K-NN and Decision Trees

- K - Nearest Neighbours (KNN)
- Decision Trees
- Criteria for Splitting Nodes in Decision Tree
- Advantages and Disadvantages of Decision Tree

K - Nearest Neighbour

It is a Supervised Learning algorithm used for both classification and regression and one of the simplest.
It is basically **“Tell me about your friends/neighbours and I’ll tell who you are”**.

This technique implements classification by considering the majority of vote among the **“k-closest points”** to the unlabeled data point.

Properties:

1. **Lazy-Learning Algorithm -** It doesn’t learn anything from the training data, it just memorizes it.
2. **Non-parametric Learning Algorithm -** It doesn’t assume anything about the underlying data or make any assumptions about it.

How do we consider closest?

Distance functions

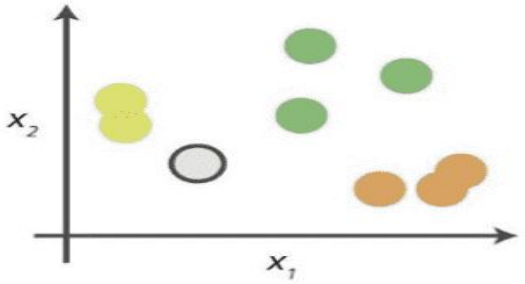
Euclidean $$\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$$

Manhattan $$\sum_{i=1}^{k}|x_i - y_i|$$

Minkowski $$\left(\sum_{i=1}^{k}(|x_i - y_i|)^q\right)^{1/q}$$

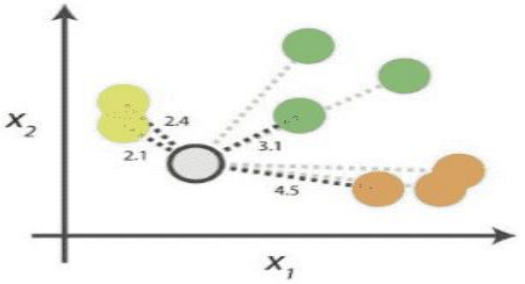
Generally Euclidean or Manhattan distance is used for calculating distance between the data points.

0. Look at the data



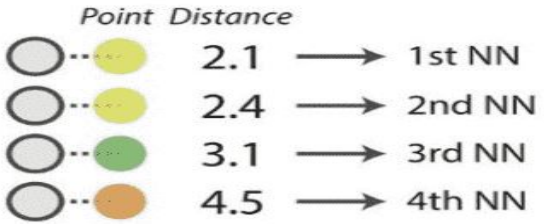
Say you want to classify the grey point into a class. Here, there are three potential classes - lime green, green and orange.

1. Calculate distances



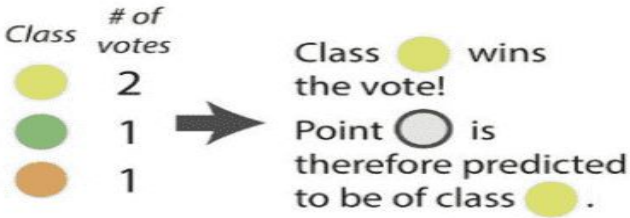
Start by calculating the distances between the grey point and all other points.

2. Find neighbours



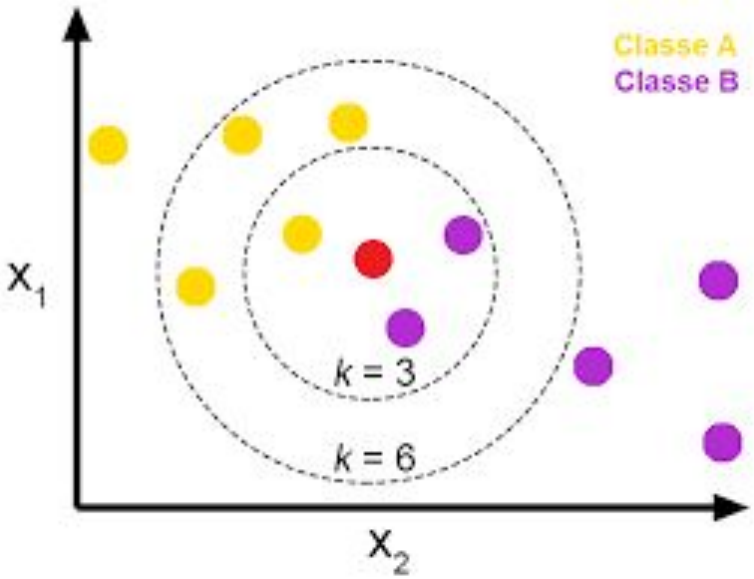
Next, find the nearest neighbours by ranking points by increasing distance. The nearest neighbours (NNs) of the grey point are the ones closest in dataspace.

3. Vote on labels



Vote on the predicted class labels based on the classes of the k nearest neighbours. Here, the labels were predicted based on the k=3 nearest neighbours.

How to choose K?



Red circle is the unlabeled data point.

When k=3

- Closest 3 points are taken
- 2 are purple, 1 is yellow
- By majority vote, red circle is Class B.

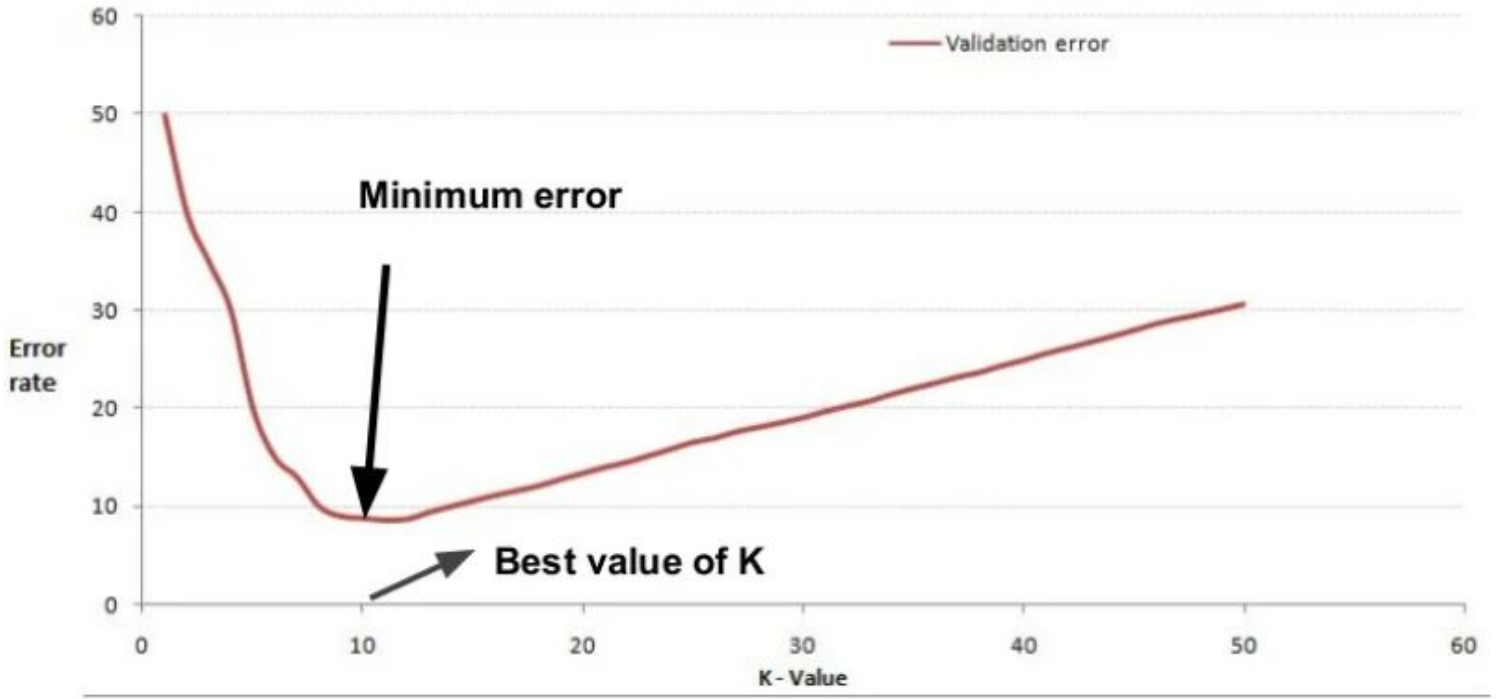
When k=6

- Closest 6 points are taken
- 2 are purple, 4 are yellow
- By majority vote, red circle is Class A.

Then how do we choose K?

1. If K is small, it is too sensitive to noise/outliers.
2. If K is large, it may include majority points from other classes.

K- value vs Validation Error Curve (Elbow Method)



Advantages of KNN:

1. Very easy to Implement as it only needs k value and a distance function.
2. No training is required as it makes predictions from the training dataset directly (Lazy-Learning) making it faster.

Disadvantages of KNN:

1. Computationally Expensive and does not work well with large dataset.
2. Lot of space is consumed as all the training data points are stored.
3. Sensitive to noisy data, missing values or outliers.

Applications of KNN:

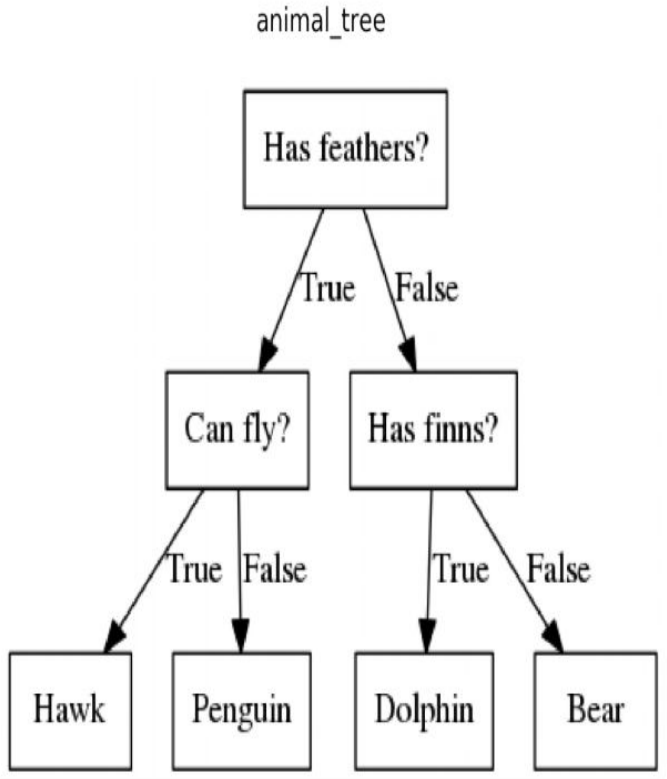
1. Recommendation Systems
2. Pattern Recognition
3. Banking Systems (Credit Score, Loan Approval)

Decision Trees

A decision tree builds the classification or regression model in the form of a tree structure. It is basically a **“Nested If-else condition classifier”.**

Each **node represents a feature(attribute),**
Each **branch represents a choice,**
Each **leaf represents a decision.**

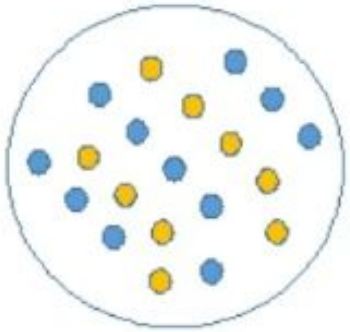
It breaks down the dataset into smaller subs with increase in depth of tree.
Final result is a tree with **decision node** and **leaf nodes.**



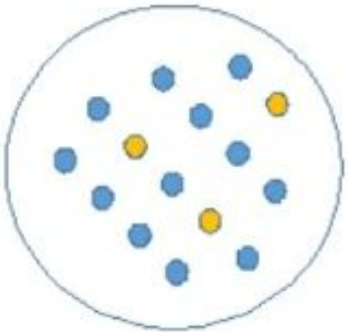
Something about Decision Tree

- While K-NN is an instance-based method and Linear/Logistic Regression is a geometric method, Decision tree is a condition-based method
- Hyperplanes in Decision tree are **axis-parallel.**
- Decision Tree uses the concepts of **Entropy** and **Information Gain** to decide the decision node.

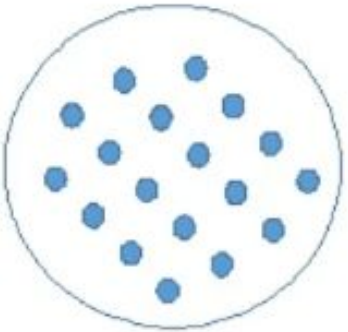
Which node can be described easily?



A



B



C

Class A:
Class B:

Entropy

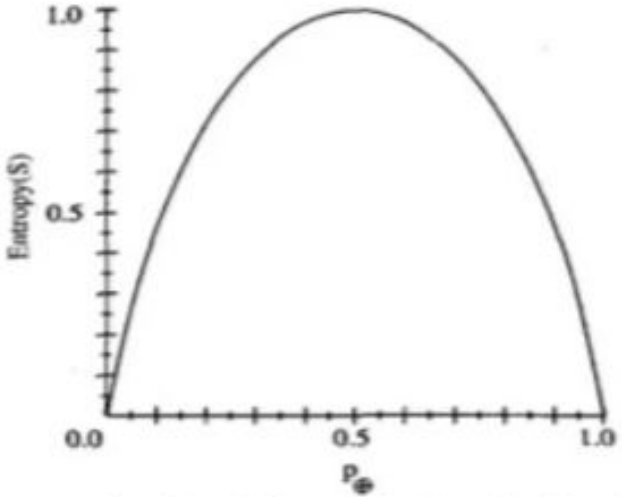
We concluded that:

- Less Impure Nodes requires less information to describe it.
- More impure Nodes requires more information.

In information theory, **Entropy** is measure to define the **degree of disorganization or uncertainty** in a system.
Given a dataset S containing some positive and negative examples of some target variable, the entropy of S is given by:

$$Entropy = \sum_{i=1}^{C} -p_i * \log_2(p_i)$$

where, pi is the proportion of S belonging to class i



The entropy function relative to a boolean classification, as the proportion, $p_\oplus$, of positive examples varies between 0 and 1.

- **Entropy** is **0** if all the members of S belong to the same class.
- **Entropy** is **1** when the collection contains an equal no. of +ve and -ve examples.
- **Entropy** is **between 0 and 1** if the collection contains unequal no. of +ve and -ve examples.

$$Entropy(S) = -p_+ log_2 p_+ - p_- log_2 p_-$$

Information Gain

It is the parameter that decides which attribute goes into the decision node.

To minimize the tree depth, the attribute with most entropy reduction (more information gain) is the best choice.

Constructing a decision tree is all about **finding attribute that returns the highest information gain**(i.e. the most homogeneous branches).

The information gain, Gain(S,A) of an attribute A,

$$Gain(S, A) = \underbrace{Entropy(S)}_{\text{original entropy of S}} - \underbrace{\sum_{v \in values(A)} \frac{|S_v|}{|S|} \cdot Entropy(S_v)}_{\text{relative entropy of S}}$$

Where:

- S is each value v of all possible values of attribute A
- Sv = subset of S for which attribute A has value v
- |Sv| = number of elements in Sv
- |S| = number of elements in S

Play Tennis Dataset

Outlook	Temp	Humidity	Wind	PlayTennis
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rain	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rain	Mild	High	Strong	No

Target Variable

Building a Decision Tree

The basic algorithm used in decision trees is known as the ID3 (by Quinlan) algorithm. The ID3 algorithm builds decision trees using a top-down, greedy approach.

Step 1: Calculate Entropy of the Target

$$\begin{aligned}\textbf{Entropy(PlayGolf)} &= \text{Entropy}\ (5,9)\\ &= \text{Entropy}\ (0.36, 0.64)\\ &= -\ (0.36 \log_2 0.36) - (0.64 \log_2 0.64)\\ &= 0.94\end{aligned}$$

Step 2:

- Dataset is split into different attributes.
- Entropy for each branch is calculated then it is added proportionally to get the total entropy of the split.
- Resulting entropy is subtracted from the entropy before the split which is equivalent to the Information Gain

$S = [9+, 5-]$ $\quad Entropy(S) = -\frac{9}{14} log_2 \frac{9}{14} - \frac{5}{14} log_2 \frac{5}{14} = 0.94$

$S_{Sunny} \leftarrow [2+, 3-]$ $\quad Entropy(S_{Sunny}) = -\frac{2}{5} log_2 \frac{2}{5} - \frac{3}{5} log_2 \frac{3}{5} = 0.971$

$S_{Overcast} \leftarrow [4+, 0-]$ $\quad Entropy(S_{Overcast}) = -\frac{4}{4} log_2 \frac{4}{4} - \frac{0}{4} log_2 \frac{0}{4} = 0$

$S_{Rain} \leftarrow [3+, 2-]$ $\quad Entropy(S_{Rain}) = -\frac{3}{5} log_2 \frac{3}{5} - \frac{2}{5} log_2 \frac{2}{5} = 0.971$

$$Gain\ (S, Outlook) = Entropy(S) - \sum_{v \in \{Sunny, Overcast, Rain\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, Outlook)$$

$$= Entropy(S) - \frac{5}{14} Entropy(S_{Sunny}) - \frac{4}{14} Entropy(S_{Overcast})$$

$$- \frac{5}{14} Entropy(S_{Rain})$$

$$Gain(S, Outlook) = 0.94 - \frac{5}{14} 0.971 - \frac{4}{14}\ 0 - \frac{5}{14} 0.971 = 0.2464$$

Step 3:

Chose the attribute with the largest information gain as the decision node.

Divide the dataset by its branches.

Repeat the same process for each of the branch

$$Gain(S, Outlook) = 0.2464$$

$$Gain(S, Temp) = 0.0289$$

$$Gain(S, Humidity) = 0.1516$$

$$Gain(S, Wind) = 0.0478$$

Sunny

Overcast

Rainy

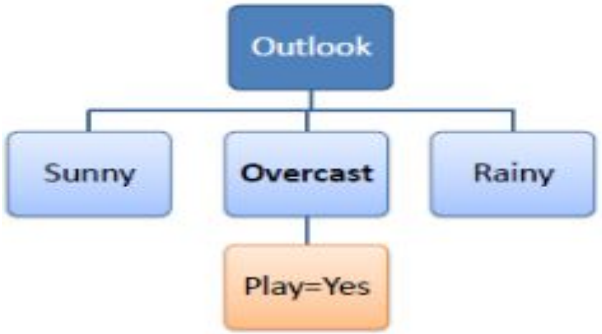
Outlook	Temperature	Humidity	Windy	PlayTennis
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Sunny	Mild	Normal	True	Yes

Outlook	Temperature	Humidity	Windy	PlayTennis
Overcast	Hot	High	False	Yes
Overcast	Cool	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes

Outlook	Temperature	Humidity	Windy	PlayTennis
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Rainy	Mild	Normal	False	Yes
Rainy	Mild	High	True	No

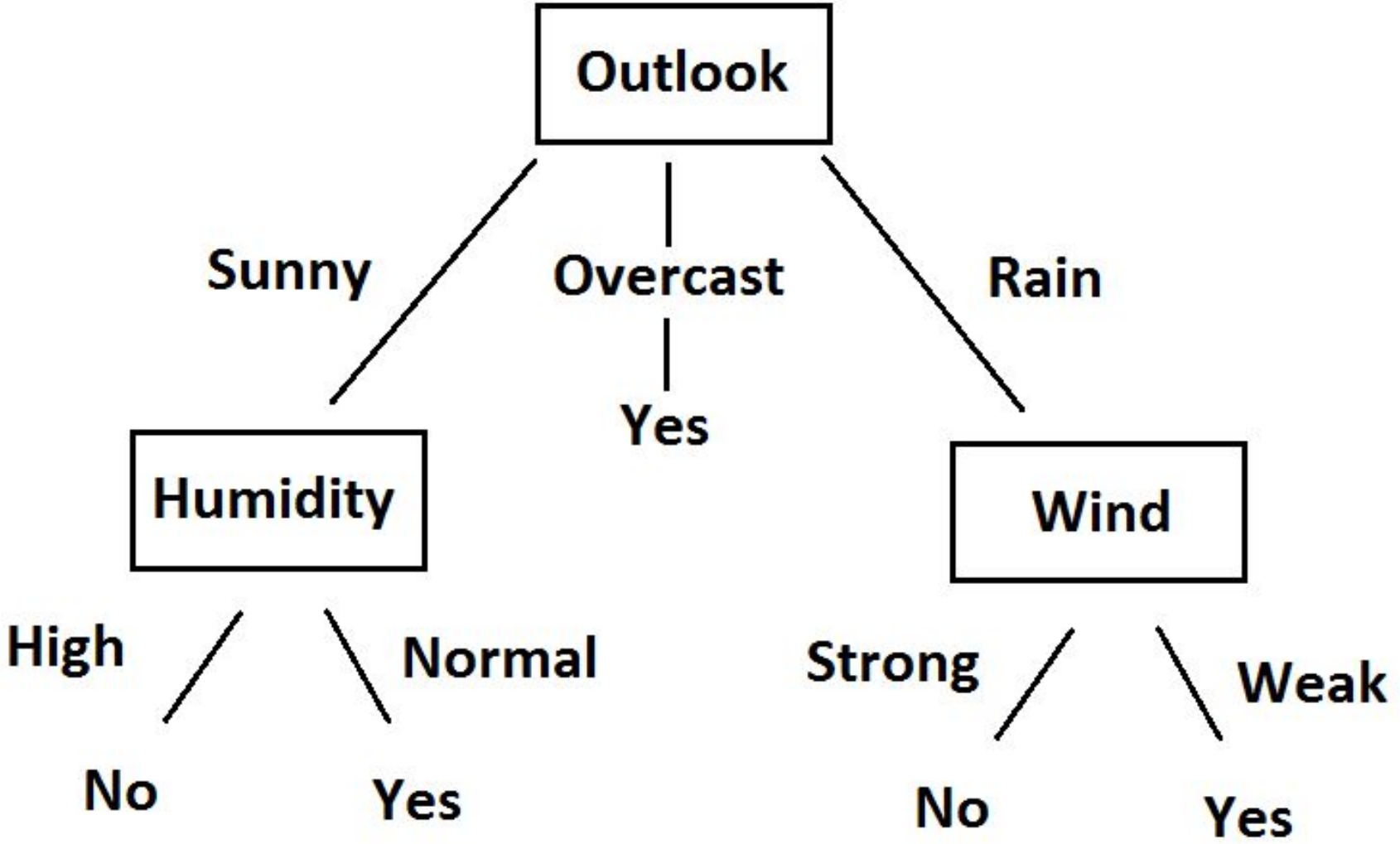
Step 4a:

- A branch with an entropy of 0 is a leaf node.



Step 4b:

- A branch with entropy more than 0 requires further splitting.


Outlook
Sunny
Overcast
Rain
Yes
Humidity
Wind
High
Normal
Strong
Weak
No
Yes
No
Yes

Advantages of Decision Tree:

- Easy to understand and requires little data preprocessing
- New features can be added very easily.
- Can handle both categorical and numerical data.
- Can easily handle irrelevant attributes (Gain = 0)

Disadvantages of Decision Tree:

- Prone to overfitting (**High Variance Model**, but has **Low Bias**)
- Very susceptible to Imbalanced data or outliers as they impact entropy calculation
- Unstable as a small variation in training data may lead to a completely different tree to get generated.

Questions:

- How deeply to grow the tree?
- How are continuous attributes handled?
- Is calculating entropy computationally efficient?