



# COS10022 – DATA SCIENCE PRINCIPLES

Dr Pei-Wei Tsai (Lecturer, Unit Convenor)  
ptsai@swin.edu.au, EN508d

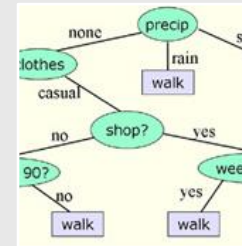
SWIN  
BUR  
NE

SWINBURNE  
UNIVERSITY OF  
TECHNOLOGY

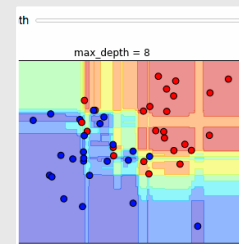
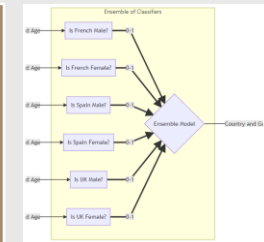


# Week 05

## Decision Tree



## Ensemble Learning



## Random Forest





# DECISION TREE



Root

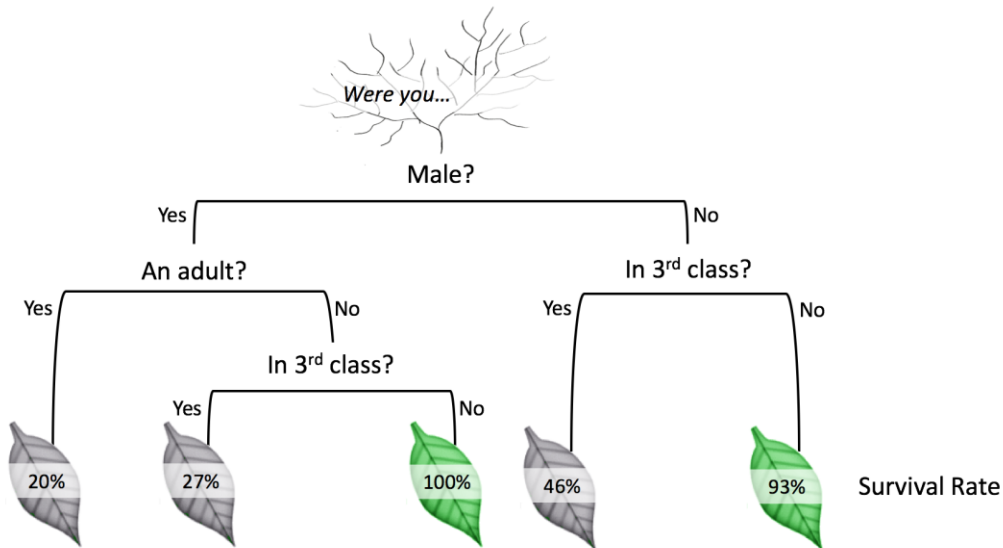


Trunk

Decision Node

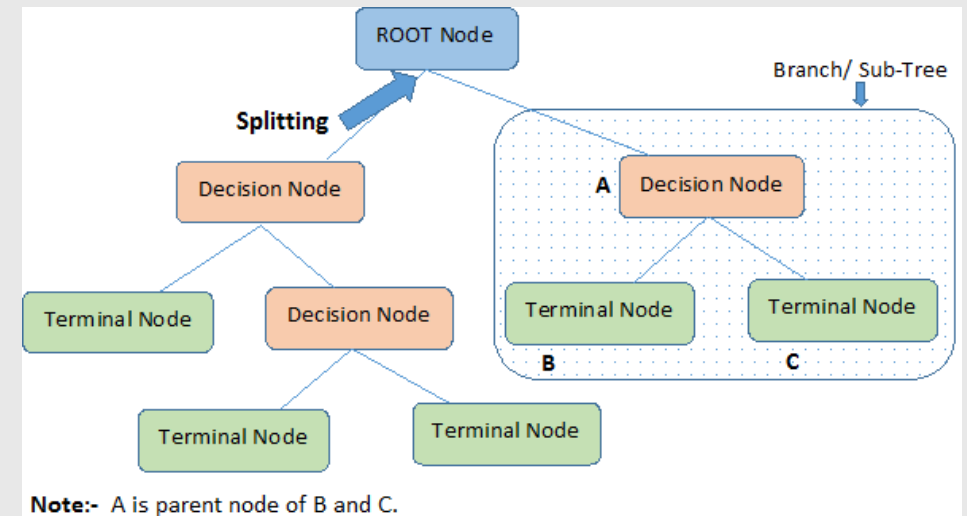


Terminal Node (Leaf)



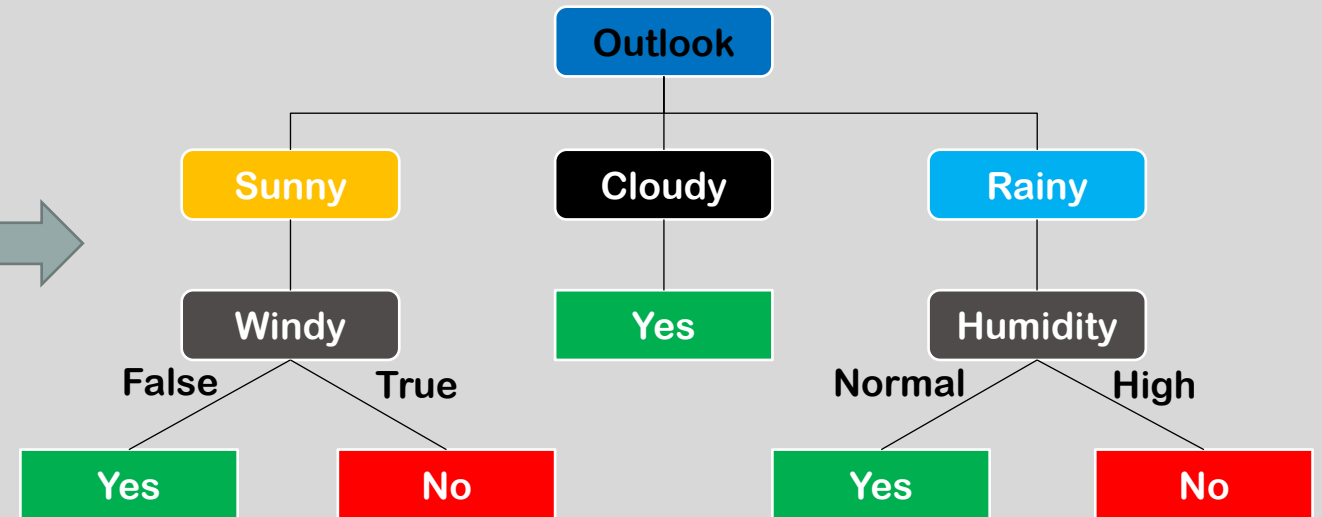
# Decision Tree

Learning Type	Processing data types	Splitting Rules	Pruning Techniques
•Supervised Learning	•Discrete •Continuous  (depending on the algorithm)	•Entropy •Gain Ratio •Chi-Square Test •Gini Index •etc.	•Predicting Error Rate •Entire Error Rate (Training and Predicting) •No pruning.



- Goal:  
Predict/classify the result based on the given criteria.

Predictors				Target
Outlook	Temp	Humidity	Windy	Outdoor Exercise
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Cloudy	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Cloudy	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Cloudy	Mild	High	True	Yes
Cloudy	Hot	Normal	False	Yes
Sunny	Mild	High	True	No



## Decision Tree (2)

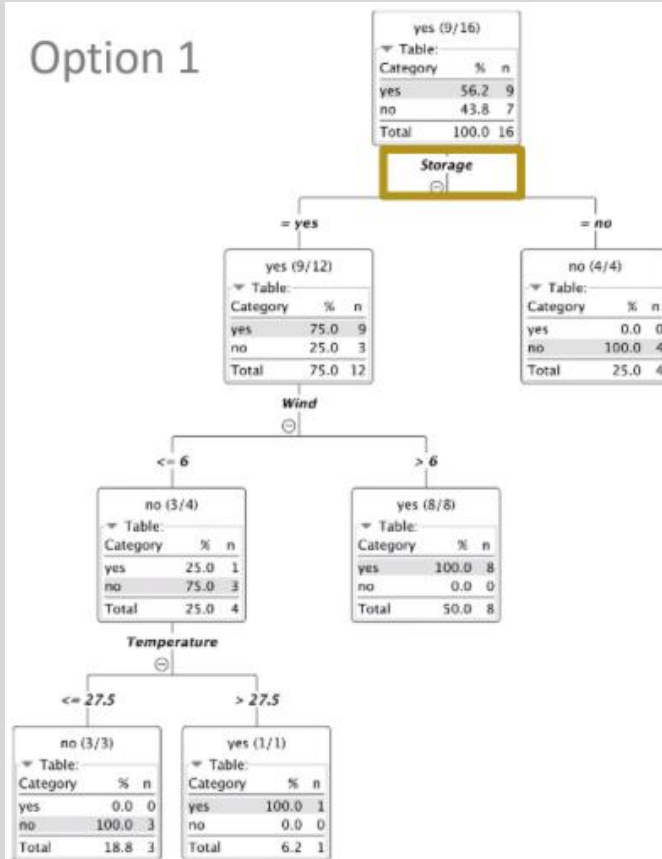
- Using algorithms such as ID3 allows users to build decision tree easily.



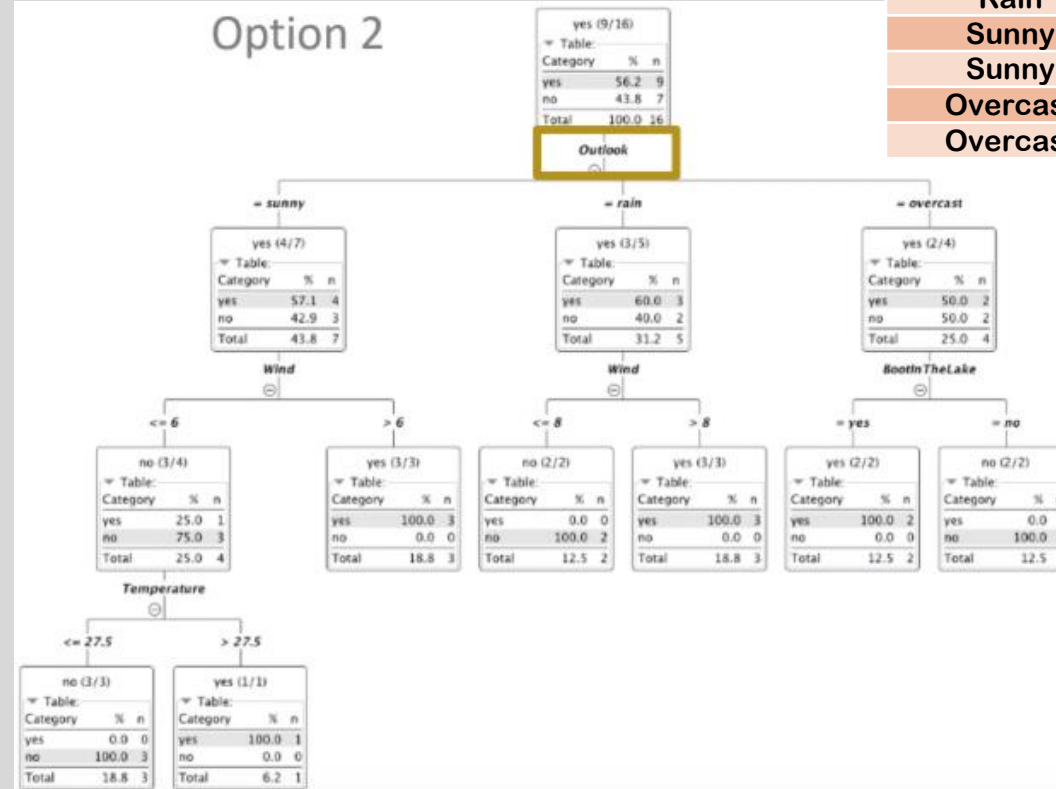
# Decision Tree (2)

Outlook	Wind	Temp	Storage	Sailing
Sunny	3	30	Yes	Yes
Sunny	3	25	Yes	No
Rain	12	15	Yes	Yes
Overcast	15	2	No	No
Rain	16	25	Yes	Yes
Sunny	14	18	Yes	Yes
Rain	3	5	No	No
Sunny	9	20	Yes	Yes
Overcast	14	5	No	No
Sunny	1	7	No	No
Rain	4	25	Yes	No
Rain	14	24	Yes	Yes
Sunny	11	20	Yes	Yes
Sunny	2	18	Yes	No
Overcast	8	22	Yes	Yes
Overcast	13	24	Yes	Yes

Option 1



Option 2



# A Clever Choice of the Attribute for Splitting would Benefit Your Result

- ID3, CART, etc.

How to construct a decision tree?



- Splitting the dataset in a way that the subsets are as pure as possible in terms of the target variable.

Purpose:



- Entropy
- Information Gain Ratio
- Gini Index
- Etc.

How does the algorithm decide when to use which feature in order to split the dataset?





*entropy*

**Entropy:**

$$H(p) = - \sum_{i=1}^n p_i \log_2 p_i$$

for  $p \in \mathbb{Q}^n$

where  $H$  (Greek capital letter eta) defines the entropy,  $p_i$  indicates the probability mass function, and  $n$  is the number of output states.

8

- **Where can we use Entropy?**
  - Entropy is one of the measurements in the information theory, which is a field of study concerned with quantifying information for communication.
- **Mathematically, it is defined as the negative of the sum over the probability for each class multiplied by the logarithm of it.**

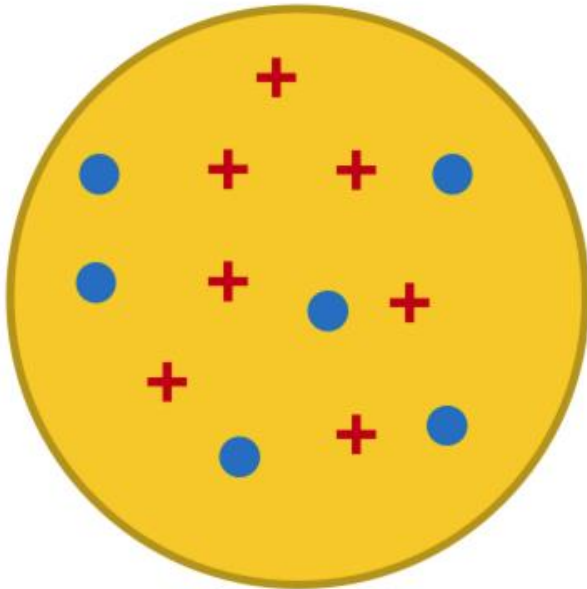


# Entropy Calculation

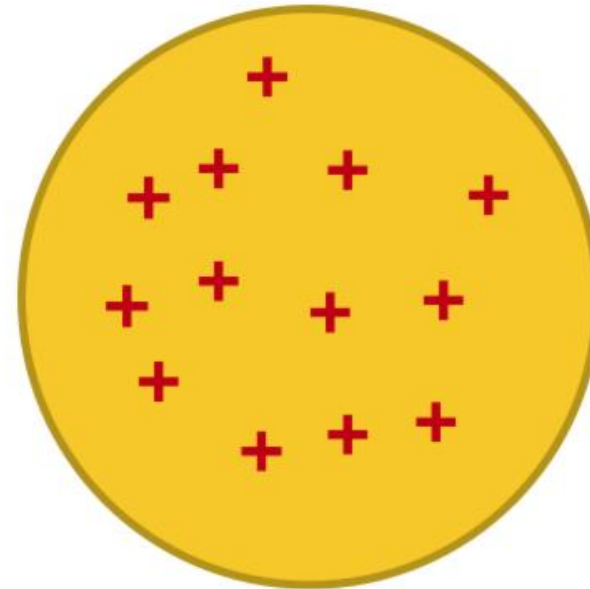
$$H(p) = -\sum_{i=1}^n p_i \log_2 p_i \text{ for } p \in \mathbb{Q}^n$$

$$p_1 = 7/13$$

$$p_2 = 6/13$$



$$H(p) = -\frac{7}{13} \log_2 \left( \frac{7}{13} \right) - \frac{6}{13} \log_2 \left( \frac{6}{13} \right) = 0.995$$



$$p_1 = 13/13 = 1$$

$$p_2 = 0/13 = 0$$

$$H(p) = -\frac{13}{13} \log_2 \left( \frac{13}{13} \right) - \frac{0}{13} \log_2 \left( \frac{0}{13} \right) = 0$$

# Entropy Calculation Example

Outlook	Wind	Temp	Storage	Sailing
Sunny	3	30	Yes	Yes
Sunny	3	25	Yes	No
Rain	12	15	Yes	Yes
Overcast	15	2	No	No
Rain	16	25	Yes	Yes
Sunny	14	18	Yes	Yes
Rain	3	5	No	No
Sunny	9	20	Yes	Yes
Overcast	14	5	No	No
Sunny	1	7	No	No
Rain	4	25	Yes	No
Rain	14	24	Yes	Yes
Sunny	11	20	Yes	Yes
Sunny	2	18	Yes	No
Overcast	8	22	Yes	Yes
Overcast	13	24	Yes	Yes

- Calculate the Entropy to find go sailing from all observations.

- $H(9^+, 7^-)$

$$= -\frac{9}{16} \log_2 \frac{9}{16} - \frac{7}{16} \log_2 \frac{7}{16}$$
$$= 0.9887$$

# Entropy Calculation Example 2

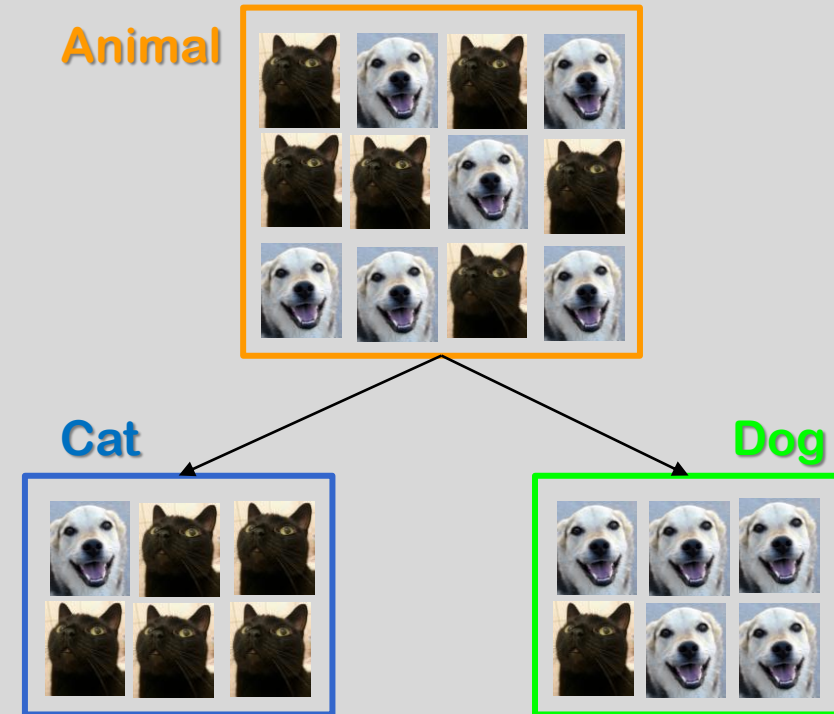
- Assume a pool of animal (cats and dogs) are somehow split into two groups.
- Let's find the entropy for each group.

$$H_A(p) = -\sum_{i=1}^2 p_i \log_2 p_i = -\frac{6}{12} \log_2 \frac{6}{12} - \frac{6}{12} \log_2 \frac{6}{12} = 1$$

$$H_C(p) = -\sum_{i=1}^2 p_i \log_2 p_i = -\frac{5}{6} \log_2 \frac{5}{6} - \frac{1}{6} \log_2 \frac{1}{6} = 0.65$$

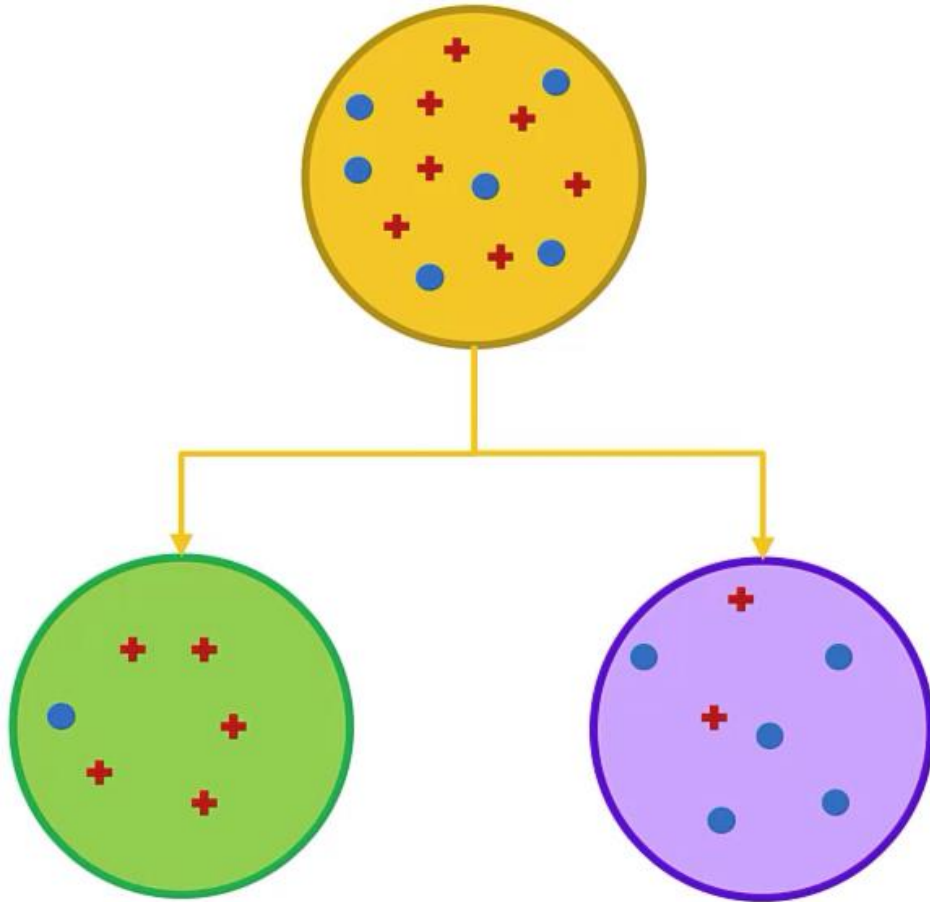
$$H_D(p) = -\sum_{i=1}^2 p_i \log_2 p_i = -\frac{5}{6} \log_2 \frac{5}{6} - \frac{1}{6} \log_2 \frac{1}{6} = 0.65$$

- Entropy reduces when the data in a set is more pure.**
- The differences between the new and the original entropies is defined as the **information gain**.





$$H_0(p) = -\frac{7}{13} \log_2 \left( \frac{7}{13} \right) - \frac{6}{13} \log_2 \left( \frac{6}{13} \right) = 0.995$$



$$H_1(p) = -\frac{5}{6} \log_2 \left( \frac{5}{6} \right) - \frac{1}{6} \log_2 \left( \frac{1}{6} \right) = 0.65$$

$$w_1 = \frac{6}{13} = 0.462$$

$$H_2(p) = -\frac{2}{7} \log_2 \left( \frac{2}{7} \right) - \frac{5}{7} \log_2 \left( \frac{5}{7} \right) = 0.863$$

$$w_2 = \frac{7}{13} = 0.539$$

**Solution: Using Gain Ratio in C4.5 algorithm to replace ID3**

## Information Gain – One of the Possible Split Criteria

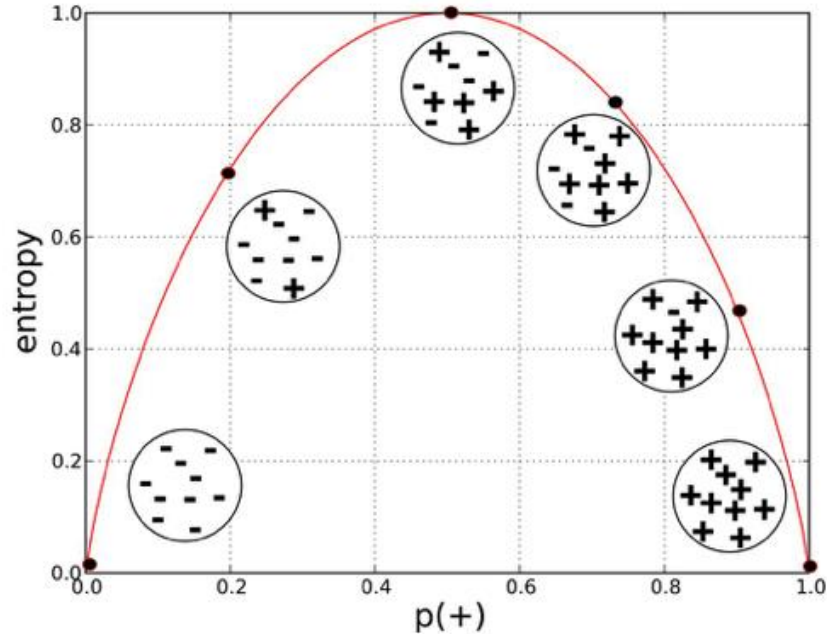
- In ID3 algorithm (one of the decision tree construction algorithms), the information gain, which is the differences of entropies before and after applying the dataset splitting rule.
- Information Gain after the split:
  - $Gain = H_0 - (w_1 H_1 + w_2 H_2)$ 

$$= 0.995 - (0.462 \times 0.65) - (0.539 \times 0.83)$$

$$= 0.995 - 0.766$$

$$= 0.23$$
- Next splitting feature:
  - Starting from the one with the highest Gain.

**Disadvantage:**  
Prefers highly represented features.

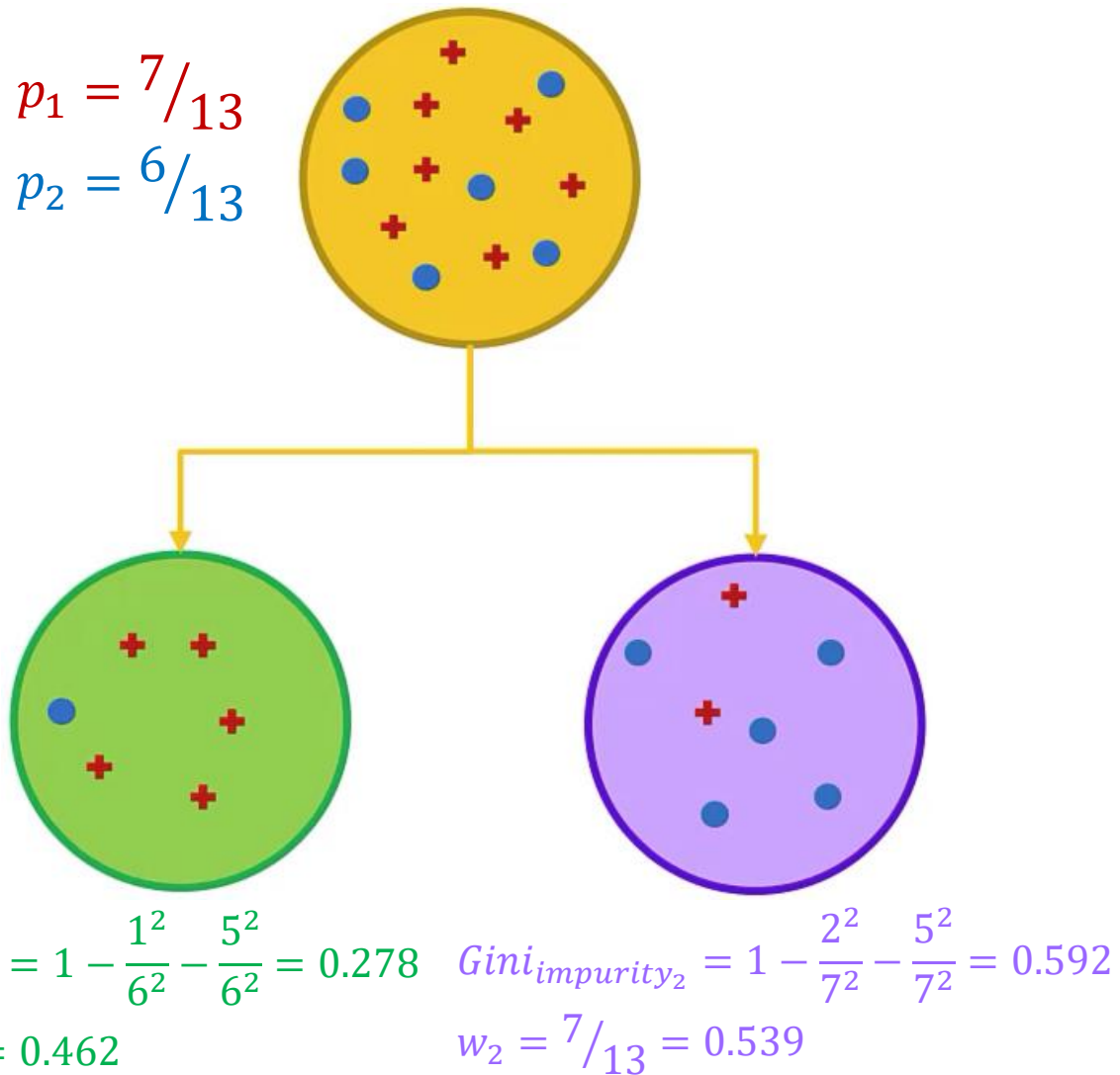


**Question:**  
Can Entropy be greater than 1?

# C4.5 Algorithm

- To overcome the disadvantage in ID3, C4.5 algorithm introduces the SplitInfo concept with the Gain Ratio.
- The SplitInfo is defined as the sum over the weights multiplied by the logarithm of the weights.
- The Gain Ratio measure is then calculated by dividing the gain from the ID3 algorithm by the SplitInfo.

$$GainRatio = \frac{Gain}{SplitInfo} = \frac{H_0 - \sum_{i=1}^k w_i H_i}{\sum_{i=1}^k w_i \log_2 w_i}$$



# CART Algorithm

- Another measurement for measuring the purity or the actual impurity, which is used by CART algorithm, is called the Gini index.

- Split criterion in CART algorithm:

$$Gini_{index} = \sum_{i=1}^n w_i Gini_{impurity_i}$$

- Gini index is designed based on Gini impurity:

$$Gini_{impurity}(p) = 1 - \sum_{i=1}^n p_i^2 \text{ for } p \in \mathbb{Q}^n$$

- In this case:

$$Gini_{index} = \frac{6}{13} Gini_{impurity_1} + \frac{7}{13} Gini_{impurity_2}$$

$$Gini_{impurity}(p) = 1 - \frac{7^2}{13^2} - \frac{6^2}{13^2} = 0.497$$

- Next splitting feature:  
Feature with the lowest  $Gini_{index}$



# Gini Index Calculation Example

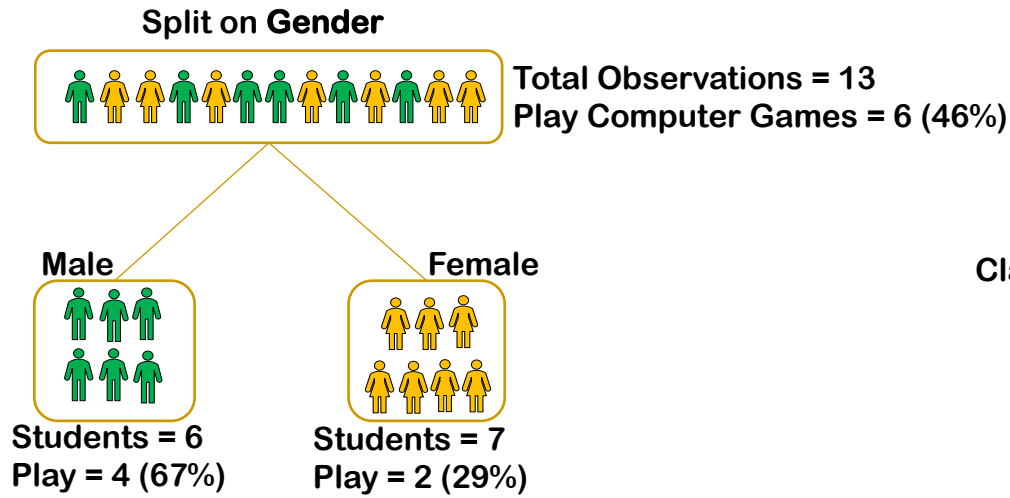
S/N	Gender	Class	Play Computer Games
1	M	IV	Yes
2	F	IV	Yes
3	F	IV	Yes
4	M	IV	No
5	F	IV	No
6	M	IV	Yes
7	M	IV	No
8	F	V	No
9	M	V	Yes
10	F	V	No
11	M	V	Yes
12	F	V	No
13	F	V	No

- Assume that we want to segregate students based on an specific target variable (playing computer games).
- We split the population using two input variables: Gender and Class.
- Now, we want to identify which split is producing **more homogeneous** sub-nodes using Gini index.



Total Observations = 13  
Play Computer Games = 6 (46%)

S/N	Gender	Class	Play Computer Games
1	M	IV	Yes
2	F	IV	Yes
3	F	IV	Yes
4	M	IV	No
5	F	IV	No
6	M	IV	Yes
7	M	IV	No
8	F	V	No
9	M	V	Yes
10	F	V	No
11	M	V	Yes
12	F	V	No
13	F	V	No

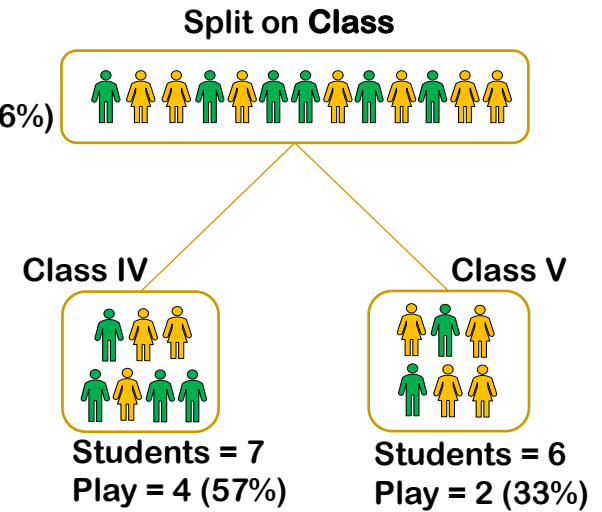


$$Gini_{impurity_M} = 1 - (0.67^2 + 0.33^2) = 0.44$$

$$Gini_{impurity_F} = 1 - (0.29^2 + 0.71^2) = 0.41$$

$$Gini_{index} = \frac{6}{13} \times 0.44 + \frac{7}{13} \times 0.41 = 0.42$$

## Better Split




$$Gini_{impurity_{IV}} = 1 - (0.57^2 + 0.43^2) = 0.49$$

$$Gini_{impurity_V} = 1 - (0.33^2 + 0.66^2) = 0.46$$

$$Gini_{index} = \frac{7}{13} \times 0.49 + \frac{6}{13} \times 0.46 = 0.48$$

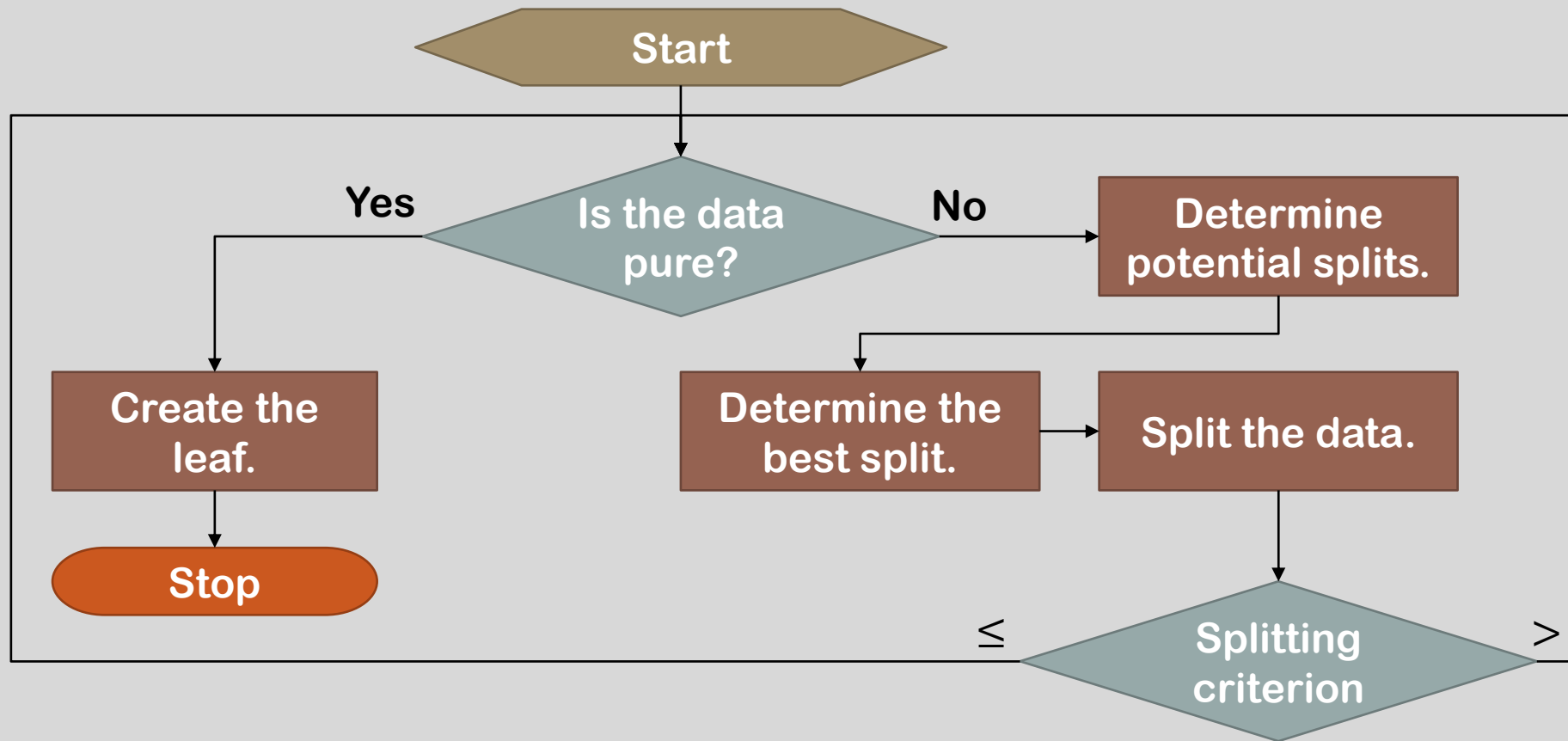
# GINI INDEX CALCULATION EXAMPLE

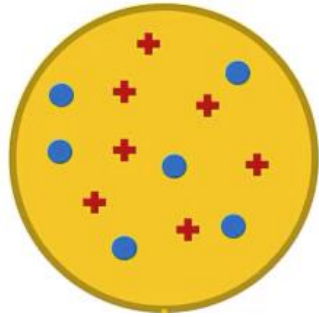
# Tree Construction Summary

- 
- Which question to ask?
  - How much a question helps to unmix the labels?
  - Quantify the amount of uncertainty at a single node using the Gini impurity metric.
  - Quantify how much a question reduces the uncertainty using Information Gain.
  - Repeating to ask questions until the data can not be divided further.



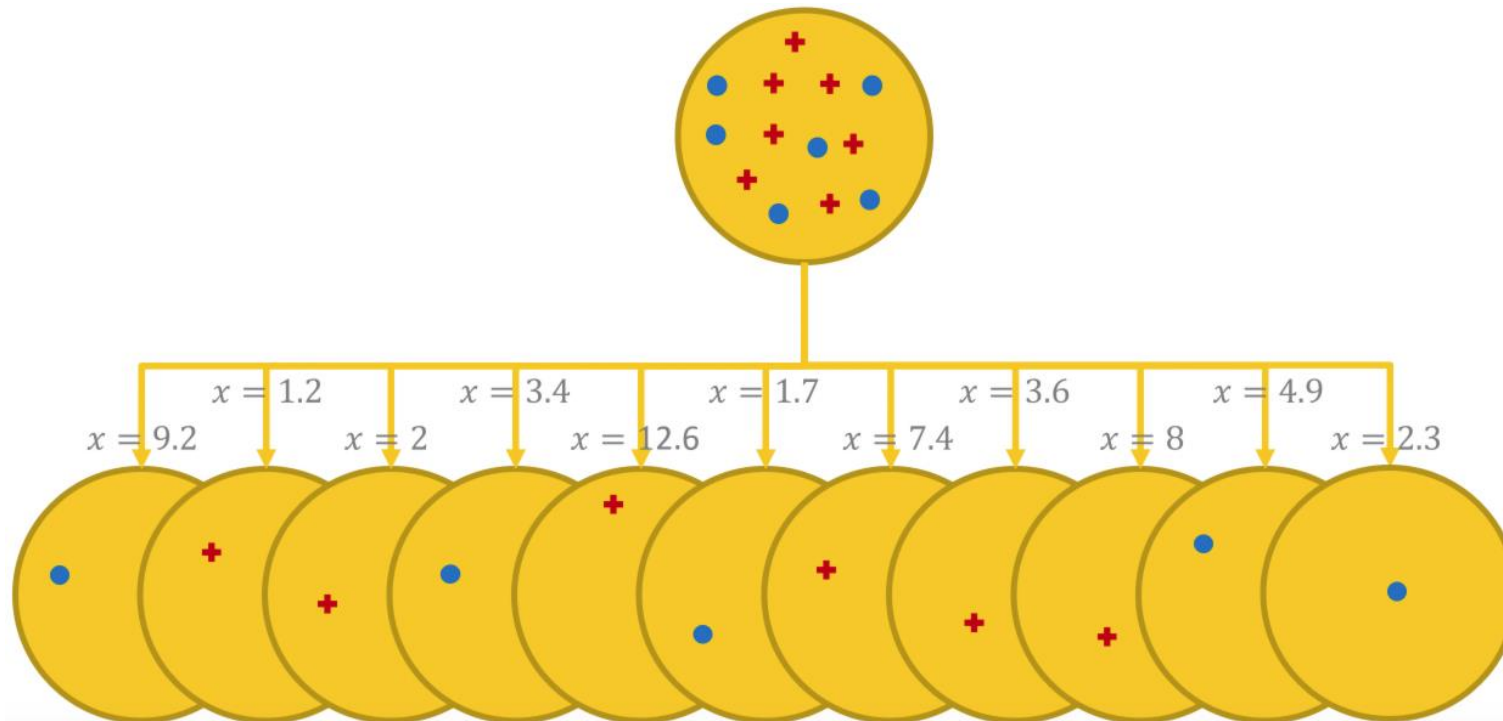
# General Steps for Tree Construction





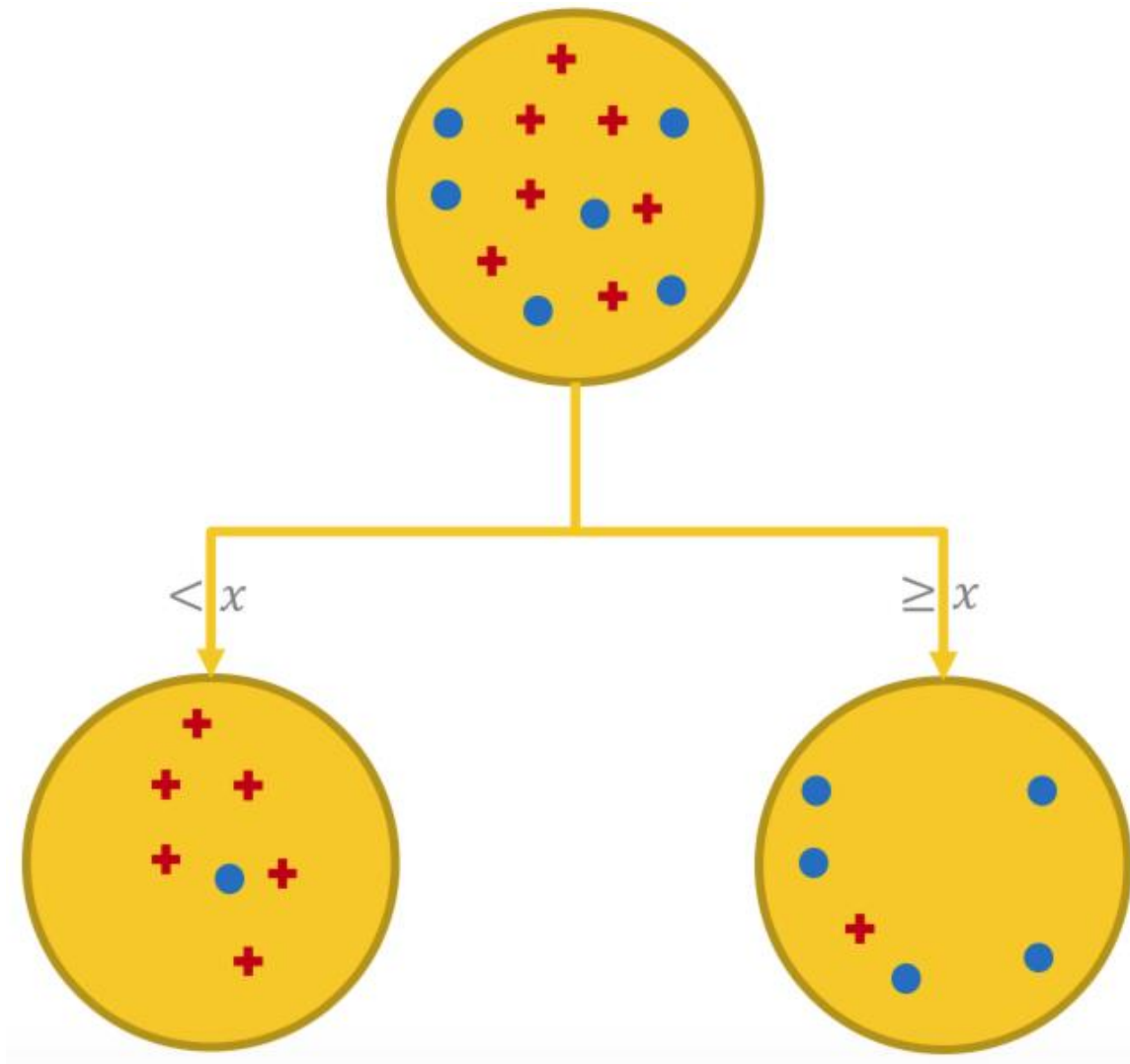
$$Gini_{impurity}(p) = 1 - \sum_{i=1}^n p_i^2 \text{ for } p \in \mathbb{Q}^n$$

$$Gini_{index} = \sum_{i=1}^n w_i Gini_{impurity_i}$$



## Numerical Feature Handling

- Is the impurity equal to 0 a good thing in all cases?
- If a cluster only contains a single data (having one branch for each possible numerical value), there is no meaning for having the cluster.

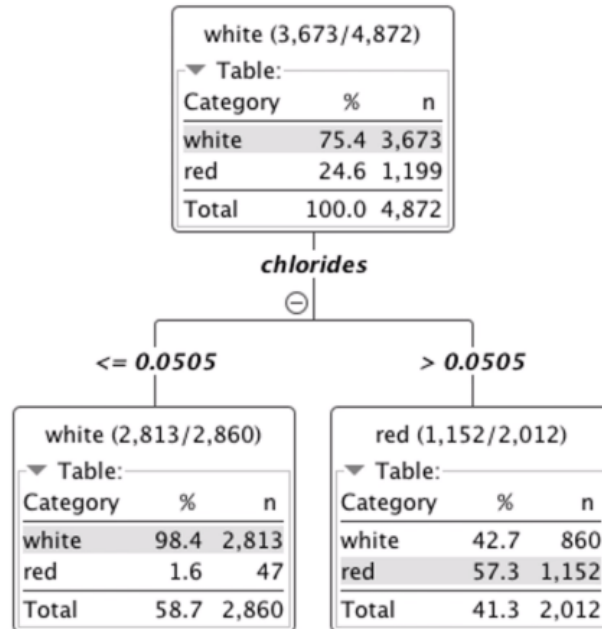


# Numerical Feature Handling

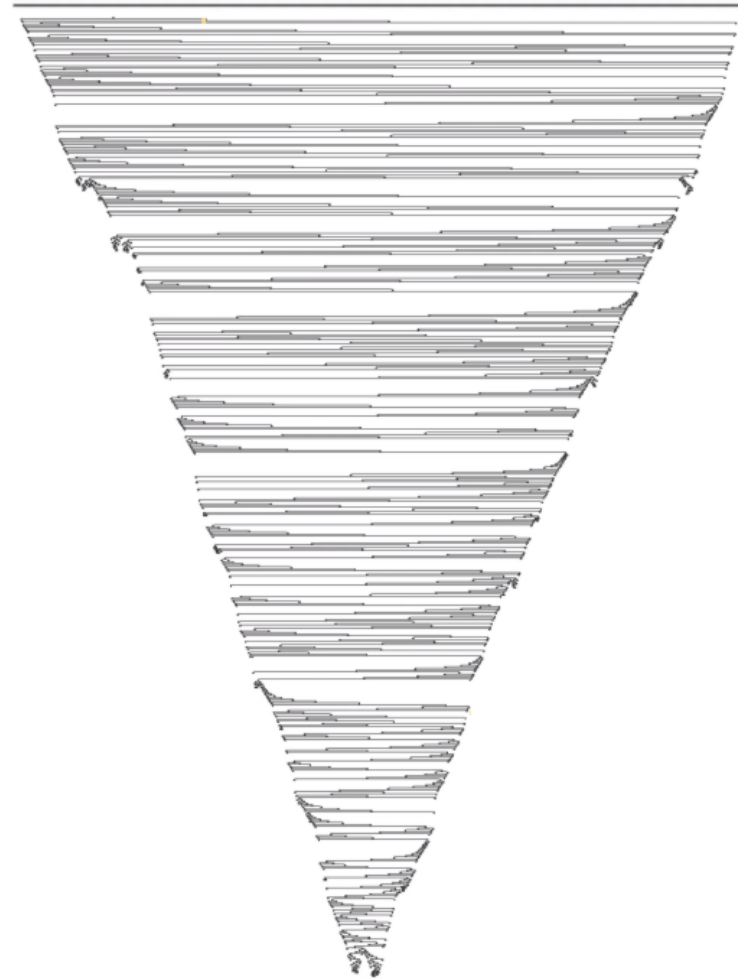
- Solution: Binary splits
  - In case of numerical features, the split is always a binary split.
- Candidate split points:
  - Mid points between consecutive values.



## Too shallow



## Too deep



## Right size of the Tree?

- Like many other machine learning algorithms, Decision Trees are subject to potentially overfitting the training data.
- Too shallow
  - Too simple model that don't fit the data.
- Too deep
  - Too detailed and don't generalize on new data.
- Solution: Pruning.



#### Pre-pruning:

- Limit the minimum samples in a group.
- Limit the depth of the tree.

#### Post-pruning:

- Let it grow first and then prune it back.

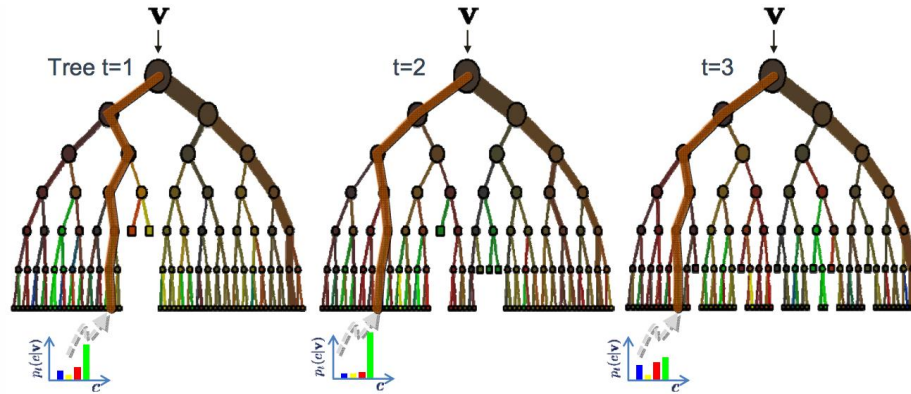
# Pruning

- **Goal**
  - Tree that generalizes better.
- **Idea**
  - Cut branches that seems as a result from overfitting the training set.
- **Common pruning methods**
  - **Reduced Error Pruning**
    - Prunes a branch if this doesn't decrease the accuracy for the training set.
  - **Minimum Description Length Pruning**
    - Prunes a branch if this decreases the description length ( $\#bits(\text{tree}) + \#bits(\text{misclassified samples})$ ) for the training set.

The description length is defined as the number of bits that are needed to encode a tree plus the number of bits to encode the samples of the training set that have been misclassified.

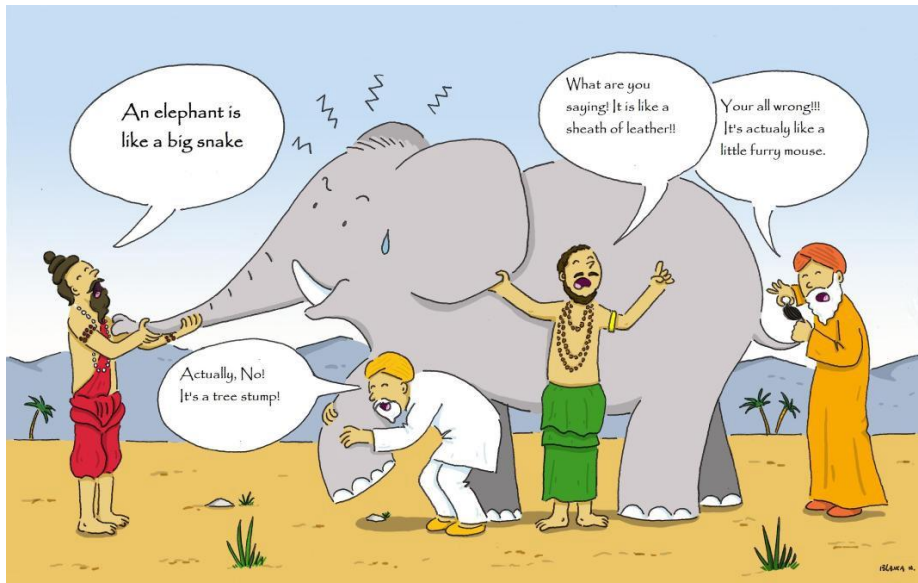
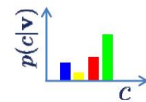






The ensemble model

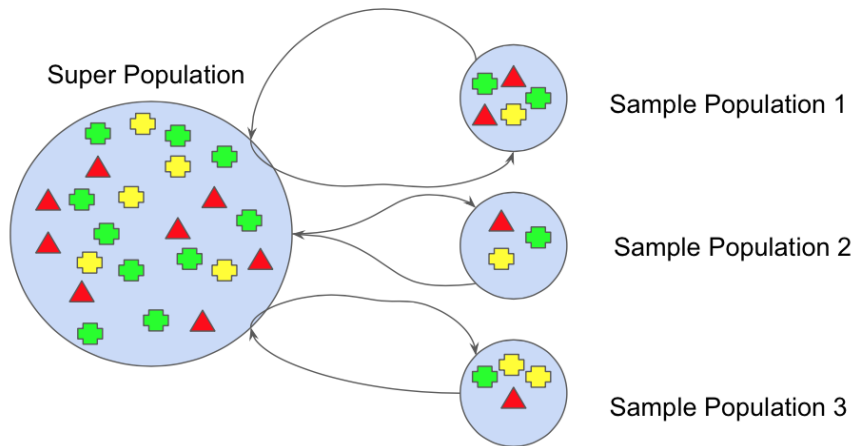
Forest output probability  $p(c|\mathbf{v}) = \frac{1}{T} \sum_t p_t(c|\mathbf{v})$



# Ensemble Learning

- Why is it necessary to ensemble?
  - To train a super strong classifier is relatively difficult comparing to train multiple weak classifiers.
  - The final result can be obtained by voting or by taking the average of the output.
- Commonly used methods:
  - Bagging (Bootstrapping)
  - Boosting
  - Stacking





# Bootstrapping

## Bootstrap refers to

- Random sampling with replacement.

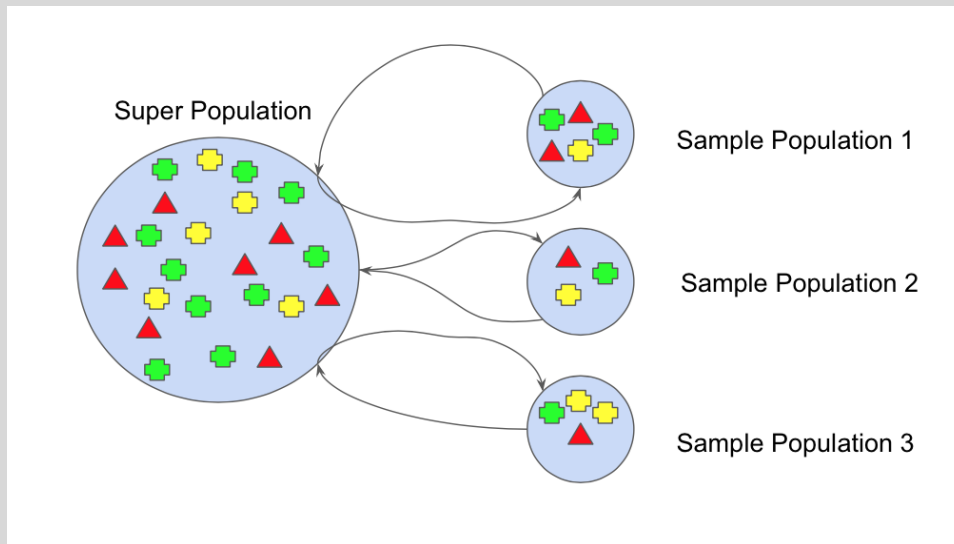
## Strong points

- To better understand the bias, mean, the variance, and the standard deviation from the dataset.

## Key operations

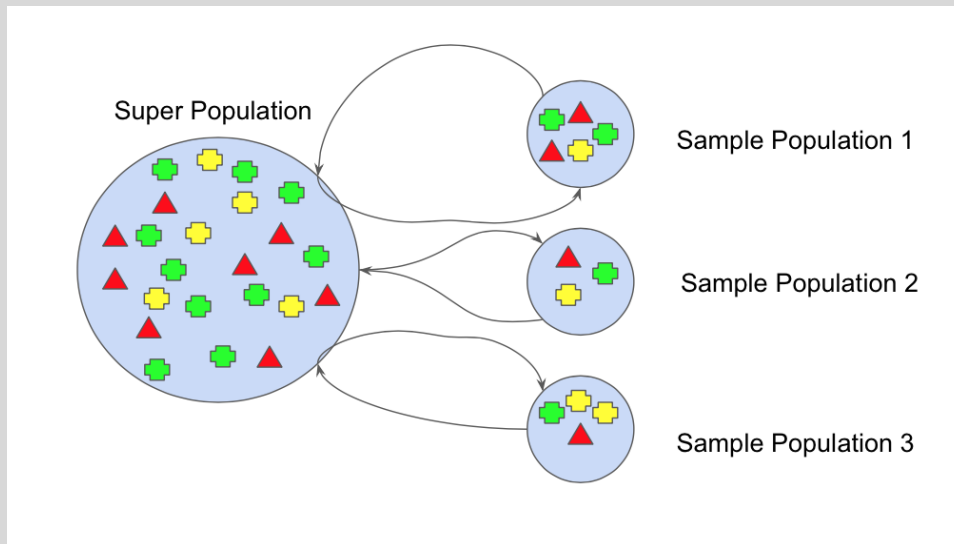
- Random sampling of small subset of data from the dataset.
- The subset can be replaced.
- The selection of all examples in the dataset has equal probability.

# Bootstrapping (2)



- Assume we have a set of data  $X = \{x_1, x_2, \dots, x_n\}$  containing  $n$  values, where  $n$  is not a relatively large number.
- If we simply calculate the mean, standard deviation, etc. by taking the whole dataset into account, the result can be biased from the actual situation.
- $$\text{mean}(X) = \frac{\sum_{i=1}^n x_i}{n}$$
- We know that our sample is small and thus the mean has error in it.
- The estimation can be improved by bootstrapping.

# Bootstrapping (3)



## Create Subsamples

- Create many (.e.g.  $m$ ) random sub-samples from the dataset with replacement (meaning we can select the same value multiple times).

## Mean Calculation

- Calculate the mean of each sub-sample.

## Ensemble Results

- Calculate the average of all collected means and use them as the estimated mean for the data.

# Bootstrapping Application Example

- Assume we have a small sample dataset:  $X = \{13, 21, 17, 22, 4, 5, 9, 28\}$  extracted from the population  $\hat{X} = \{17, 10, 6, 13, 9, 12, 28, 13, 0, 4, 21, 5, 22, 9, 13, 4, 21, 16, 3, 22\}$ .
- You'll find that the mean value of this dataset is  $mean(X) = \frac{\sum_{i=1}^n x_i}{n}$ , where  $n = 8$  in this case and  $mean(X) = 14.88$
- To calibrate the mean value, we can randomly take  $m$  sub-samples. Let's say  $m = 5$  as an example.

$$X_1 = \{13, 22, 4, 28\}$$

$$X_2 = \{13, 21, 5, 9, 28\}$$

$$X_3 = \{17, 4, 9\}$$

$$X_4 = \{21, 22, 5, 28\}$$

$$X_5 = \{13, 17, 4, 9\}$$

Find the means



$$s_1 = mean(X_1) = \frac{13 + 22 + 4 + 28}{4} = \frac{67}{4} = 16.75$$

$$s_2 = mean(X_2) = \frac{13 + 21 + 5 + 9 + 28}{5} = \frac{76}{5} = 15.2$$

$$s_3 = mean(X_3) = \frac{17 + 4 + 9}{3} = \frac{30}{3} = 10$$

$$s_4 = mean(X_4) = \frac{21 + 22 + 5 + 28}{4} = \frac{76}{4} = 19$$

$$s_5 = mean(X_5) = \frac{13 + 17 + 4 + 9}{4} = \frac{43}{4} = 10.75$$

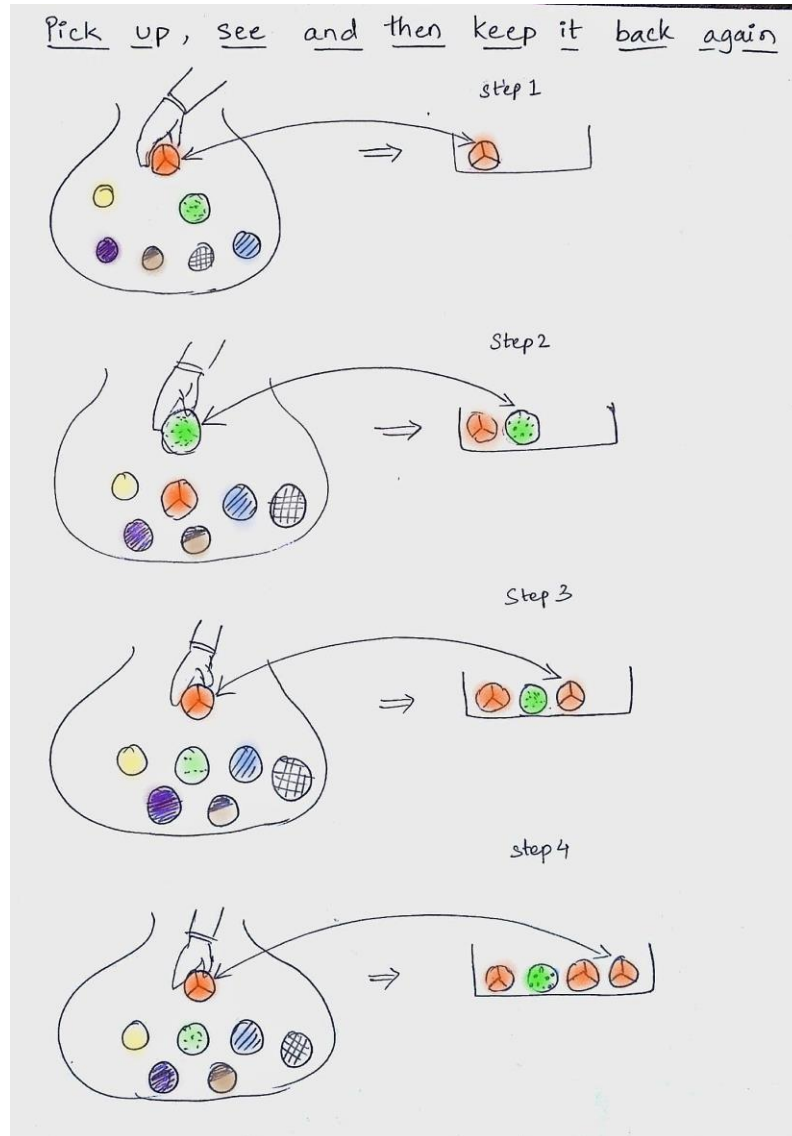
- Ensemble the final results:

$$\text{Calibrated Mean} = \frac{\sum_{j=1}^m s_j}{m} = \frac{16.75 + 15.2 + 10 + 19 + 10.75}{5} = \frac{71.7}{5} = 14.34$$

$$\text{Population Mean} = 12.4$$

Is this always the case?



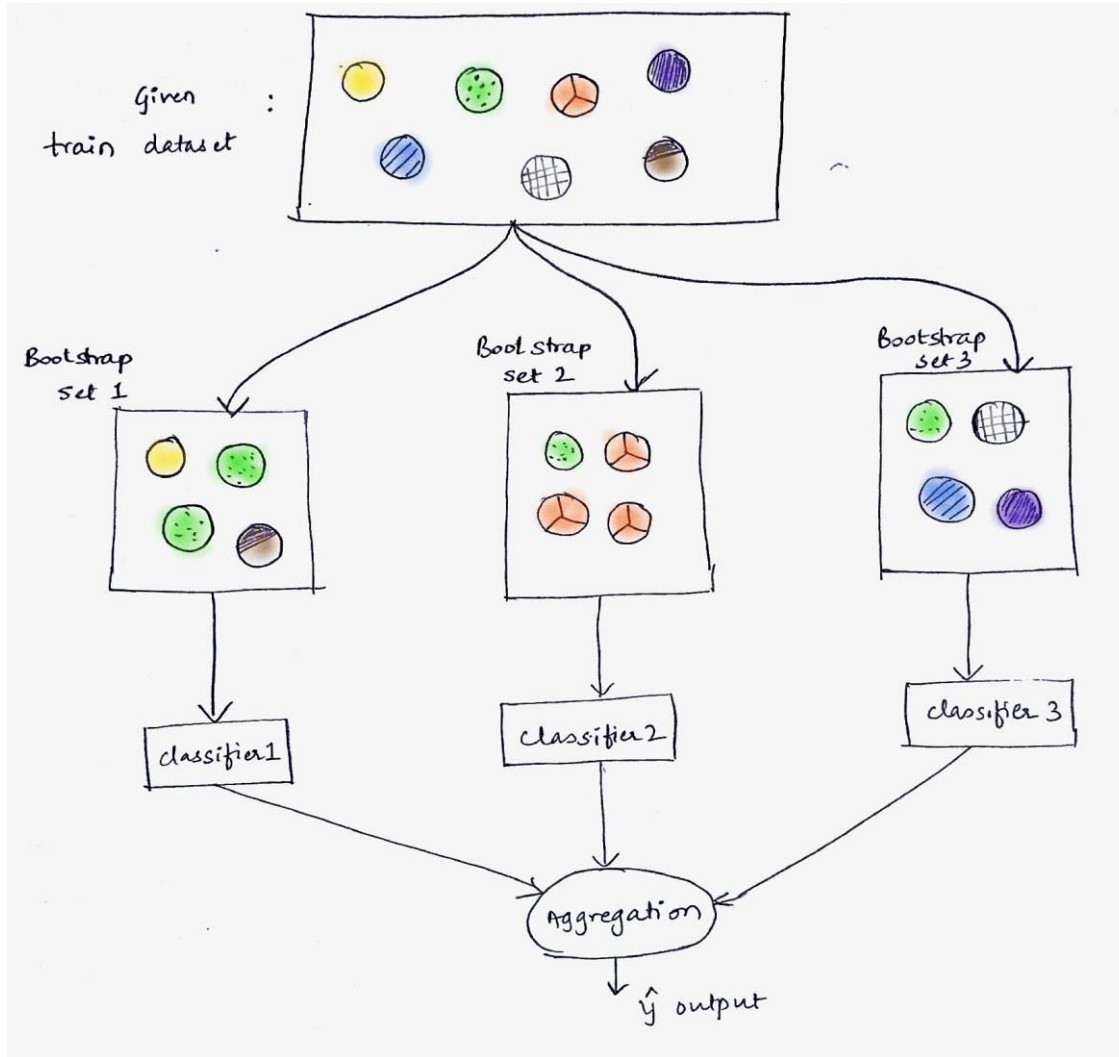


# Bagging

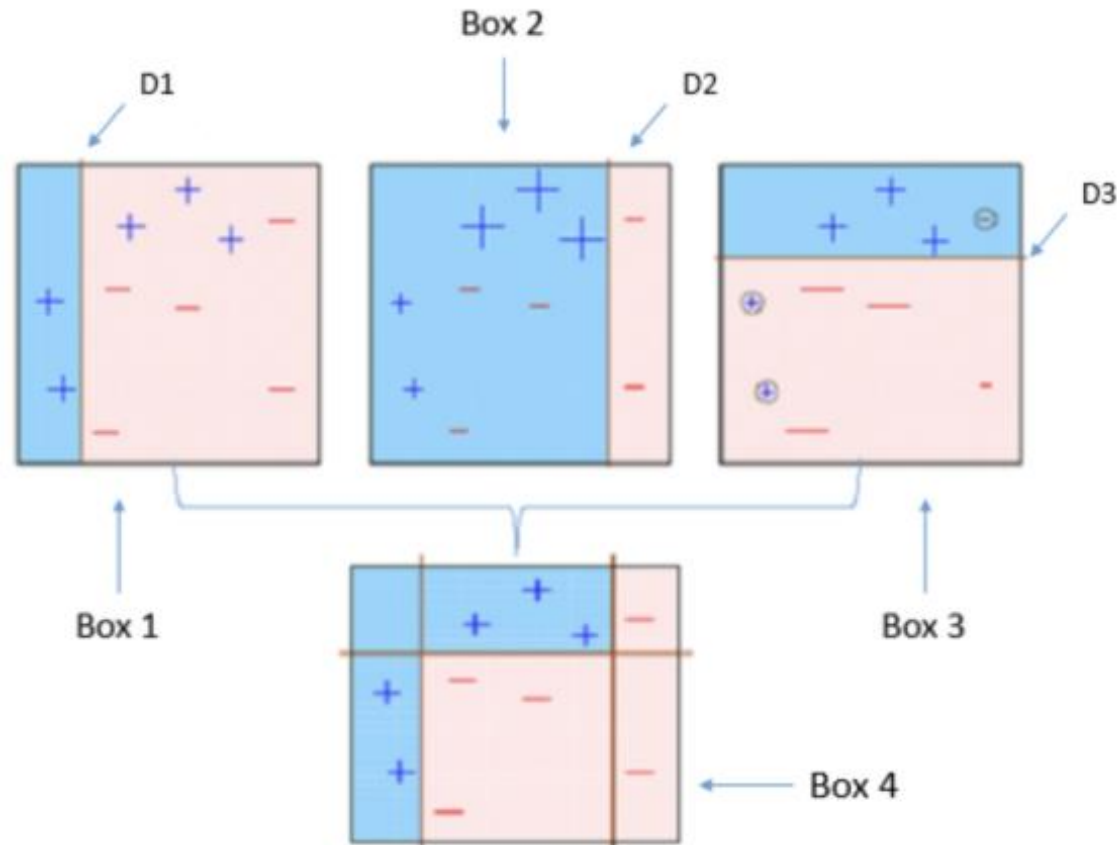
- **Bagging** is also known as **Bootstrap Aggregation**.
- A simple but a powerful ensemble method.
- It is typically an application of the Bootstrap procedure to **decision tree**, which is a high-variance machine learning algorithm.
- In bagging, training instances can be **sampled several times for the same predictor**.

# Bagging (2)

- Moving on to bagging ensemble learning, the output of each predictor (classifier) is summarised by the aggregation function.
- The aggregation function is typically the statistical **mode** for **classification** and **average** for **regression**.
- It helps to reduce both bias and variance.



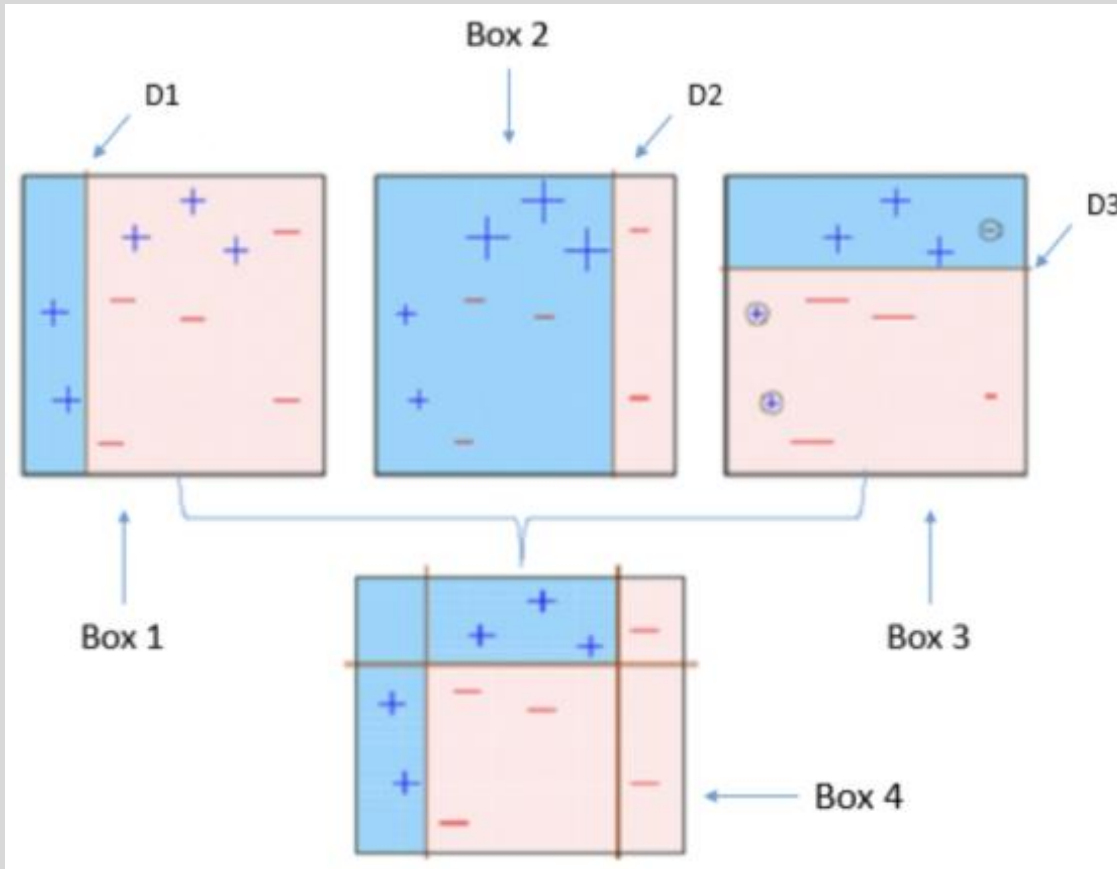
# Boosting



- In boosting algorithms, they utilise the weighted averages to mark weak learners into strong learners.
- Different from bagging, which runs each model independently before aggregating the outputs at the end without preference to any model, boosting is all about “teamwork.”
- Each model that runs in boosting, dictates what features the next model will focus on.

# Boosting

- **Data points**
  - + and - samples are equally weighted at the beginning. (The weight is denoted by the size of the symbols.)
- **Box1**
  - The decision stump (D1) generates a vertical line to classify the data points.
  - For the three + on the right side, higher weights are assigned to those + on the right side and the classification process is handed over to D2.
- **Box2**
  - Since there are three + having higher weights, the second decision strum (D2) will try to predict them correctly.
  - A vertical line on the right side of box2 has classified the three mis-classified + correctly.
  - But again, it causes another mis-classification errors on the three - on the left side. Thus, those three - would gain weight in D3.





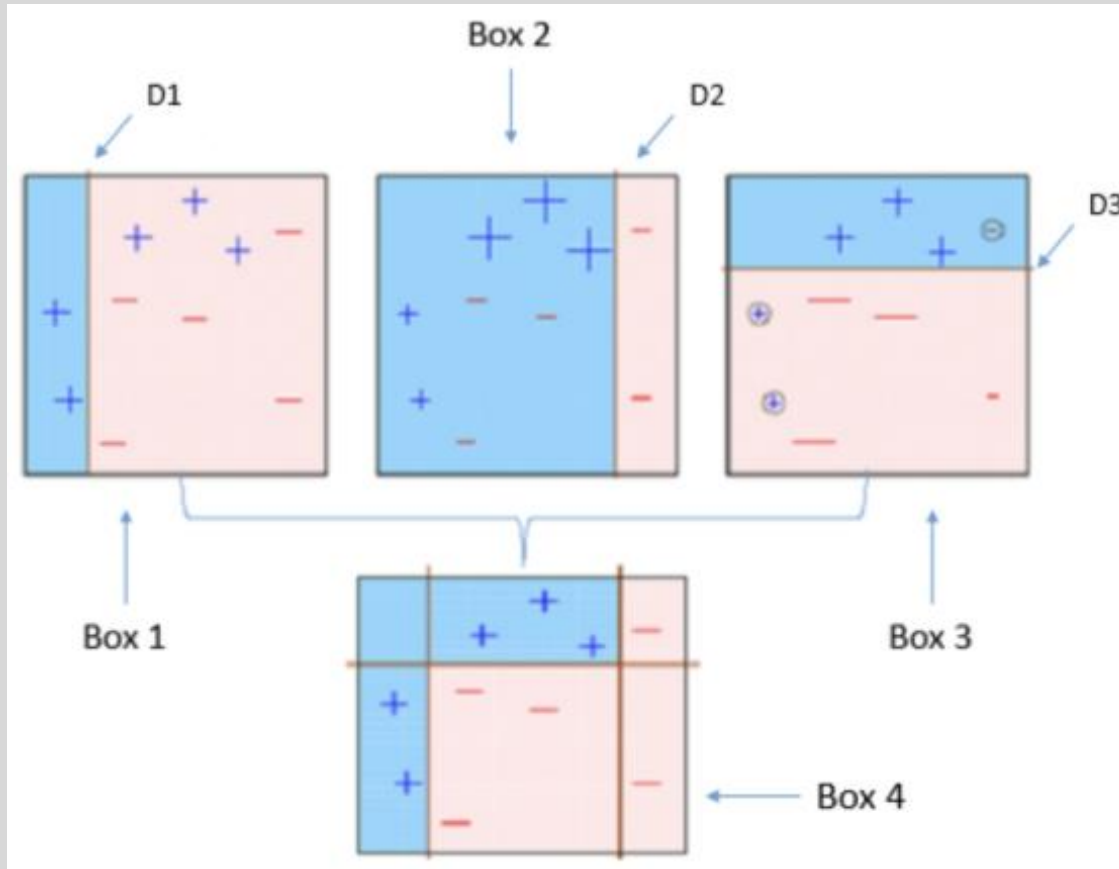
# Boosting

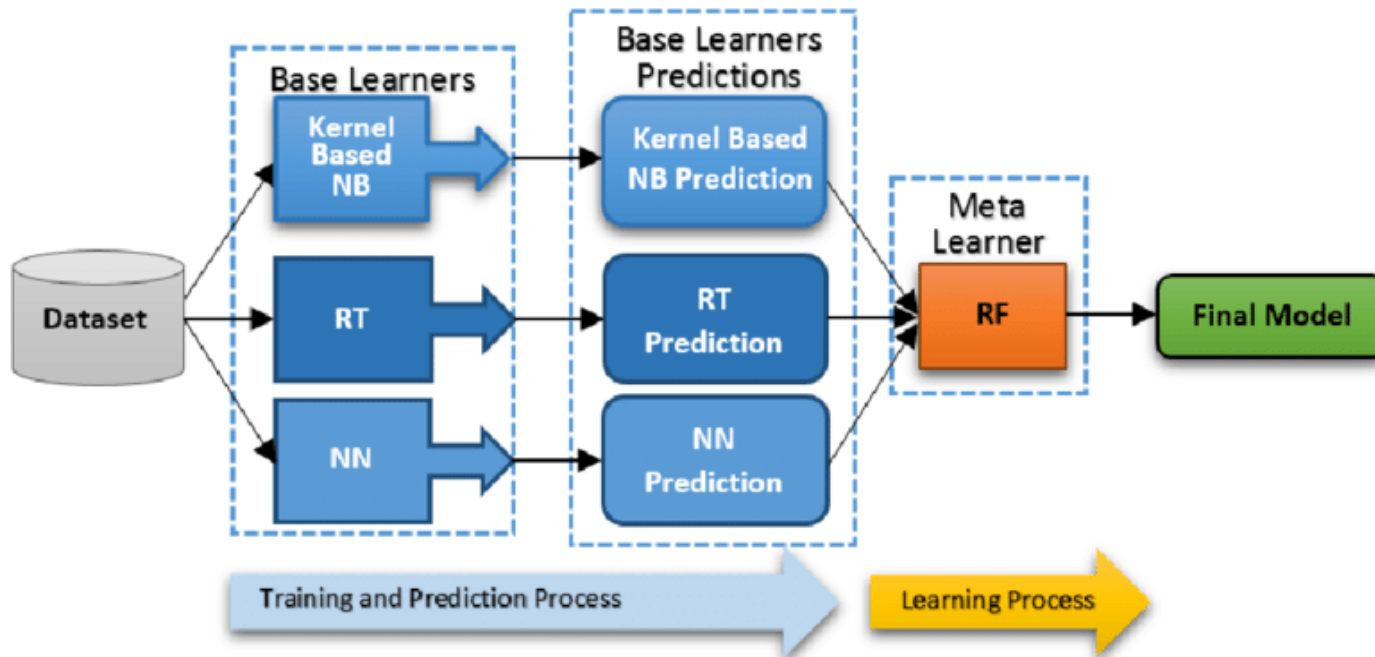
- **Box3**

- Since the three mis-classified – are given higher weights, a decision stump (D3) is applied to predict these mis-classified observation correctly.
- This time, a horizontal line is generated to classify + and – based on higher weight of mis-classified observations.

- **Box4**

- By combining D1, D2, and D3, a strong predictor, which has complex rules as compared to individual weak learner, is generated.



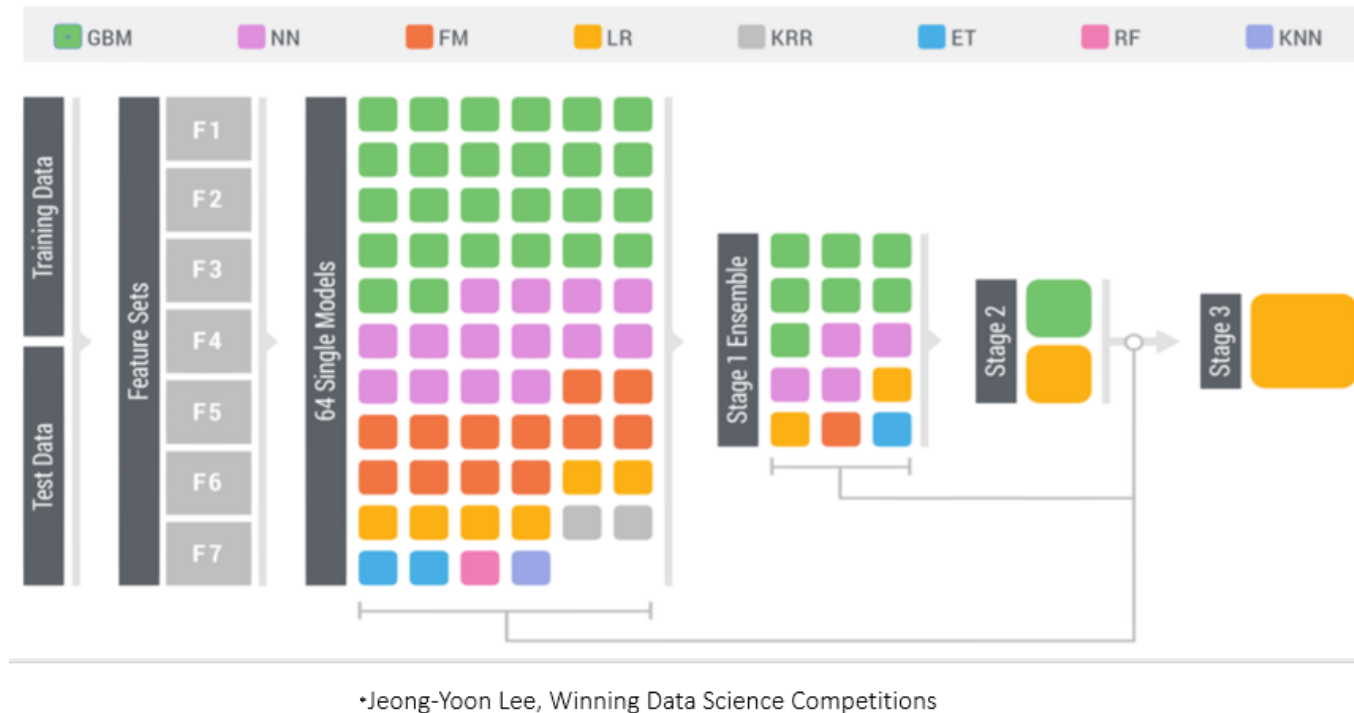


# Stacking

- Stacking often considers **heterogeneous** weak learners, learns them in parallel and combines them by training a **meta-model** to output a prediction based on the different weak models predictions.
- In recent years, a more modern approach is to create an ensemble of a well-chosen collection of strong yet diverse models in stacking.

# Stacking

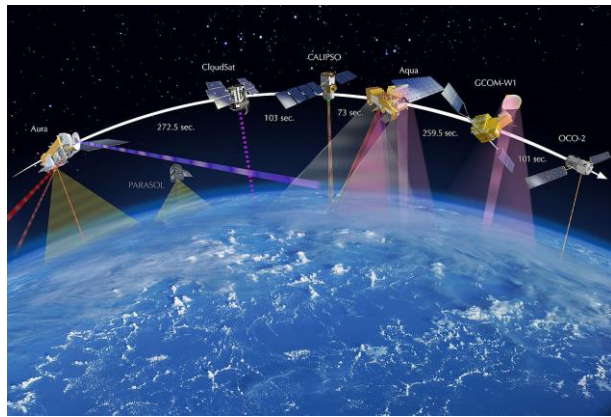
- On the popular data science competition site Kaggle (<https://www.kaggle.com/>), you can explore numerous winning solutions through its discussion forums to get a flavour of the state of the art.
- Another popular data science competition is the KDD Cup (<http://www.kdd.org/kdd-cup>). The figure shown on the left is the winning solution for the 2015 competition, which used a three-stage stacked modelling approach.





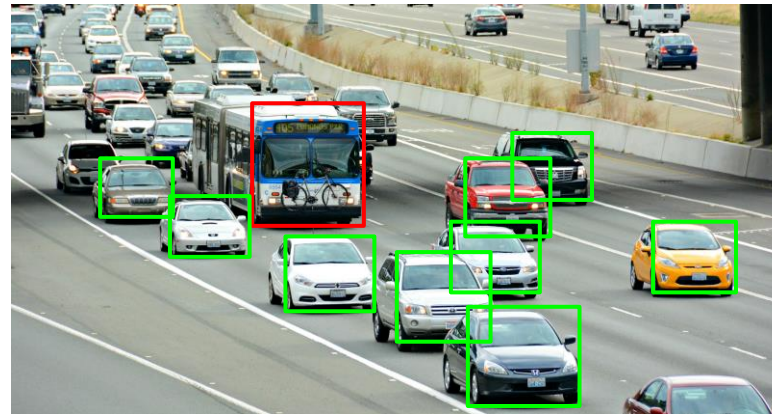


# RANDOM FOREST



Using Enhanced Thematic Mapping (ETM) devices to acquire images of the earth's surface.

The accuracy is higher while the training time is less comparing to other ML methods.



Multiclass object detection is done using Random Forest algorithms

Provides better detection in complicated environments



Random Forest is used in the KINECT game console.

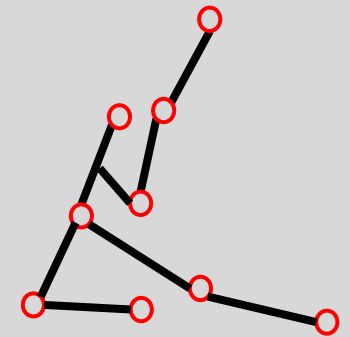
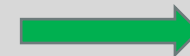
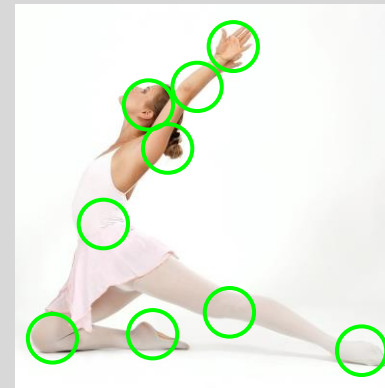
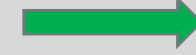
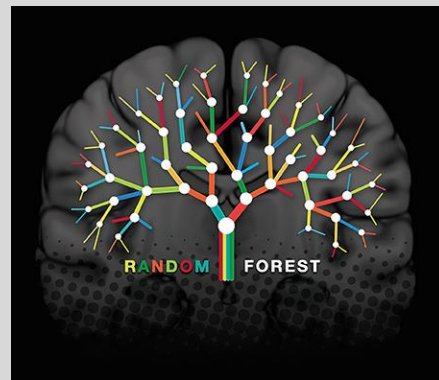
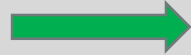
Tracks body movements and recreates it in the game

# Where to use Random Forest?

- Remote Sensing
- Object Detection
- Gaming console (Object Detection and Tracking)



# Application of Random Forest



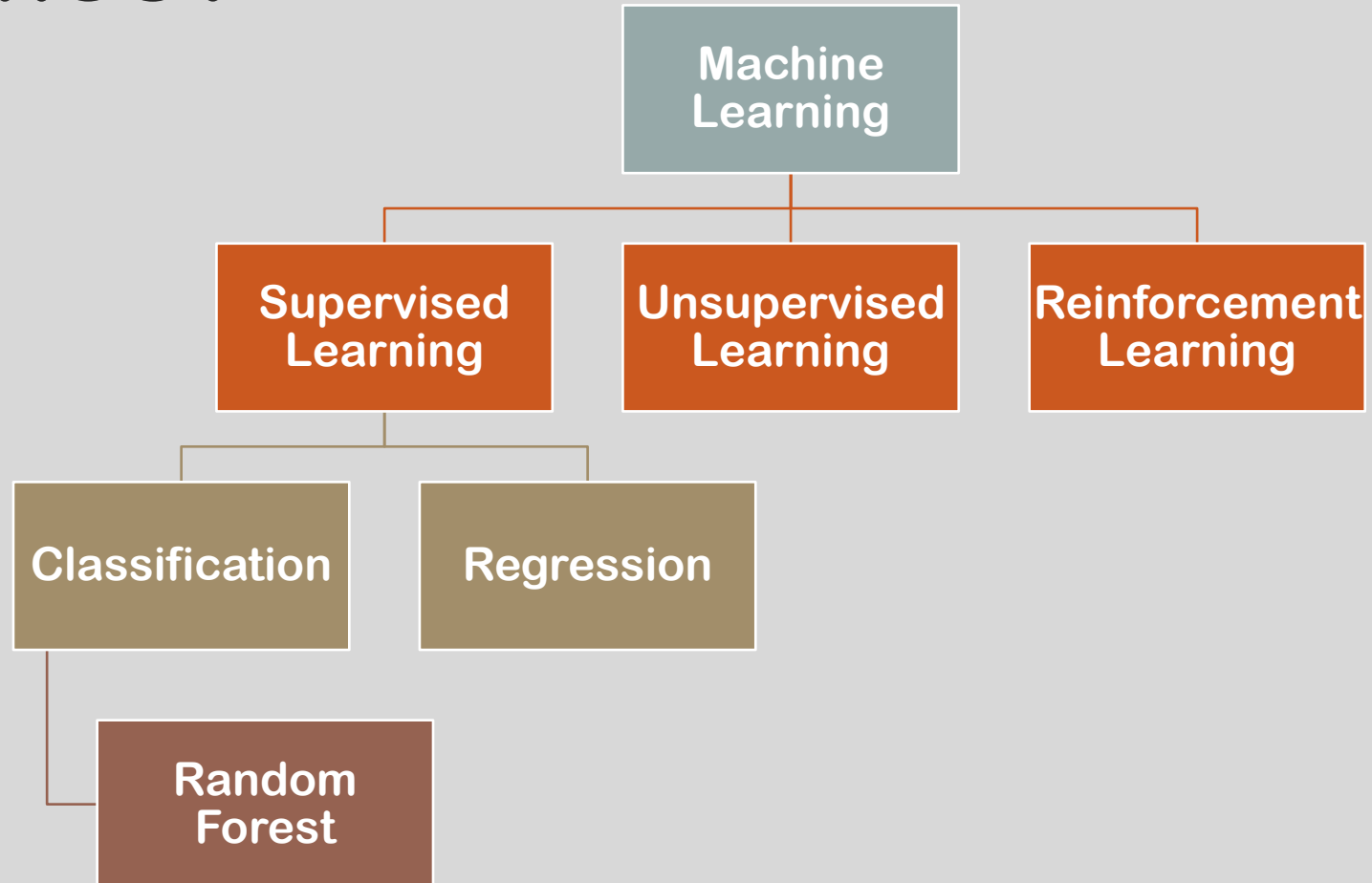
Training set for  
identifying body parts

Classifier Learning

Identify body parts

Mapping

# Where does Random Forest fit in Data Science?



# Why using Random Forest?

## No Overfitting

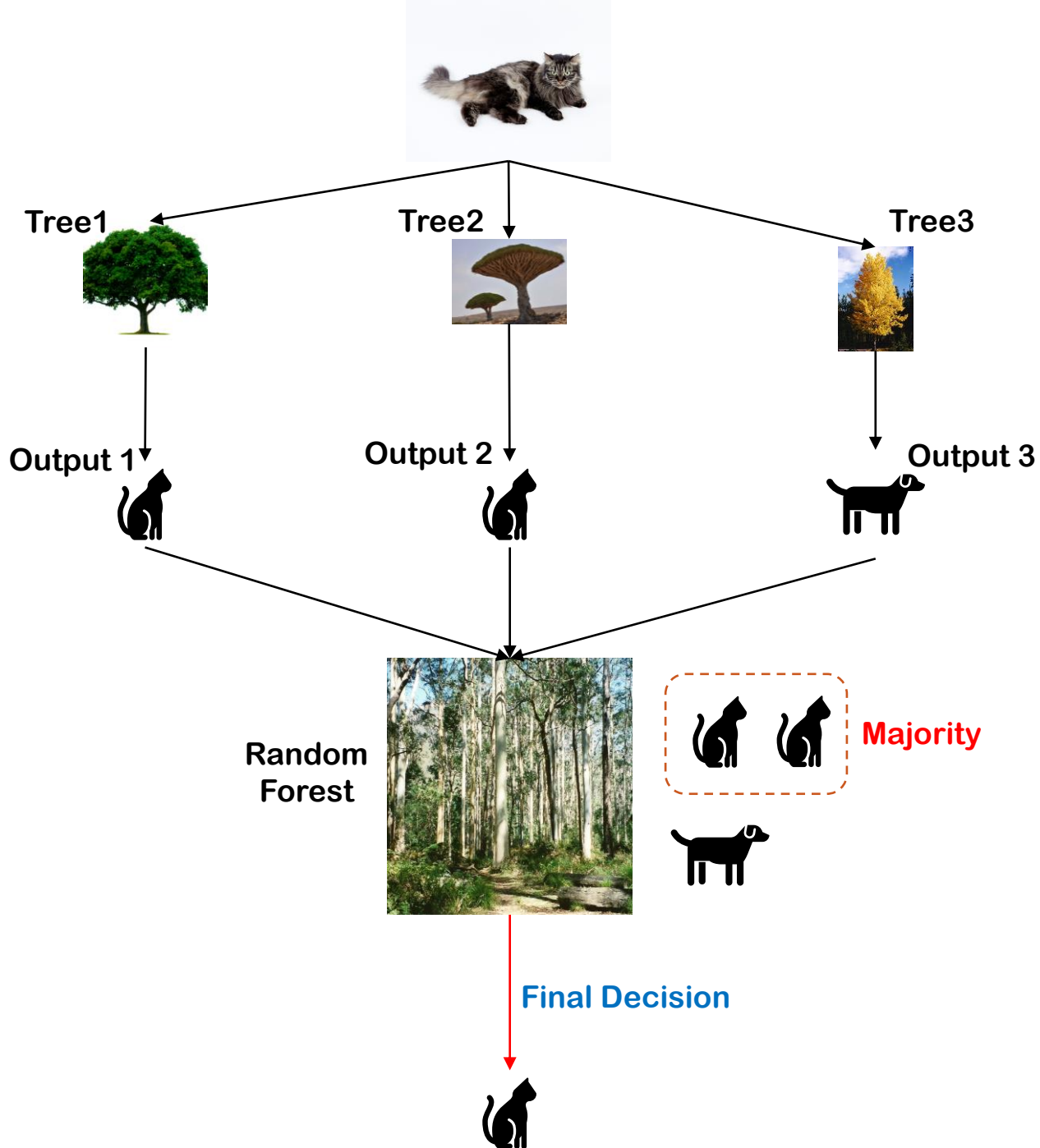
- Use of multiple trees to reduce the risk of overfitting.
- Less training time.

## High Accuracy

- Runs efficiently on large dataset.
- Produces highly accurate predictions in large dataset.

## Estimates missing data

- Random Forest can maintain accuracy when a large proportion of data is missing.



# What is Random Forest?

## Definition

- Random Forest or Random Decision Forest is a method that operates by constructing multiple Decision Trees during training phases.

## Operation

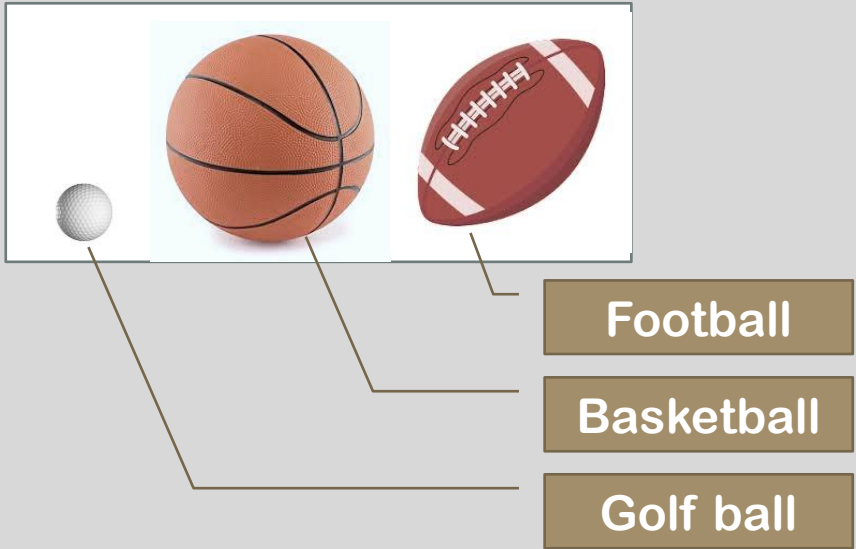
- It's an application of stacking.



# HOW DOES RANDOM FOREST WORK?



# Let's Review How Decision Tree Works, First.

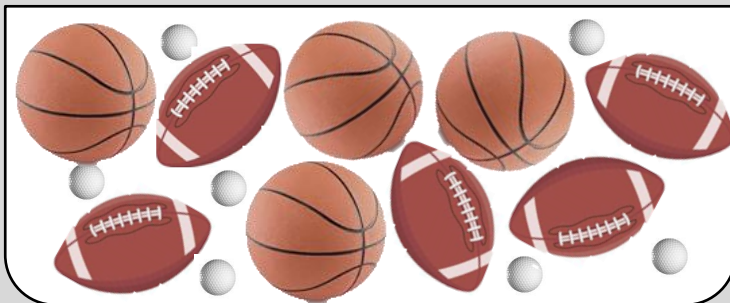


Classification Criteria

S/N	Conditions
1	Colour == Brown?
2	Diameter < 8?
3	Colour == White?
4	Diameter == 30?

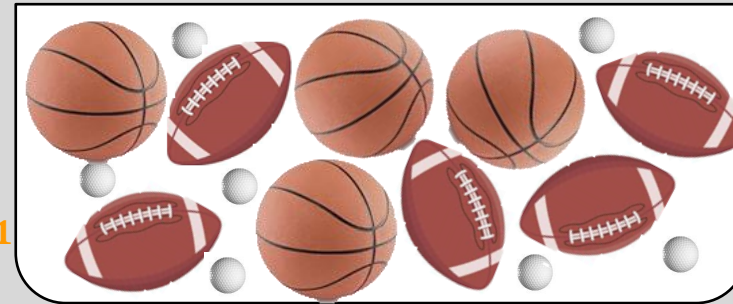
Training Sample

Colour	Diameter	Shape	Label
White	5	Round	Golf ball
White	5	Round	Golf ball
Brown	30	Round	Basketball
Brown	32	Oval	Football
Brown	32	Oval	Football
White	5	Round	Golf ball
White	5	Round	Golf ball
Brown	30	Round	Basketball
Brown	30	Round	Basketball
Brown	32	Oval	Football



# Let's Review How Decision Tree Works, First.

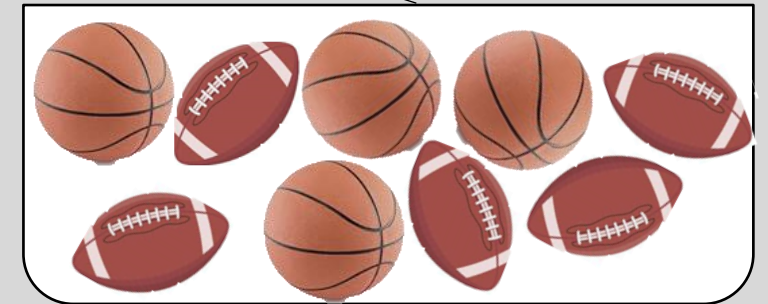
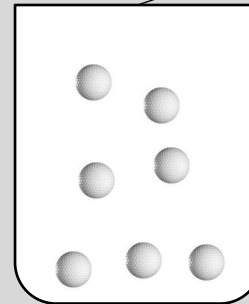
$$H_0(p) = -\sum_{i=1}^3 p_i \log_2 p_i = -\frac{7}{16} \log_2 \frac{7}{16} - \frac{4}{16} \log_2 \frac{4}{16} - \frac{5}{16} \log_2 \frac{5}{16} = 1.55 > 1$$



Is the diameter  $\leq 8$ ?

True

False



$$H_{11}(p) = -\sum_{i=1}^1 p_i \log_2 p_i = -\frac{7}{7} \log_2 \frac{7}{7} = 0$$

$$H_{12}(p) = -\frac{4}{9} \log_2 \frac{4}{9} - \frac{5}{9} \log_2 \frac{5}{9} = 0.99$$

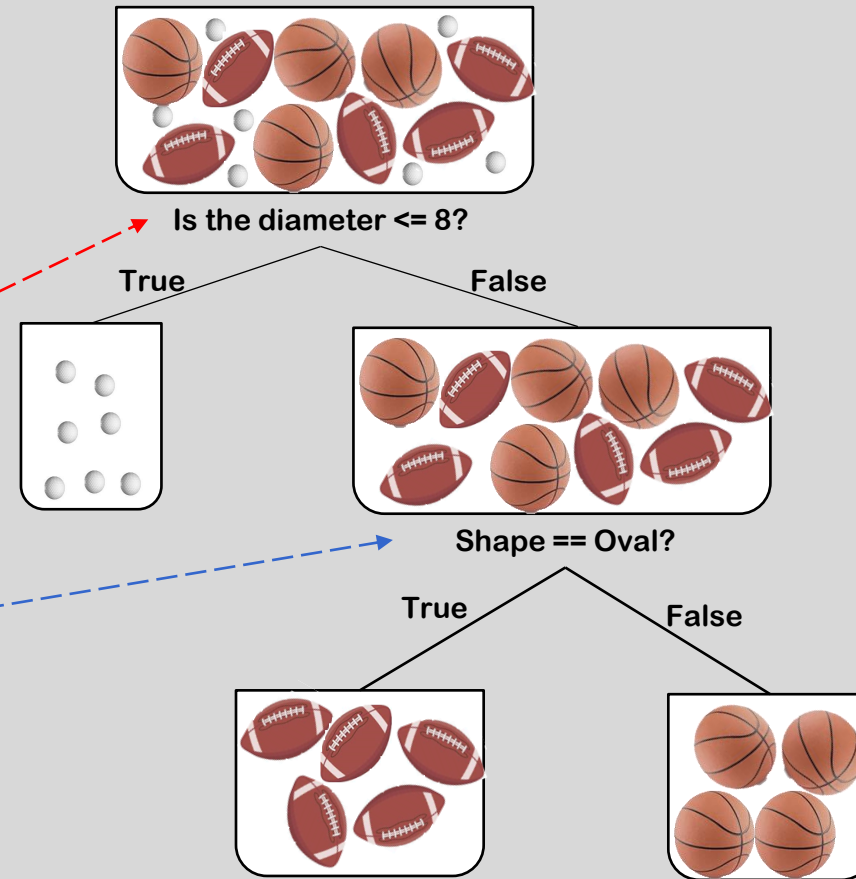
S/N	Conditions
1	Colour == Brown?
2	Diameter < 8?
3	Colour == White?
4	Shape == Oval?

Colour	Diameter	Shape	Label
White	5	Round	Golf ball
White	5	Round	Golf ball
Brown	30	Round	Basketball
Brown	32	Oval	Football
Brown	32	Oval	Football
White	5	Round	Golf ball
White	5	Round	Golf ball
Brown	30	Round	Basketball
Brown	30	Round	Basketball
Brown	32	Oval	Football

# Let's Review How Decision Tree Works, First.

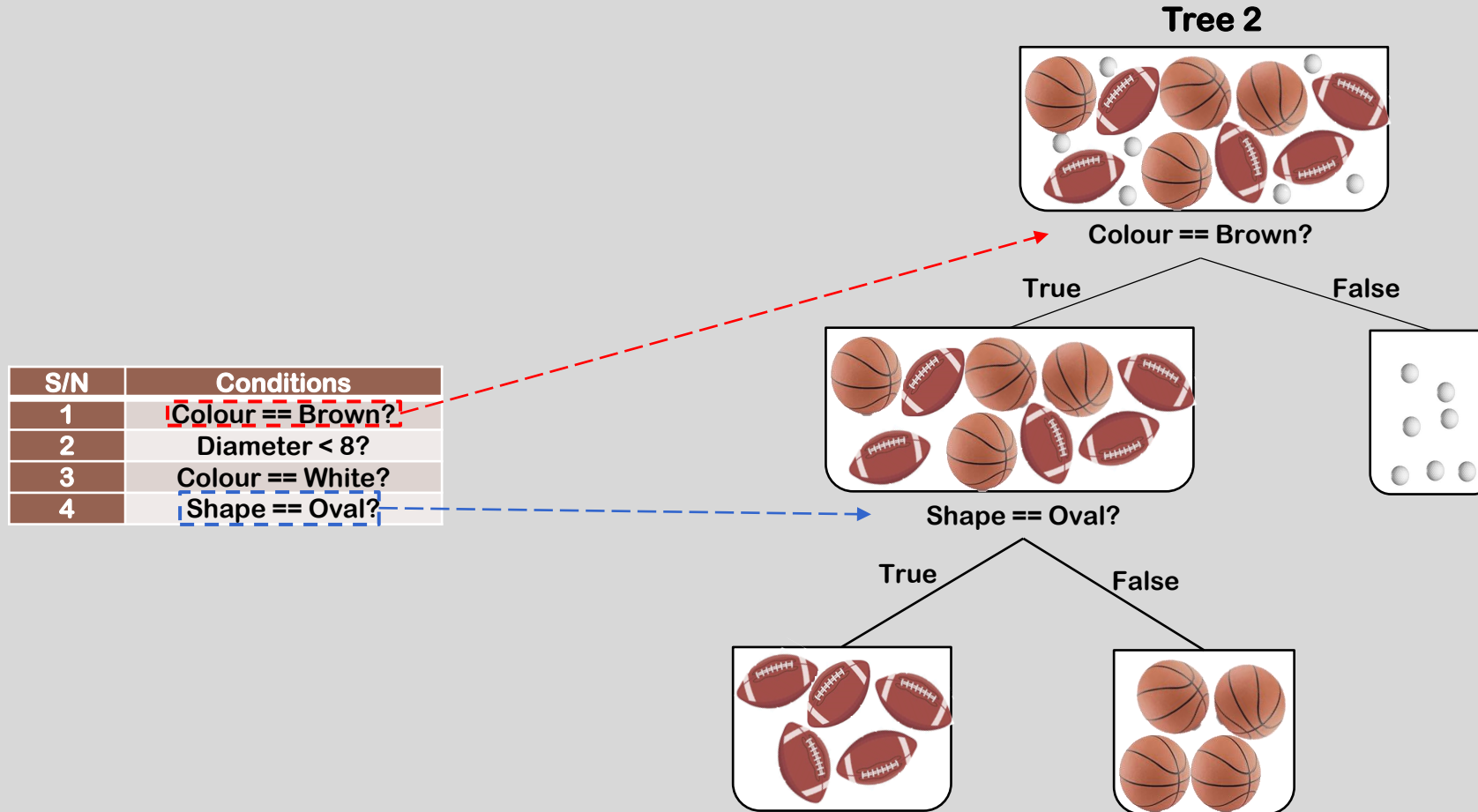
S/N	Conditions
1	Colour == Brown?
2	Diameter < 8?
3	Colour == White?
4	Shape == Oval?

Colour	Diameter	Shape	Label
White	5	Round	Golf ball
White	5	Round	Golf ball
Brown	30	Round	Basketball
Brown	32	Oval	Football
Brown	32	Oval	Football
White	5	Round	Golf ball
White	5	Round	Golf ball
Brown	30	Round	Basketball
Brown	30	Round	Basketball
Brown	32	Oval	Football

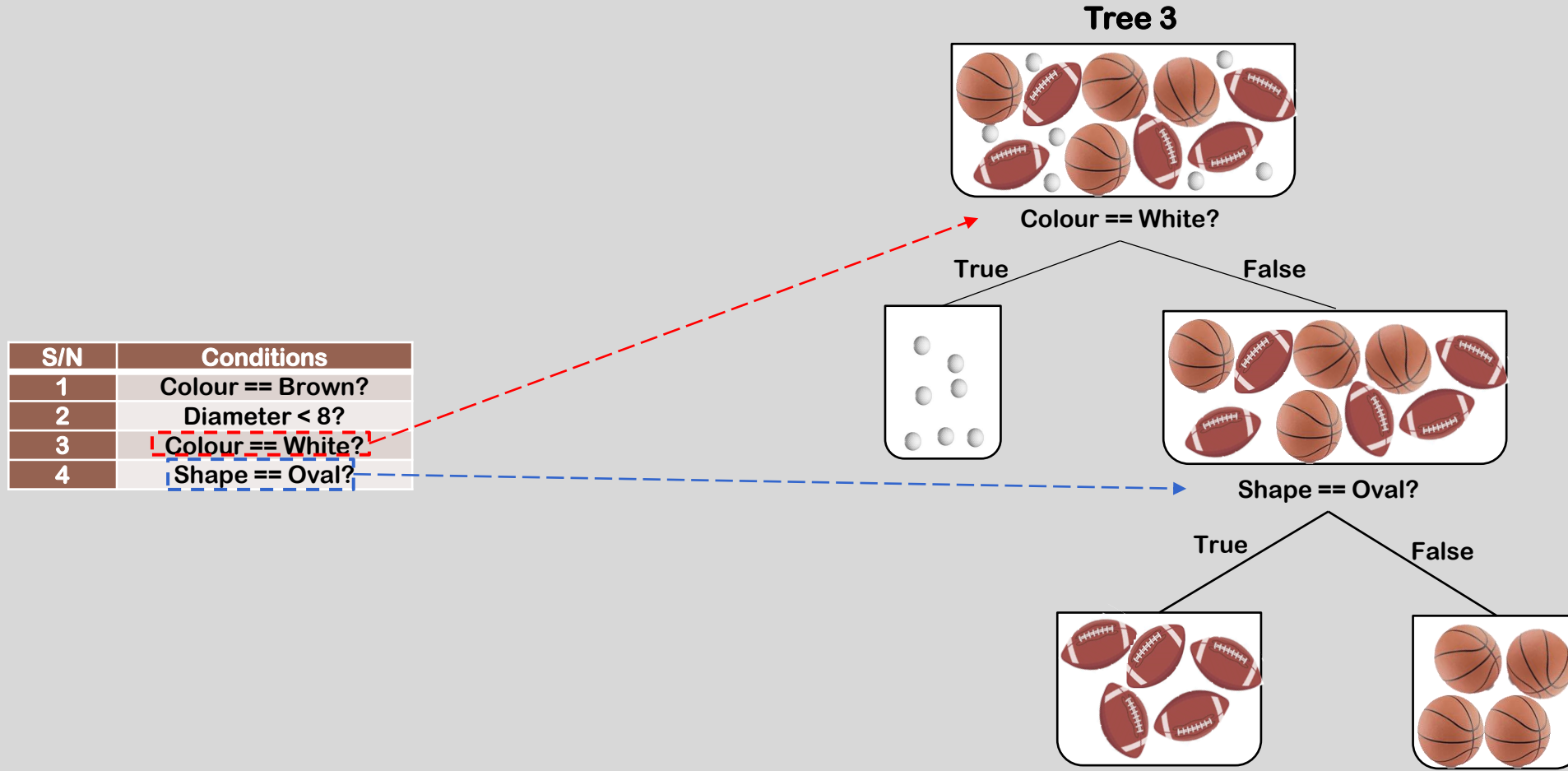


Now we have 3 baskets, whose entropies are all 0, with 100% prediction accuracy.

# How Does Random Forest Work?



# How Does Random Forest Work?



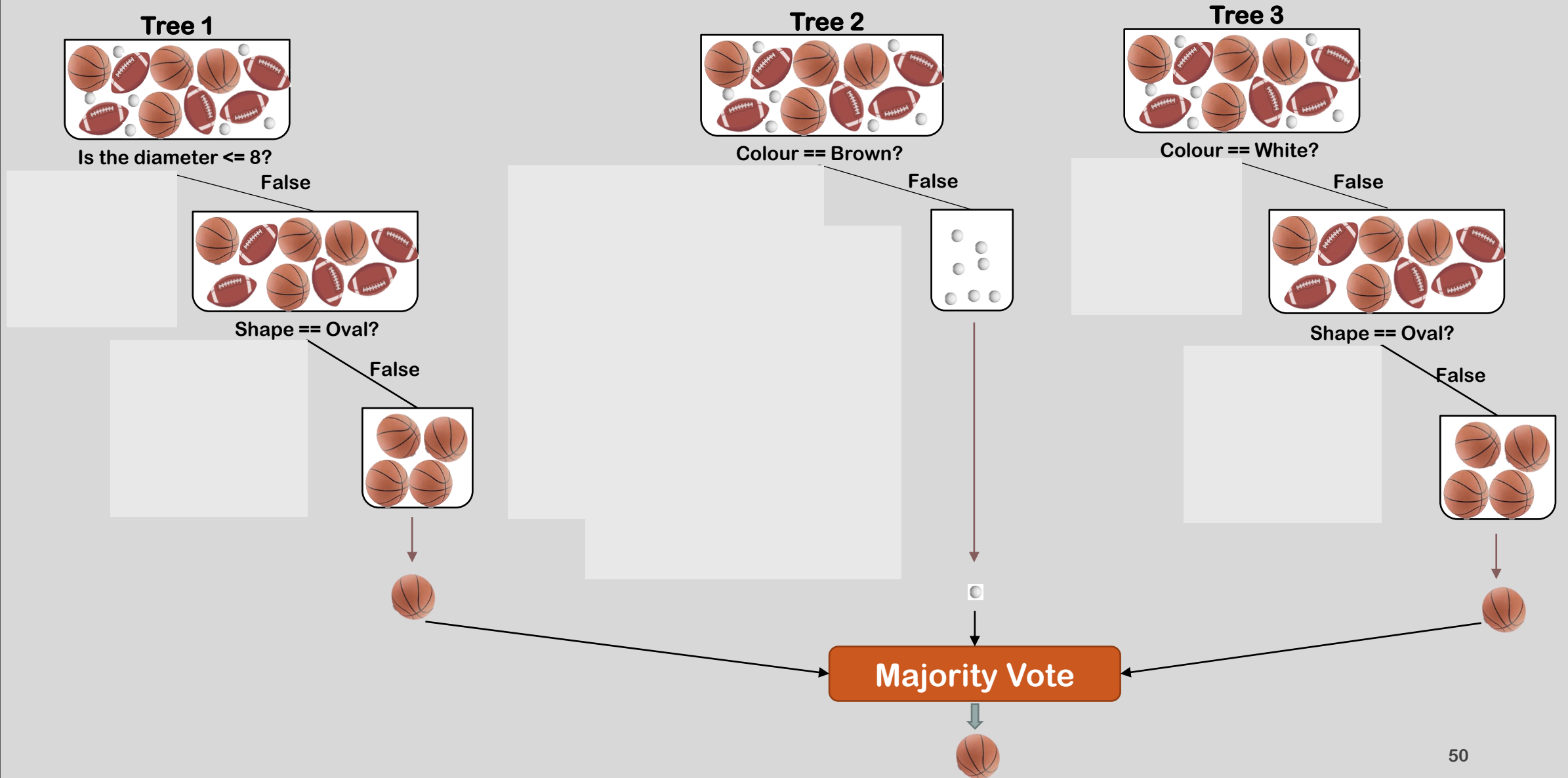


# How Does Random Forest Work?

- Assume we have a new test data with no colour information.
- Let the default answer to be false if there is no judge on the particular classifying criterion.
- Let's guess what this input will be.



# How Does Random Forest Work?





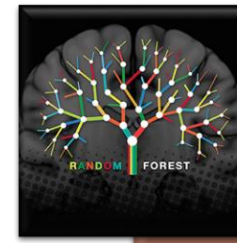
## Decision Tree

- Entropy
- Information Gain
- Gini Index
- Creation
- Pruning



## Ensemble Learning

- Bagging
- Boosting
- Stacking



## Random Forest

- Definition
- Majority Vote

## Summary

