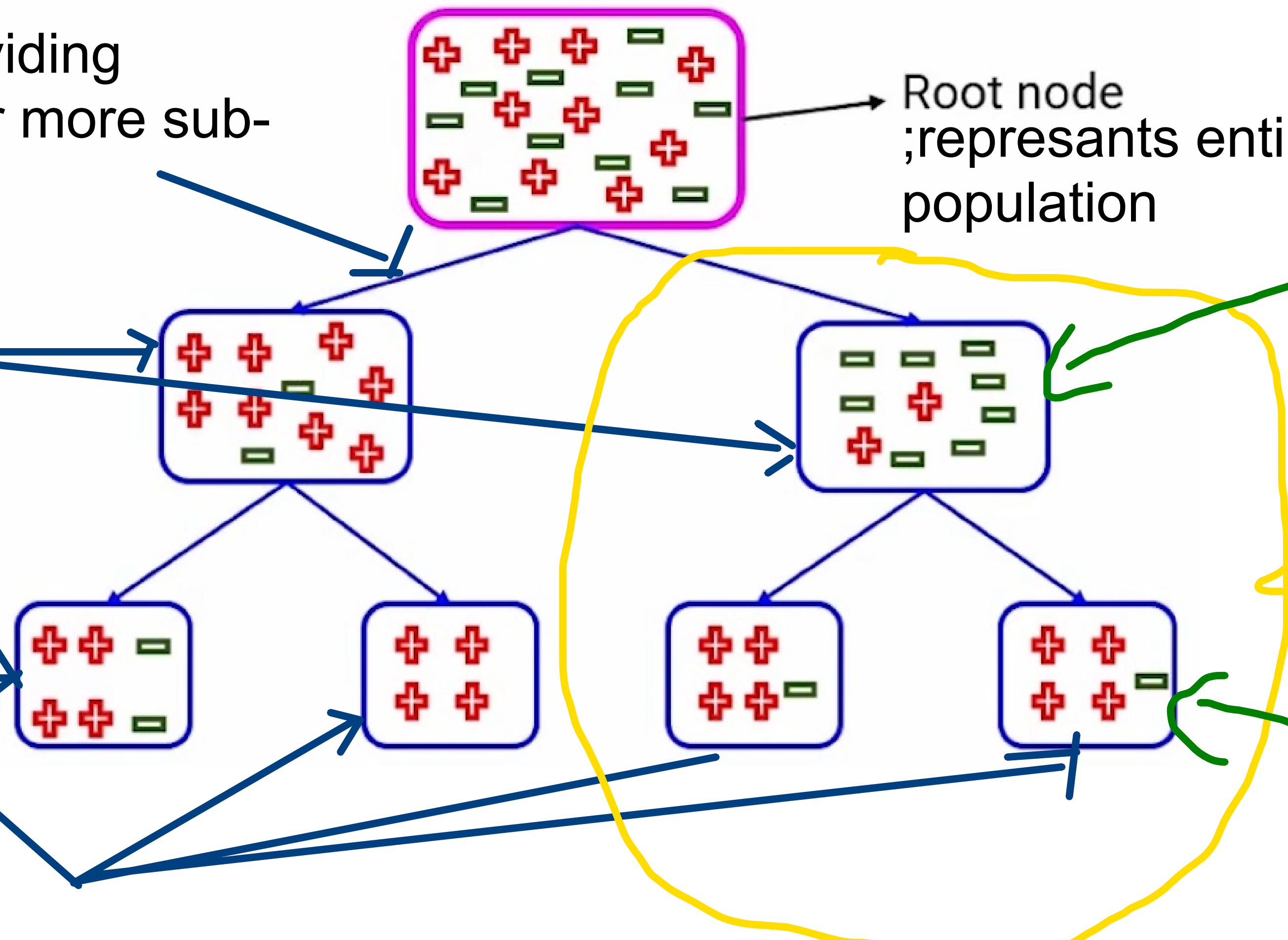
Decision Tree

splitting
;process of deviding nodes in two or more sub-nodes

Decision node



;represants entire population

parent node
;which has further devided iinto sub-node

Branch/sub tree

Leaf/terminal nodes
;which are not further devidable

child node

Depth of the tree = 2
;the longest path from root node to leaf node

- Select the split which results in most homogeneous sub-nodes

- there are multiple algo./techniques to decide the best split for tree

Information Gain

3

1

Gini Impurity

Gini impurity = 1 - Gini

Probability that randomly picked points belonging to the same class
(0 <= Gini <= 1)

- Lower the gini impurity, higher the the homogenety

- this only works with categorical targest
- only for binary splits

2

Chi- Square

- it is to measure statistical significance of differances between child nodes and their parent nodes

Chi-Square = √[(Actual - Expected)² / Expected]

- Only works for Categorical targets
- can split into two or more nodes

Steps to calcilate Gini impurity

- Calculate the gini impurity for sub-nodes :

$$Gini\ Impurity = 1 - Gini$$

- Gini = Sum of square of probabilities for each class/category

$$Gini = (p_1^2 + p_2^2 + p_3^2 + ... + p_n^2)$$

- To calculate the gini impurity for split, take weighted gini impurity of both sub-nodes of that split

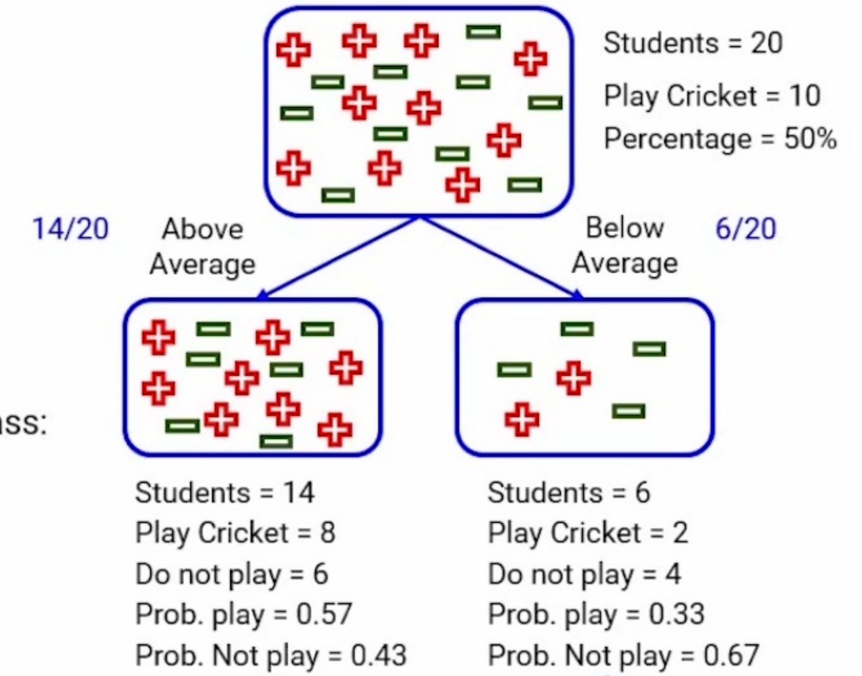
calculation for spliting based on performance of the class

Split on Performance in Class

- Gini Impurity: sub-node Above Average:
1 - [(0.57)*(0.57) + (0.43)*(0.43)] = 0.49

- Gini Impurity: sub-node Below Average:
1 - [(0.33)*(0.33) + (0.67)*(0.67)] = 0.44

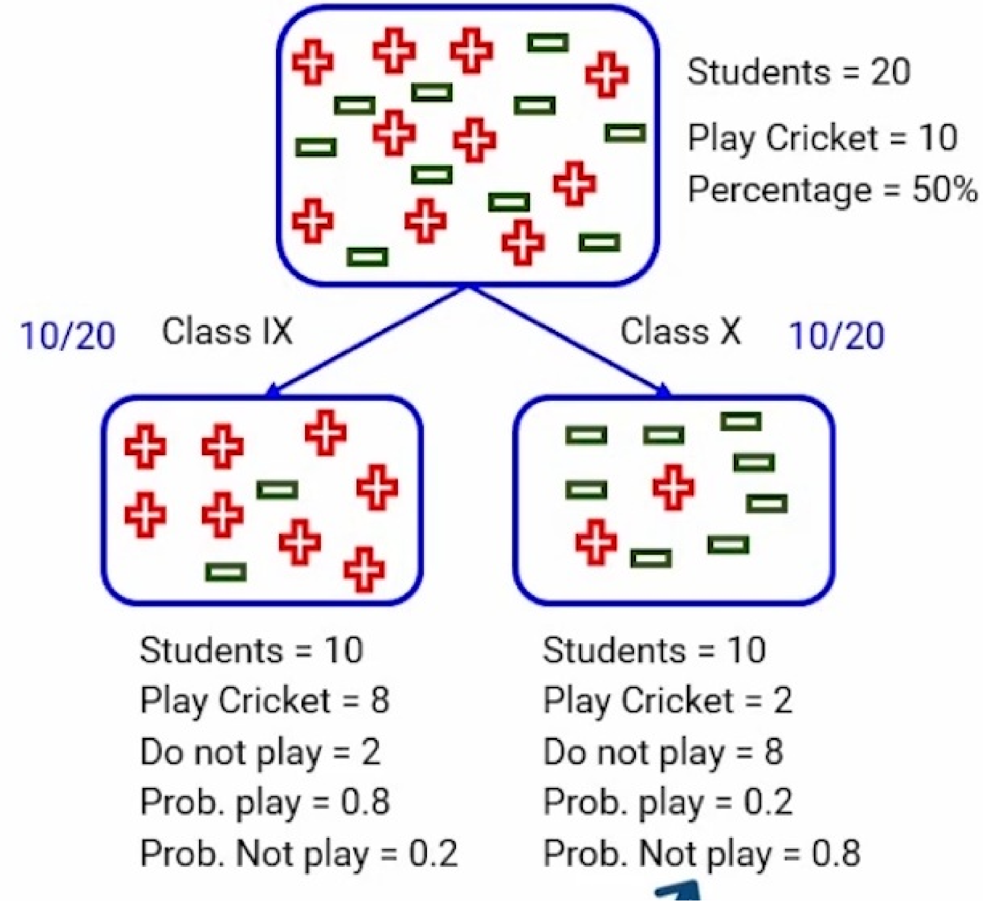
- Weighted Gini Impurity: Performance in Class:
(14/20)*0.49 + (6/20)*0.44 = 0.475



calculation for spliting based on class

Split on Class

- Gini Impurity: sub-node Class IX:
  1 - [(0.8)*(0.8) + (0.2)*(0.2)] = 0.32
- Gini Impurity: sub-node Class X:
  1 - [(0.2)*(0.2) + (0.8)*(0.8)] = 0.32
- Weighted Gini Impurity: Class:
  (10/20)*0.32 + (10/20)*0.32 = 0.32



Split	Weighted Gini Impurity
Performance in Class	0.475
Class	0.32

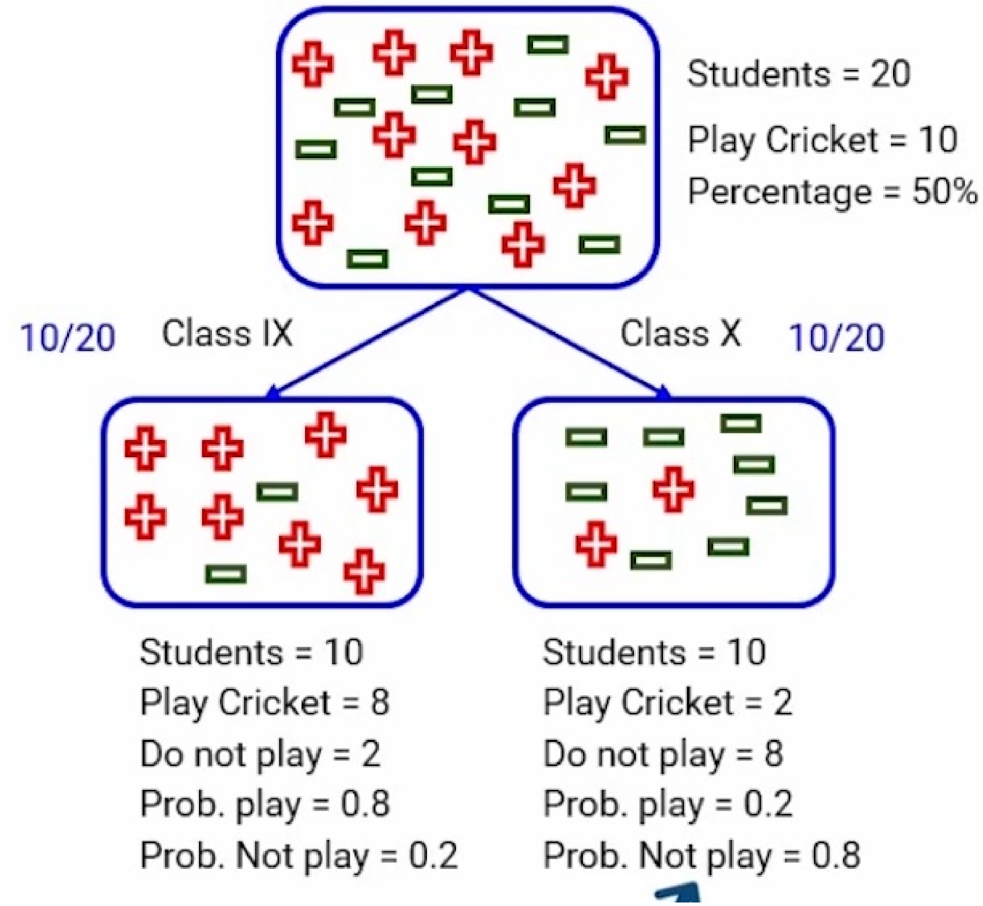
Lower the impurity-- hogher the homogeneous node

Hence, class will be the first split on this tree

calculation for spliting based on class

Split on Class

- Gini Impurity: sub-node Class IX:
  1 - [(0.8)*(0.8) + (0.2)*(0.2)] = 0.32
- Gini Impurity: sub-node Class X:
  1 - [(0.2)*(0.2) + (0.8)*(0.8)] = 0.32
- Weighted Gini Impurity: Class:
  (10/20)*0.32 + (10/20)*0.32 = 0.32



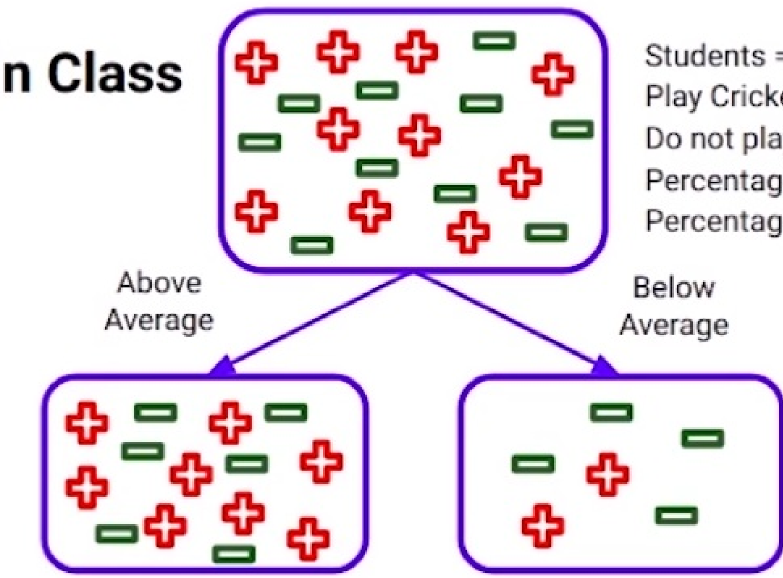
Split	Weighted Gini Impurity
Performance in Class	0.475
Class	0.32

Lower the impurity-- hogher the homogeneous node

Hence, class will be the first split on this tree

How to calculate Chi-square

Split on Performance in Class



Students = 20
Play Cricket = 10
Do not play = 10
Percentage play = 50%
Percentage not play = 50%

Students = 14

Expected Play = 7
Play Cricket = 8

Expected Not Play = 7
Not Play Cricket = 6

Students = 6

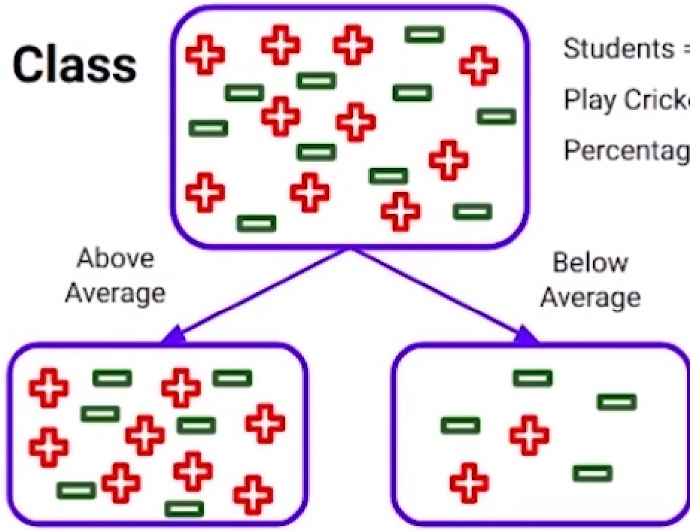
Expected Play = 3
Play Cricket = 2

Expected Not Play = 3
Not Play Cricket = 4

$$\text{Chi-Square} = \sqrt{[(\text{Actual} - \text{Expected})^2 / \text{Expected}]}$$

- **Higher the value of Chi-Square -- we are in the direction to purifie the node**
- **Higher the value of Chi-Square -- More will be the purity of nodes after splitting**

Split on Performance in Class



Students = 20

Play Cricket = 10

Percentage = 50%

Students = 14

Expected Play = 7
Play Cricket = 8

Expected Not Play = 7
Not Play Cricket = 6

Students = 6

Expected Play = 3
Play Cricket = 2

Expected Not Play = 3
Not Play Cricket = 4

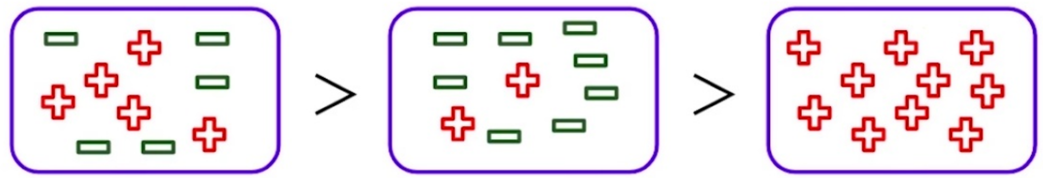
Node	Actual Play	Actual Not Play	Expected Play	Expected Not Play	Deviation Play	Deviation Not Play	Chi-Square (Play)	Chi-Square (Not Play)
Above Average	8	6	7	7	1	-1	0.38	0.38
Below Average	2	4	3	3	-1	1	0.58	0.58

Chi- Sqaure = 0.38 + 0.38 + 0.58 + 0.58
= 1.92

This was just for one variable, similarly cal. for other and than see which one has higher one

Split	Chi-Square
Performance in Class	1.92
Class	5.36

Information Gain



Node 1 Node 2 Node 3

→ More impure node will require more information to describe nodes

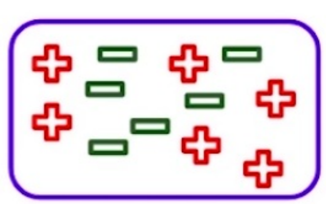
→ Higher information gain leads to pure nodes

$$\boxed{\text{Information Gain} = 1 - \text{Entropy}}$$

Entropy

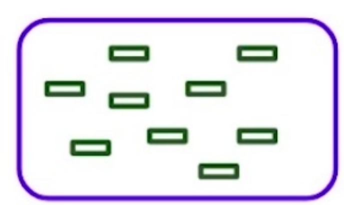
$$- p_1{*}\log_2 p_1 - p_2{*}\log_2 p_2 - p_3{*}\log_2 p_3 - \ldots\ldots - p_n{*}\log_2 p_n$$

(p = %of each class in the node)



% Play = 0.50
% Not play = 0.50

Entropy = 1



% Play = 0
% Not play = 1

Entropy = 0

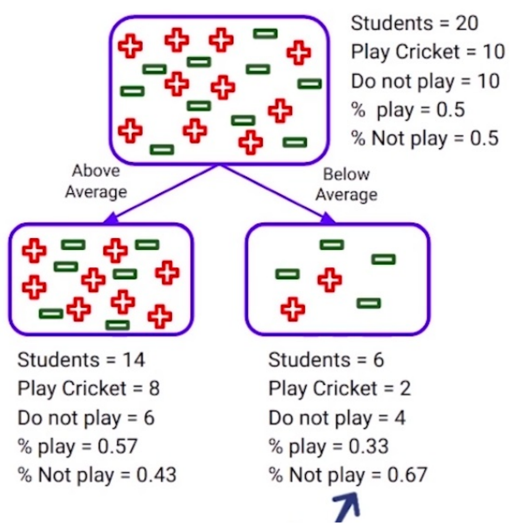
How to calculate entropy and split by it

- Calculate the entropy of the parent node
- Calculate the entropy of each child node
- Calculate the weighted average entropy of the split

- for splitting node further
child node's Entropy must be less then parent node's entropy

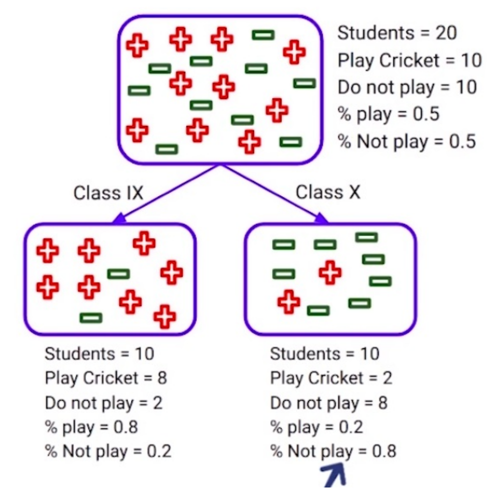
Split on Performance in Class

- Entropy for Parent node:
$-(0.5)*\log_2(0.5) -(0.5)*\log_2(0.5) = 1$
- Entropy for sub-node Above Average:
$-(0.57)*\log_2(0.57) -(0.43)*\log_2(0.43) = 0.98$
- Entropy for sub-node Below Average:
$-(0.33)*\log_2(0.33) -(0.67)*\log_2(0.67) = 0.91$
- Weighted Entropy: Performance in Class:
$(14/20)*0.98 + (6/20)*0.91 = 0.959$



Split on Class

- Entropy for Parent node:
$-(0.5)*\log_2(0,5) -(0.5)*\log_2(0.5) = 1$
- Entropy for sub-node Class IX:
$-(0.8)*\log_2(0.8) -(0.2)*\log_2(0.2) = 0.722$
- Entropy for sub-node Class X:
$-(0.2)*\log_2(0.2) -(0.8)*\log_2(0.8) = 0.722$
- Weighted Entropy: Class:
$(10/20)*0.722 + (10/20)*0.722 = 0.722$

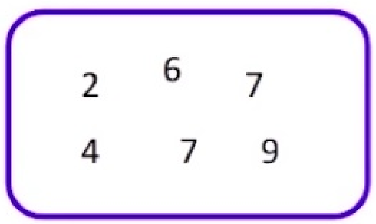


Split	Entropy	Information Gain
Performance in Class	0.959	0.041
Class	0.722	0.278

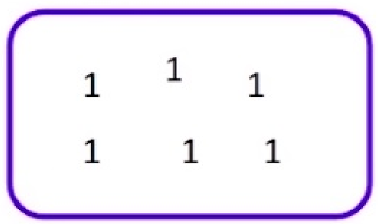
If we want to use Decision tree for contineous values then we should use belowed algo for splitting the nodes.

Reduction in variance

$$\text{Variance} = \Sigma\,[(X - \mu)^2]\,/\,n$$



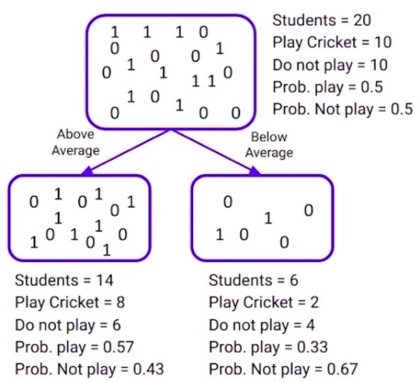
Variance ~ 6



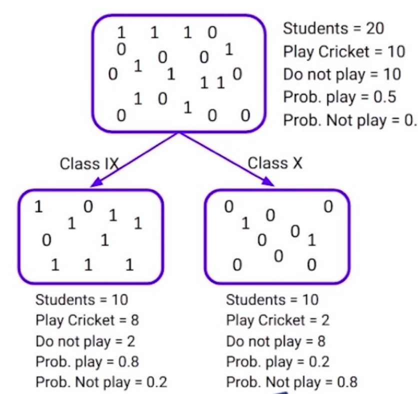
Variance = 0

Lower value of variance - higher the purity of nodes

- Above Average node:
  - Mean = (8*1 + 6*0) / 14 = 0.57
  - Variance = $[8*(1-0.57)^2 + 6*(0-0.57)^2]$ / 14 = 0.245
- Below Average node:
  - Mean = (2*1 + 4*0) / 6 = 0.33
  - Variance = $[2*(1-0.33)^2 + 4*(0-0.33)^2]$ / 6 = 0.222
- Variance: Performance in Class: (14/20)*0.245 + (6/20)*0.222 = 0.238



- Class IX node:
  - Mean = (8*1 + 2*0) / 10 = 0.8
  - Variance = $[8*(1-0.8)^2 + 2*(0-0.8)^2]$ / 10 = 0.16
- Class X node:
  - Mean = (2*1 + 8*0) / 10 = 0.2
  - Variance = $[2*(1-0.2)^2 + 8*(0-0.2)^2]$ / 10 = 0.16
- Variance: Class: (10/20)*0.16 + (10/20)*0.16 = 0.16

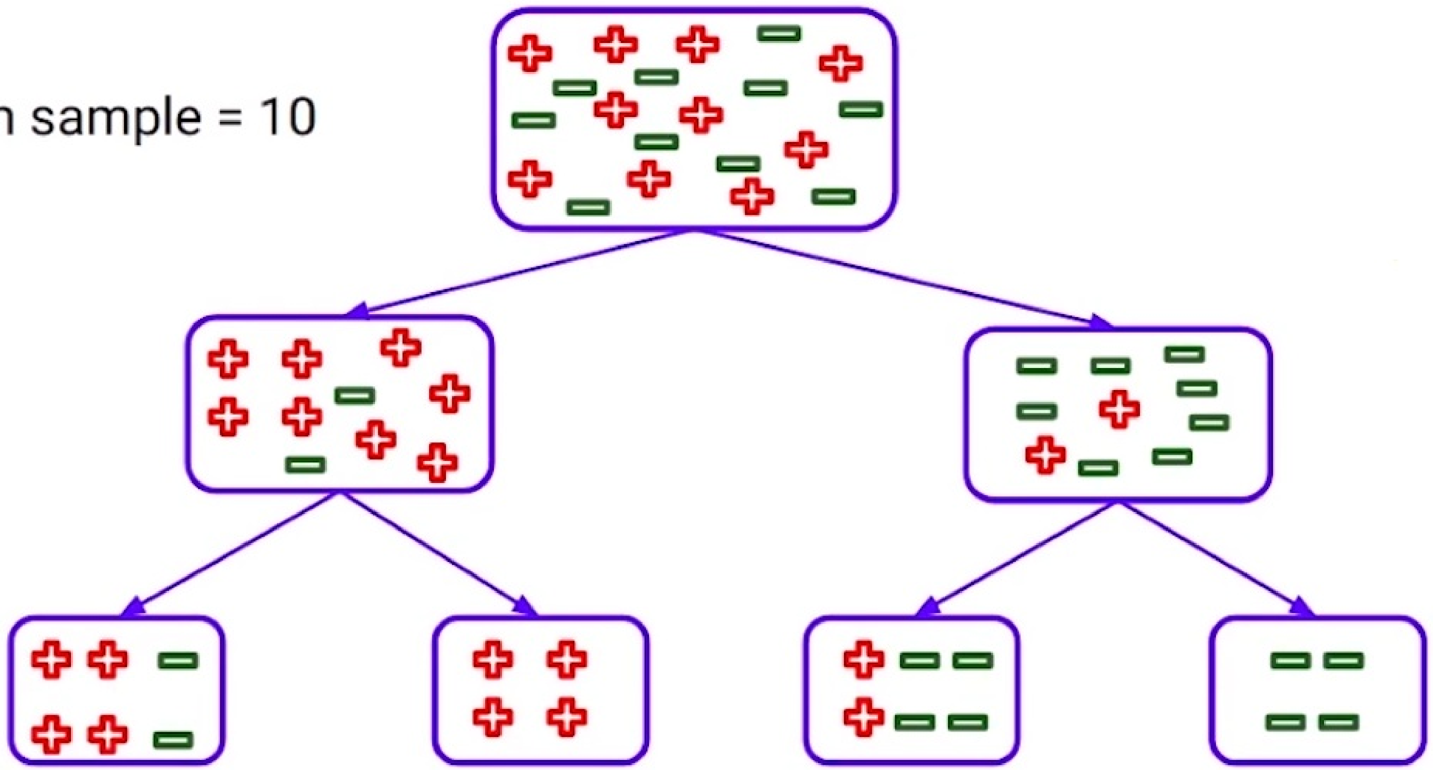


Split	Variance
Performance in Class	0.238
Class	0.16

Optimizing performance of a decision tree

1. Minimum samples for a node split
   a. Higher values controls overfitting
   b. Too high values can lead to underfitting

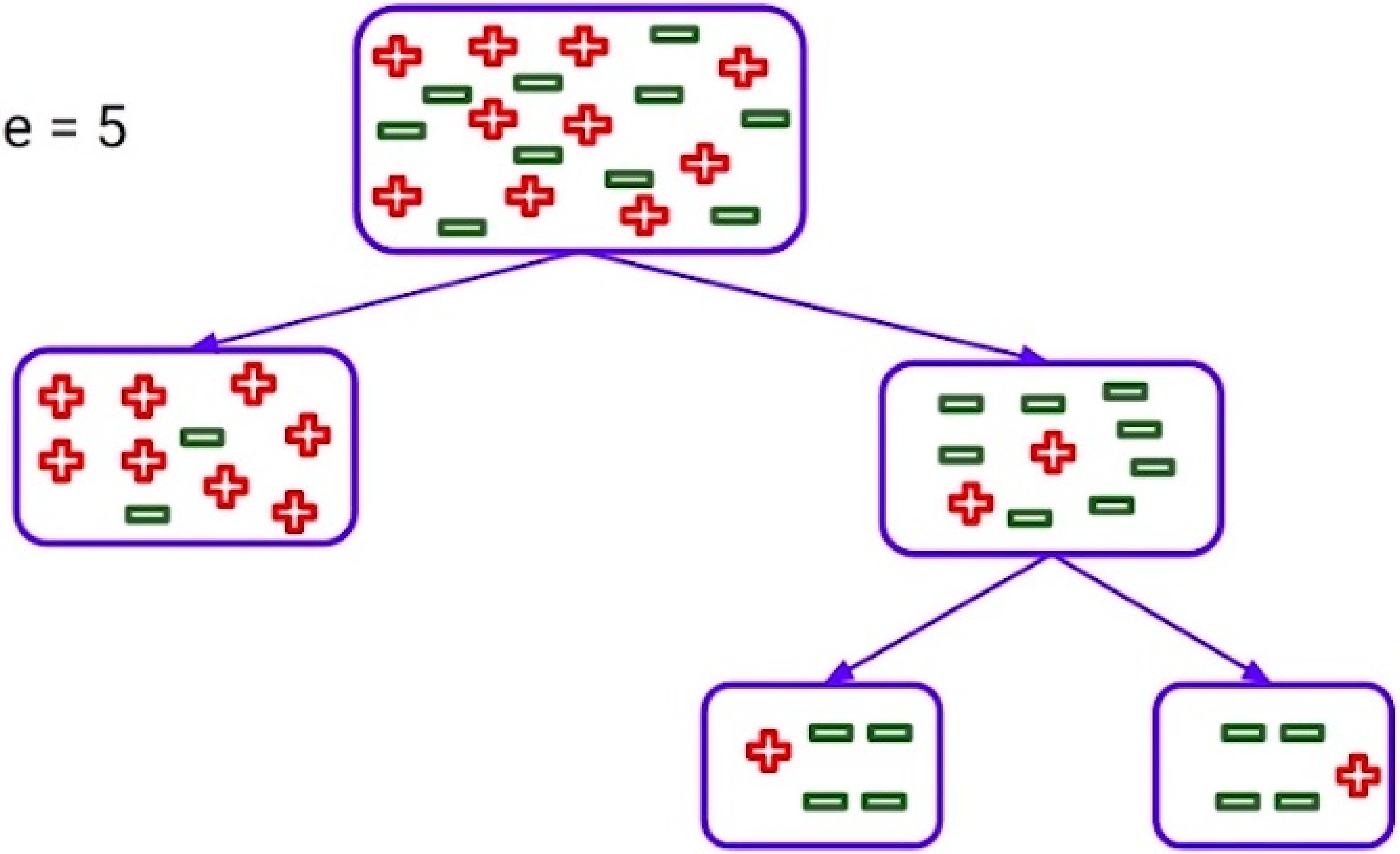
Minimum sample = 10



2. Minimum samples for a terminal node

   a. Higher value controls overfitting

Minimum sample = 5

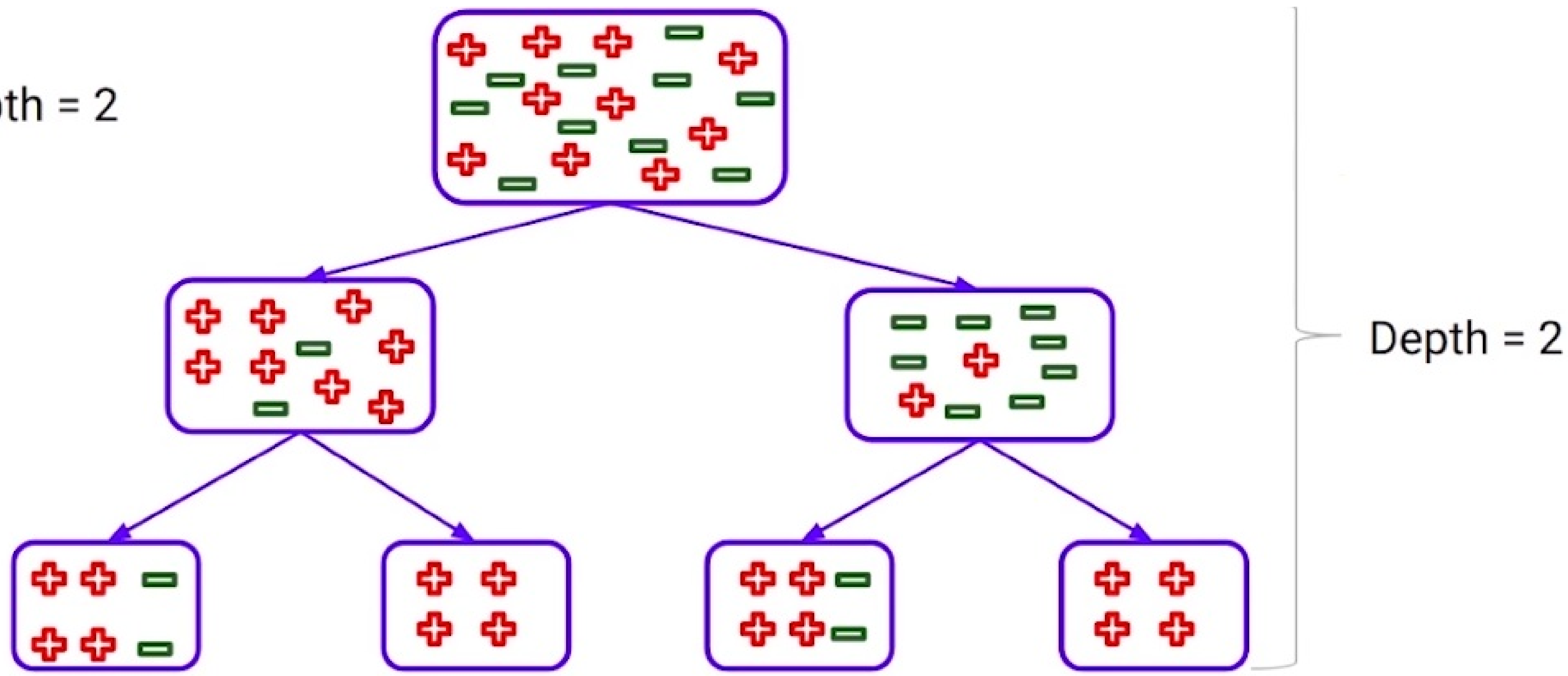


3. Maximum depth of tree

   a. Higher depth can lead to overfitting

   b. Lower depth can lead to underfitting

Max Depth = 2



4. Maximum number of terminal nodes

Max Terminal nodes = 2