Decision Tree Introduction

- Decision tree builds classification or regression models in the form of a tree structure.
- It breaks down a dataset into smaller subsets with increase in depth of tree.
- The final result is a tree with **decision nodes** and **leaf nodes**

- A decision node (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy).
- Leaf node (e.g., Play) represents a classification or decision.

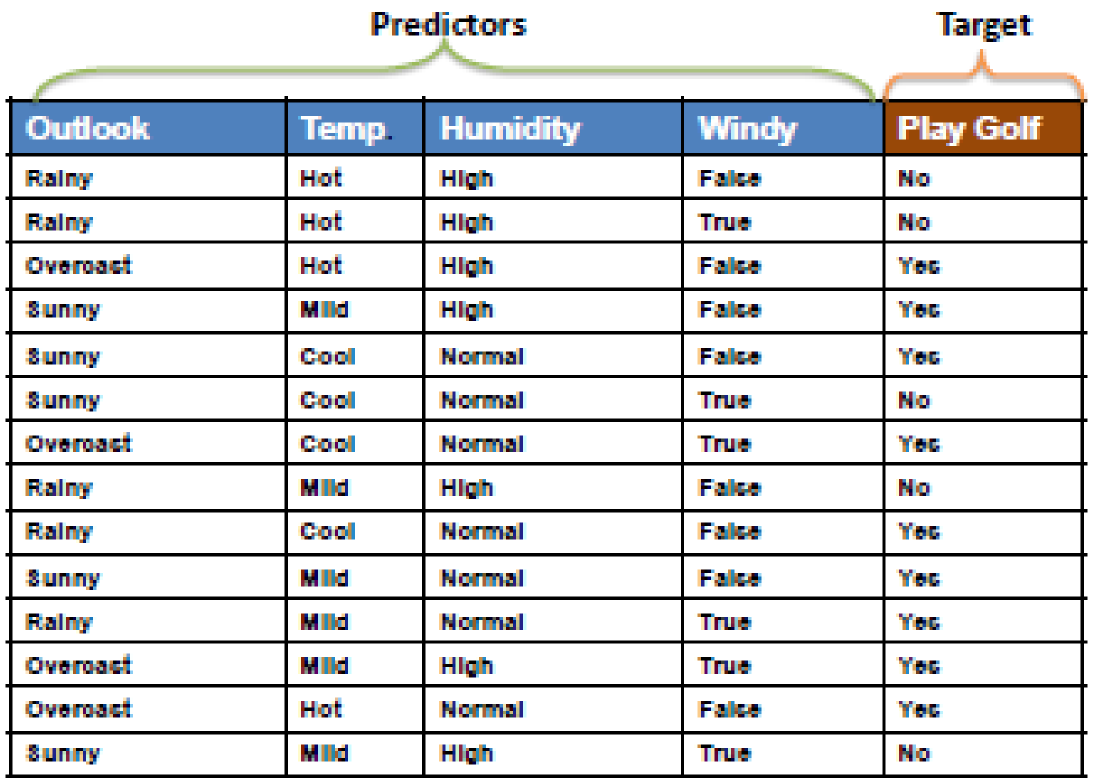
- The topmost decision node in a tree which corresponds to the best predictor is called **root node**.
- Decision trees can handle both categorical and numerical data.

Decision Tree - Types

Categorical Variable Decision Tree: Decision Tree which has categorical target variable then it called as categorical variable decision tree.

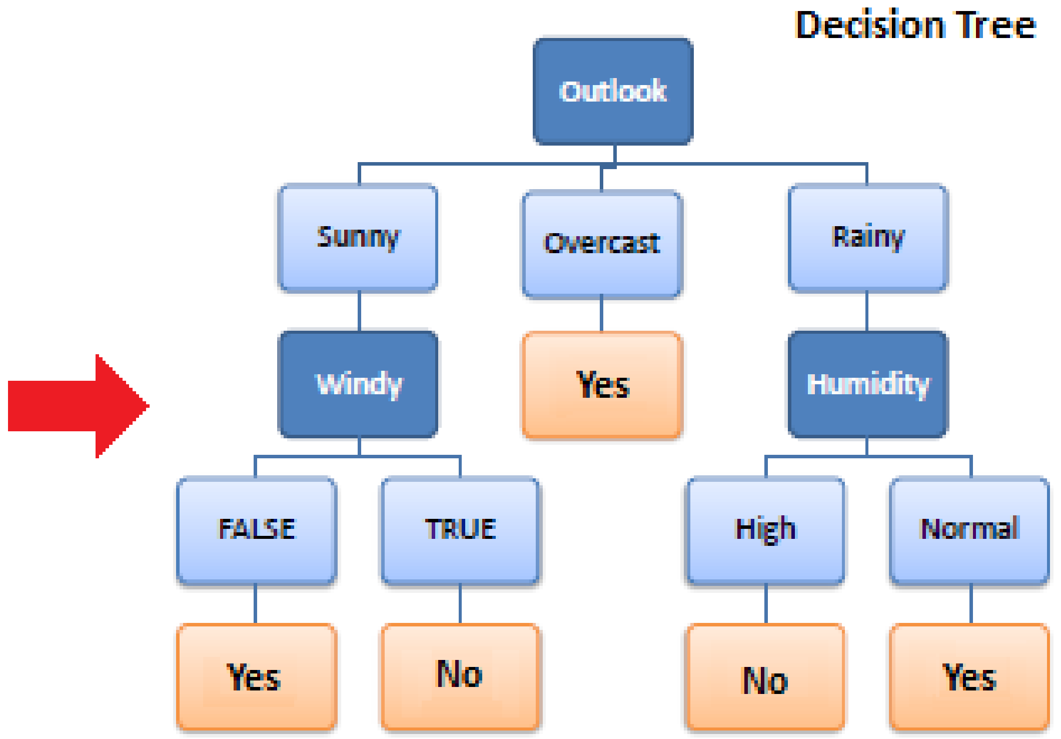
Continuous Variable Decision Tree: Decision Tree which has continuous target variable then it is called as Continuous Variable Decision Tree.

Decision Tree Introduction



Predictors | Target

Outlook	Temp.	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No



Decision Tree – Important Terminologies

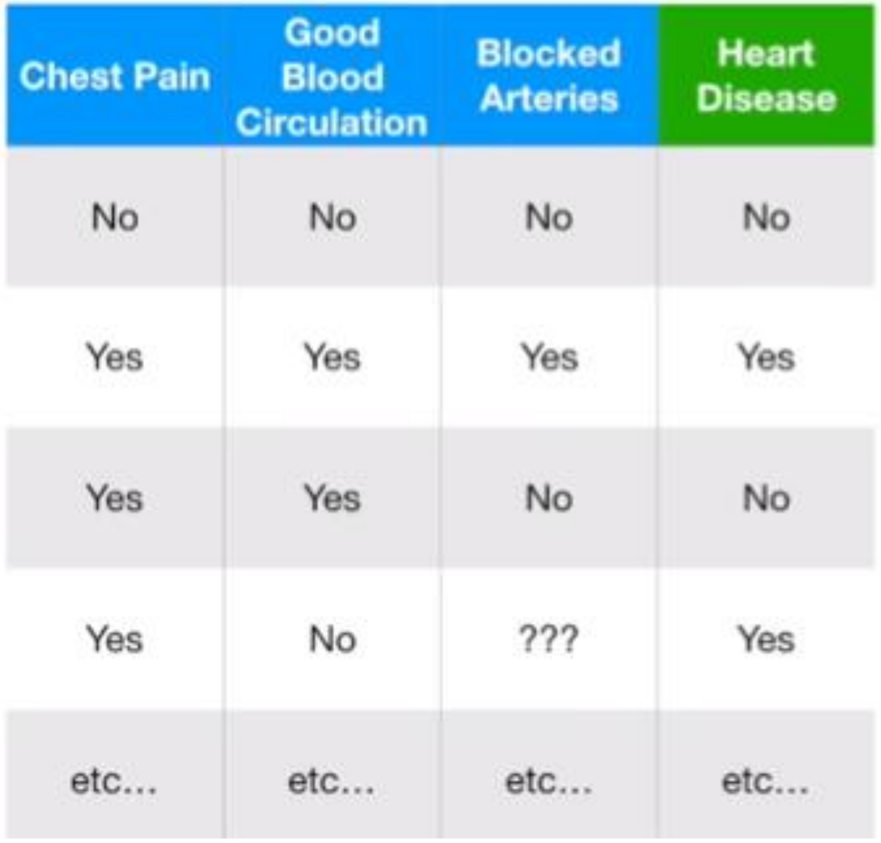
- **Root Node:** It represents entire population or sample and this further gets divided into two or more homogeneous sets.
- **Splitting:** It is a process of dividing a node into two or more sub-nodes.
- **Decision Node:** When a sub-node splits into further sub-nodes, then it is called decision node.
- **Leaf/ Terminal Node:** Nodes with no children (no further split) is called Leaf or Terminal node.
- **Pruning:** When we reduce the size of decision trees by removing nodes (opposite of Splitting), the process is called pruning.
- **Branch / Sub-Tree:** A sub section of decision tree is called branch or sub-tree.
- **Parent and Child Node:** A node, which is divided into sub-nodes is called parent node of sub-nodes where as sub-nodes are the child of parent node.

Decision Tree Introduction

Lets take an example of a patient's health data.
(Total 303 observations)

In this example we want to create a tree that uses chest pain, good blood circulation and blocked arteries status to predict whether the person has a heart disease or not.

The first thing we want to know is,
what should be our root node ?
Chest Pain, Good Blood Circulation or Blocked Arteries ?

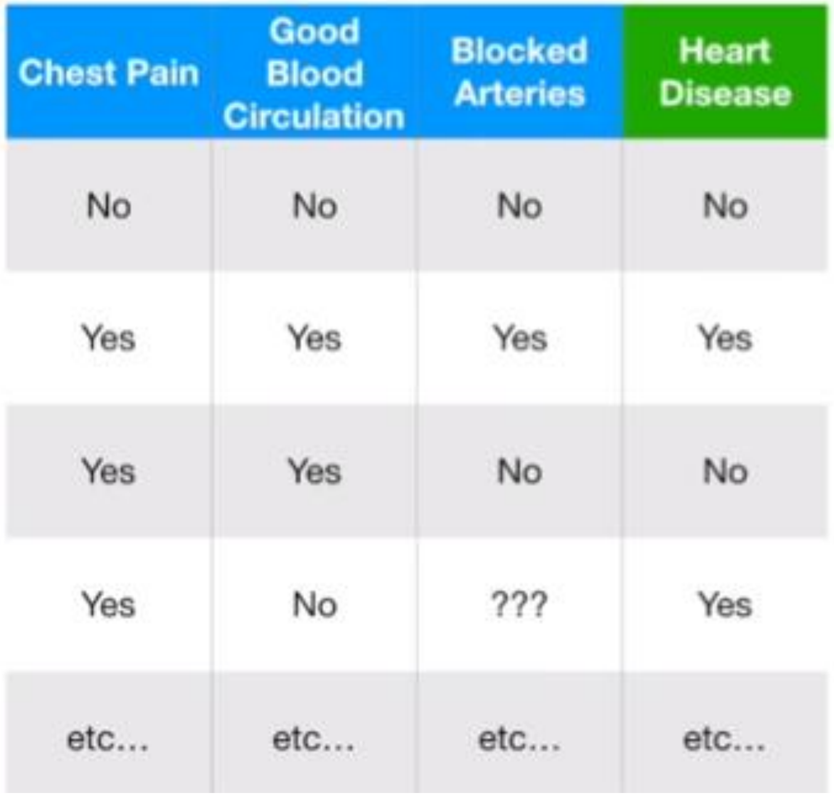


Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...

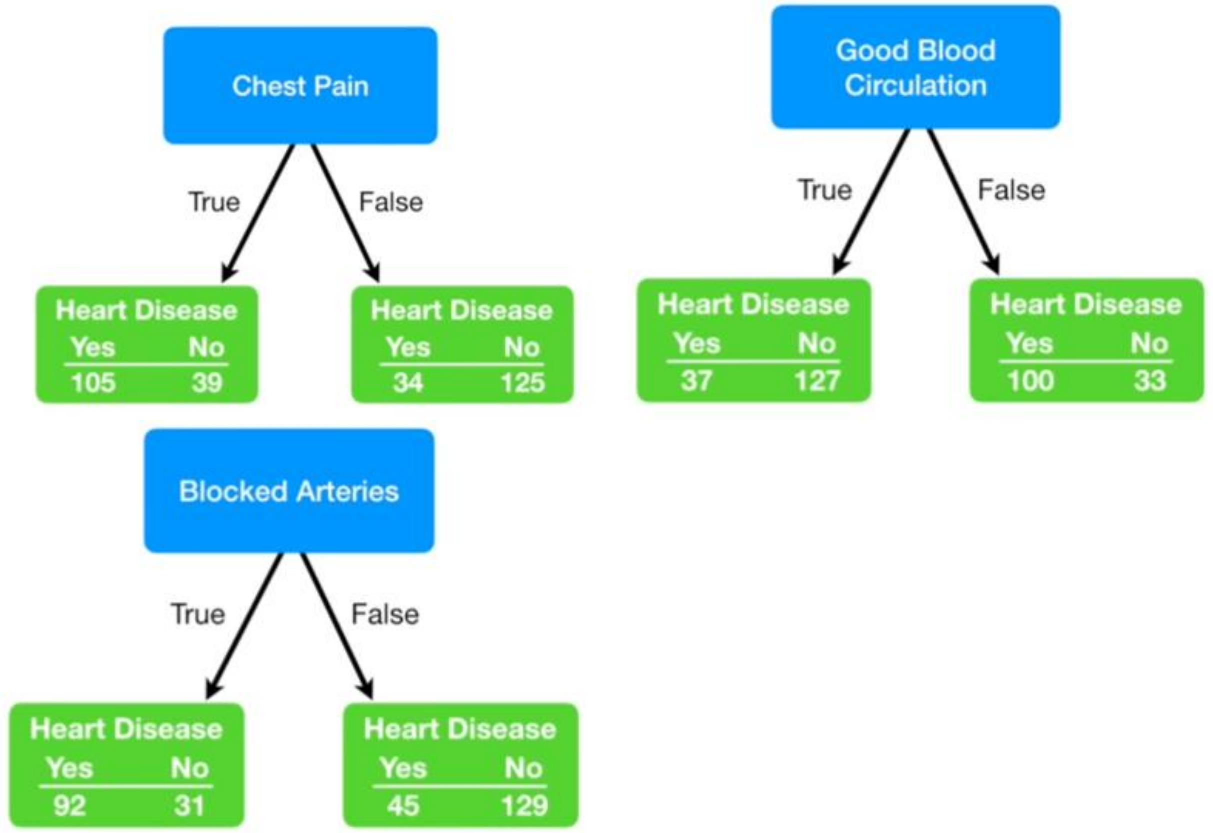
Decision Tree Introduction

The first thing we want to know is, **what should be our root node ?**
Chest Pain, Good Blood Circulation or Blocked Arteries ?

Lets start by looking each feature separately.



Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...


Chest Pain
True
False
Heart Disease Yes No 105 39
Heart Disease Yes No 34 125
Good Blood Circulation
True
False
Heart Disease Yes No 37 127
Heart Disease Yes No 100 33
Blocked Arteries
True
False
Heart Disease Yes No 92 31
Heart Disease Yes No 45 129

Decision Tree Introduction

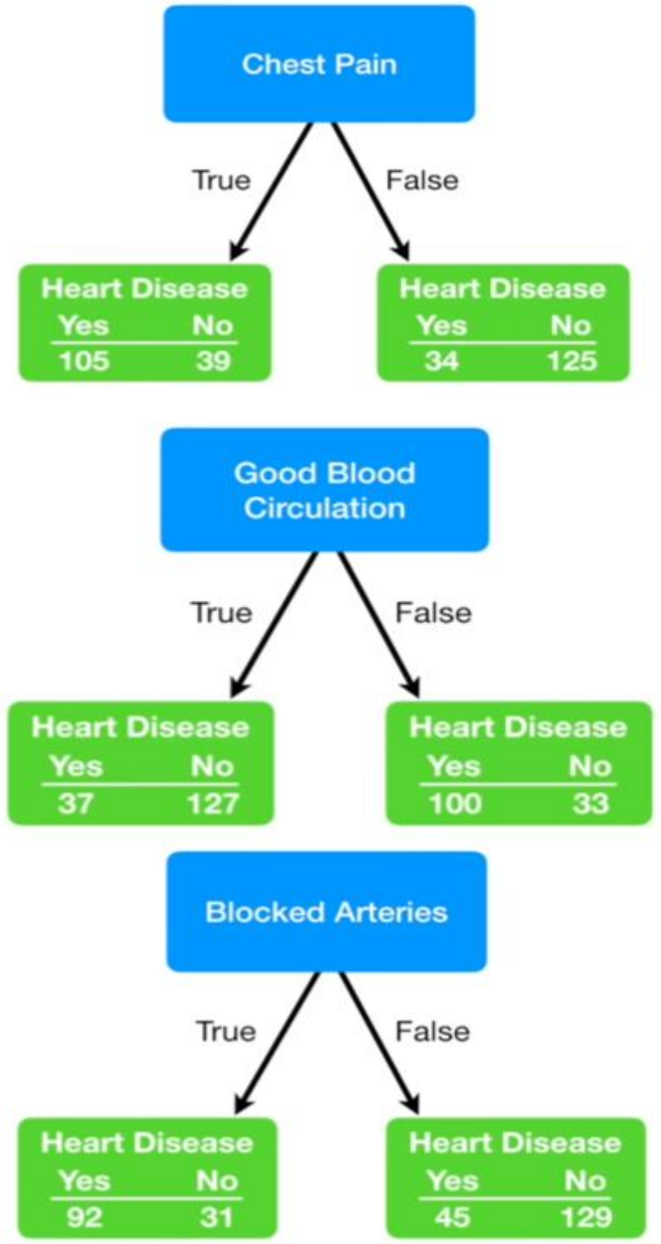
The Goal is: **what should be our root node ?**
Chest Pain, Good Blood Circulation or Blocked Arteries ?

We looked at how well the '**Chest Pain'** separated patients with and without heart disease. **It did 'OK' but it was not perfect.**

All the leaf nodes contains both 'Yes – Heart Disease' as well as 'No – Heart Disease'.
Same is the scenario for other features as well.

Because none of the lead nodes are 100% "YES Heart Disease" or "100% - NO Hear Disease", they are all considered as **impure.**

To determine which separation is best, we need a way to measure and compare this **impurity.**

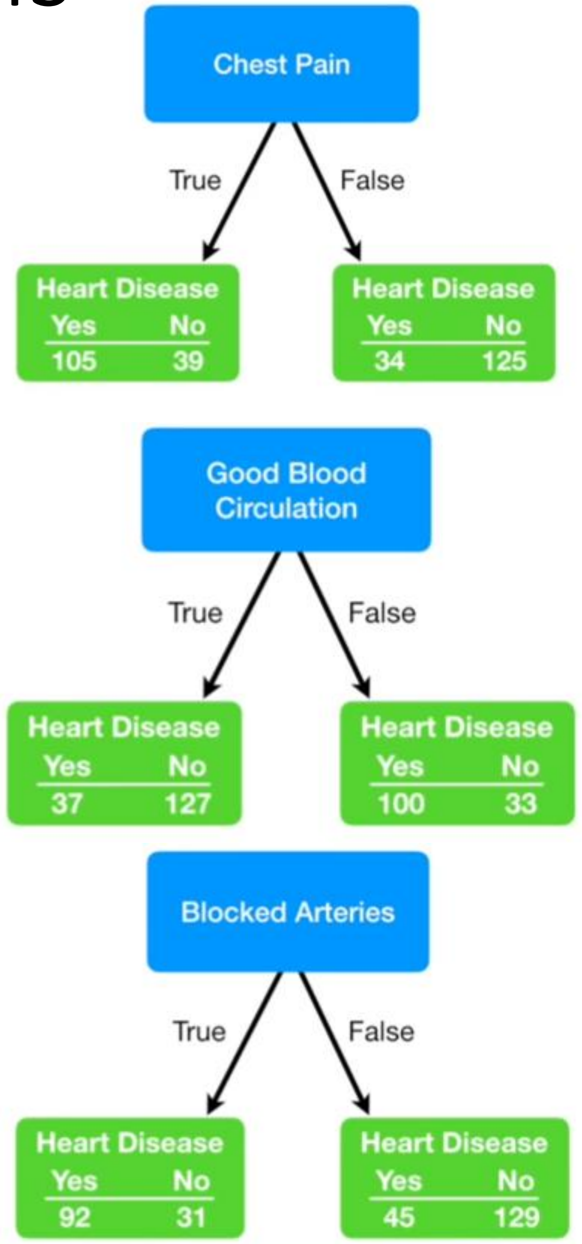


Decision Tree – Algorithms

The Goal is: **what should be our root node ?**
Chest Pain, Good Blood Circulation or Blocked Arteries ?

Following algorithms are used to measure impurity :

- Gini Index / Gini Coefficient / Gini Ratio
- ID3
- Chi Squared
- Reduction in Variation



Decision Tree – Gini Index

Gini index says, if we select two items from a population at random then they must be of same class and probability for this is 1 if population is pure.

- It works with categorical target variable “Success” or “Failure” and Numerical Data.
- It performs only Binary splits
- Higher the value of Gini higher the homogeneity.
- CART (Classification and Regression Tree) uses Gini method to create binary splits.

Steps to Calculate Gini for a split

1. Calculate Gini for sub-nodes, using formula $(1 - p^2 - q^2)$.
    - Where, p = probability of “yes”
    - q = probability of “no”

2. Calculate Gini for split using weighted Gini score of each node of that split

Note: Gini index is often used in finance/economic decision making.
The **Gini index** is a statistical measure of distribution often used as a gauge of economic inequality.
A society that scores 0.0 on the **Gini** scale has perfect equality in income distribution. Higher the number over 0 higher the inequality, and the score of 1.0 (or 100) indicates total inequality where only one person corners all the income

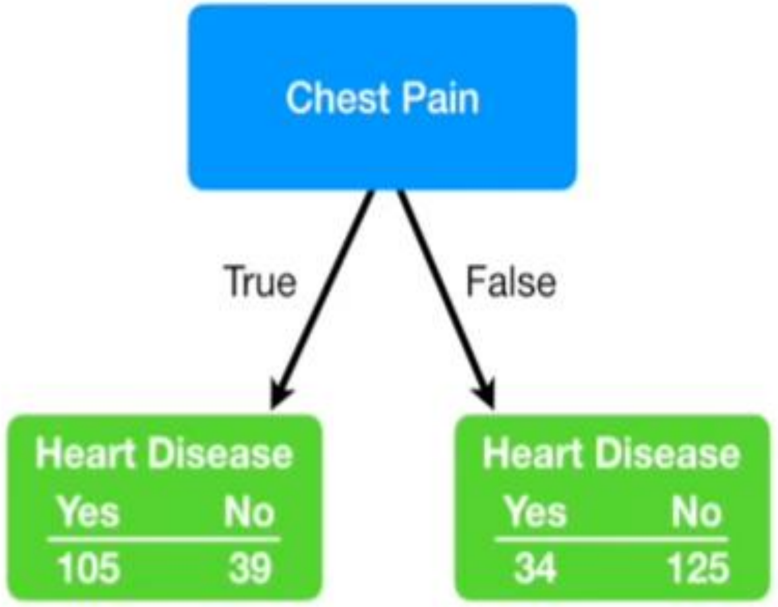
Decision Tree – Gini Index Categorical

Step 1: Gini index calculation for root node

Gini Impurity for 'Chest Pain',

For Left Leaf Node:
Gini impurity = 1 – (probability of "yes")^2 – (probability of "no")^2
Gini impurity = 1 – (105 / 105 + 39)^2 – (39/105+39)^2
Gini impurity = 0.395

For Right Leaf Node:
Gini impurity = 1 – (probability of "yes")^2 – (probability of "no")^2
Gini impurity = 1 – (34/34+125)^2 – (125/34+125)^2
Gini impurity = 0.336



The left leaf node = 144 patients and right leaf node = 159 patients,
Thus the total Gini impurity for 'Chest Pain' to separate patients with and without Heart disease is:
Total Gini impurity = Weighted average of the leaf node impurities

$$\textit{Total Gini Impurity} = \frac{\textit{total observations in left node}}{\textit{total observations in both nodes}}\ \text{Gini left} + \frac{\textit{total observations in right node}}{\textit{total observations in both nodes}}\ \text{Gini right}$$

Total Gini Impurity = $\frac{144}{144+159} 0.395 + \frac{159}{144+159}$ **0.336 = 0.364**

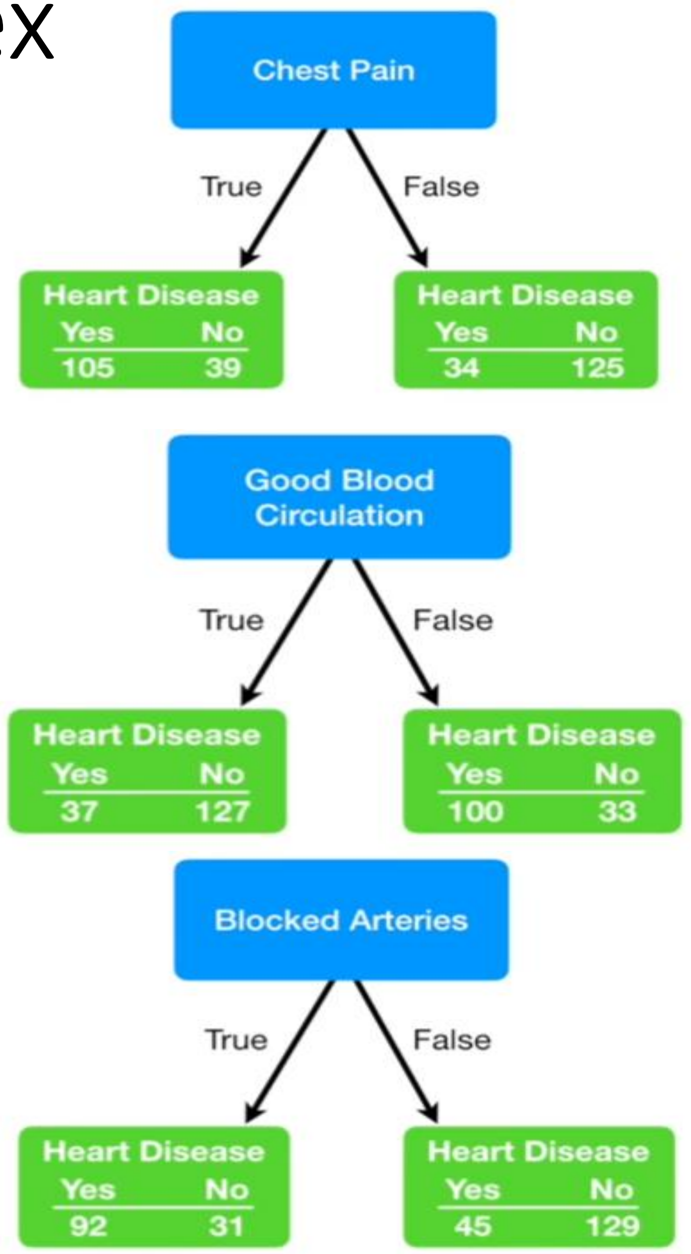
Decision Tree – Gini Index

Step 1: Gini index calculation for root node

- Gini impurity for Chest Pain = 0.364
- Gini impurity for Good Blood Circulation = 0.360
- Gini impurity for Blocked Arteries = 0.381

Good Blood Circulation has the lowest impurity
i.e.
It separates patients with and without heart disease the best.

Hence we will use it as the root of the tree.

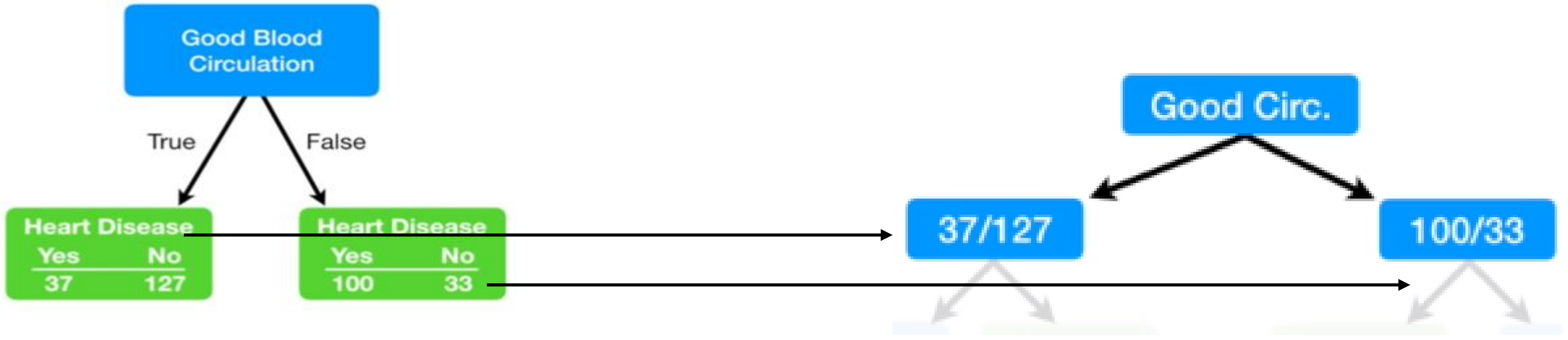


Decision Tree – Gini Index Categorical

Step 1: Gini index calculation for root node

When we divide all of the patients using **Good Blood Circulation,** we ended up with “impure” leaf nodes.

Each leaf contained a mixture of patients with and without Heart Disease.



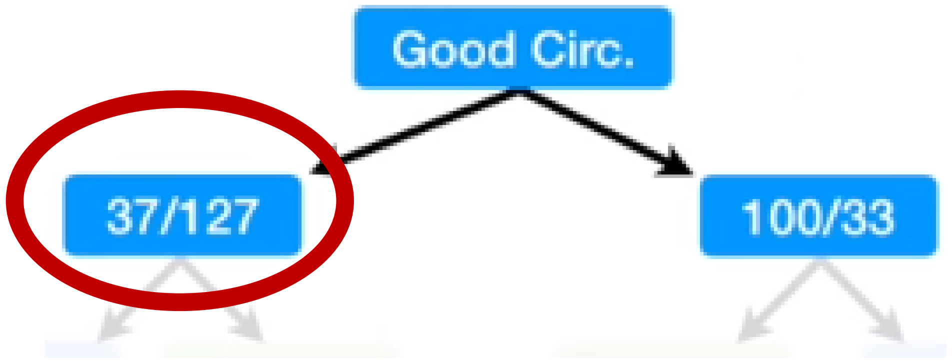
Decision Tree – Gini Index Categorical

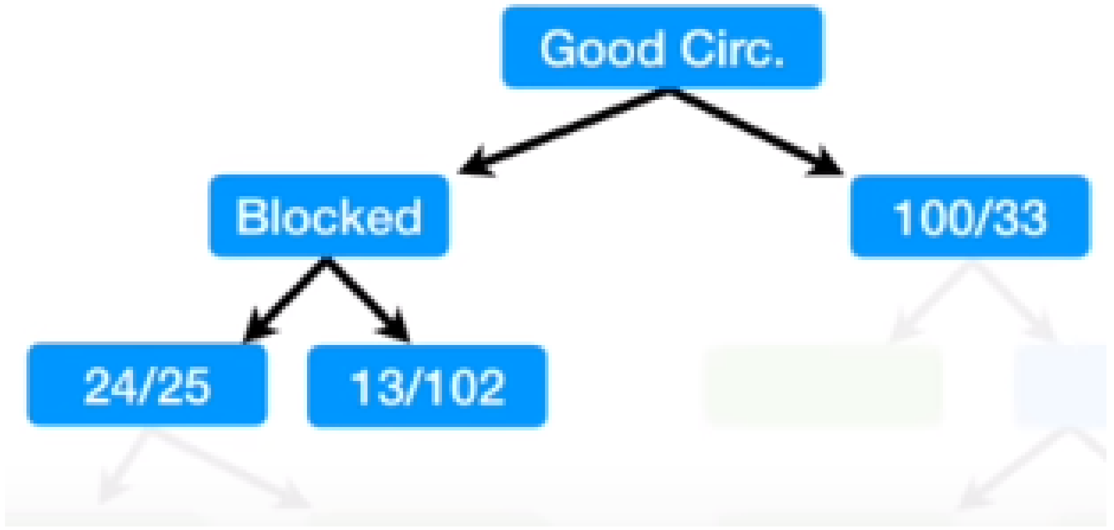
Step 2: Gini index calculation for decision node

Now we need to figure how well **Chest Pain** and **Blocked Arteries** separate these 164 (37+127) patients

- Gini impurity for Chest Pain = 0.3
- Gini impurity for Blocked Arteries = 0.290

Since Blocked Arteries has lower Gini impurity, we will use it at this node to separate patients





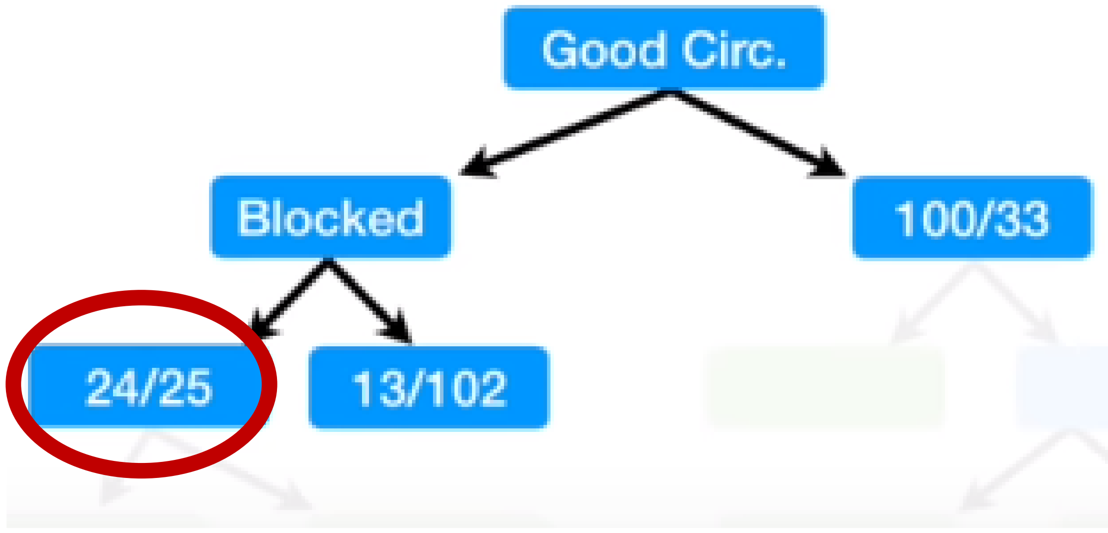
Decision Tree – Gini Index Categorical

Step 3: Gini index calculation for last decision node

Now we need to figure how well **Chest Pain** separate these 49 (24 + 25) patients:

- Gini impurity calculated at 2nd level for Chest Pain = 0.3

Now we calculate Gini impurity for left node.
IF : Gini impurity of left node > Gini impurity of Chest Pain (0.3)
We further classify the left node based on the remaining feature, here Chest Pain
ELSE: (Gini index of left node < Gini impurity of Chest Pain (0.3))
That means, the current node fairly classify the variables, Hence we do not further classify the observations in the node and stop there.



Decision Tree – Gini Index Categorical

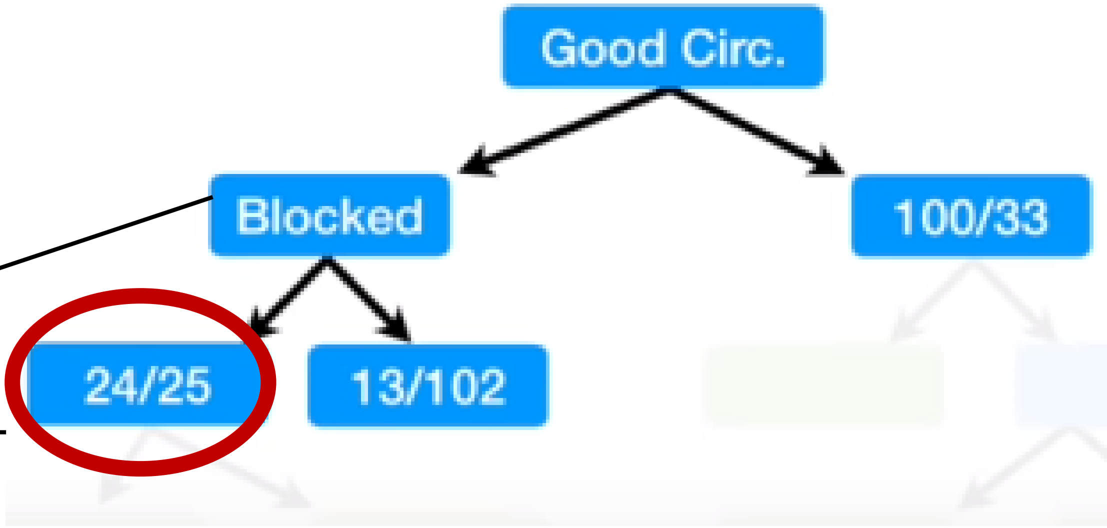
Step 3: Gini index calculation for last decision node

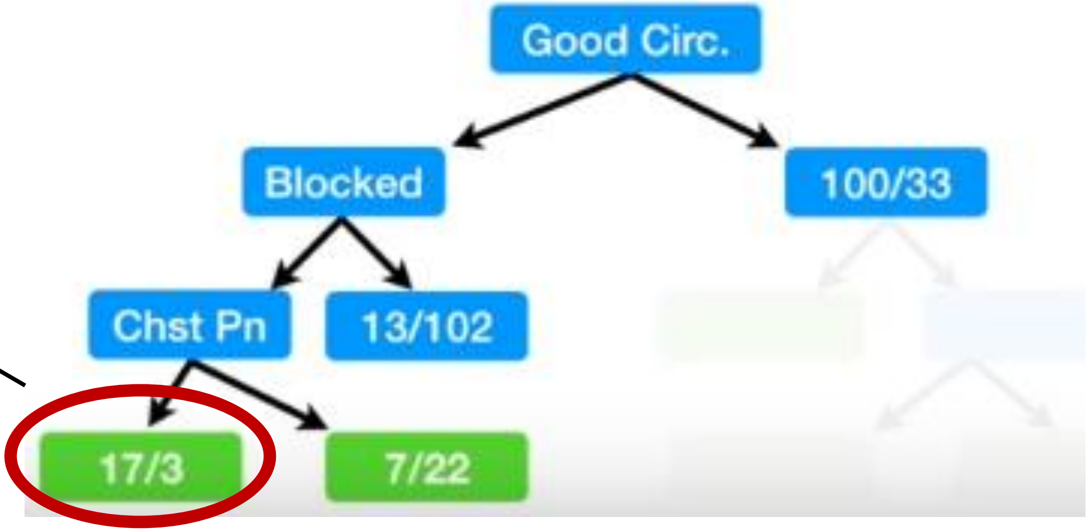
Now we need to figure how well **Chest Pain** separate these 49 (24 + 25) patients:

Gini impurity calculated at 2nd level for Chest Pain = 0.3
Gini index of left node = 0.4998

Hence we further classify the left node based on ‘Chest Pain’

Now, Gini index of last left node = 0.2550
Which is less than Gini index of ‘Chest Pain’ calculated above (0.4998). Hence we stop here.





Decision Tree – Gini Index Categorical

Step 3: Gini index calculation for last decision node

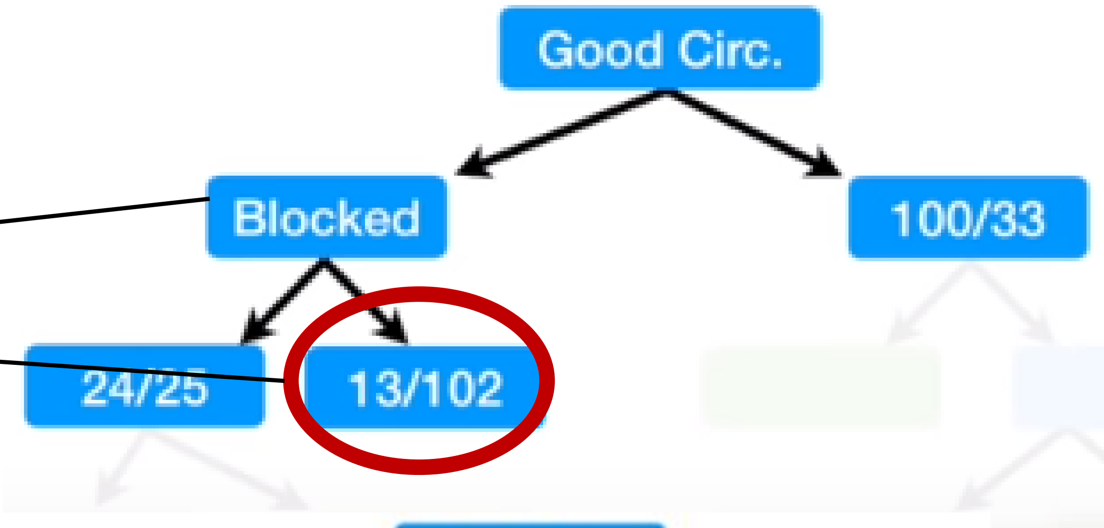
Gini impurity calculated at 2nd level for Chest Pain = 0.3

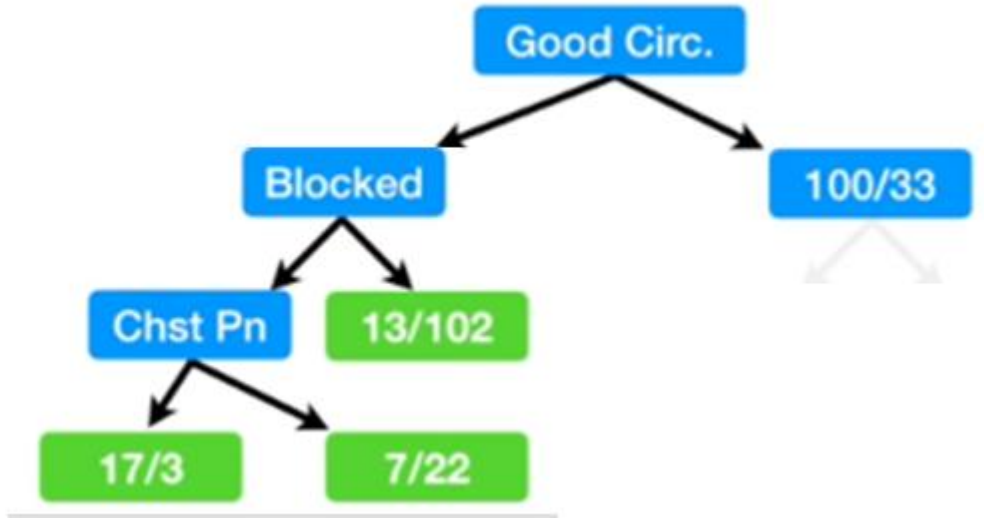
Gini index of left node = **0.1987**

Here impurity of lower node is (0.1987) < upper node (0.3)

Hence the impurity is lower if we do not separate patients using "Chest Pain".

Hence we make it a leaf node and do not classify it further.



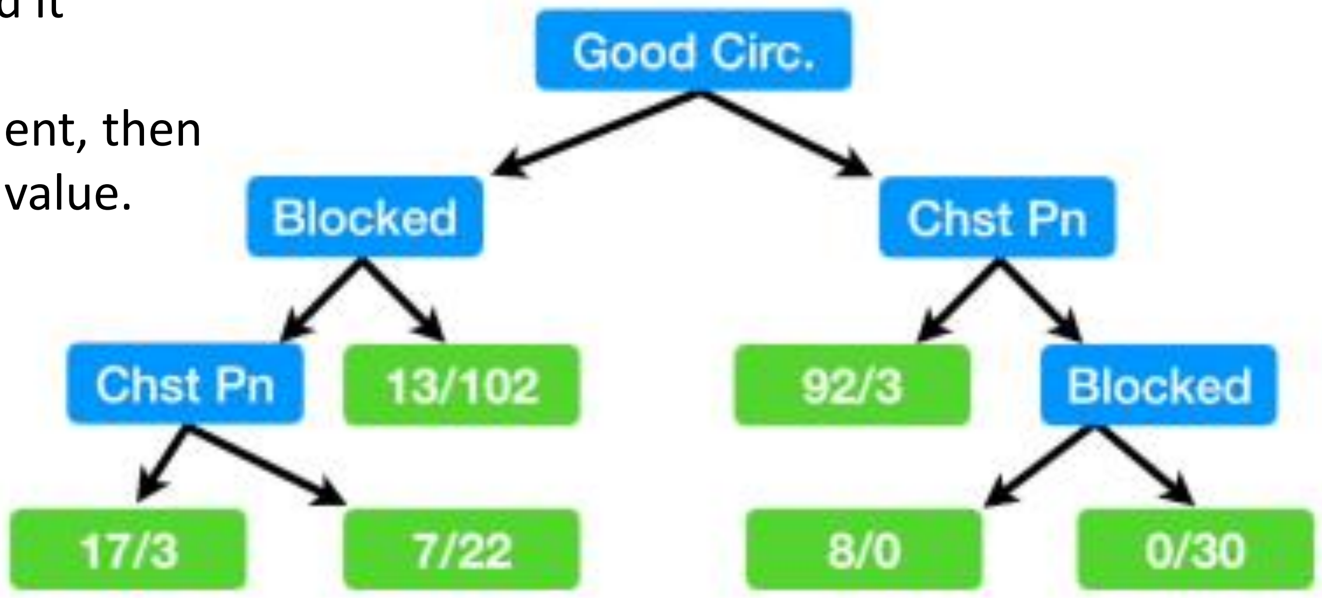


Decision Tree – Gini Index Categorical

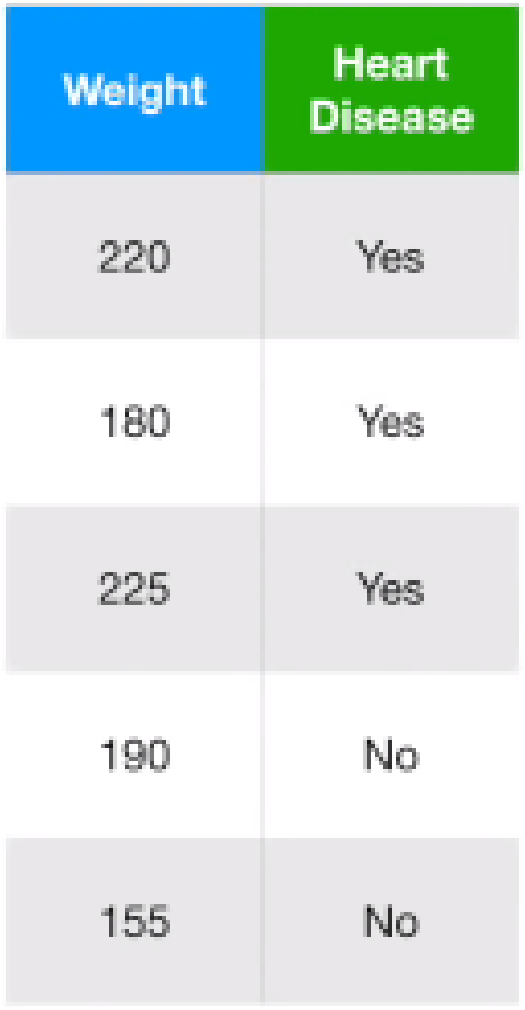
Step 4: Gini index calculation for Right Side

We follow the exact same steps as we did on the left side:

1. Calculate all of the Gini impurity scores.
2. If the node itself has the lowest score, then there is no point in separating the patients any more and it becomes a leaf node.
3. If separating the data results in an improvement, then pick the separation with the lowest impurity value.



Decision Tree – Gini Index Numeric Data

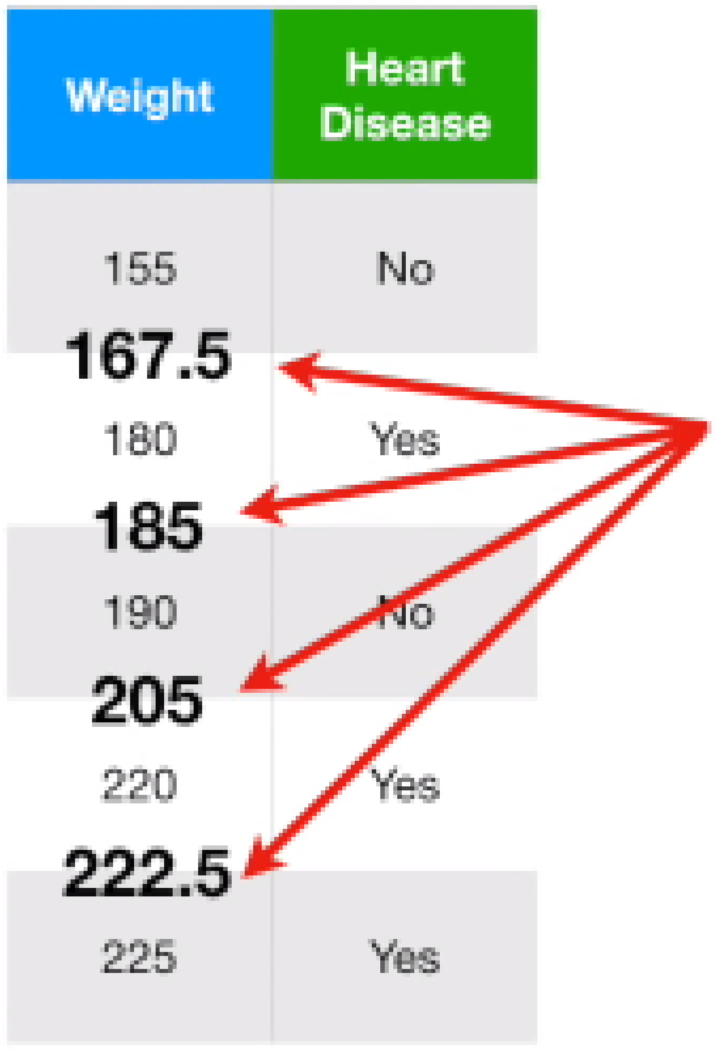


Weight	Heart Disease
220	Yes
180	Yes
225	Yes
190	No
155	No

Suppose we have Numeric Data like this. Then how do Gini Index work on Numeric Data ?

It performs the following steps:

Decision Tree – Gini Index Numeric Data



Weight	Heart Disease
155	No
167.5	
180	Yes
185	
190	No
205	
220	Yes
222.5	
225	Yes

Step 1: Sort the Patients by weight, lowest to highest

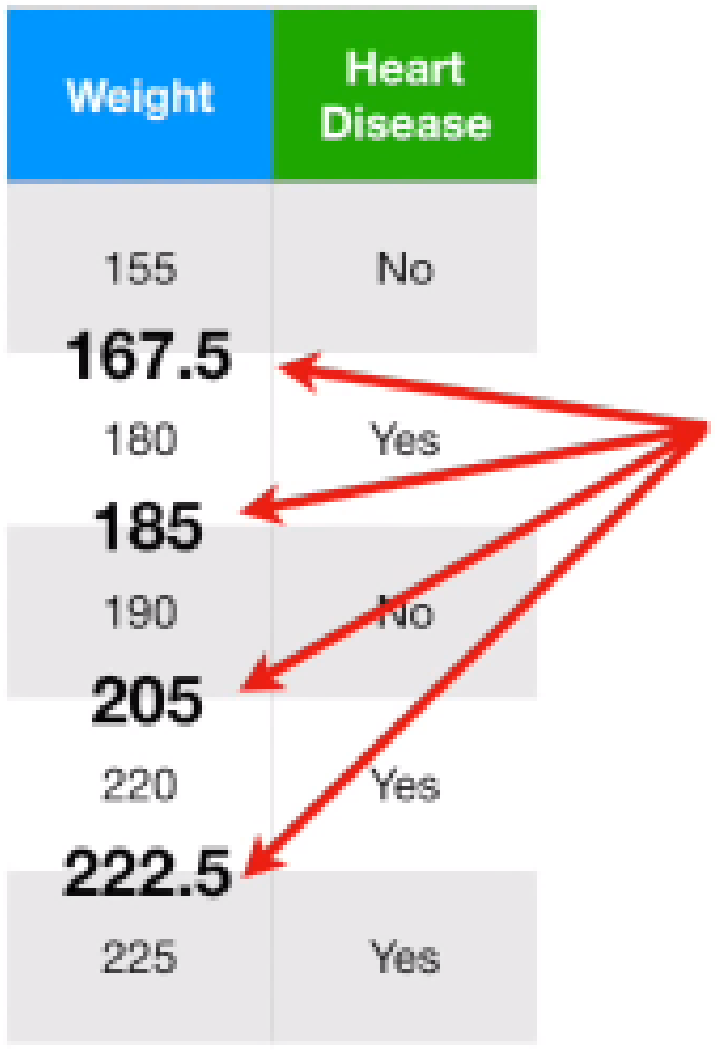
Step 2: Calculate the adjacent weights

(155 + 180)/2 = 167.5

(180 + 190)/2 = 185

…

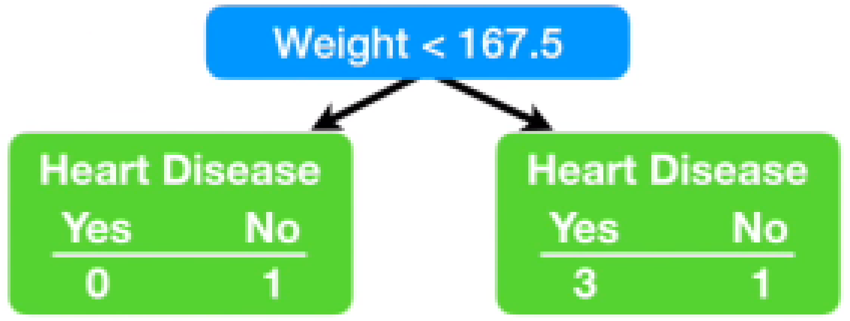
Decision Tree – Gini Index Numeric Data



Weight	Heart Disease
155	No
167.5	
180	Yes
185	
190	No
205	
220	Yes
222.5	
225	Yes

Step 3: Calculate Gini impurity for each average weight.

e.g



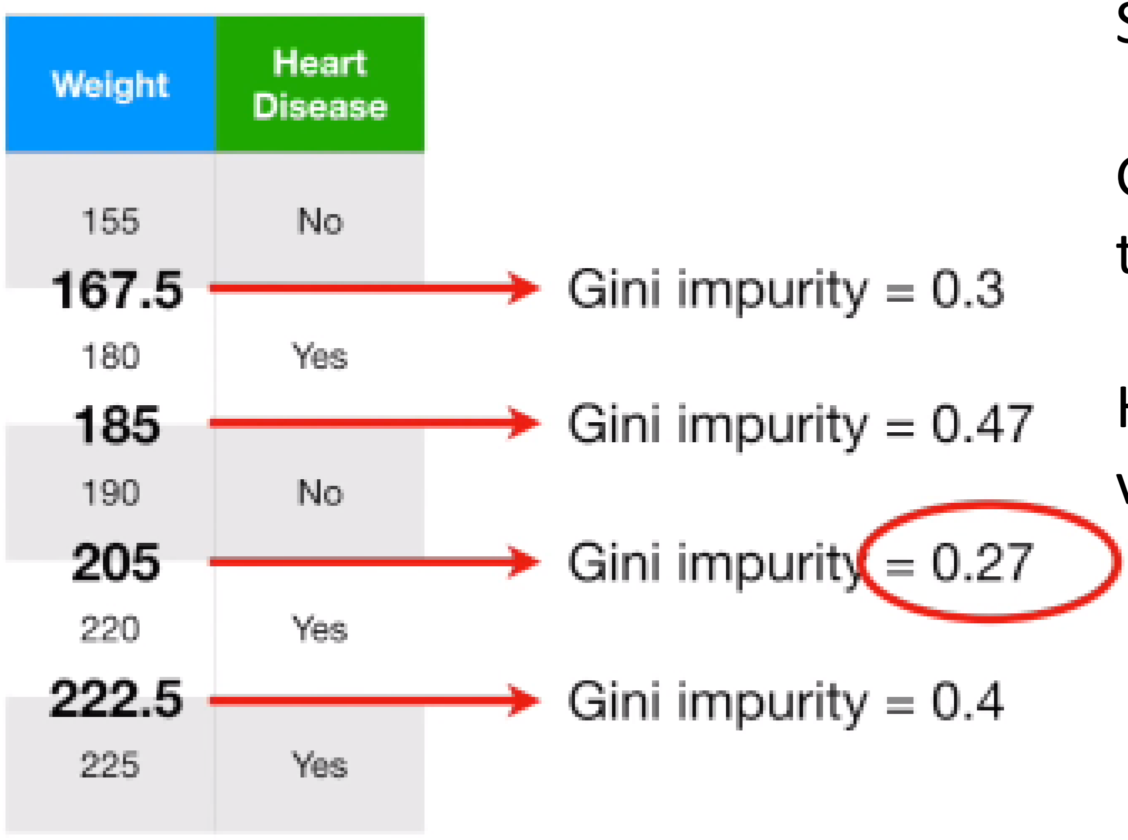
Gini impurity = 0

Gini impurity = 0.375

Gini impurity for Weight < 167.5 = Weighted average of the two leaves

Gini impurity for Weight < 167.5 = 0.336 ~ 0.3

Decision Tree – Gini Index Numeric Data



Weight	Heart Disease
155	No
180	Yes
190	No
220	Yes
225	Yes

Step 4: Find Root Element

Get the lowest impurity value and separate using the same.

Here for weight 205, we get the lowest impurity value, hence we use this as the root element.

Decision Tree – ID3

ID3 is the core algorithm for building decision trees.
ID3 uses *Entropy* and *Information Gain* to construct a decision tree.

Entropy
A decision tree is built top-down from a root node and involves partitioning the data into subsets that contain instances with similar values (homogeneous).
ID3 algorithm uses entropy to calculate the homogeneity of a sample.

If the sample is completely homogeneous the entropy is zero
If the sample is equally divided then it has entropy of one.

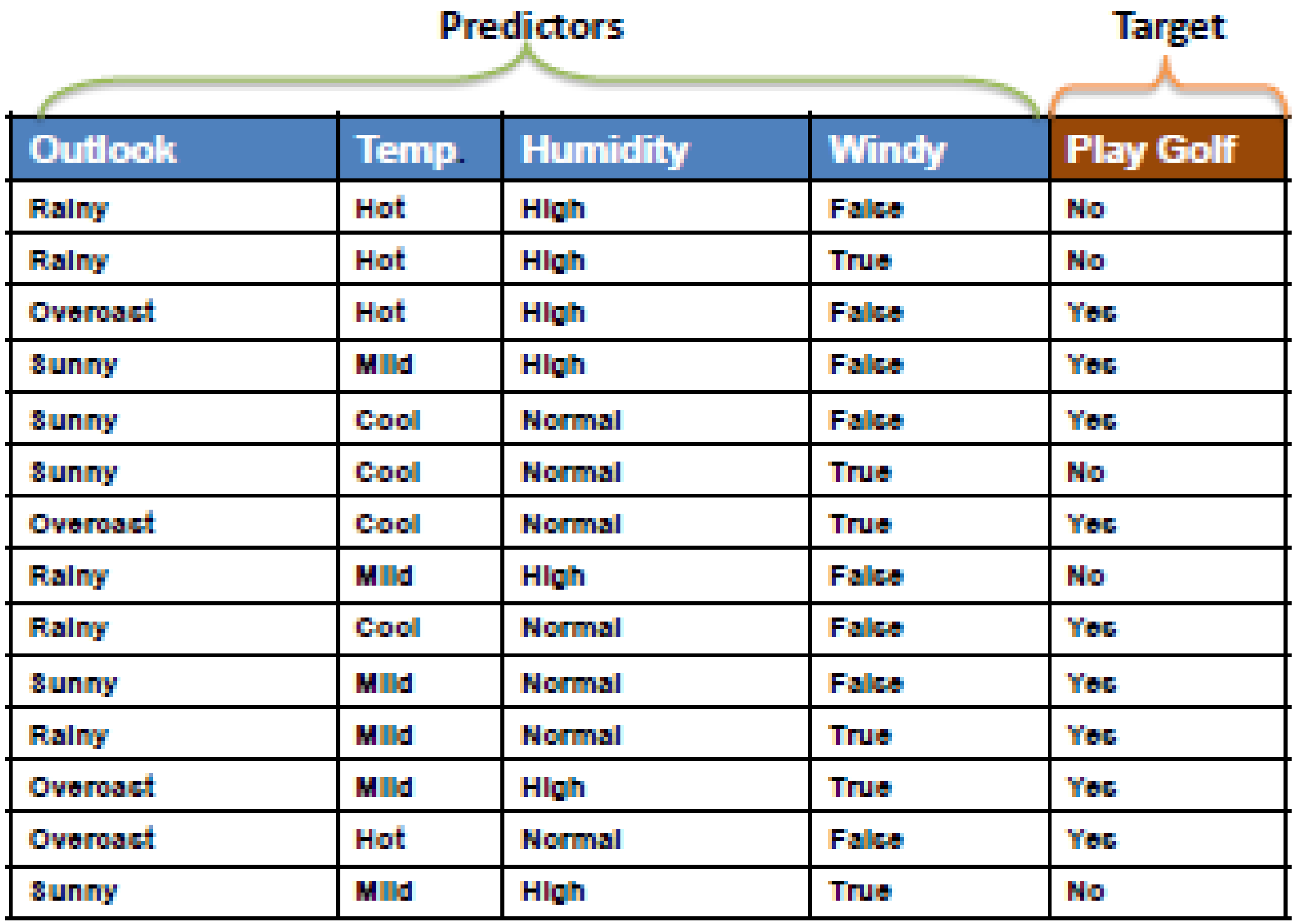
Decision Tree – ID3

ID3 is the core algorithm for building decision trees.
ID3 uses *Entropy* and *Information Gain* to construct a decision tree.

Information Gain
The information gain is based on the decrease in entropy after a data-set is split on an attribute.
Constructing a decision tree is all about finding attribute that returns the highest information gain
(i.e., the most homogeneous branches)

Decision Tree – ID3

Example: Say we have the below dataset



Predictors

Target

Outlook	Temp.	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

Decision Tree – ID3

ID3 performs the following steps to select the root node and further decision nodes.

1. Sample Entropy of target

$$E(S) = \sum_{i=1}^{c} -p_i \log_2 p_i$$

5/(5 + 9) = 0.3571 ~ 0.36
9/(5 + 9) = 0.6428 ~ 0.64

Play Golf	
Yes	No
9	5

Entropy(PlayGolf) = Entropy (5,9)
= Entropy (0.36, 0.64)
= - (0.36 $\log_2$ 0.36) - (0.64 $\log_2$ 0.64)
= 0.94

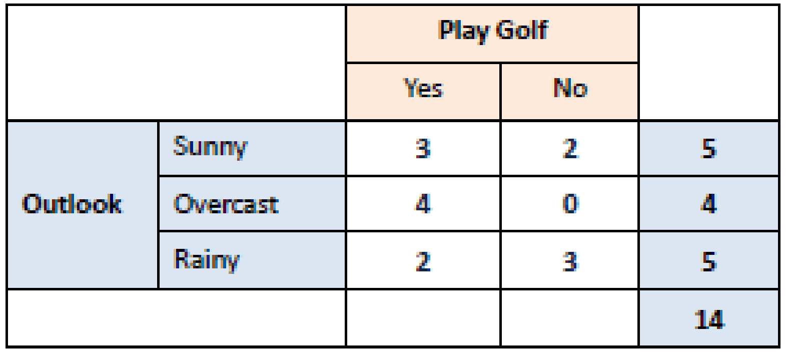
Decision Tree – ID3

Step 2: Calculate Information Gain

This is done in 2 steps:

Step 2.1: Calculate Entropy of each feature with target as below:

$$E(T,X) = \sum_{c \in X} P(c)E(c)$$



		Play Golf		
		Yes	No	
Outlook	Sunny	3	2	5
	Overcast	4	0	4
	Rainy	2	3	5
				14

E(PlayGolf, Outlook) = **P**(Sunny)***E**(3,2) + **P**(Overcast)***E**(4,0) + **P**(Rainy)***E**(2,3)

= (5/14)*0.971 + (4/14)*0.0 + (5/14)*0.971

= 0.693

Decision Tree – ID3

Step 2: Calculate Information Gain

Step 2.2 : Calculate Information gain of each feature with target as below:

$$Gain(T,X) = Entropy(T) - Entropy(T,X)$$

G(PlayGolf, Outlook) = **E**(PlayGolf) – **E**(PlayGolf, Outlook)

= 0.940 – 0.693 = 0.247

By the end of Step 2, we get Gain for each variable with the target

		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			

		Play Golf	
		Yes	No
Temp.	Hot	2	2
	Mild	4	2
	Cool	3	1
Gain = 0.029			

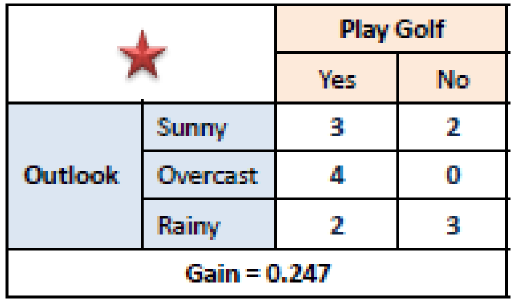
		Play Golf	
		Yes	No
Humidity	High	3	4
	Normal	6	1
Gain = 0.152			

		Play Golf	
		Yes	No
Windy	False	6	2
	True	3	3
Gain = 0.048			

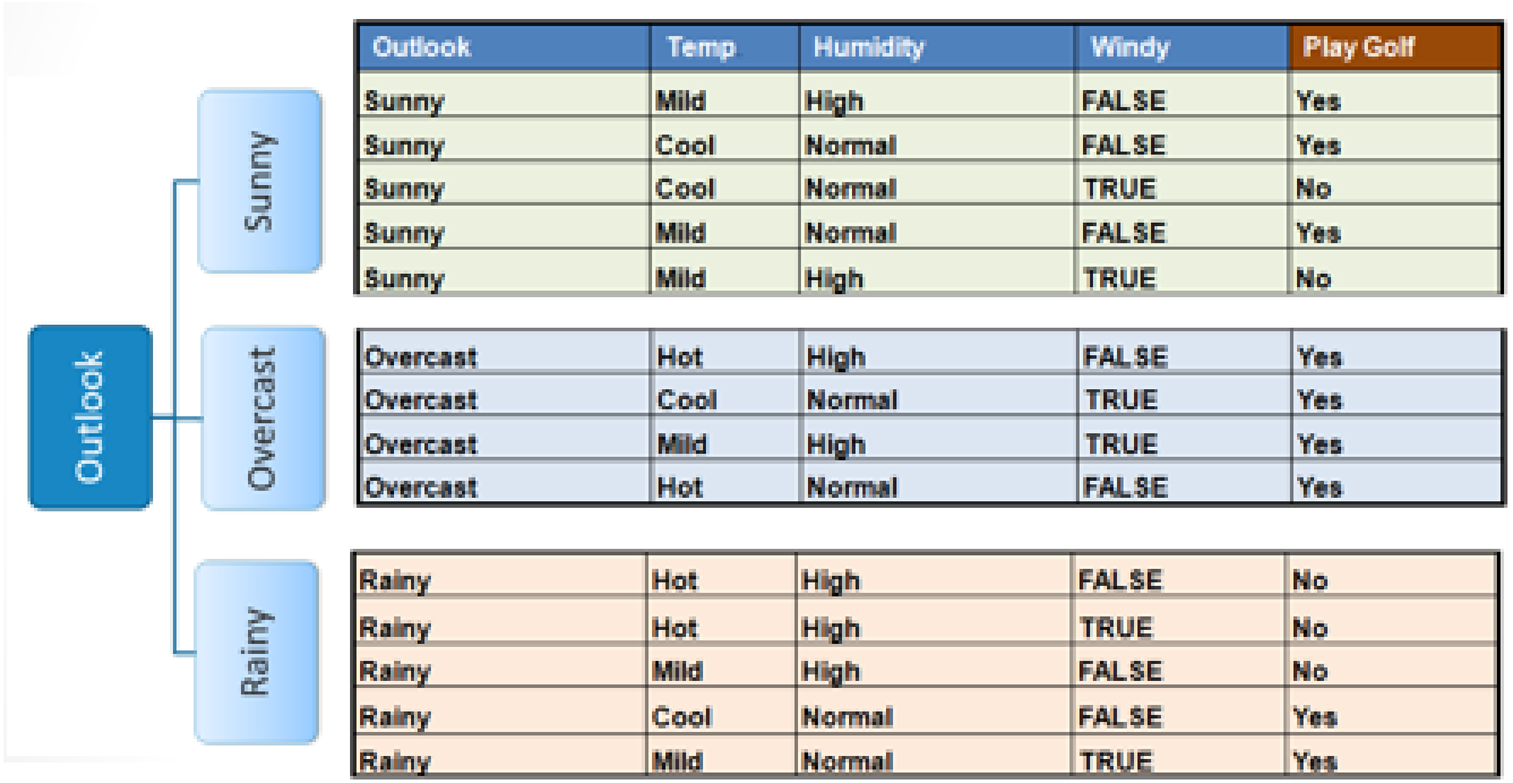
Decision Tree – ID3

Step 3:

Choose attribute with the largest information gain as the decision node, divide the dataset by its branches and repeat the same process on every branch



★		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			



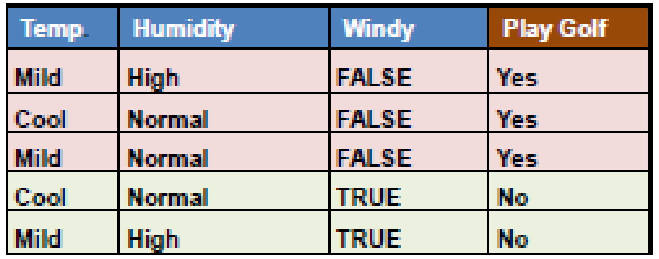
Outlook	Temp	Humidity	Windy	Play Golf
Sunny	Mild	High	FALSE	Yes
Sunny	Cool	Normal	FALSE	Yes
Sunny	Cool	Normal	TRUE	No
Sunny	Mild	Normal	FALSE	Yes
Sunny	Mild	High	TRUE	No
Overcast	Hot	High	FALSE	Yes
Overcast	Cool	Normal	TRUE	Yes
Overcast	Mild	High	TRUE	Yes
Overcast	Hot	Normal	FALSE	Yes
Rainy	Hot	High	FALSE	No
Rainy	Hot	High	TRUE	No
Rainy	Mild	High	FALSE	No
Rainy	Cool	Normal	FALSE	Yes
Rainy	Mild	Normal	TRUE	Yes

Decision Tree – ID3

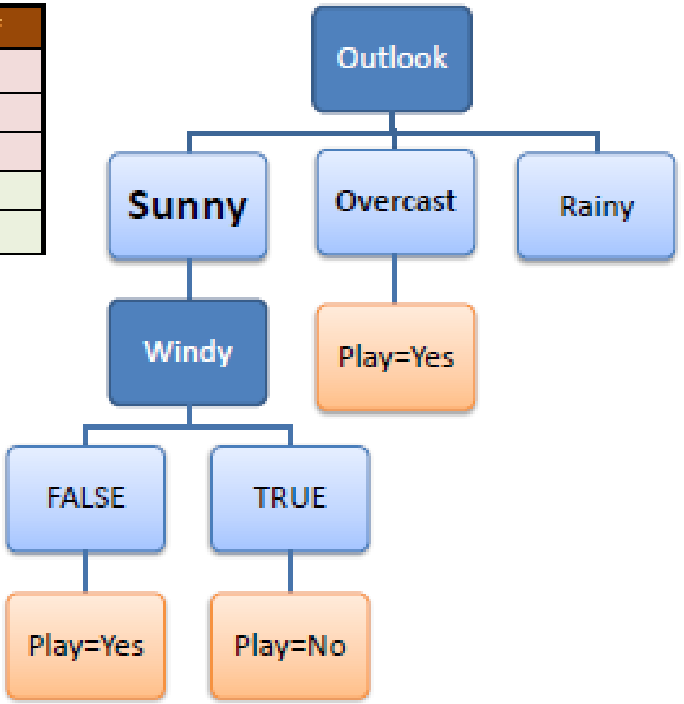
Step 4a: A branch with entropy of 0 is a leaf node.

Step 4b: A branch with entropy more than 0 needs further splitting.

Step 5: The ID3 algorithm is run recursively on the non-leaf branches, until all data is classified.



Temp	Humidity	Windy	Play Golf
Mild	High	FALSE	Yes
Cool	Normal	FALSE	Yes
Mild	Normal	FALSE	Yes
Cool	Normal	TRUE	No
Mild	High	TRUE	No



Decision Tree – Horizontal Effect & Pruning

It is really difficult to get the optimal size in a decision tree.

A tree that is too large risks overfitting the training data and poorly generalizing to new samples.

A small tree might not capture important structural information about the sample space.
However, it is hard to tell when a tree algorithm should stop because it is impossible to tell if the addition of a single extra node will dramatically decrease error.

This problem is known as the horizon effect.

A common strategy is to grow the tree until each node contains a small number of instances then use pruning to remove nodes that do not provide additional information.

Pruning should reduce the size of a learning tree without reducing predictive accuracy.

Decision Tree – Horizontal Effect & Pruning

There are many techniques for tree pruning.
Some are:

- Reduced Error Pruning
- Cost Complexity Pruning

Reduced Error Pruning:

- One of the simplest forms of pruning is reduced error pruning.
- Starting at the leaves, each node is replaced with its most popular class.
- If the prediction accuracy is not affected then the change is kept.
- While somewhat naive, reduced error pruning has the advantage of simplicity and speed.

Decision Tree – Advantages

Less data cleaning required: It requires less data cleaning compared to some other modelling techniques.

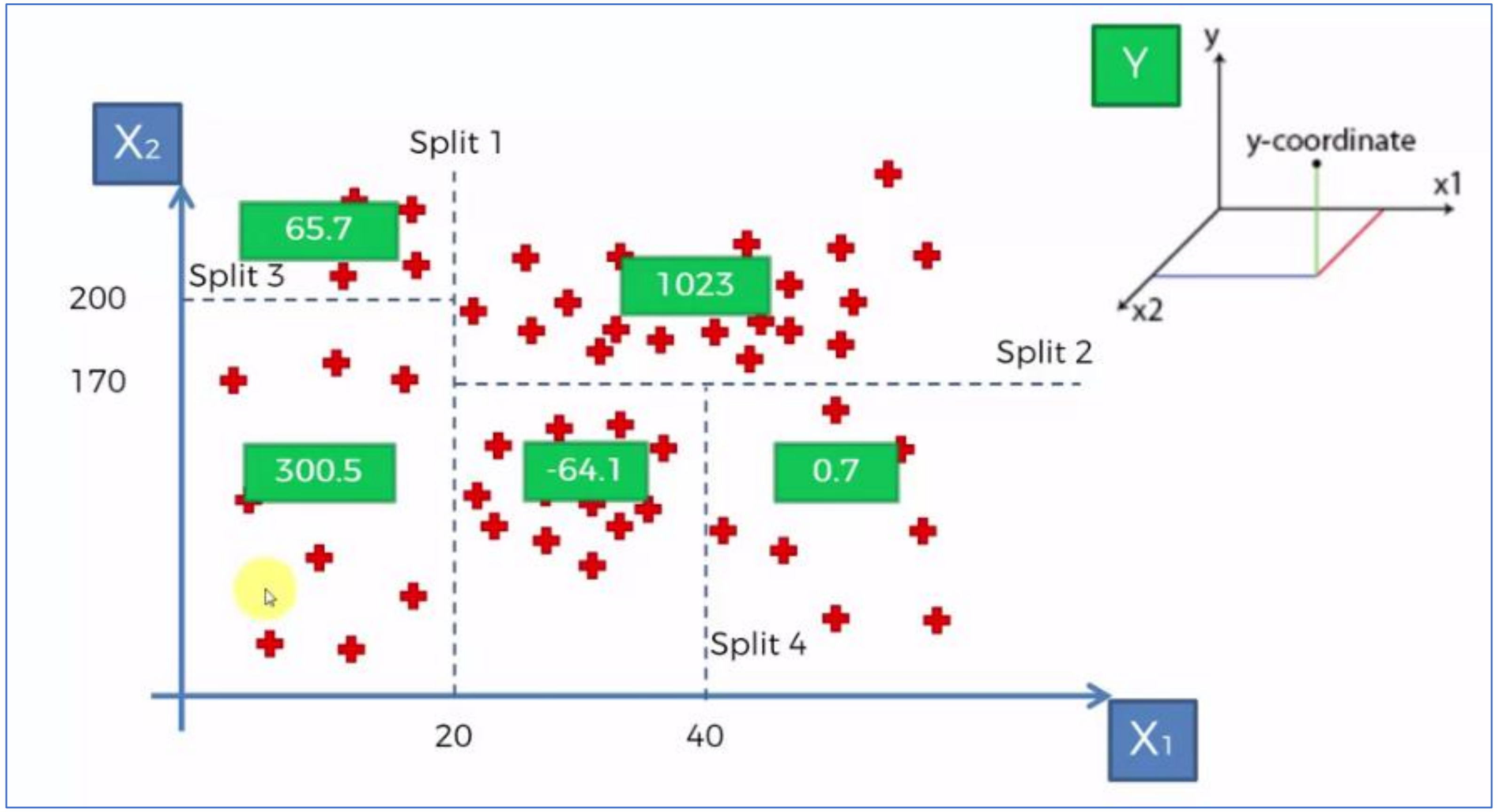
Fair Resistance to Outliers: It is not influenced by outliers and missing values to a fair degree.

Data type is not a constraint: It can handle both numerical and categorical variables.

Non Parametric Method: Decision tree is considered to be a non-parametric method. This means that decision trees have no assumptions about the space distribution and the classifier structure

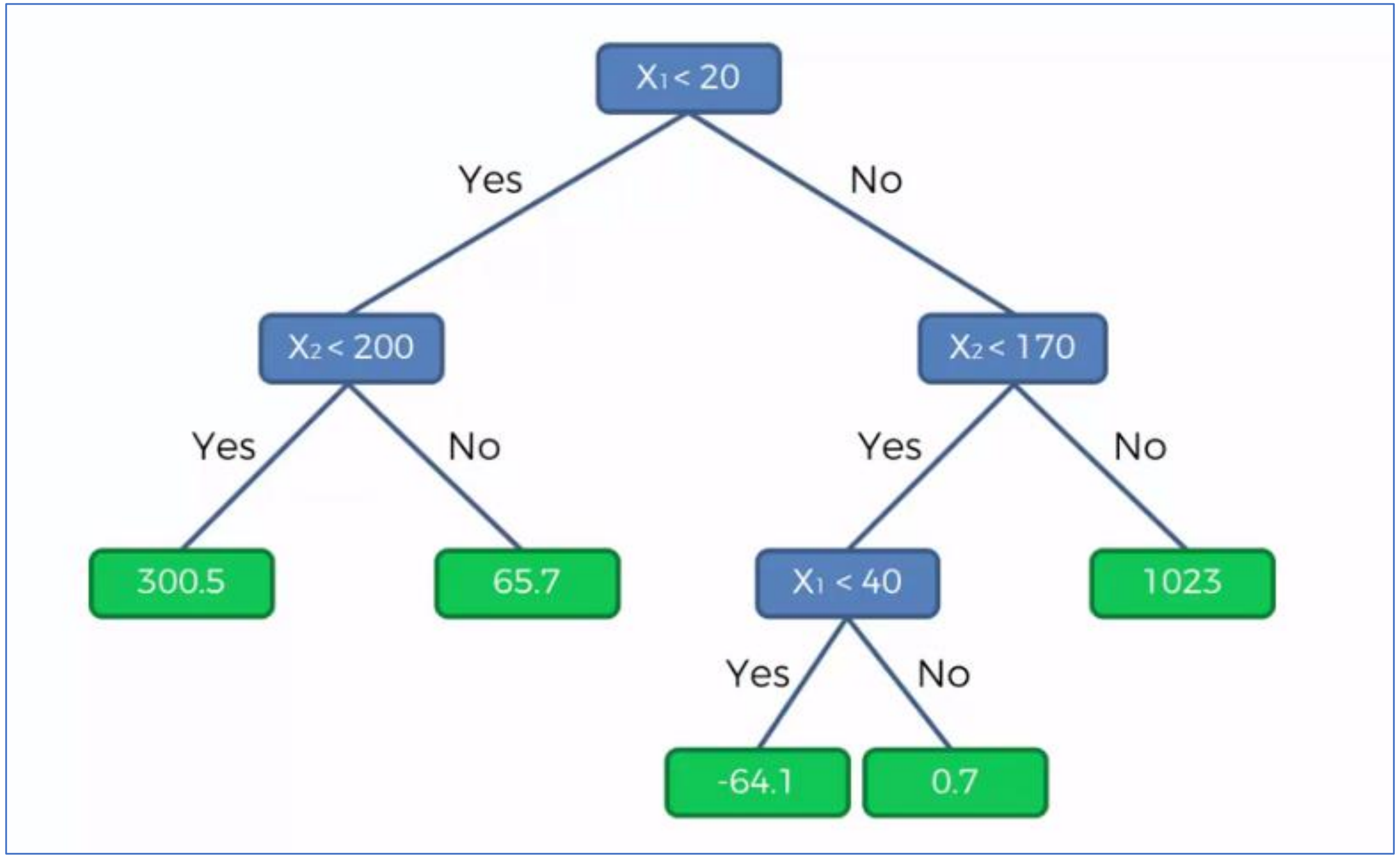
Decision Tree Regressor

X1, X2 are independent variables.
We need to predict y
Here splits happen and then the average of each leaf is taken for predicting the value.



Decision Tree Regressor

X1, X2 are independent variables.
We need to predict y
Here splits happen and then the average of each leaf is taken for predicting the value.


$X_1 < 20$
Yes
No
$X_2 < 200$
$X_2 < 170$
Yes
No
Yes
No
300.5
65.7
$X_1 < 40$
1023
Yes
No
-64.1
0.7