Decision Tree

A Decision Tree is a Supervised learning algorithm that can be used for both classification and Regression problems, but most of the time it's preferred for solving a classification problem. A Decision tree is a flowchart-l**ike tree structure**, where internal nodes represent the feature of a dataset, branches represent the decision rules and each leaf node represents the outcome.

The Purpose of the Decision Tree is to create a training modal that can be used to predict the class or value of a target variable by learning simple decision rules inferred from training data.

Decision Tree Terminologies

Root node: The root node is from where the decision tree starts, it represents the entire population or simply and this further gets divided into two or more homogeneous sets.

Leaf Node: Final output nodes are called leaf nodes, and a tree cannot be split further after getting a leaf node.

Splitting: the process of making a node into two or more sub-node.

Branch/ Sub-Tree: a subtree of the main tree is called a branch or sub-tree.

Pruning: It is the process of reducing the unwanted branches from the tree.

Parent/Child Node: A node that is divided into a sub-node is called the parent node of sub-nodes whereas sub-nodes are called a child of a parent node.

Decision Tree algorithm working Procedure:

Decision trees use various algorithms to decide the root and to split a node into sub-nodes.

i.e. **ID3(Interactive Dichotomiser 3)** Algorithm use of deciding root nodes and splitting a node into a sub node.

If we want to learn the ID3 algorithm we need to know **Entropy (H)** and **Gain (G)**

Entropy:

In data science, entropy is used as a way to measure how “mixed” a column is. Specifically, entropy is used to measure disorder.

Through entropy, we understand how well partitioning a data set can partition out the target variable.

Partitioning means taking different values of the target variable for different values of a feature.

The formula of Entropy:

H(S) = $-\left(P_{i+} log_2 P_{i+} + P_{i-} log_2 P_{i-}\right)$

Where, H(S) is used to find out the entropy of the current dataset

H(S) = Entropy of the current/main dataset

P_{i+}= Probability of Positive (Yes) Class in S

P_{i-}- = Probability of Negative (No) Class in S

Information Gain:

Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an **attribute**. It calculates how much information a feature provides us about a class.

We will select as the root who has more gain than others.

The formula of Gain:

Gain (S, F) = $H(S) - \sum_{v \in F} P(v) \times H(S_v)$

S = Target dataset

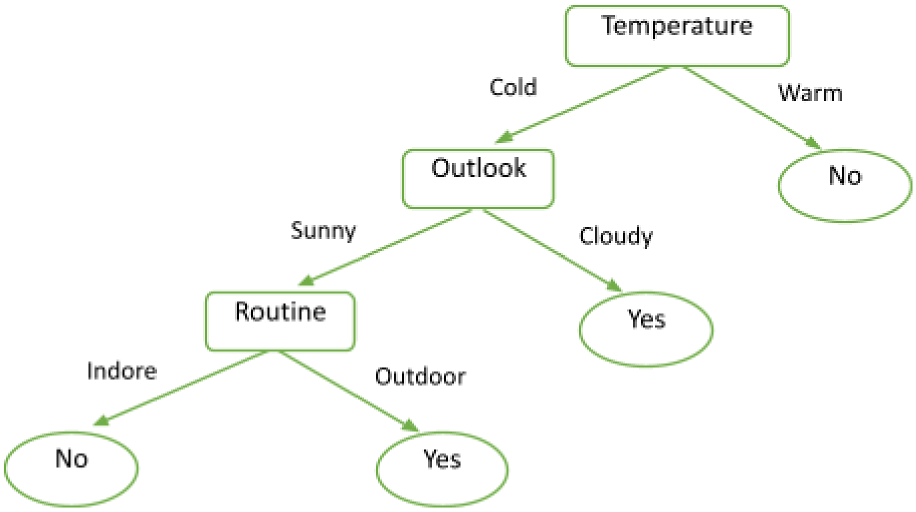
F = Feature

$v \in F$= Each value of Feature F

$H(S_v)$ = Entropy for selected attribute of feature F

$P(v)$ = Probability of selected attribute of feature F

Process flow chart:



Application of Decision Tree:

1. **Healthcare industries:** In healthcare industries, decision tree can tell whether a patient is suffering from a disease or not based on his weight, sex, age, and another factor.
2. **Educational sector:** In school, college, university, a student eligible for a scholarship or not based on result, financial status, family income, etc. can be decided on a decision tree.

3. **Banking sector:** A person eligible for a loan or not based on his salary, family member or financial status, etc. it can be decided on a decision tree.,

Research paper based on Decision Tree:

1. Research on anti-money laundering based on core decision tree algorithm
   Published in: 2011 Chinese Control and Decision Conference (CCDC)

2. An improved ID3 decision tree algorithm
   Published in: 2009 4th International Conference on Computer Science & Education

3. Research on incremental decision tree algorithm
   Published in: Proceedings of 2011 International Conference on Electronic & Mechanical Engineering and Information Technology

4. A comparative study of Reduced Error Pruning method in decision tree algorithm
   Published in: 2012 IEEE International Conference on Control System, Computing and Engineering

5. Predicting Student Performance in Higher Education Institutions Using Decision Tree Analysis
   Published in: International Journal of Interactive Multimedia and Artificial Intelligence 5.2 (2018)

Example:

Let’s consider the dataset named **S** in the table below.

Day	Outlook	Temperature	Routine	Play
Day1	Sunny	Cold	Indoor	No
Day2	Sunny	Warm	Outdoor	No
Day3	Cloudy	Warm	Indoor	No
Day4	Sunny	Warm	Indore	No
Day5	Cloudy	Cold	Indore	Yes
Day6	Cloudy	Cold	Outdoor	Yes
Day7	Sunny	Cold	Outdoor	Yes

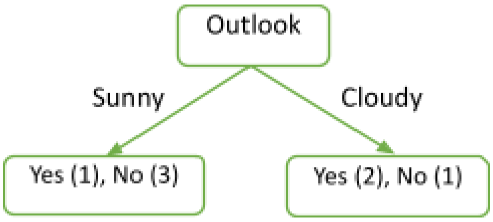
Here, *we have 3 features: Outlook, Temperature, and Routine*. We will make a separate partition and calculate the different gains for these three different features. We will take the one who has more gain as the root.

Before partitioning, Entropy is H(S) = $-\ (P_{i+}log_2P_{i+} + P_{i-}log_2P_{i-})$

= - ($\frac{3}{7}log_2\frac{3}{7} + \frac{4}{7}log_2\frac{4}{7}) = 0.985$ [- (-.985)]

[**Explanation**: Here, The dataset **S has 7 rows** and target columns(Play) has 3 Positive(**Yes**) and 4 Negative(**No**) classes so probability of positive(Yes) class P_{i+} = 3/7 and probability of Negative(No) class P_{i-} = 4/7]

For Outlook Feature:



H(Outlook_Sunny) = $-\ (P_{i+}log_2P_{i+} + P_{i-}log_2P_{i-})\ =$ **- (**$\frac{1}{4}log_2\frac{1}{4} + \frac{3}{4}log_2\frac{3}{4}$**)** $=$ **0.811**

[**Explanation**: Here, Outlook has 4 sunny , and sunny has 1 positive(Yes) target value, so the probability of sunny(Yes) is ¼, and 3 negative (No) target value, so the probability of sunny(No) is ¾ . See the above dataset]

H(Outlook_Cloudy) = $-\ (P_{i+}log_2P_{i+}\ +\ P_{i-}log_2P_{i-})\ \ =$ - $(\frac{2}{3}log_2\frac{2}{3}\ +\ \frac{1}{3}log_2\frac{1}{3}) = 0.918$

[Explanation: Here, Outlook has 3 cloudy, and cloudy has 2 positive(Yes) target values, so the probability of cloudy(Yes) is ⅔ and 1 negative (No) target value, so the probability of cloudy(No) ⅓. See the above dataset**]**

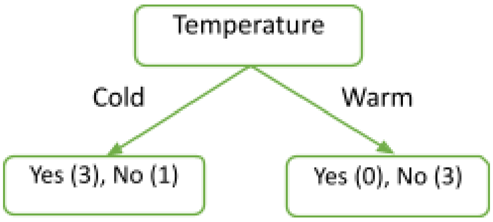
Gain (S, Outlook) = $H(S)\ -\ \sum_{v\,\epsilon\,F} P(v)\times H(S_v)$ **=** H (S)- $\frac{4}{7}\times 0.811\ -\ \frac{3}{7}\times 0.918$

= 0.985 $-\ \frac{4}{7}\times 0.811\ -\ \frac{3}{7}\times 0.918 =$ **0.128** **[Gain for Outlook Feature]**

[Explanation: Outlook has two attributes sunny and cloudy, there are probability of attribute sunny is 4/7 and Entropy of sunny is .811, and probability of cloudy is 3/7 and entropy 0.918**]**

Similarly:

For Temperature Feature:

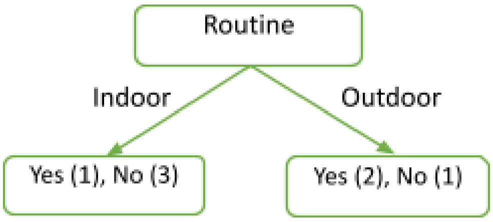


H(Temperature_Cold) = - $(\frac{3}{4}log_2\frac{3}{4}\ +\ \frac{1}{4}log_2\frac{1}{4}) = 0.811$

H(Temperature_Warm) = - $(\frac{0}{3}log_2\frac{0}{3}\ +\ \frac{3}{3}log_2\frac{3}{3}) = 0$

Gain (S, Temperature) = H (S) - $(\frac{4}{7}\times 0.811\ +\ \frac{3}{7}\times 0)\ = 0.985 - (\frac{4}{7}\times 0.811\ +\ \frac{3}{7}\times 0) = 0.521$

For Routine Feature:



H(Routine_Indoor) = - $(\frac{1}{4}log_2\frac{1}{4}\ +\ \frac{3}{4}log_2\frac{3}{4}) = 0.811$

H(Routine_Outdoor) = - $(\frac{2}{3}log_2\frac{2}{3}\ +\ \frac{1}{3}log_2\frac{1}{3}) = 0.918$

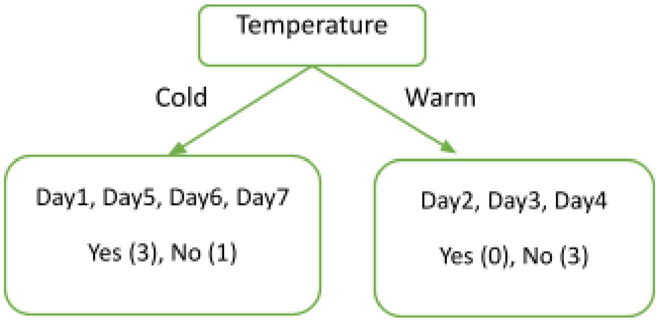
Gain (S, Outlook) = H (S) - $(\frac{4}{7}\times 0.811\ +\ \frac{3}{7}\times 0.918)$

$$= 0.985 - (\frac{4}{7}\times 0.811 + \frac{3}{7}\times 0.918) = 0.128$$

Summary of gain for dataset S:

Gain	**Outlook**	**Temperature**	**Routine**
	0.128	0.521	0.128

From the above table, we see that the temperature gain is the highest. So, we will select the temperature feature as a root node. So, we found a partial Decision Tree.



Here, The Dataset is divided into two subsets. One of these (right one) entropies always will be zero, in this node, target column(Play) will always be No, so we don't need this one. Again, we have to do partitioning with the left one's dataset (day1, day5, day6, day7). **Consider this new subset as S1.**

Day	Outlook	Temperature	Routine	Play
Day1	Sunny	Cold	Indoor	No
Day5	Cloudy	Cold	Indore	Yes
Day6	Cloudy	Cold	Outdoor	Yes
Day7	Sunny	Cold	Outdoor	Yes

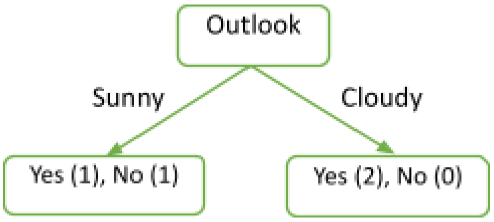
Now assume the **S1** subset as the **root** dataset, and do partitioning as previous. We already used temperature, so now we will be partitioning **Outlook** and **Routine**.

[If any feature is used, this feature will not be considered again.]

The entropy of S1 is **H(S1) =** $-(P_{i+}log_2P_{i+} + P_{i-}log_2P_{i-})$

$$= \quad -\left(\frac{3}{4}log_2\frac{3}{4} + \frac{1}{4}log_2\frac{1}{4}\right) = 0.811$$

For outlook feature:



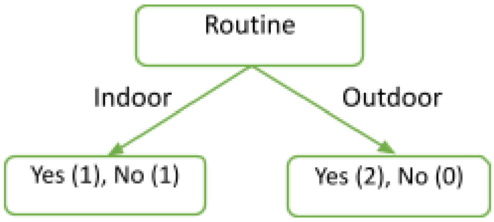
H(Outlook_Sunny) = - $(\frac{1}{2}log_2\frac{1}{2} + \frac{1}{2}log_2\frac{1}{2}) = 1$

H(Outlook_Cloudy) = - $(\frac{2}{2}log_2\frac{2}{2} + \frac{0}{2}log_2\frac{0}{2})$ =0

Gain (S, Outlook) = H (S)- $(\frac{2}{4}\times1 + \frac{2}{4}\times0)$

= 0.985 $- (\frac{2}{4}\times1 + \frac{2}{4}\times0) = 0.311$

For Routine Feature:



H(Routine_Indoor) = - $(\frac{1}{2}log_2\frac{1}{2} + \frac{1}{2}log_2\frac{1}{2}) = 1$

H(Routine_Outdoor) = - $(\frac{2}{2}log_2\frac{2}{2} + \frac{0}{2}log_2\frac{0}{2})$ =0

Gain (S, Outlook) = H (S)- $(\frac{2}{4}\times1 + \frac{2}{4}\times0)$

= 0.985 $- (\frac{2}{4}\times1 + \frac{2}{4}\times0) = 0.311$

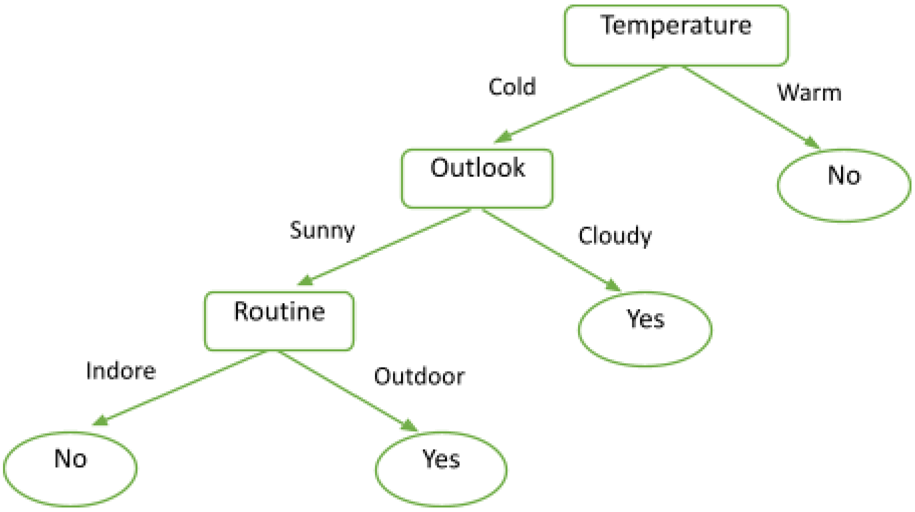
Summary of gain for dataset S:

Gain	**Outlook**	**Routine**
	0.311	0.311

From the above table, we see that the gain of **Outlook** and **Routine** are the same. So, We can select any one as a node for the decision tree.

Now one feature left is **Routine,** So we don't need to calculate like above.

The final Decision tree given in the image below:



Implementation of this example using Python (Jupyter notebook):

Step 1: Read dataset through pandas

In [13]:

```
import pandas as pd
dataset = pd.read_csv('Weather_Condition_vs_Play.csv')
dataset
```

Out[13]:

Day	Outlook	Temperature	Routine	Play	
0	Day1	Sunny	Cold	Indoor	No
1	Day2	Sunny	Warm	Outdoor	No
2	Day3	Cloudy	Warm	Indoor	No
3	Day4	Sunny	Warm	Indoor	No
4	Day5	Cloudy	Cold	Indoor	Yes
5	Day6	Cloudy	Cold	Outdoor	Yes
6	Day7	Sunny	Cold	Outdoor	Yes

Step 2: Check missing value, if any then handle it

In [14]:

```
dataset.isnull().sum()
```

Out[14]:

```
Day            0
Outlook        0
Temperature    0
Routine        0
Play           0
dtype: int64
```

see , there is no null value.

Step 3: Data preprocessing,

Machine can not work with string value so, we need to convert String to numeric value.that's why preprocessing is required

Many ways to preprocess data, one of these label encoding from sklearn

In [15]:

```
from sklearn.preprocessing import LabelEncoder

le_x = LabelEncoder()
x =
dataset[['Outlook','Temperature','Routine']].apply(LabelEncoder().fit_transfo
rm)
x
```

Out[15]:

	Outlook	Temperature	Routine
0	1	0	0
1	1	1	1
2	0	1	0
3	1	1	0
4	0	0	0
5	0	0	1
6	1	0	1

step 4: create decisionTreeClassifier modal and train it

In [16]:

```
from sklearn.tree import DecisionTreeClassifier

modal = DecisionTreeClassifier()
modal.fit(x, dataset.Play)
```

Out[16]:

```
DecisionTreeClassifier()
```

step 5: predict data using some data

In [20]:

```
import numpy as np
x_test = np.array([1,0,1]) # 1-> Sunny, 0-> Cold, 1-> Outdoor; according to
preprocessed table

modal.predict([x_test])[0]
```

Out[20]:

```
'Yes'
```

result is Yes