Classification Trees

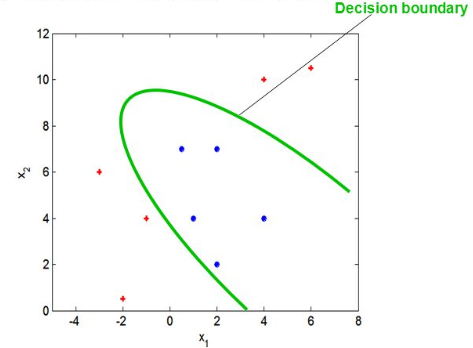
Son Nguyen

10/14/2022

## Reading Materials

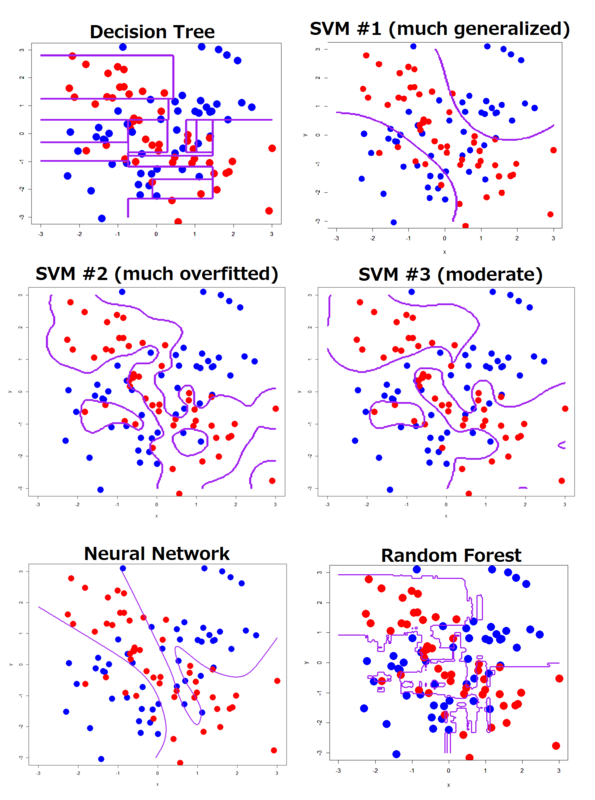
* Max Kuhn. Chapter 14. Section 14.1

## Decision Boundary in Classification



Classification is a process of finding the **decision boundary** that best separate two classes

## Decision Boundary in Classification

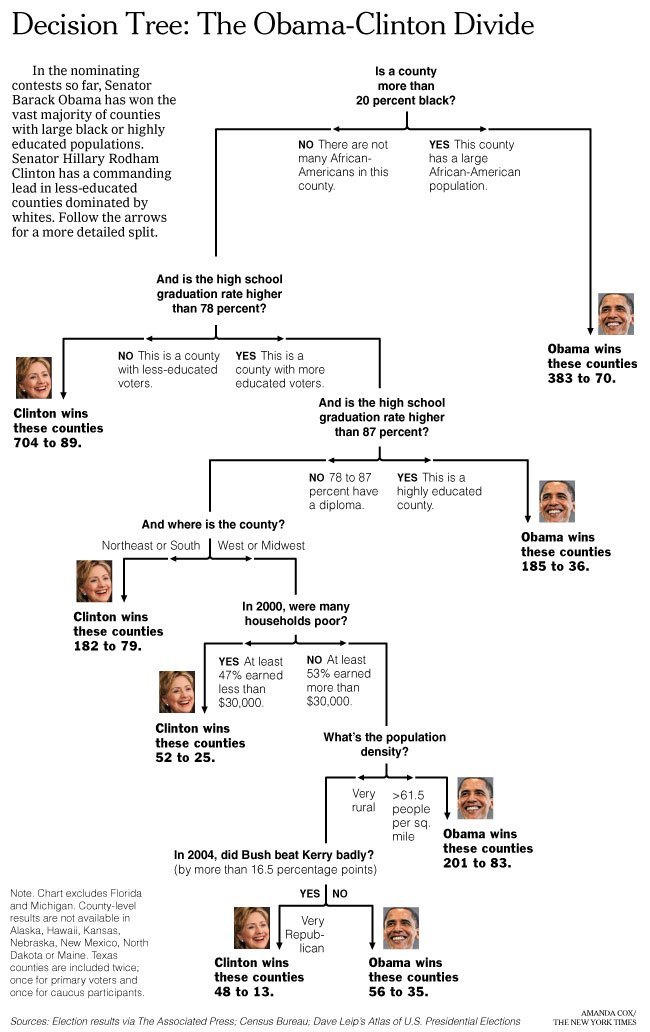


SVM = Support Vector Machine

## Decision Tree

* Decision Tree for classification is **Classification Tree**
* Decision Tree for Regression is **Regression Tree**

## Example of Classification Tree

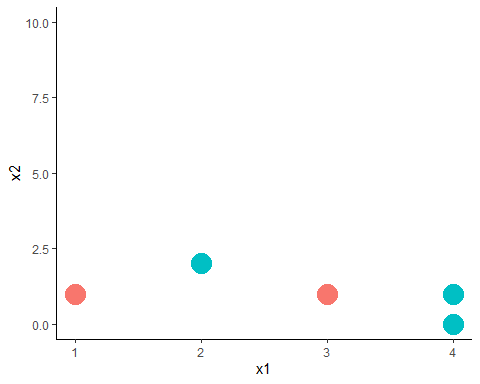


[Link] (<http://graphics8.nytimes.com/images/2008/04/16/us/0416-nat-subOBAMA.jpg>)

## Classification Tree

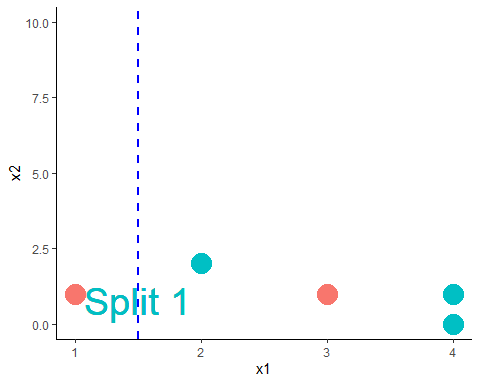
* In two dimension, classification Tree’s decision boundary is a collection of horiontal and vertical line

## Data

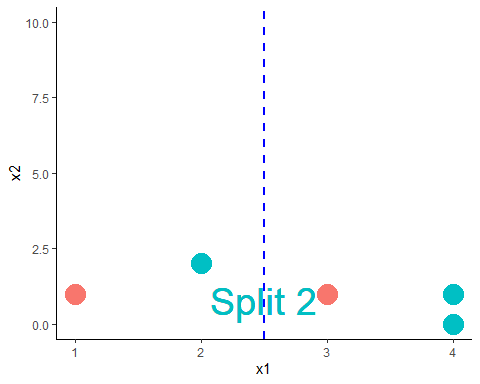


* The tree starts by a vertical or horizontal line that **best** seperate the data
* **Question**: Find a vertical line that best seperate **red** and **green**.

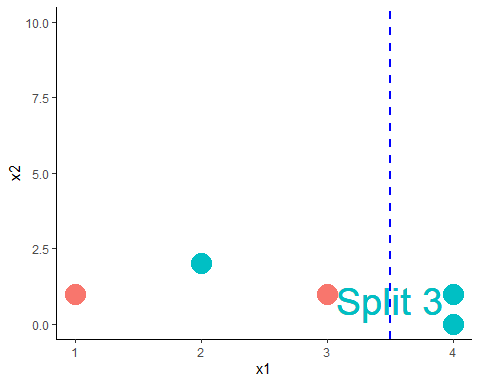
## One way to seperate the reds and greens



## One way to seperate the reds and greens



## One way to seperate the reds and greens



## Question

* **Question**: Which is the best split?

## Partial Answer

* It looks like Split 1 and 3 are better than Split 2 since it misclassifies less
* Which is the better split between Split 1 and Split 3?
* We need to find a way to measure *how good a split is*

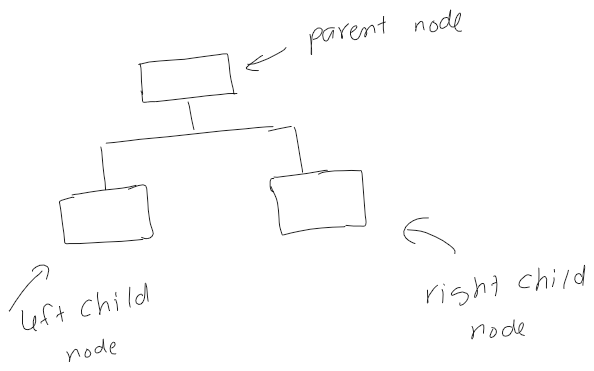
## Impurity Measure

* The impurity of a node (**a node = a subset of the data or the original data**) measure how uncertain the node is.
* For example, node A with 50% reds and 50% greens would be more uncertained than node B with 90% reds and 10% greens. Thus, node A has greater impurity than node B.
* More uncertained Greater impurity

## Impurity Measure

* A split that *gains* more impurity is the **better split**!

## Impurity Gain



* IG is Impurity Gain of the split
* and are the number of points in the left child node and right child node, respectively.

## Impurity Measure

* Impurity can be measured by: classification error, Gini Index, and Entropy.

## Impurity Measure

* Let and be the proportion of class 0 and class 1 in a node.

$$

$$

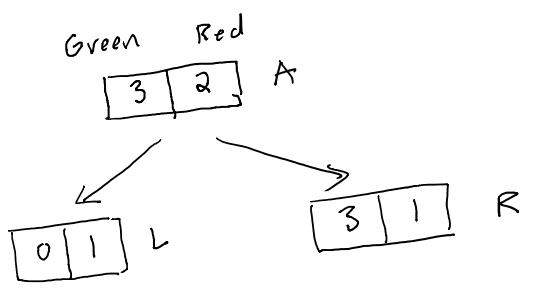
## Calculation

* Let’s calculate the impurity gain of the three splits to decide which split is the best

## IG By Classification Error

* Let **green** and **red** be class 0 and class 1, respectively.

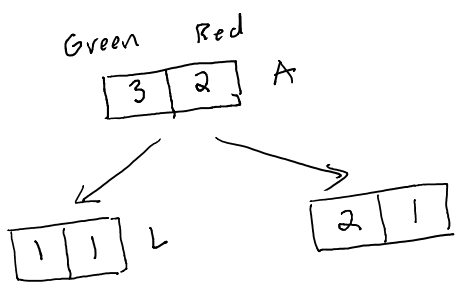
For Split 1:



* Node *parent,* A: . Thus,
* Node *child left,* L: . Thus,
* Node *child right,* R: . Thus,
* Impurity Gain of Split 1:

## IG By Classification Error

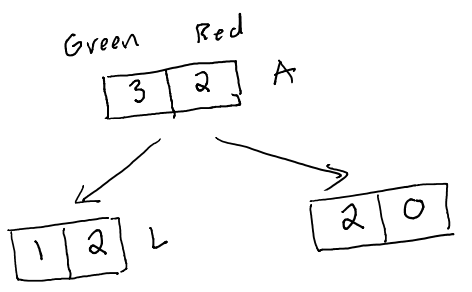
For Split 2:



* Node *parent,* A: . Thus,
* Node *child left,* L: . Thus,
* Node *child right,* R: . Thus,
* Impurity Gain of Split 2:

## IG By Classification Error

For Split 3:



* Node *parent,* A: . Thus,
* Node *child left,* L: . Thus,
* Node *child right,* R: . Thus,
* Impurity Gain of Split 3:

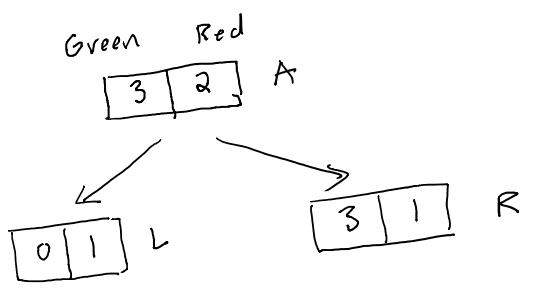
## Comparing IG By Classification Error

|  | IG |
| --- | --- |
| Split 1 | 0.2 |
| Split 2 | 0 |
| Split 3 | 0.2 |

* By classification error, Split 1 and Split 3 are tie as the best because they have the same impurity gain.

## IG By Gini Index

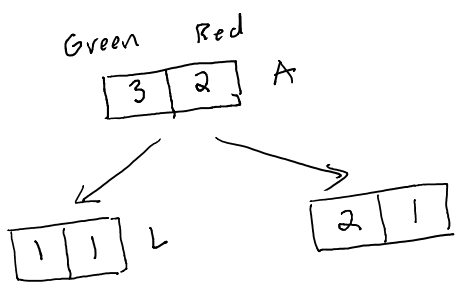
For Split 1:



* Node *parent,* A: . Thus,
* Node *child left,* L: . Thus,
* Node *child right,* R: . Thus,
* Impurity Gain of Split 1:

## IG By Gini Index

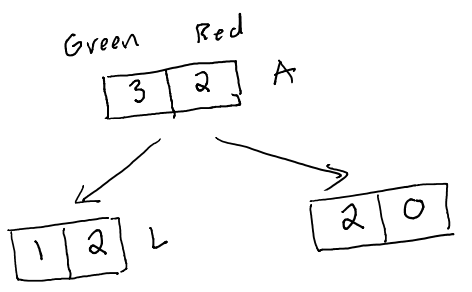
For Split 2:



* Node *parent,* A: . Thus,
* Node *child left,* L: . Thus,
* Node *child right,* R: . Thus,
* Impurity Gain of Split 2:

## IG By Gini Index

For Split 3:



* Node *parent,* A:
* Node *child left,* L: . Thus,
* Node *child right,* R: . Thus,
* Impurity Gain of Split 3:

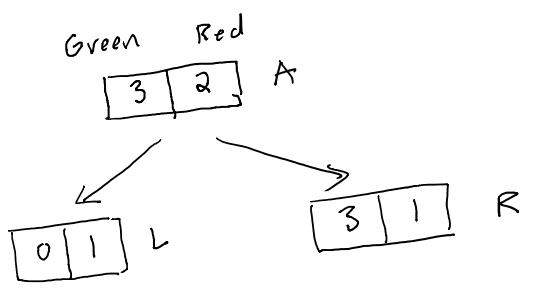
## Comparing IG By Gini Index

|  | IG |
| --- | --- |
| Split 1 | 0.18 |
| Split 2 | 0.016 |
| Split 3 | 0.216 |

* By Gini Index, Split 3 is the best because it has the greatest impurity gain.

## IG By Entropy

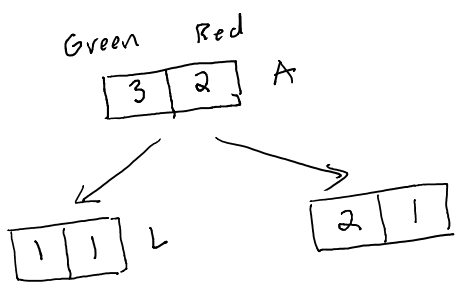
For Split 1:



* Node *parent,* A: . Thus,
* Node *child left,* L: . Thus,
* Node *child right,* R: . Thus,
* Impurity Gain of Split 1:

## IG By Entropy

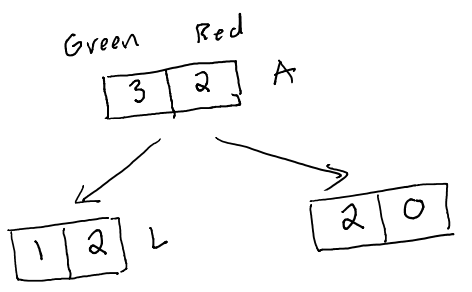
For Split 2:



* Node *parent,* A: . Thus,
* Node *child left,* L: . Thus,
* Node *child right,* R: . Thus,
* Impurity Gain of Split 2:

## IG By Entropy

For Split 3:



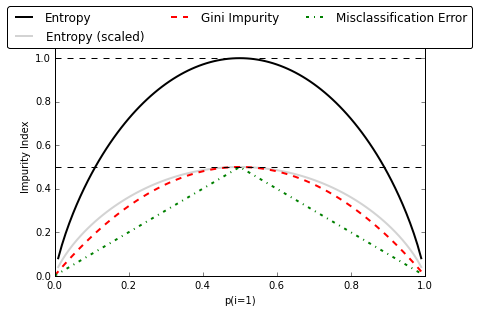
* Node *parent,* A:
* Node *child left,* L: . Thus,
* Node *child right,* R: . Thus,
* Impurity Gain of Split 3:

## Comparing IG By Entropy

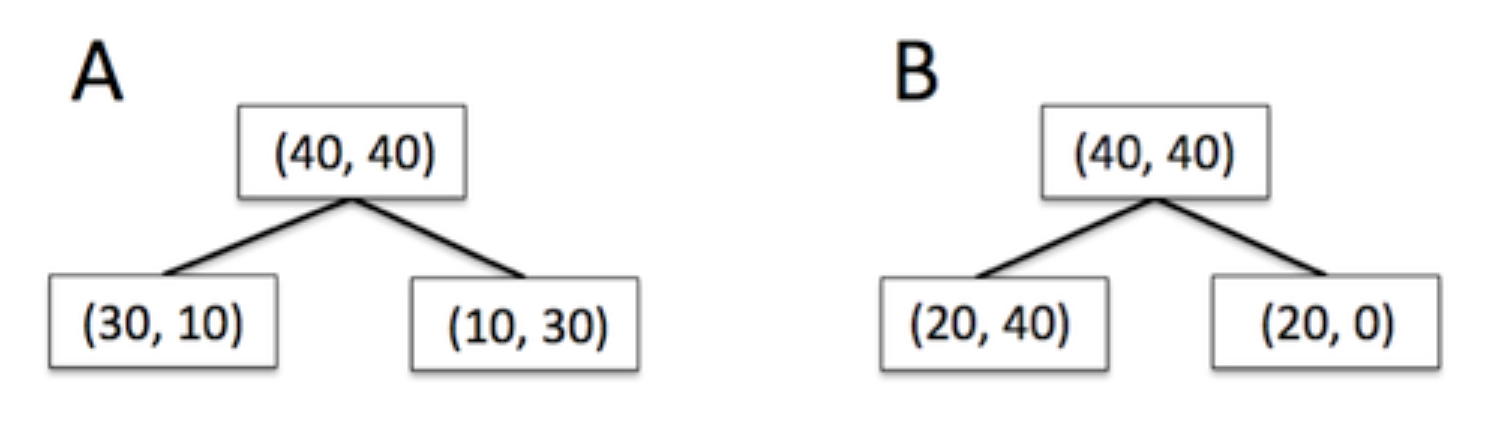
|  | IG |
| --- | --- |
| Split 1 | 0.322 |
| Split 2 | 0.02 |
| Split 3 | 0.42 |

* By Gini Index, Split 3 is the best because it has the greatest impurity gain.

## Comparing Impurity Measures

 - Relation between impurity and the class probabilities. All impurity measures are maximized at and minimized at and .

## Another Example

 - Which split is better?

## Decide the best split using Chi-Square test of Independence

* Besides impurity gain, one can use the Chi-square, , test of independence to decide the best split.

## Review of Chi-Square test of Independence

* Let and be two categorical variables.
* We want to test if and are independent/associated
  + : and are independent
  + and are dependent
* Test statistic:

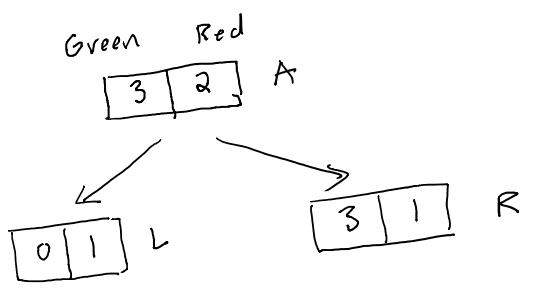
## Review of Chi-Square test of Independence

* In our context, the greater the value, the smaller the
* The smaller the , the more dependent the two variables are. Thus the better the split is.
* Therefore, we look for the split with the **greatest value.**

## Applying to Our Example

* We will calculate the values of the three splits.
* The best split is the split with the greatest value.

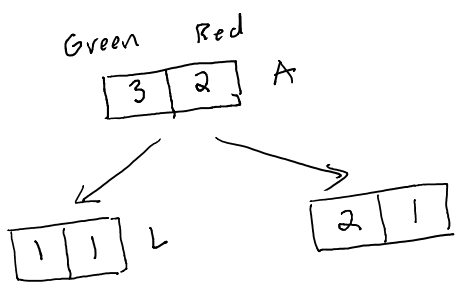
## Split 1



|  | Greens | Reds |  |
| --- | --- | --- | --- |
| Left Branch | 0 (Cell 1) | 1 (Cell 2) | 1 |
| Right Branch | 3 (Cell 3) | 1 (Cell 4) | 4 |
|  | 3 | 2 |  |

* (Cell 1): ,
* (Cell 2): ,
* (Cell 3): ,
* (Cell 4): ,
* Plug in, we have:

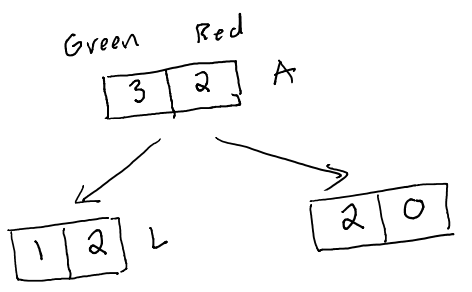
## Split 2



|  | Greens | Reds |  |
| --- | --- | --- | --- |
| Left Branch | 1 (Cell 1) | 1 (Cell 2) | 2 |
| Right Branch | 2 (Cell 3) | 1 (Cell 4) | 3 |
|  | 3 | 2 |  |

* (Cell 1): ,
* (Cell 2): ,
* (Cell 3): ,
* (Cell 4): ,
* Plug in, we have:

## Split 3



|  | Greens | Reds |  |
| --- | --- | --- | --- |
| Left Branch | 1 (Cell 1) | 2 (Cell 2) | 3 |
| Right Branch | 2 (Cell 3) | 0 (Cell 4) | 2 |
|  | 3 | 2 |  |

* (Cell 1): ,
* (Cell 2): ,
* (Cell 3): ,
* (Cell 4): ,
* Plug in, we have:

## Comparing the three splits

|  |  |
| --- | --- |
| Split 1 | 1.875 |
| Split 2 | 0.139 |
| Split 3 | 2.222 |

* Split 3 is the best because it has the greatest !

## Logworth

* The quality of the split can be measured by **Logworth**
* Formula:
* The greater the logworth, the better the split

## Logworth

|  |  | p-value | logworth |
| --- | --- | --- | --- |
| Split 1 | 1.875 | 0.114 | 0.943 |
| Split 2 | 0.139 | 0.998 | 0.0008 |
| Split 3 | 2.222 | 0.088 | 1.055 |

* Greatest = Lowest = Greatest logworth = Best Split
* Split 3 is the best split!

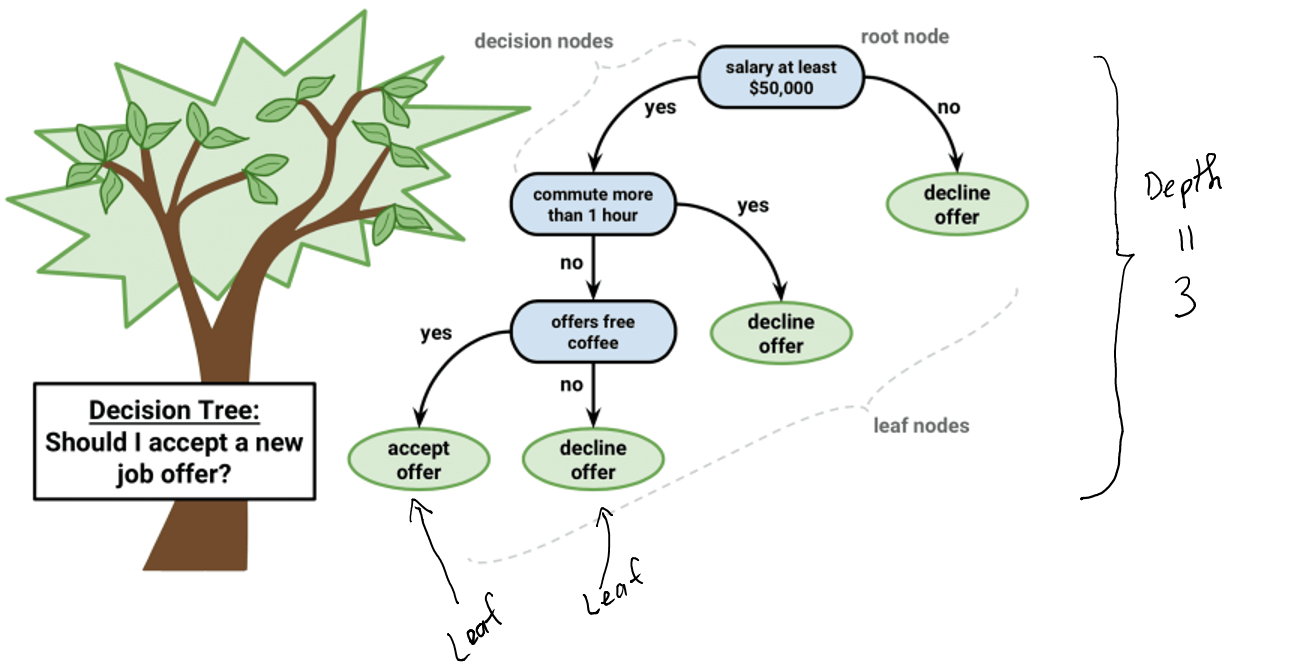
## What happens after the first split?

* After the first split, the data are divided into to subsets.
* The splitting process is repeated for each subset.
* The process ends when a stopping criteria is satisfied

## Stopping Criteria

* Minimum Leaf Size: The minimum of observations in the leaves
* Maximum Number of Leaves
* Maximum Depth
* Others

## Stopping Criteria



## Decision Tree Algorithm - How to grow a tree

* Step 1: Calculate the impurity gain or of all possible splits at all variables
* Step 2: Select the split that give the maximum impurity gain or lowest to split the data into two subdata and
* Repeat *Step 1* and *Step 2* to both and .
* Until a stopping criteria is satisfied

## Complexity of Decision Tree

* A complexity of a tree can be measured by the number of leaves the tree has
* The more leaves a tree has, the more complex the tree is.
* A complex tree may be **overfitted**, i.e. having low training error but high testing error.

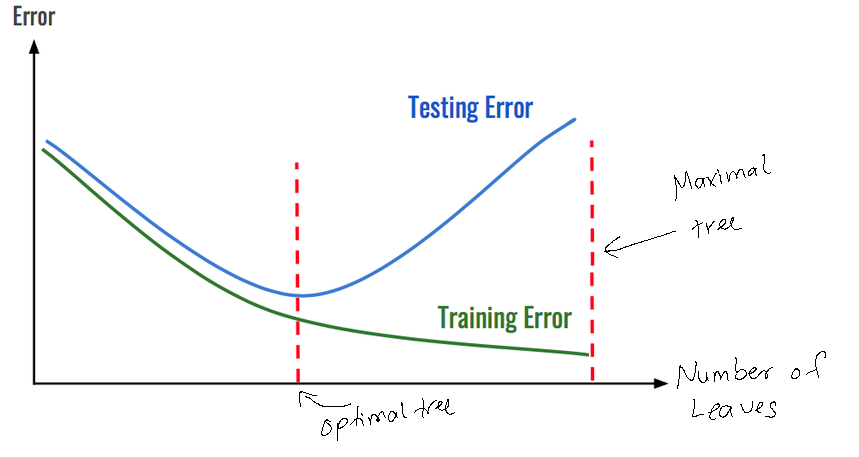
## Prunning a tree

* For any given data, one can construct a tree that achives 0 misclassification on training data
* After growing the tree one needs to prune it to avoid overfittted

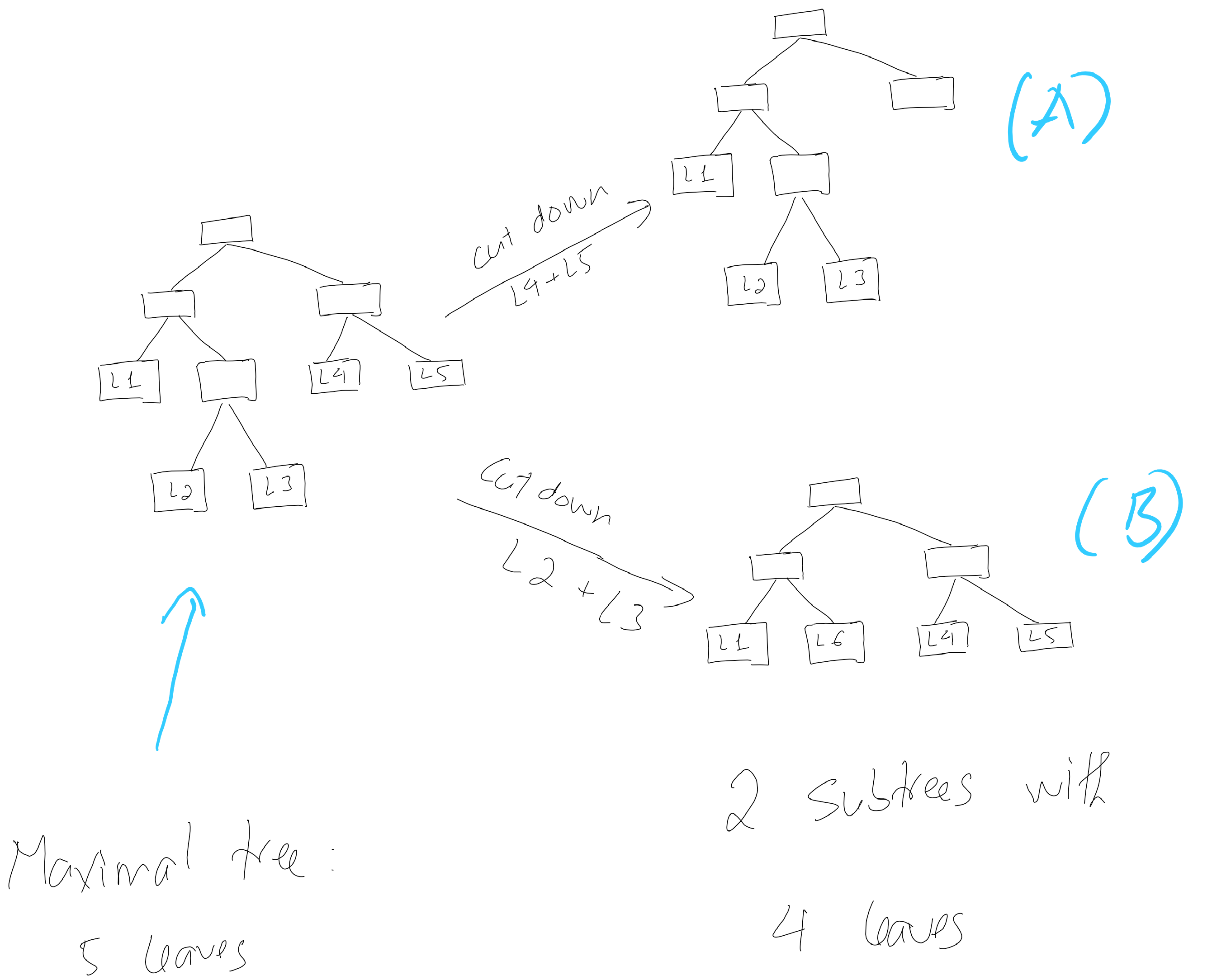
## Prunning a tree

* The tree with maximum number of leaves is called the **maximal tree** (still satisfied the stopping rule)
* From the **maximal tree**, leaves are cut down, one by one, to obatined all possible subtrees
* The subtree with lowest error on validation data, is the **optimal tree**

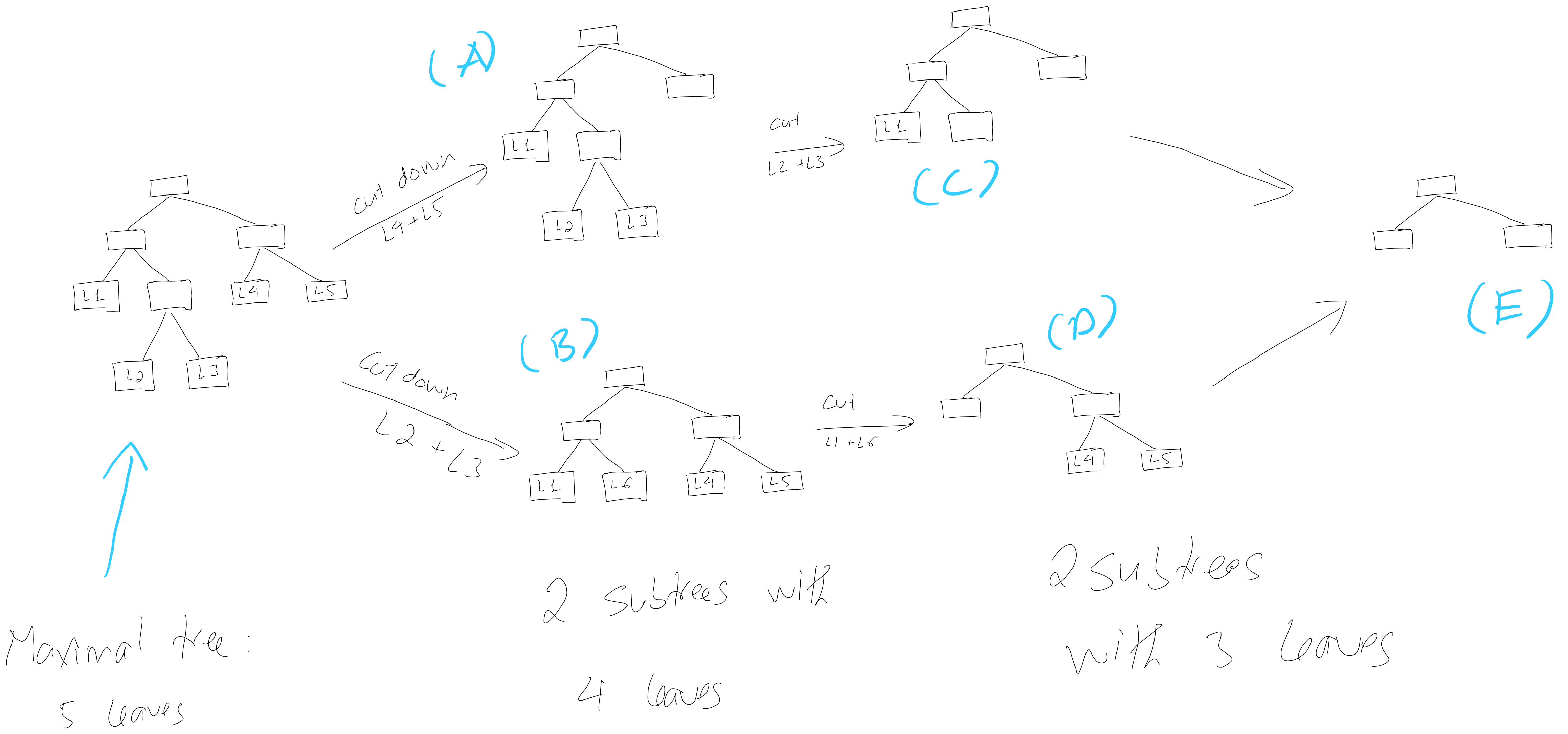
## Maximal vs Optimal Tree



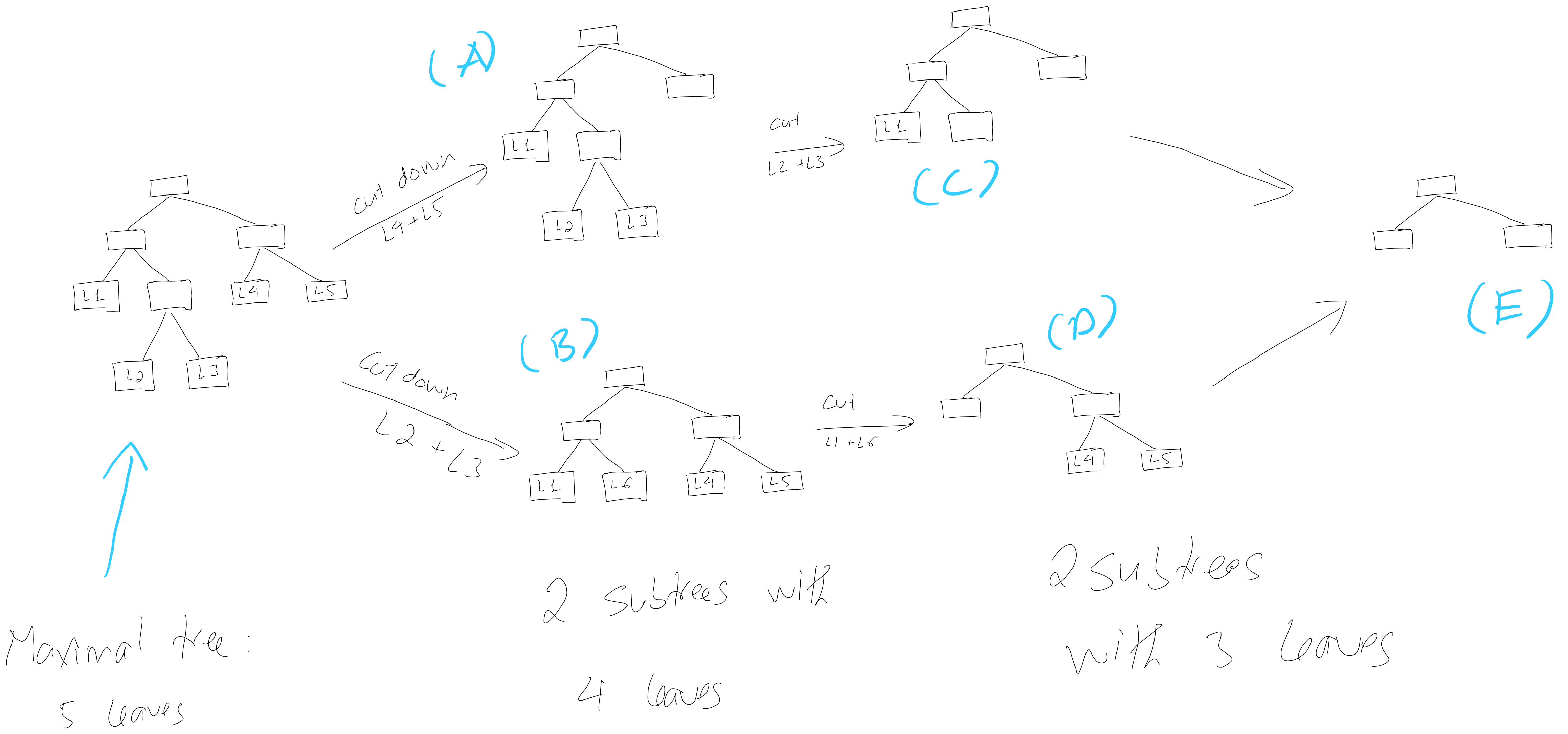
## Example of Tree Prunning



## Example of Tree Prunning

 - All the subtrees A, B, C, D, and E will be validated with the validation data to find the **optimal tree** - The **optimal tree** could be the **maximal tree**!

## Question

 - What if both B and C give the lowest error on the validation data? Which tree should be selected as the final model?