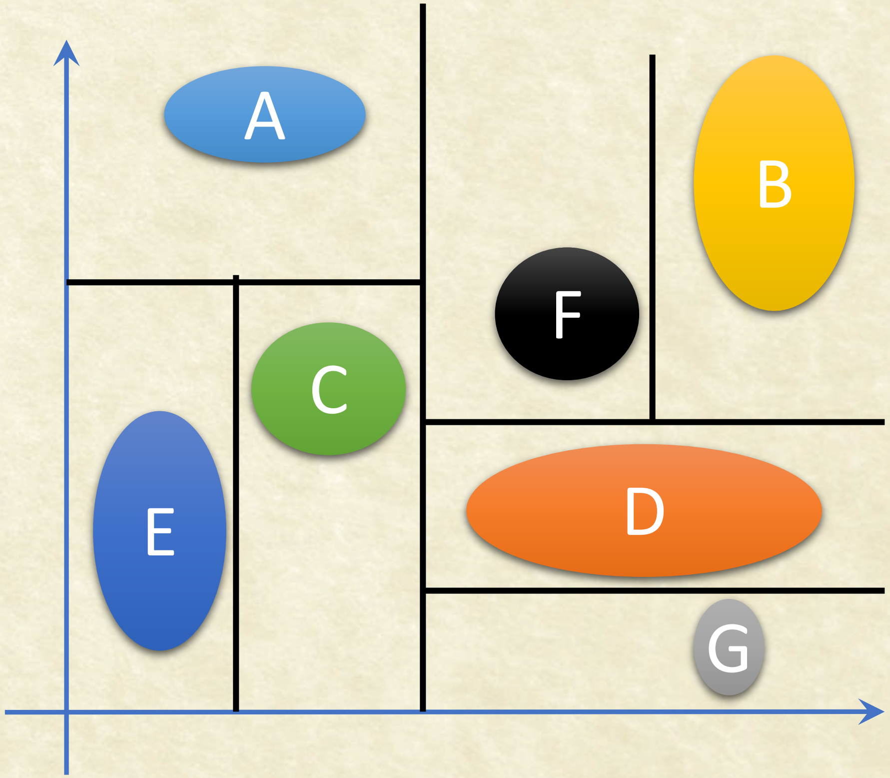
Computational Complexity

- How to find split points for real attributes?
  - Sort samples at each node [$O(n \log n)$]
  - Try between consecutive samples
- Training Complexity at each level:
  - Root Node: $O(d\, n \log n)$
  - First Level: $O(d\, n \log(n/2))$
  - Second Level: $O(d\, n \log(n/4))$ .. log n levels
- Average Training Complexity: $O(d\, n \log^2 n)$
- Testing Complexity: $O(\log n)$

Twoing Criterion

- Multi-class binary tree training
  - Let the list of categories be: $C = \{\omega_1, \omega_2, ..\omega_c\}$.
  - At each node we split C into two super-categories
    - $C_1 = \{\omega_{i1}, \omega_{i2}, ..\omega_{ik}\}$, and $C_2 = C - C_1$
  - For every candidate split,
    - compute: $\Delta i(N, C_1)$ as in a two-class problem
    - Find the supercategory C_1^* that maximises Δi.





Avoiding Overfitting

- Make the tree very short (say height = 2 or 3)
  - Avoids overfitting
  - Lower accuracy
  - Higher speed
- Train a large number of such trees
  - Careful selection to give complimentary strengths
  - Could takes lesser time to train !!
- Combine the trees together
  - Multiple trees together form a forest: Random Forest
- We will explore this idea next week.

Questions?