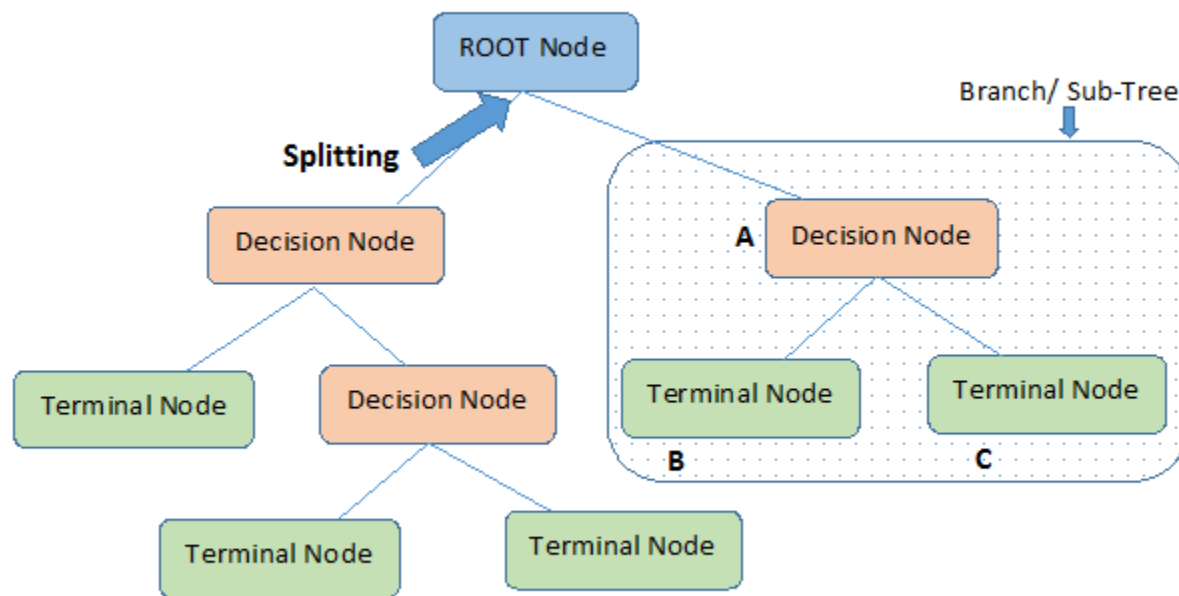


Decision Tree – Entropy and Gini Index



Note:- A is parent node of B and C.

Definitions

- 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.

Definitions

- 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.

Types of Algorithms in Decision Tree

Algorithm used in decision trees:

- **entropy, Information Gain**
- **Gini Index**

Entropy and Information Gain

Entropy and Information Gain

Entropy :

- A measure of Homogeneity of the set of examples
- If the sample is completely homogeneous the entropy=0 and if the sample is equally divided it has entropy =1

Information Gain:

- Measures how well an given attribute separates the training examples according to their target classification
- This measure is used to select among the candidate attributes at each step while growing the tree

Entropy

- **Entropy is used for calculating Information gain.**
- **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).

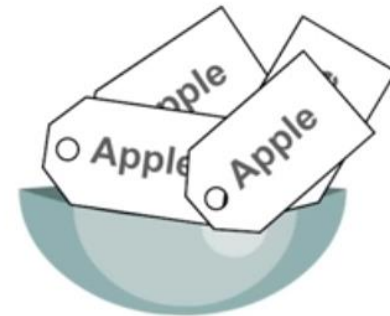
Entropy is used for measuring Impurity

1. If the sample is completely homogeneous the **entropy is zero (If your data is matching- that means no impurity)**
2. If the sample is equally divided then it has **entropy of one. (contains equal divisions of different data). No matching data. Then Impurity is high so entropy is 1.**

Information Gain

- The information gain is the decrease in entropy after a dataset is split the basis of an attribute.
- Constructing a decision tree is all about finding attribute that returns the **highest information gain**. That is split at the attribute that has the highest information gain.
- Information gain is calculated based on entropy.

What is Impurity (in Entropy)



TASK: FRUITS should have matching Labels

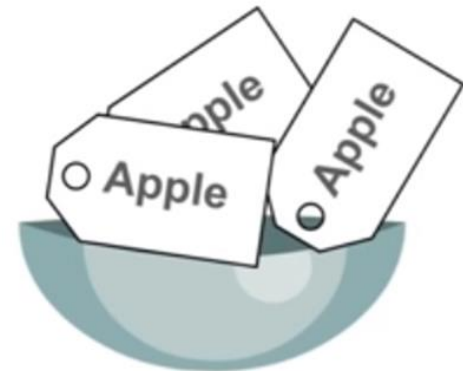
Example: Apples should have the same label (Apples).

If you have to pick Apples and Labels from 2 baskets, you will always get matching labels. So the impurity is 0

Impurity is also the degree of Randomness. In the Basket the degree of Randomness is Less, so impurity is zero



Impurity = 0



Entropy is 0

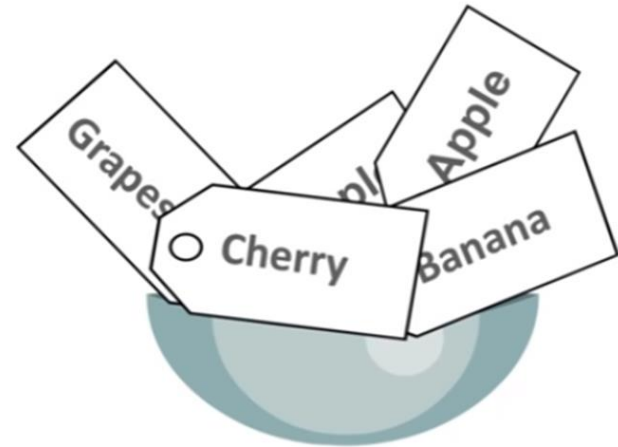
Because Apple and Apple Label Match

TASK: FRUITS should have matching Labels

Example: Apples should have the same label (Apples).

If you have 2 baskets with **different fruits and different labels**.

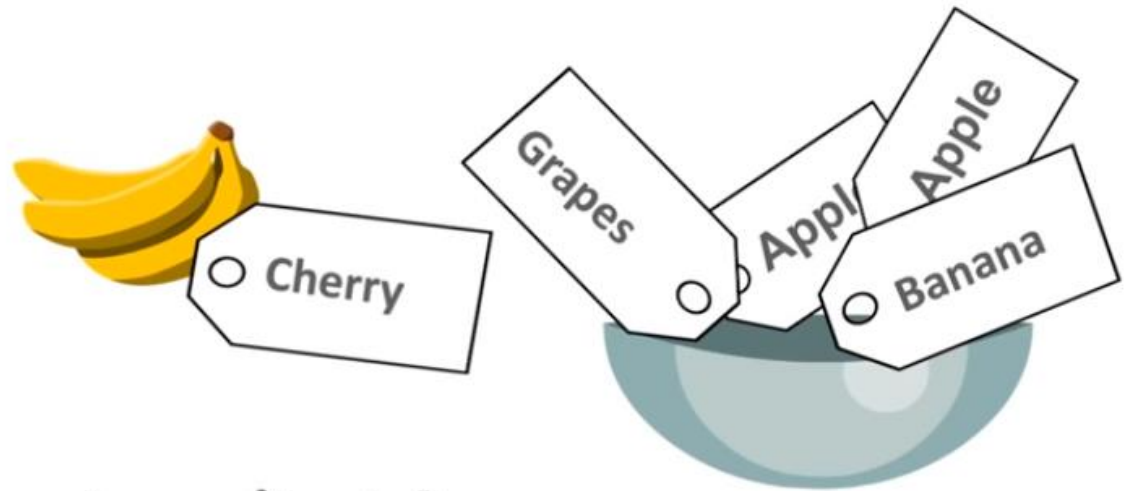
If you have to pick any Random FRUIT or LABEL , it may or may not match



TASK: FRUITS should have matching Labels

Example: Apples should have the same label (Apples).

It may happen a Banana is match with a Cherry Label
So impurity is not equal to zero



Impurity $\neq 0$

Entropy not equal to 0

Entropy is more than 0 or closer to 1

Because Fruits and Label Do not Match

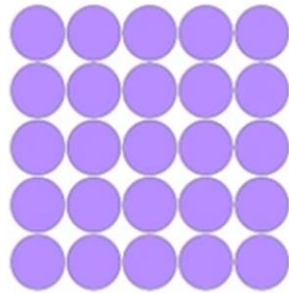
Another Example of Entropy

Let us say you have a bag of balls.

1. You grab one ball from the bag and put it on the table. How many ways can you arrange that ball? The answer: one way.
 2. What if we grab two balls and ask the same question? Now there are more ways to arrange the two balls.
 3. We keep doing this until all the balls are on the table. At this point, there are so many ways to arrange the bag of balls, you might not even be able to count the number of ways.
- This situation is very much like entropy.
 - The higher the entropy (meaning the more ways the system can be arranged), the more the system is disordered.

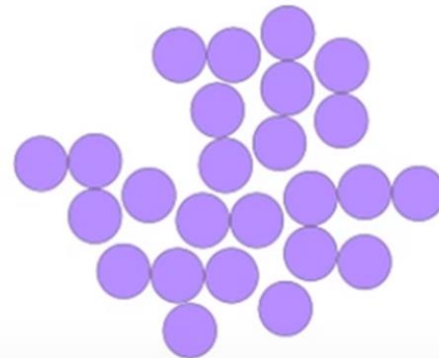
Entropy

- Defines randomness in the data
- **Entropy** is just a metric which measures the impurity or
- The first step to solve the problem of a decision tree



Low Entropy

Data Contains all Yes
Data Contains similar Types
More Pure



High Entropy

Data Contains mix of Yes and No
Data Contains Mix Types
Less Pure

Entropy

$$\text{Entropy}(s) = -P(\text{yes}) \log_2 P(\text{yes}) - P(\text{no}) \log_2 P(\text{no})$$

Where,

- S is the total sample space,
- $P(\text{yes})$ is probability of yes

If number of yes = number of no ie $P(S) = 0.5$

$$\Rightarrow \text{Entropy}(s) = 1$$

If it contains all yes or all no ie $P(S) = 1$ or 0

$$\Rightarrow \text{Entropy}(s) = 0$$

Half Yes, Half No, Entropy = 1

$$E(S) = -P(\text{Yes}) \log_2 P(\text{Yes})$$

When $P(\text{Yes}) = P(\text{No}) = 0.5$ ie YES + NO = Total Sample(S)

$$E(S) = 0.5 \log_2 0.5 - 0.5 \log_2 0.5$$

$$E(S) = 0.5(\log_2 0.5 - \log_2 0.5)$$

$$E(S) = 1$$

[Total Yes] or [Total No] Entropy] =0

$$E(S) = -P(\text{Yes}) \log_2 P(\text{Yes})$$

When $P(\text{Yes}) = 1$ ie YES = Total Sample(S)

$$E(S) = 1 \log_2 1 \quad \text{Value of Log 1 is zero}$$

$$E(S) = 0$$

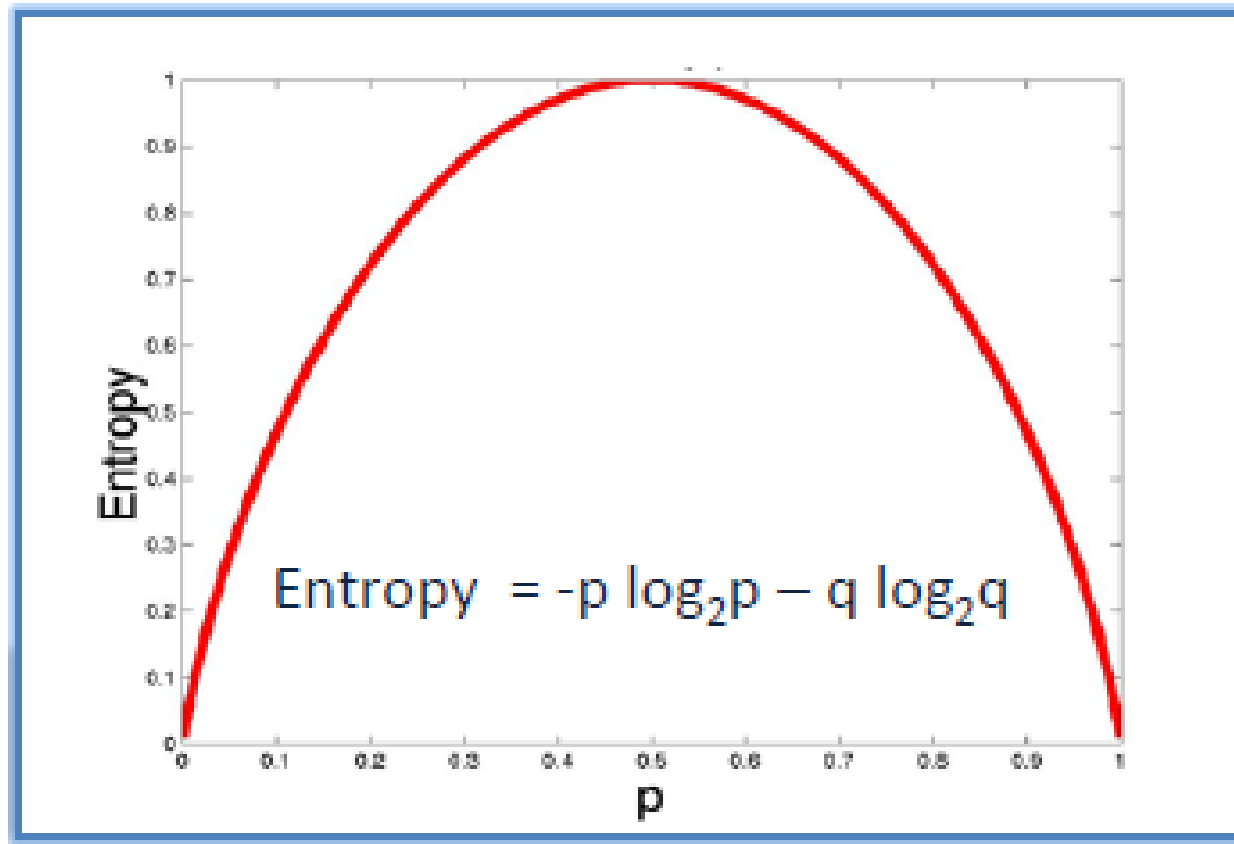
$$E(S) = -P(\text{No}) \log_2 P(\text{No})$$

When $P(\text{No}) = 1$ ie No = Total Sample(S)

$$E(S) = 1 \log_2 1$$

$$E(S) = 0 \quad \text{Value of Log 1 is zero}$$

Entropy



Entropy = 0

Entropy = 1

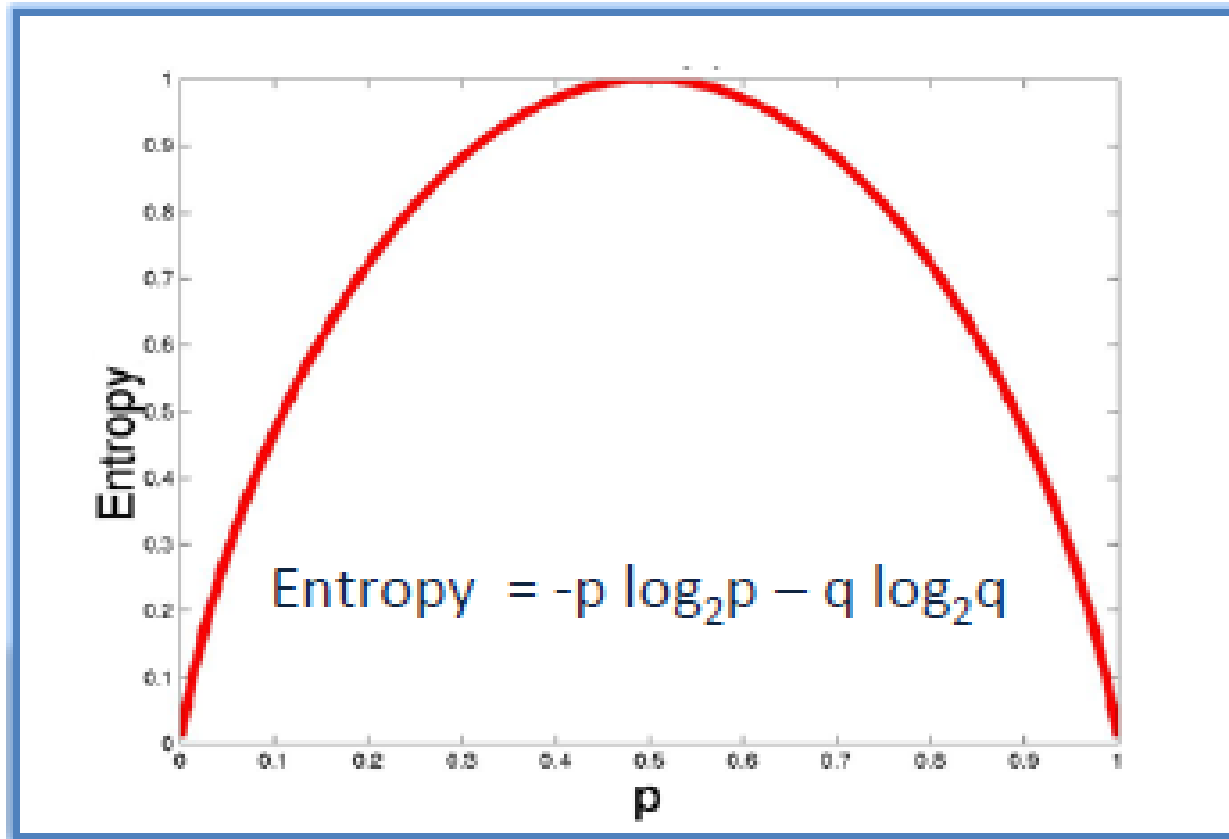
Highly Pure

Data is Pure, Randomness = 0
Data is of the Same Type, All Yes

Highly Impure

Data is Impure, Randomness is high
Data is of the 2 Types, Half Yes + Half No

Entropy



$$\text{Entropy} = -0.5 \log_2 0.5 - 0.5 \log_2 0.5 = 1$$

Information Gain

- Measures the reduction in entropy
- Decides which attribute should be selected as the decision node

If S is our total collection,

Information Gain = Entropy(S) – [(Weighted Avg) x Entropy(each feature)]

Decision Tree – Play Golf

Decision Tree

- To predict if a player will play golf that day based on the weather (Outlook, Temperature, Humidity, Windy).

Predictors

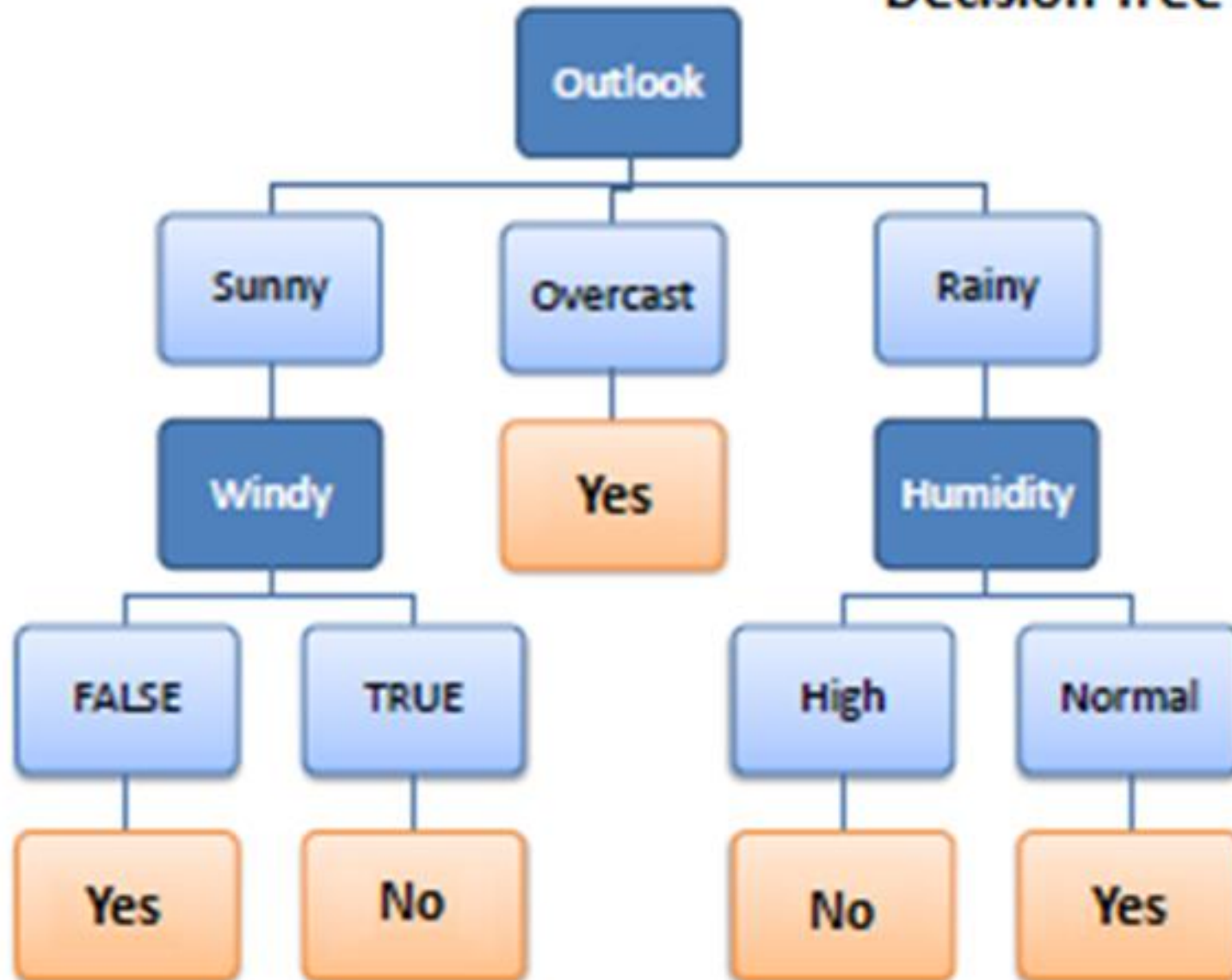
Target

	Outlook	Temp	Humid	Windy	Play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rainy	mild	high	weak	yes
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rainy	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rainy	mild	high	strong	No

	Outlook	Temp	Humid	Windy	Play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rainy	mild	high	weak	yes
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rainy	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rainy	mild	high	strong	No

We need to build a Decision Tree, how to Select the Nodes?

Decision Tree



Weather Data

	Outlook	Temp	Humid	Windy	Play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rainy	mild	high	weak	yes
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rainy	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rainy	mild	high	strong	No

Entropy – Information Gain Steps

Step 1: Calculate entropy of the target (Play Golf) =.94

Step 2: Calculate Entropy and then information gain for each attribute

Step 3: Chose the attribute with the largest information gain as the decision node (Outlook Gain =0.247)

Step 4a: A Branch with Entropy of 0 is a leaf node

Step 4b: A Branch with Entropy of more than 0 needs further splitting

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

Compute Entropy Of Target (Play)

Rows = 14

Out of 14 instances we have 9 YES and 5 NO

So we have the formula,

$$E(S) = -P(\text{Yes}) \log_2 P(\text{Yes}) - P(\text{No}) \log_2 P(\text{No})$$

$$E(S) = - (9/14) * \log_2 9/14 - (5/14) * \log_2 5/14$$

$$E(S) = 0.41 + 0.53 = 0.94$$

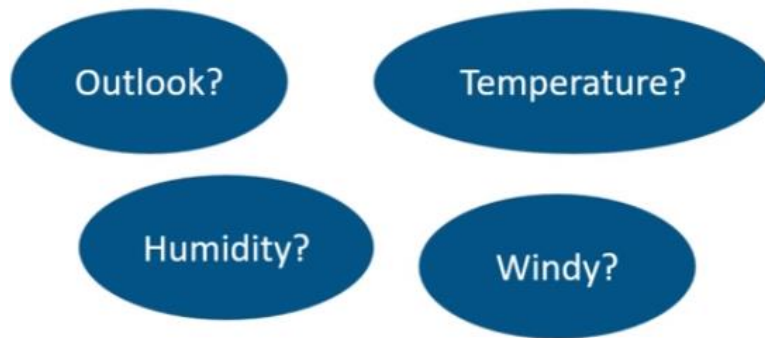
$$E(S) = 0.94$$

Entropy of the entire
dataset = 0.94

	Outlook	Temp	Humid	Windy	Play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rainy	mild	high	weak	yes
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rainy	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rainy	mild	high	strong	No

Step 1: Calculate entropy of the target (Play Golf) = .94

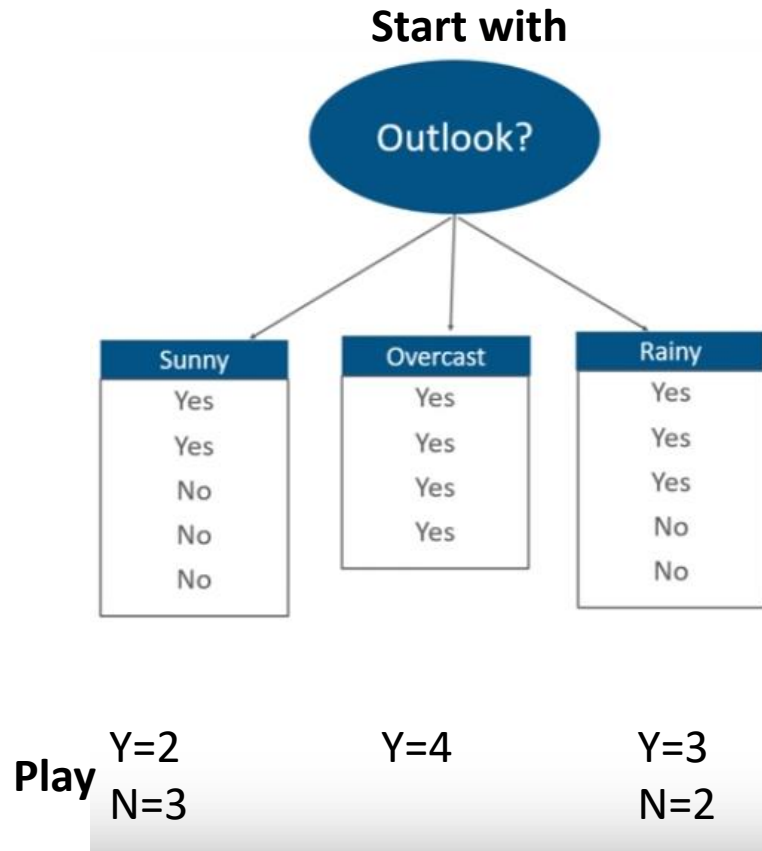
Which Node to select as Root Node



	Outlook	Temp	Humid	Windy	Play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rainy	mild	high	weak	yes
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rainy	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rainy	mild	high	strong	No

Step 2: Calculate information gain for each attribute

Calculated Entropy and Information Gain for Each Node



	Outlook	Temp	Humid	Windy	Play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rainy	mild	high	weak	yes
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rainy	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rainy	mild	high	strong	No

Which Node to select as Root Node:

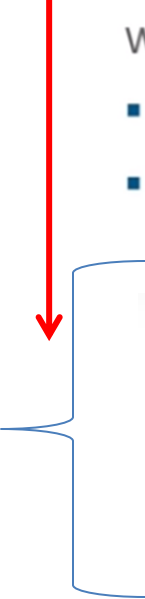
Outlook

$$\text{Entropy}(s) = -P(\text{yes}) \log_2 P(\text{yes}) - P(\text{no}) \log_2 P(\text{no})$$

Where,

- S is the total sample space,
- P(yes) is probability of yes

Let us Consider **Outlook** as Node


$$E(\text{Outlook} = \text{Sunny}) = -2/5 \log_2 2/5 - 3/5 \log_2 3/5 = 0.971$$

$$E(\text{Outlook} = \text{Overcast}) = -1 \log_2 1 - 0 \log_2 0 = 0$$

$$E(\text{Outlook} = \text{Rainy}) = -3/5 \log_2 3/5 - 2/5 \log_2 2/5 = 0.971$$

Step 2: Calculate information gain for each attribute

Information Gain for : Outlook

If S is our total collection,

Information Gain = Entropy(S) – [(Weighted Avg) x Entropy(each feature)]

Weighted Avg (Sunny) = 5/14 , Entropy(Sunny) = .971

Information from outlook,

$$I(\text{Outlook}) = \underset{\text{Sunny}}{5/14} \times 0.971 + \underset{\text{Overcast}}{4/14} \times 0 + \underset{\text{Rainy}}{5/14} \times 0.971 = 0.693$$

Information gained from outlook,

$$\text{Gain}(\text{Outlook}) = E(S) - I(\text{Outlook})$$

$$0.247 = 0.94 - 0.693$$

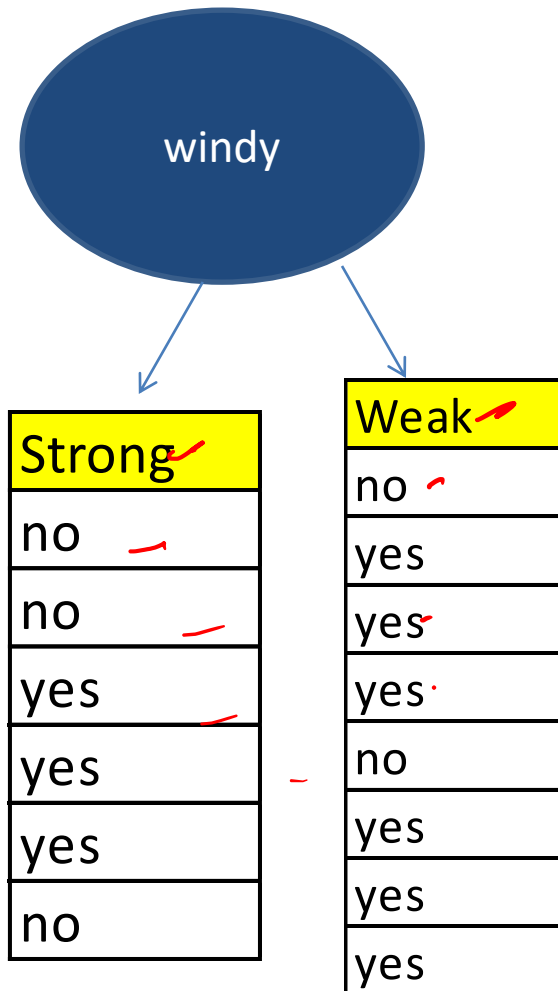


Entropy of the dataset

Step 2: Calculate information gain for each attribute

Which node to select as Root Node ?

Windy



Play
Yes=3
No=3

Yes=6
No=2

	Outlook	Temp	Humid	Windy	Play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rainy	mild	high	weak	yes
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rainy	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rainy	mild	high	strong	No

Step 2: Calculate information gain for each attribute

Information Gain for : Windy

$$E(\text{Windy}=\text{Weak})=0.811$$

$$E(\text{Windy}=\text{Strong})=1$$

Information from Windy:

$$I(\text{Windy}) = \frac{8}{14} \times 0.811 + \frac{6}{14} \times 1 = 0.892$$

Information Gain from Windy:

$$\text{Gain}(\text{Windy}) = E(S) - I(\text{Windy})$$

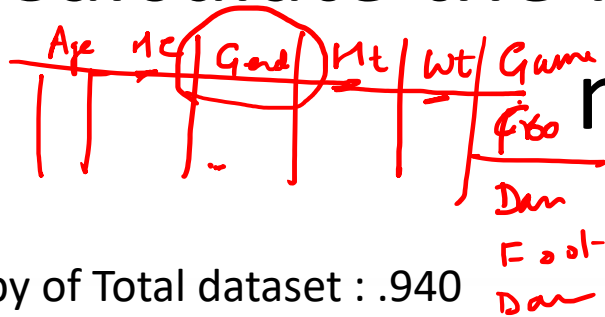
$$0.048 = 0.94 - 0.892$$

0.247 ? ? 0.048 0.94

	Outlook	Temp	Humid	Windy	Play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rainy	mild	high	weak	yes
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rainy	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rainy	mild	high	strong	No

Calculate the Info Gain for all the nodes

Class x
~~100~~
 100



Entropy of Total dataset : .940

Outlook:

Info

Gain: $0.940 - 0.693$

0.693

0.247

Temperature:

Info

Gain: $0.940 - 0.911$

0.911

0.029

Outlook has maximum gain so it is our root node

Humidity:

Info

Gain: $0.940 - 0.788$

0.788

0.152

Windy:

Info

Gain: $0.940 - 0.982$

0.892

0.048



	Outlook	Temp	Humid	Windy	Play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rainy	mild	high	weak	yes
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rainy	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rainy	mild	high	strong	No

Step 2: Calculate information gain for each attribute

Select the Node with highest Info Gain: Outlook

Entropy of Total dataset : .940

Outlook:

Info

Gain: $0.940 - 0.693$

0.693

0.247

Temperature:

Info

Gain: $0.940 - 0.911$

0.911

0.029

Outlook has maximum gain so it is our root node

Humidity:

Info

Gain: $0.940 - 0.788$

0.788

0.152

Windy:

Info

Gain: $0.940 - 0.982$

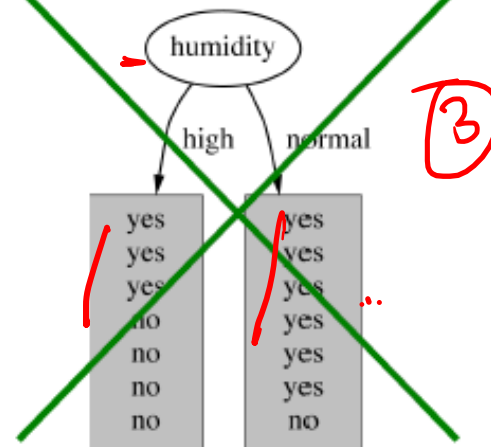
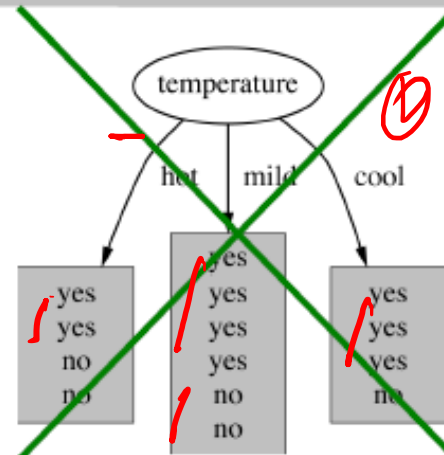
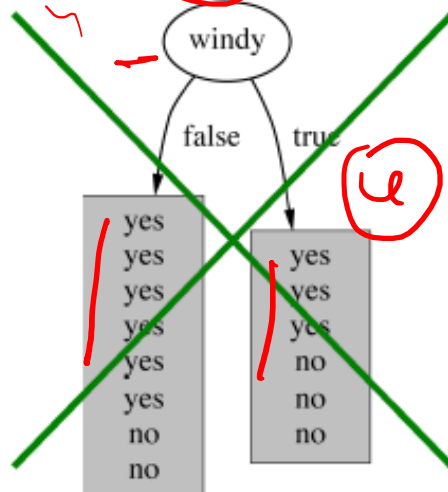
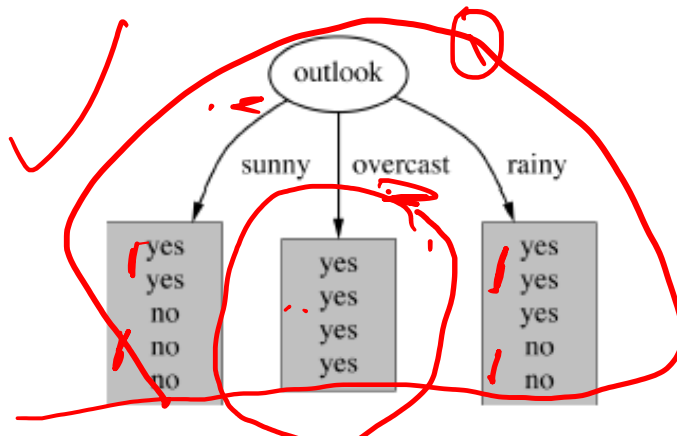
0.892

0.048

	Outlook	Temp	Humid	Windy	Play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rainy	mild	high	weak	yes
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rainy	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rainy	mild	high	strong	No

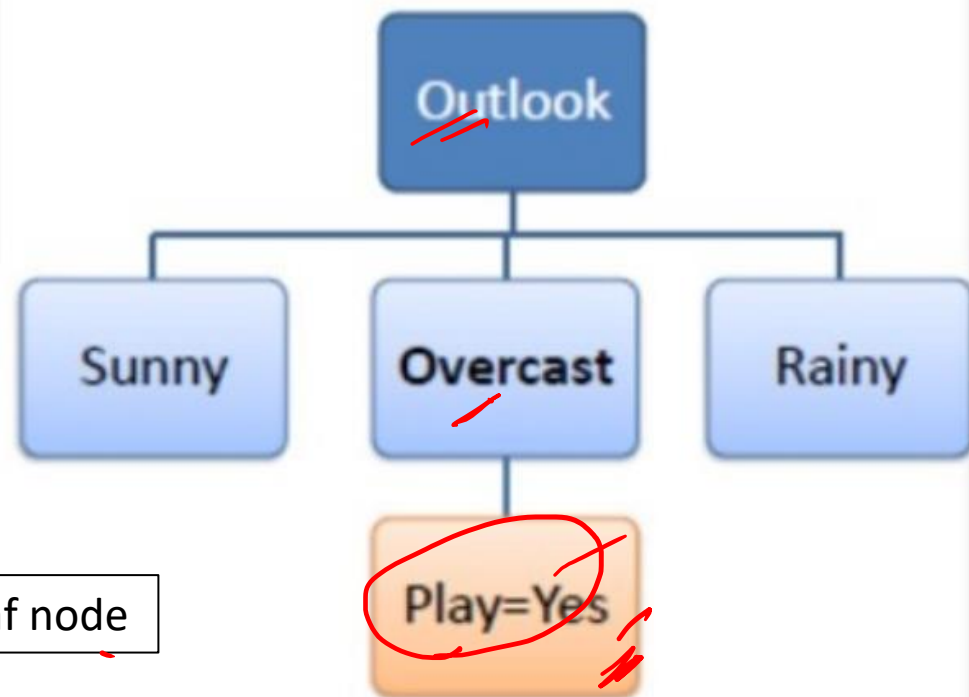
Step 3: Chose the attribute with the largest information gain as the decision node (Outlook Gain =0.247)

Which attribute to select as the root?



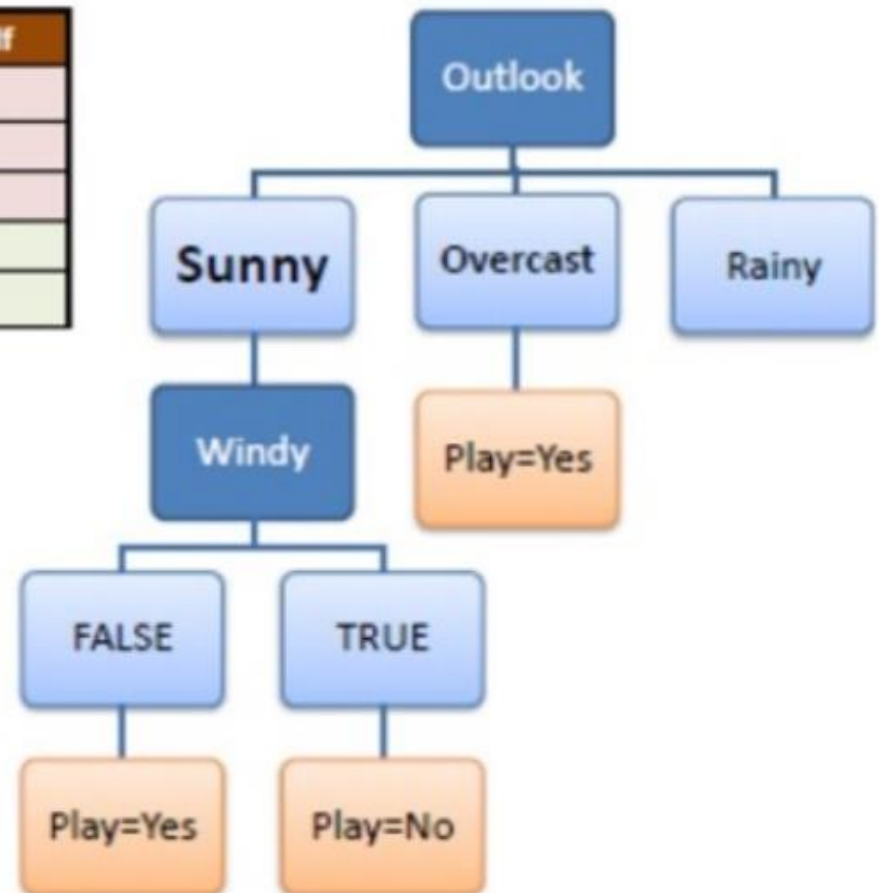
Select the node that gives more accurate answers

Temp.	Humidity	Windy	Play Golf
Hot	High	FALSE	Yes
Cool	Normal	TRUE	Yes
Mild	High	TRUE	Yes
Hot	Normal	FALSE	Yes
Hot	High	FALSE	Yes



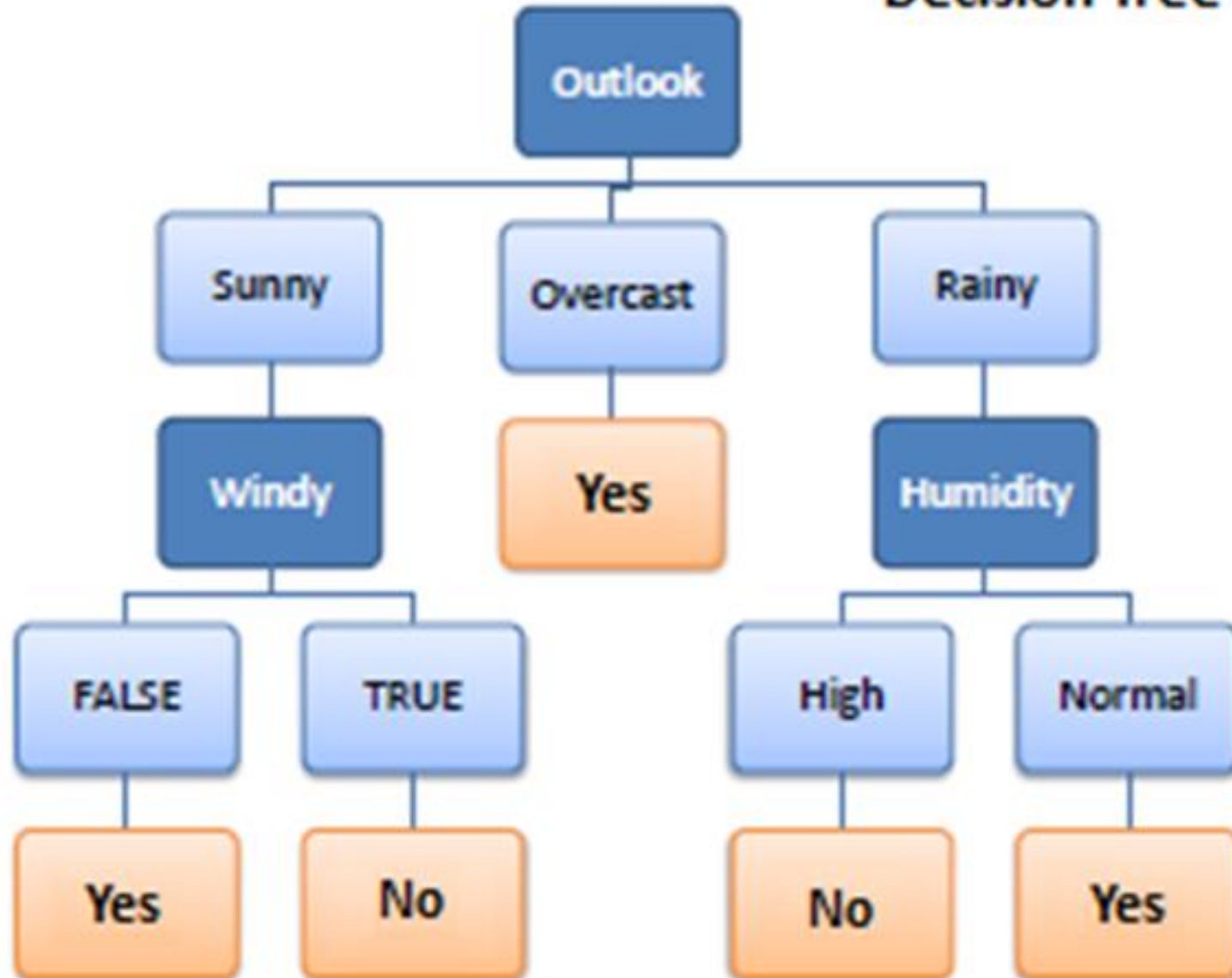
Step 4a: A Branch with Entropy of 0 is a leaf node

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



Step 4b: A Branch with Entropy of more than 0 needs further splitting

Decision Tree



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

End

Label Encoding

Food Name	Categorical #	Calories
Apple	1	95
Chicken	2	231
Broccoli	3	50

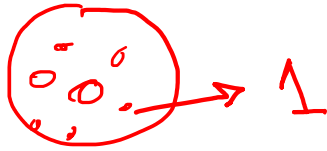


One Hot Encoding

Apple	Chicken	Broccoli	Calories
1	0	0	95
0	1	0	231
0	0	1	50

Gini Index

	Outlook	Temp	Humid	Windy	Play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rainy	mild	high	weak	yes
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rainy	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rainy	mild	high	strong	No



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”. (0 or 1) (Yes or No)
- 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.



$$p=1$$

$$q^2$$

$$p^2 + q^2$$

Calculate Gini for a split

- Calculate Gini for sub-nodes, using formula sum of square of probability for success and failure ($p^2 + q^2$).
- Calculate Gini for split using weighted Gini score of each node of that split

Gini index measures the impurity of a data partition K , formula for Gini Index can be written down as:

$$Gini(K) = 1 - \sum_{i=1}^n P_i^2$$

Where n is the number of classes, and P_i is the probability that an observation in K belongs to the class.

Gini Index assumes a binary split for each of the attribute in S , let say T_1 & T_2 . The Gini index of K given this partitioning is given by:

$$Gini_S(K) = \frac{T_1}{T} Gini(T_1) + \frac{T_2}{T} Gini(T_2)$$

Calculate Gini Index

So as the first step we will find the root node of our decision tree. For that Calculate the gini index of the class variable (Play)

- $\text{Gini}(S) = 1 - [(9/14)^2 + (5/14)^2] = 0.4591$
- $\text{Gini}(S) = 1 - [(p)^2 + (q)^2] = 0.4591$
- $p = \text{success} = \text{yes}$
- $q = \text{failure} = \text{no}$

Gini Gain

we will calculate gini gain. For that first, we will find the average weighted gini impurity of Outlook, Temperature, Humidity and Windy.

Gini Gain - Outlook

		play		
		yes	no	total
Outlook	sunny	3	2	5
	overcast	4	0	4
	rainy	2	3	5
				14

$$\begin{aligned}
 \text{Gini}(S, \text{outlook}) &= (5/14)\text{gini}(3,2) + (4/14)*\text{gini}(4,0) + (5/14)*\text{gini}(2,3) \\
 &= (5/14)(1 - [(3/5)^2 + (2/5)^2]) + (4/14)*0 + (5/14)(1 - [(2/5)^2 + (3/5)^2]) \\
 &= 0.171 + 0 + 0.171 = 0.342
 \end{aligned}$$



$$\text{Gini gain}(S, \text{outlook}) = 0.459 - 0.342 = 0.117$$

Gini Index : Play

Gini Gain – For all Predictors

Gini gain (S, outlook) = $0.459 - 0.342 = 0.117$

Gini gain(S, Temperature) = $0.459 - 0.4405 = 0.0185$

Gini gain(S, Humidity) = $0.459 - 0.3674 = 0.0916$

Gini gain(S, windy) = $0.459 - 0.4286 = 0.0304$

Choose one that having higher gini gain.

Gini gain is higher for outlook .So we can choose it as our root node.

Split on Which Column?

- Above, you can see that Gini score for *Split on Outlook* (**0.117**) is higher than *Split on other Classes(Columns)* hence, the node split will take place on Outlook

Cricket Example for Gini

there are 4 entries.

Steps to Calculate Gini for a split

- Calculate Gini for sub-nodes, using formula sum of square of probability for success and failure (p^2+q^2).
- Calculate Gini for split using weighted Gini score of each node of that split

Decision Tree for Cricket

- **Example:** — Referring to example where we want to segregate the students based on target variable (playing cricket or not). In the snapshot below, we split the population using two input variables Gender and Class. Now, I want to identify which split is producing more homogeneous sub-nodes using Gini index.

Split on Gender

Students = 30
Play Cricket = 15 (50%)



Female



Students = 10
Play Cricket = 2 (20%)

Male



Students = 20
Play Cricket = 13 (65%)

Split on Class



Class IX



Students = 14
Play Cricket = 6 (43%)

Class X



Students = 16
Play Cricket = 9 (56%)

Split on Gender or Class ?

- Above, you can see that Gini score for *Split on Gender* is higher than *Split on Class*, hence, the node split will take place on Gender.

Gini Vs Entropy

Gini Vs Entropy

- The Gini Index is calculated by subtracting the sum of the squared probabilities of each class from one. It favours larger partitions.
- Information Gain multiplies the probability of the class times the log (base=2) of that class probability. Information Gain favors smaller partitions with many distinct values.
- Ultimately, you have to experiment with your data and the splitting criterion.

End

Adding Chart Packages Adding Packages to Jupyter

Install pydot

- `import sys`
`!conda install --yes --prefix {sys.prefix}`
`pydotplus`
- Example of using Matplotlib lib:
- <https://mljar.com/blog/visualize-decision-tree/>

Installing Graphviz

Option 1

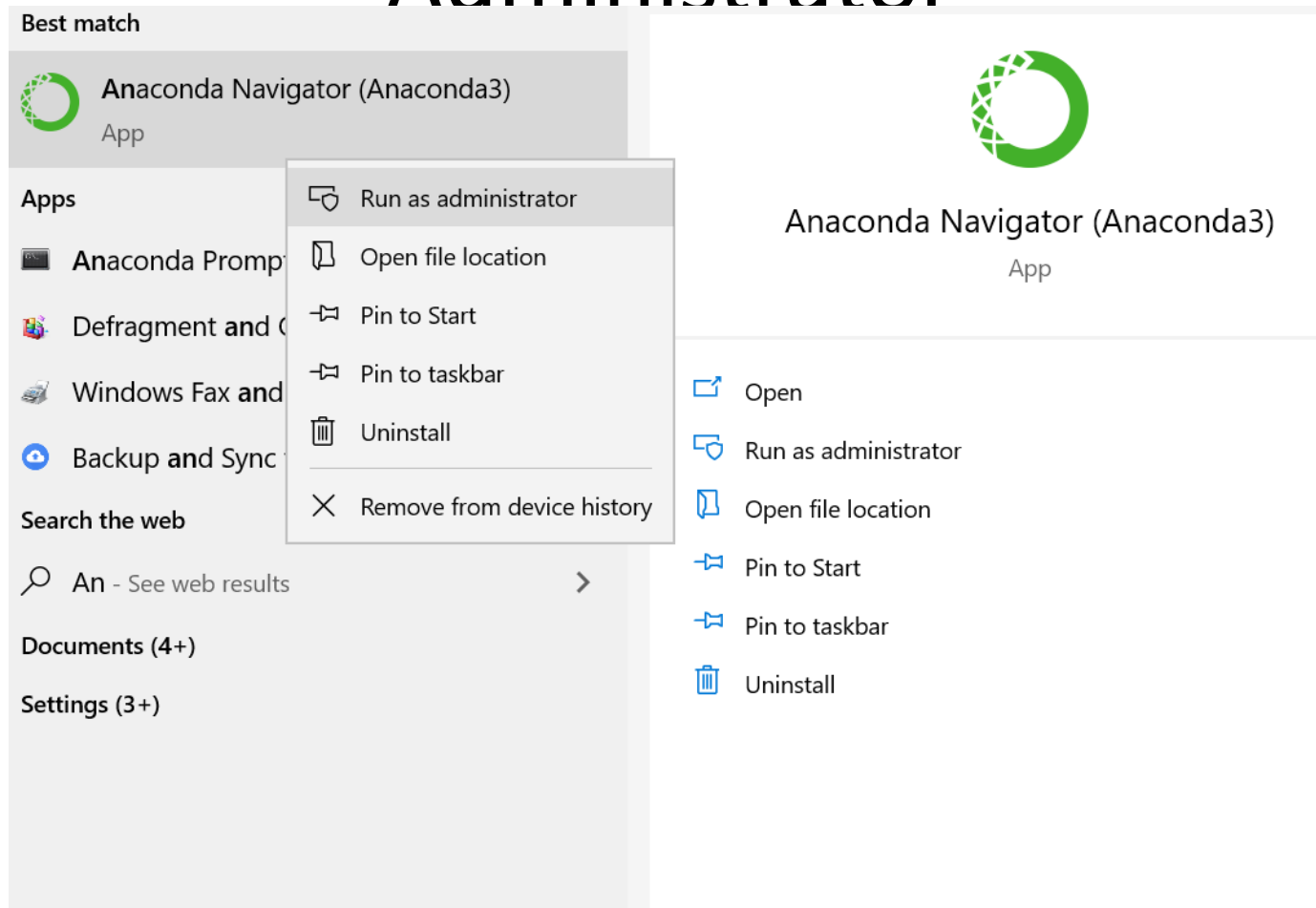
- Anaconda Prompt
- Or

Option 2 Anaconda Navigator>Terminal

conda install -c anaconda graphviz

Option 3(Recommended)

Run Anaconda Navigator as Administrator



Update the index

CONDA NAVIGATOR

Sign in to Anaconda Cloud

All





▼

Channels

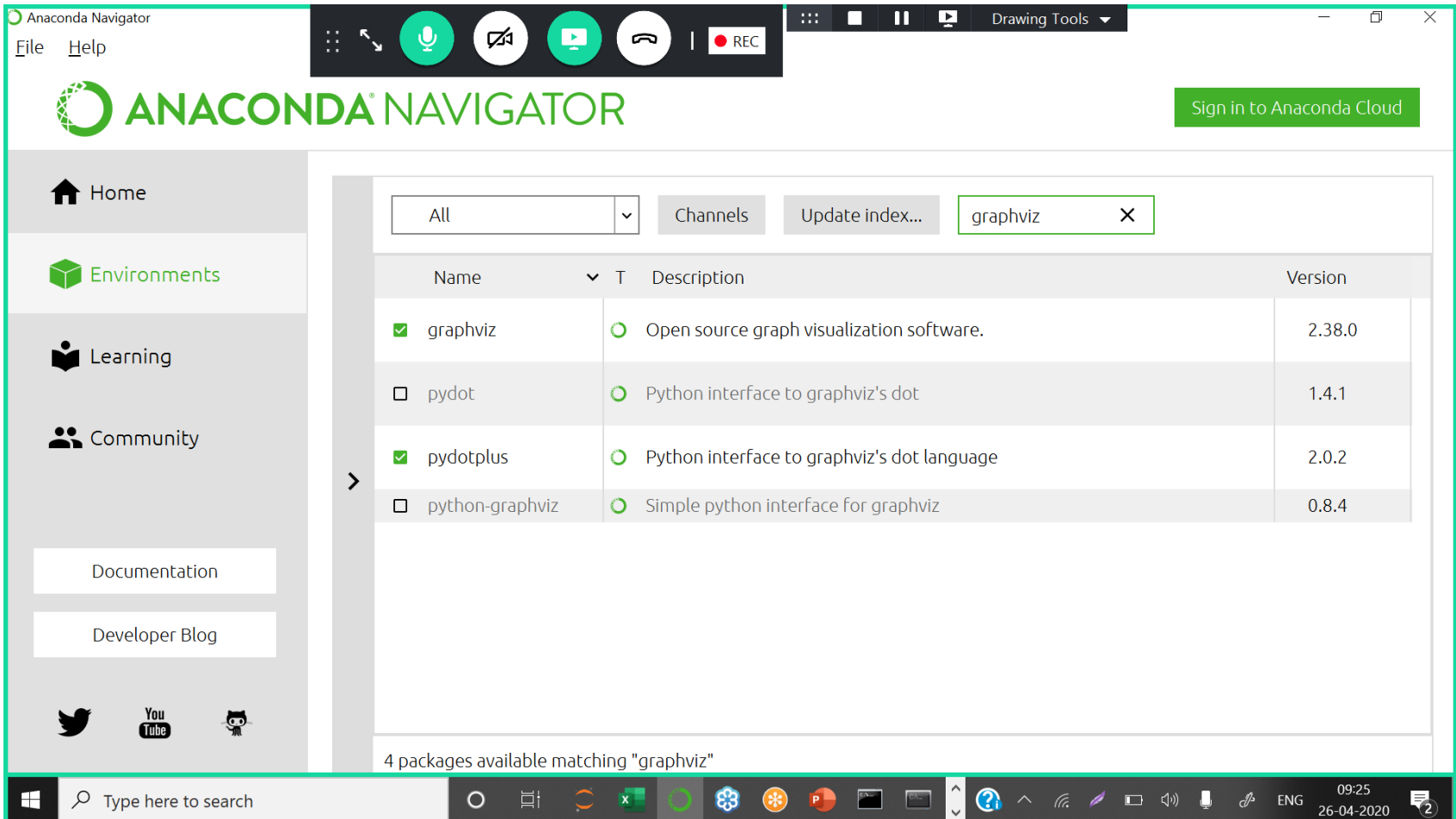
Update index...

graphviz

×

Name	▼ T	Description	Version
<input checked="" type="checkbox"/> graphviz		Open source graph visualization software.	2.38.0
<input type="checkbox"/> pydot		Python interface to graphviz's dot	1.4.1
<input checked="" type="checkbox"/> pydotplus		Python interface to graphviz's dot language	2.0.2
<input type="checkbox"/> python-graphviz		Simple python interface for graphviz	0.8.4

Install graphviz from Navigator



The screenshot shows the Anaconda Navigator application interface. The top bar includes the Anaconda Navigator logo, a 'Sign in to Anaconda Cloud' button, and a search bar. The left sidebar contains navigation links: Home, Environments, Learning, and Community. The main panel displays a search results table for the query 'graphviz'. The table has columns for Name, Description, and Version. The results show four packages: graphviz (version 2.38.0), pydot (version 1.4.1), pydotplus (version 2.0.2), and python-graphviz (version 0.8.4). The 'graphviz' and 'pydotplus' rows are checked. The bottom status bar indicates '4 packages available matching "graphviz"'. The Windows taskbar is visible at the bottom.

Anaconda Navigator

File Help

ANACONDA NAVIGATOR

Sign in to Anaconda Cloud

Home

Environments

Learning

Community

Documentation

Developer Blog

graphviz

Name	Description	Version
<input checked="" type="checkbox"/> graphviz	Open source graph visualization software.	2.38.0
<input type="checkbox"/> pydot	Python interface to graphviz's dot	1.4.1
<input checked="" type="checkbox"/> pydotplus	Python interface to graphviz's dot language	2.0.2
<input type="checkbox"/> python-graphviz	Simple python interface for graphviz	0.8.4

4 packages available matching "graphviz"

Type here to search

09:25 26-04-2020

Option 4

- Uninstall Anaconda
- Download a **New Anaconda Installation**
- Try all the steps again
- As a User (with Administrator rights) of the Laptop/Computer and then do all the installations

K-Fold is another method used for
Splitting data Training and Test data

Here for the 2nd Random forest
example we are using the K-Fold
method

K-fold

