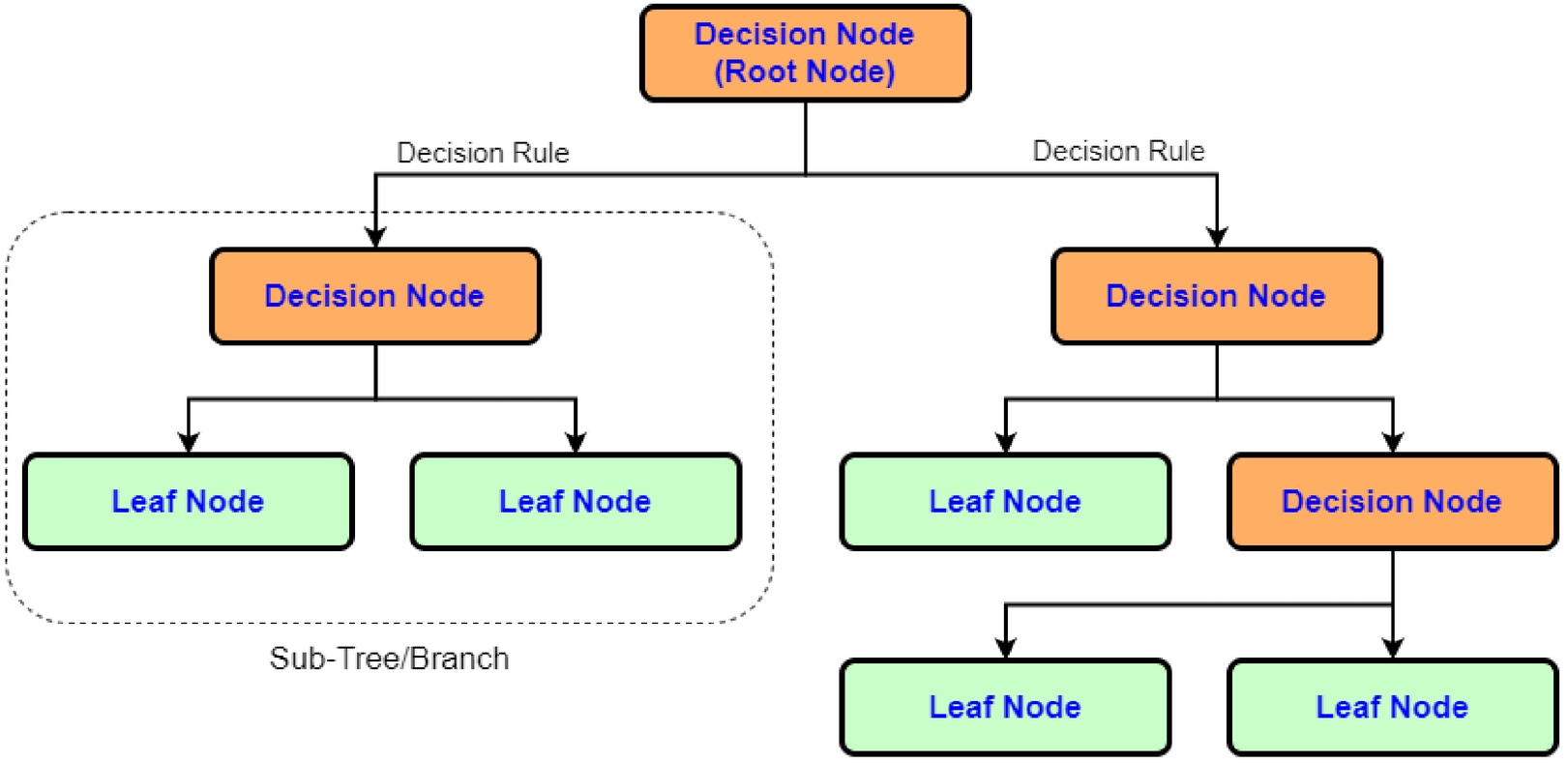
ML SIG

DAY 3

Decision Trees

- Decision tree algorithm belongs to the family of supervised learning algorithms. Unlike other supervised learning algorithms decision tree can be used to solve regression and classification problems.

- The goal of decision tree is to create training model that can predict class(single or multi) or value by learning simple decision rules from training data.



How Decision Tree Works?

Step 1: Select the best attribute to split (Feature Selection).

Step 2: Split data based on that attribute (Node Creation).

Step 3: Repeat recursively until a stopping condition is met (Tree Growth).

Step 4: Make predictions based on the final nodes (Leaf Nodes).

Questions

1. Which feature should be chosen as the root node?

2. What should be the splitting criteria?

3. When should the tree stop growing?

Splitting Criteria :The objective of decision tree is to split the data in such a way that at the end we have different groups of data which has more similarity and less randomness/impurity.

- **For Classification:**
  - Gini Impurity
  - Entropy & Information Gain
- **For Regression:**
  - Mean Squared Error (MSE)
  - Mean Absolute Error (MAE)

Technique to handle overfitting : Pre-Pruning & Post-Pruning

Entropy and Information Gain

Entropy is a measure of impurity or randomness in a dataset. It is used in information gain to decide the best feature for splitting in a decision tree.

'i' in below Entropy formula represent the target classes

$$Entropy = \Sigma_i - p_i \log_2 p_i$$

So in case of 'Entropy', decision tree will split the data using the feature that provides the highest information gain.

```
Information Gain= Entropy(Parent Decision Node)-(Average Entropy(Child Nodes))
```

Gini impurity

In case of gini impurity, we pick a random data point in our dataset. Then randomly classify it according to the class distribution in the dataset. So it becomes very important to know the accuracy of this random classification.

Gini impurity gives us the probability of incorrect classification. We'll determine the quality of the split by weighting the impurity of each branch by how many elements it has.

Resulting value is called as 'Gini Gain' or 'Gini Index'. This is what's used to pick the best split in a decision tree.
Higher the Gini Gain, better the split

'i' in below Gini formula represent the target classes

$$Gini = 1 - \Sigma_i p_i^2$$

So in case of 'gini', decision tree will split the data using the feature that provides the highest gini gain.

Person	Age	Weight (kg)	Smoker	Risk
P1	15	55	No	Low Risk
P2	12	70	No	High Risk
P3	25	68	No	Low Risk
P4	35	72	Yes	High Risk
P5	40	65	No	Low Risk
P6	38	80	Yes	High Risk
P7	45	55	No	Low Risk

$$\text{Entropy(Parent)} = 0.985$$

We divide the dataset based on age into three groups:

- **Age < 18**: 2 people (1 Low, 1 High) →

$$H = -0.5 \log_2(0.5) - 0.5 \log_2(0.5) = 1$$

- **Age 18–30**: 1 person (1 Low) →
  Pure group ⇒

$$H = 0$$

- **Age > 30**: 4 people (2 Low, 2 High) →

$$H = -0.5 \log_2(0.5) - 0.5 \log_2(0.5) = 1$$

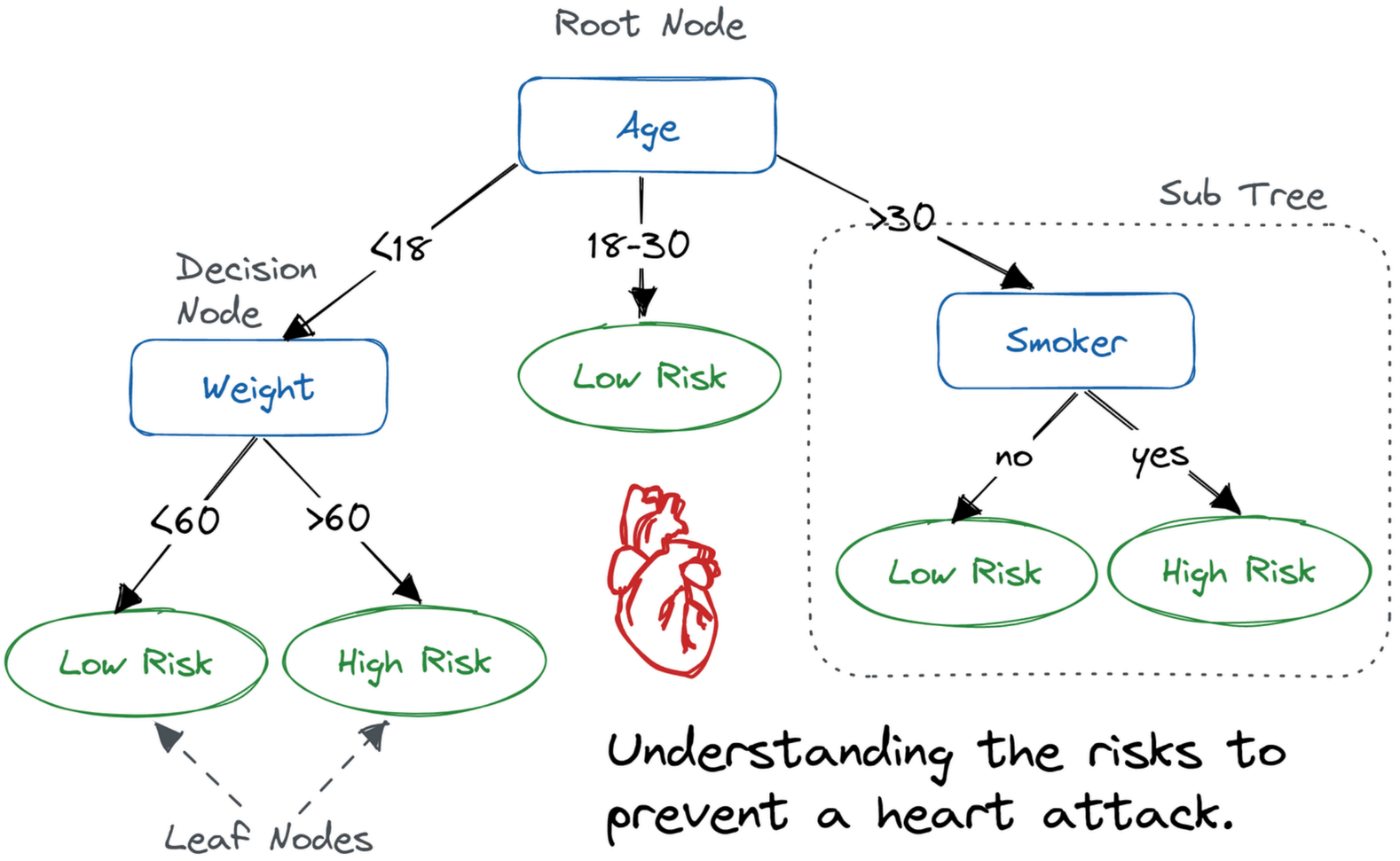
After splitting the dataset based on age, we calculate the **weighted average entropy** of the child nodes:

$$\text{Weighted Entropy} = \frac{2}{7} \cdot 1 + \frac{1}{7} \cdot 0 + \frac{4}{7} \cdot 1 = 0.857$$

Information Gain tells us how much uncertainty is reduced by making this split:

$$\text{Information Gain (Age)} = \text{Entropy (Parent)} - \text{Weighted Entropy}$$

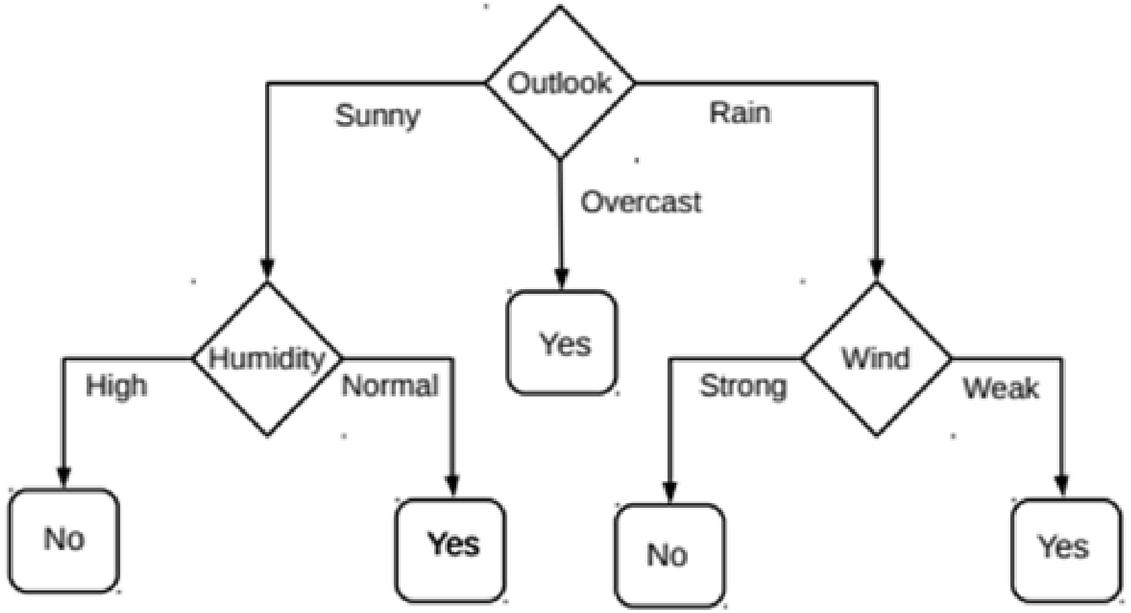
$$\text{Information Gain (Age)} = 0.985 - 0.857 = \boxed{0.128}$$


Root Node
Age
<18
18-30
>30
Decision Node
Weight
Low Risk
Sub Tree
Smoker
<60
>60
no
yes
Low Risk
High Risk
Low Risk
High Risk
Leaf Nodes
Understanding the risks to prevent a heart attack.

Example

This dataset represents a classic example of a Decision Tree classification problem where we predict whether to "play" or "not play" a game based on weather conditions.

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



Step 1:

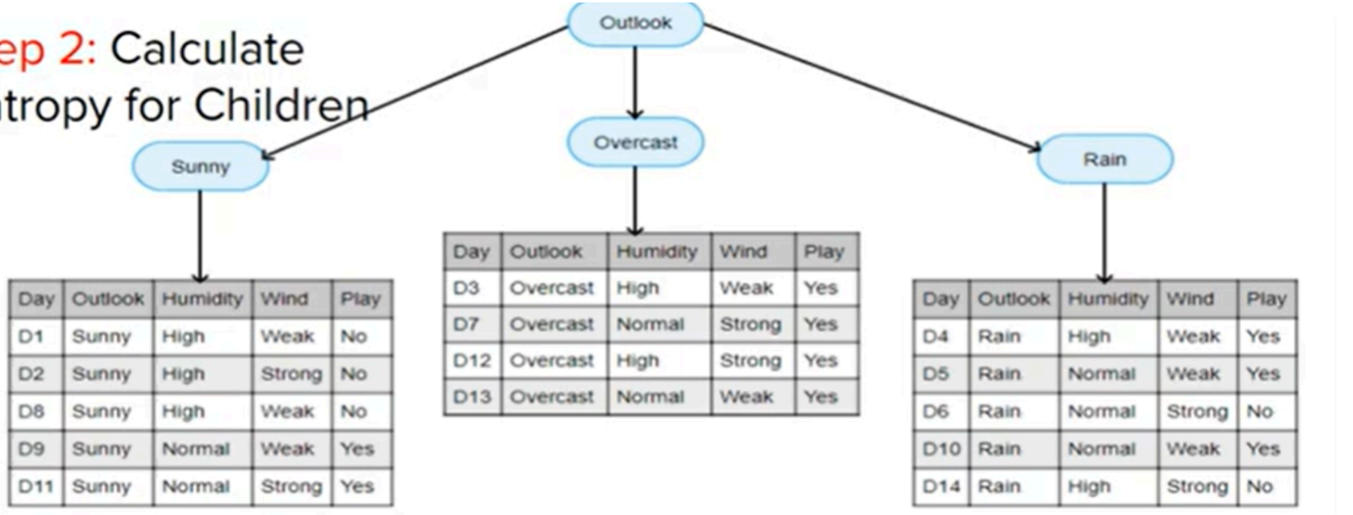
Entropy of Parent

$E(P) = -p_y \log_2(p_y) - p_n \log_2(p_n)$

$= 9/14 \log_2(9/14) - 5/14 / \log_2(5/14)$

$E(P) = \mathbf{0.94}$

Step 2: Calculate Entropy for Children



Day	Outlook	Humidity	Wind	Play
D1	Sunny	High	Weak	No
D2	Sunny	High	Strong	No
D8	Sunny	High	Weak	No
D9	Sunny	Normal	Weak	Yes
D11	Sunny	Normal	Strong	Yes

Day	Outlook	Humidity	Wind	Play
D3	Overcast	High	Weak	Yes
D7	Overcast	Normal	Strong	Yes
D12	Overcast	High	Strong	Yes
D13	Overcast	Normal	Weak	Yes

Day	Outlook	Humidity	Wind	Play
D4	Rain	High	Weak	Yes
D5	Rain	Normal	Weak	Yes
D6	Rain	Normal	Strong	No
D10	Rain	Normal	Weak	Yes
D14	Rain	High	Strong	No

E(S) = -2/5log(2/5) - 3/5log(3/5) E(S)= 0.97	E(O) = -5/5log(5/5) - 0/5log(0/5) E(O)= 0	E(R) = -3/5log(3/5) - 2/5log(2/5) E(S)= 0.97

Step 3 : Calculate weighted Entropy of Children

Weighted Entropy = 5/14 * 0.97 + 4/14 * 0 + 5/14 * 0.97

W.E(Children) = **0.69**

P(Overcast) is a leaf node as it's entropy is 0

Step 4 : Calculate Information Gain

Information Gain = E(Parent) - {Weighted Average} * E(Children)

IG = **0.97 - 0.69 = 0.28**

So the information gain(or the decrease in entropy/impurity) when you split this data on the basis of **Outlook** condition/column is **0.28**

Disadvantages of Decision Trees

Overfitting – Decision trees can become too complex and fit the training data too well, leading to poor generalization on new data.

Unstable – Small changes in data can significantly alter the tree structure, making it highly sensitive to noise.

Biased Splitting – Trees may favor features with more unique values, leading to biased decision-making.

Computationally Expensive – Training deep trees can be slow and resource-intensive, especially with large datasets.

Prone to Irrelevant Features – If irrelevant features are present, decision trees may still split on them, reducing model efficiency.

Cat classification example

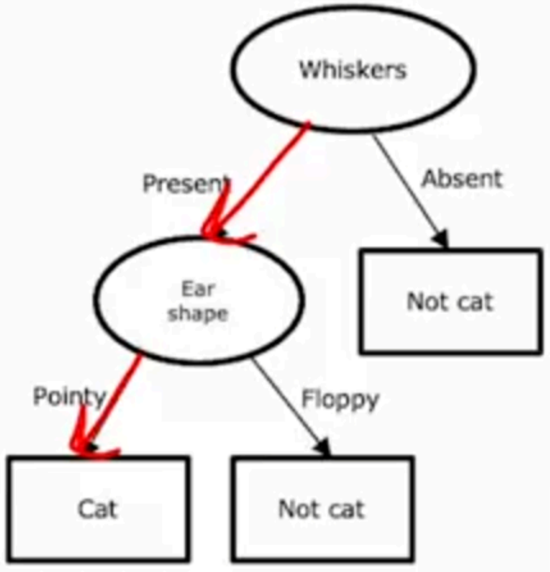
	Ear shape	Face shape	Whiskers	Cat
	Pointy	Round	Present	1
	Floppy	Not round	Present	1
	Floppy	Round	Absent	0
	Pointy	Not round	Present	0
	Pointy	Round	Present	1
	Pointy	Round	Absent	1
	Floppy	Not round	Absent	0
	Pointy	Round	Absent	1
	Floppy	Round	Absent	0
	Floppy	Round	Absent	0

Categorical (discrete values)

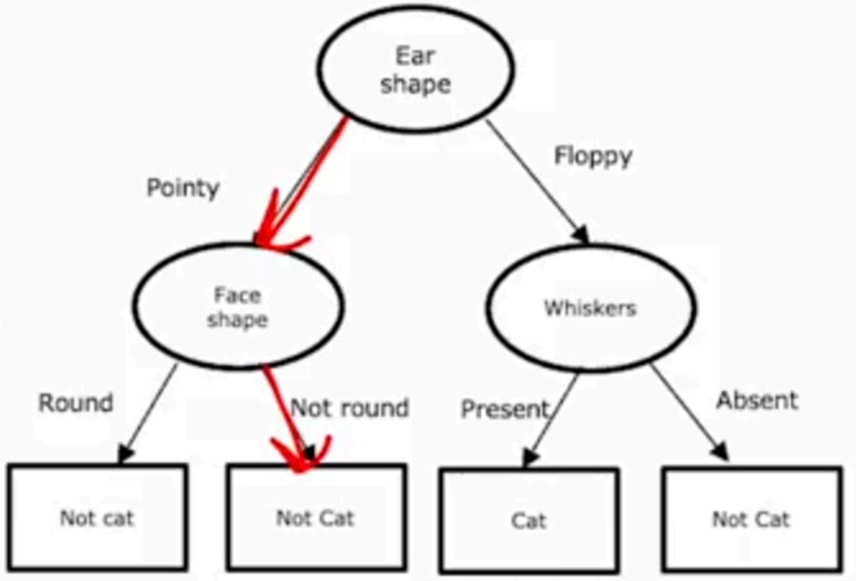
X

y

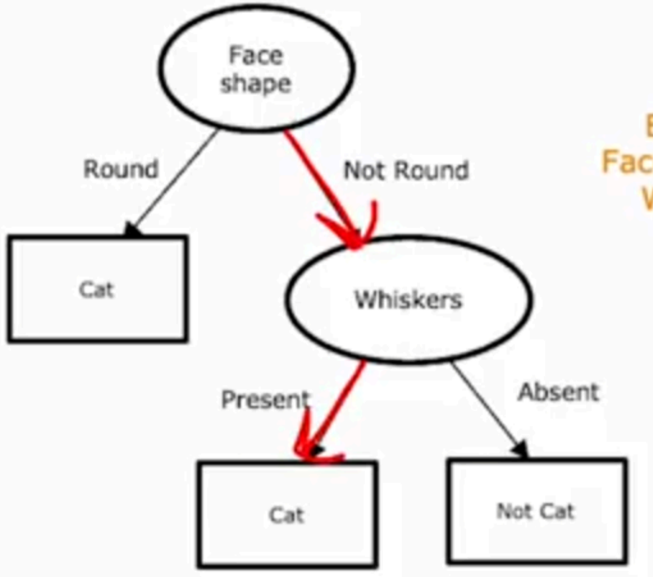
Tree ensemble



Prediction: Cat



Prediction: Not cat



Prediction: Cat

New test example

Ear shape: Pointy
Face shape: Not Round
Whiskers: Present

Random Forest

Random forest is a supervised learning algorithm. It has two variations – one is used for classification problems and other is used for regression problems.

1. The random forest uses many trees, and it makes a prediction by averaging the predictions of each component tree.
2. It generally has much better predictive accuracy than a single decision tree and it works well with default parameters.

Random Forest Algorithm
(Bagging technique)

Random forest algorithm intuition can be divided into two stages.

In the first stage, we randomly select “k” features out of total m features and build the random forest. In the first stage, we proceed as follows:-

1. Randomly select k features from a total of m features where k < m.
2. Among the k features, calculate the node d using the best split point.
3. Split the node into daughter nodes using the best split.
4. Repeat 1 to 3 steps until l number of nodes has been reached.
5. Build forest by repeating steps 1 to 4 for n number of times to create n number of trees.

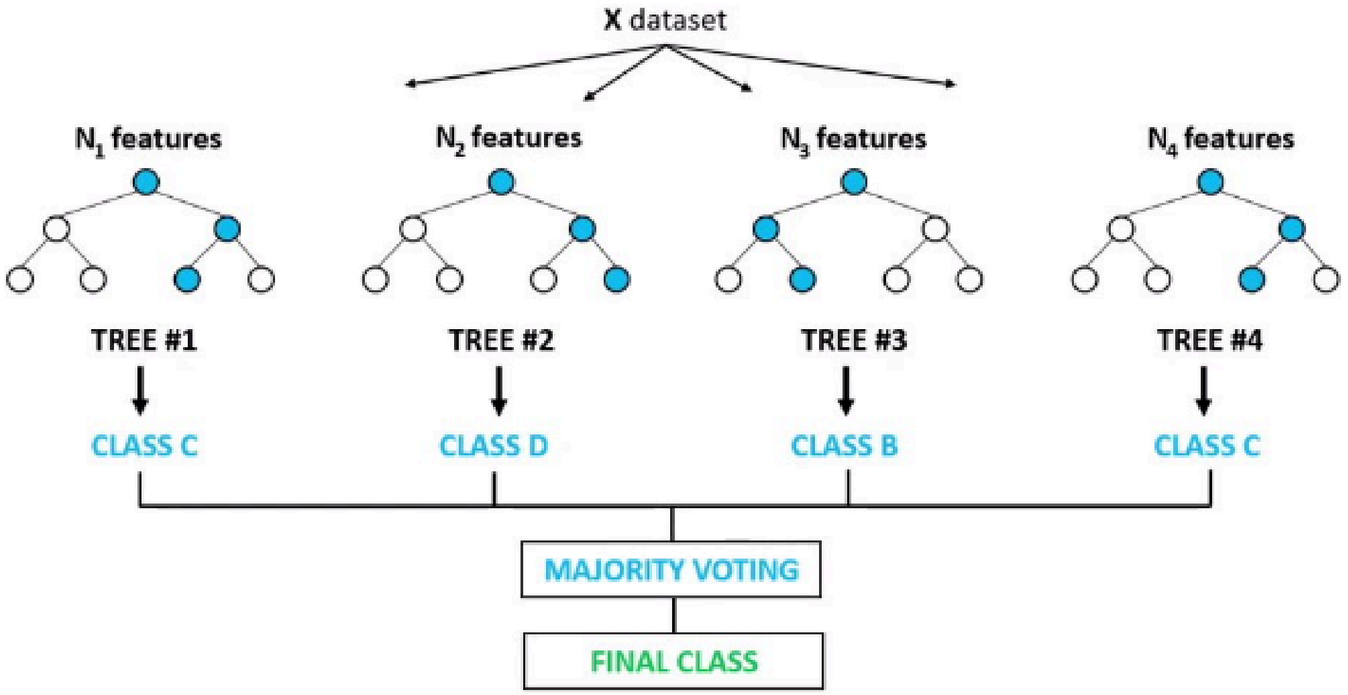
In the second stage, we make predictions using the trained random forest algorithm.

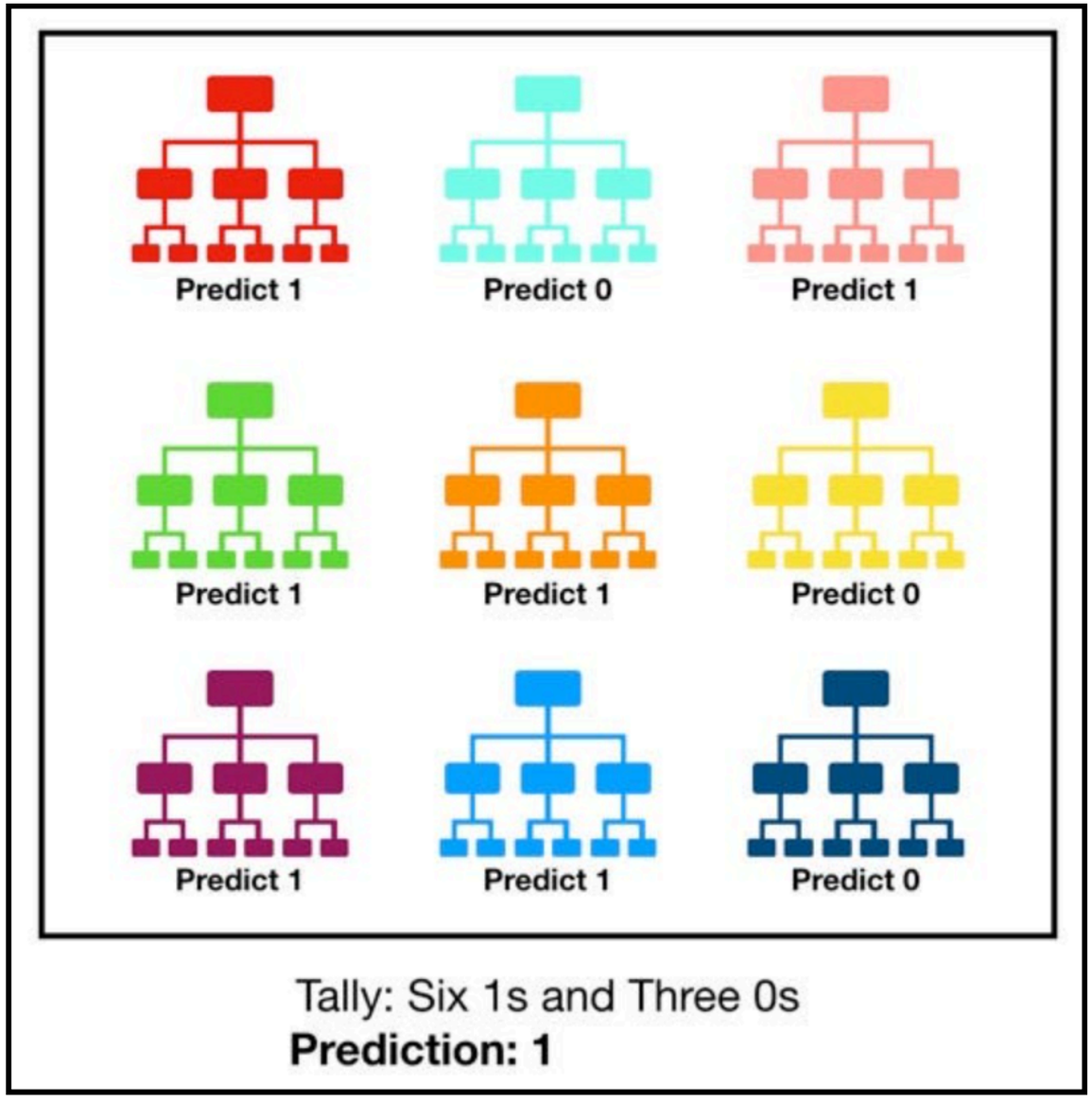
1. **We take the test features and use the rules of each randomly created decision tree to predict the outcome and stores the predicted outcome.**

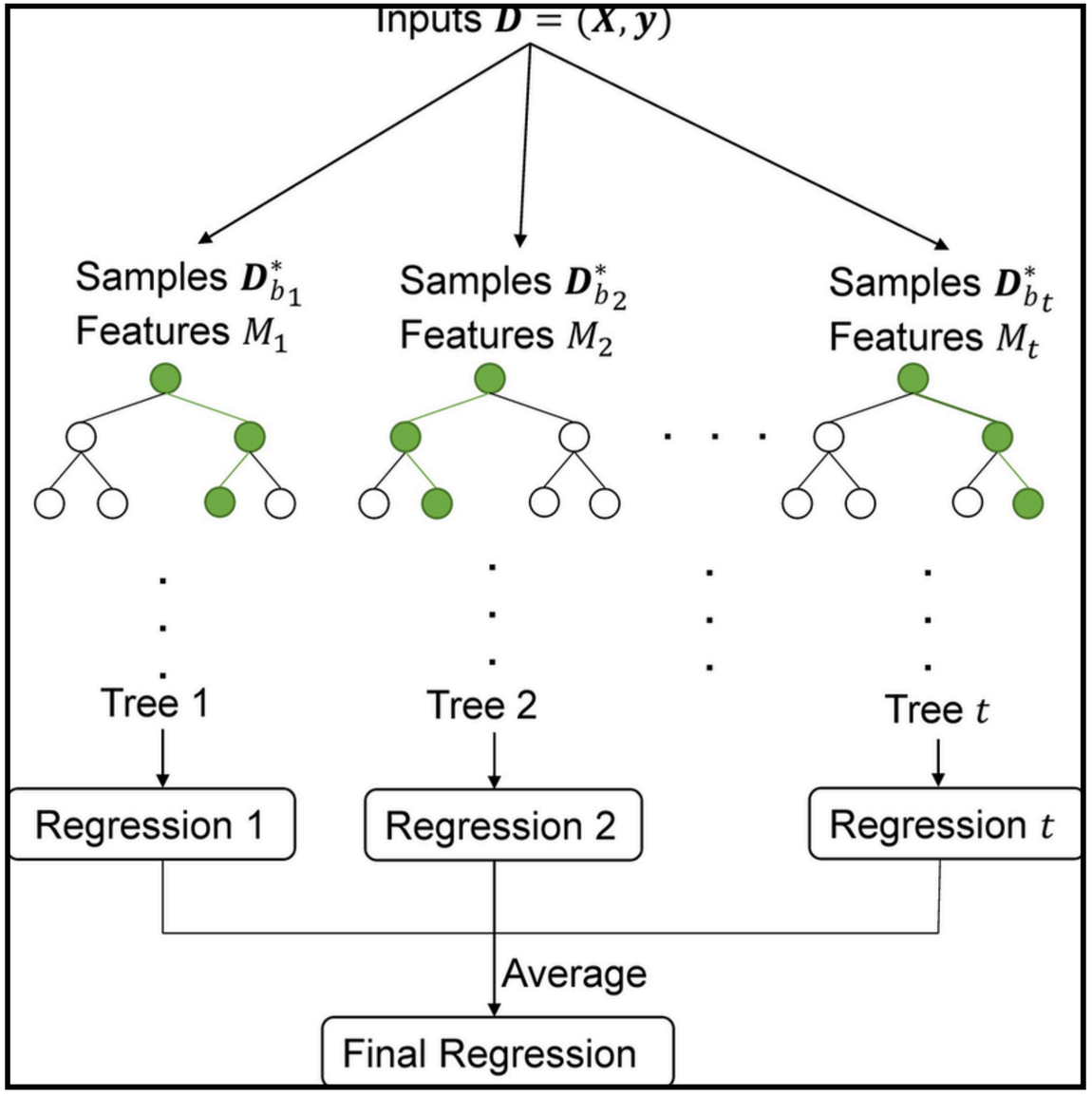
2. **Then, we calculate the votes for each predicted target.**

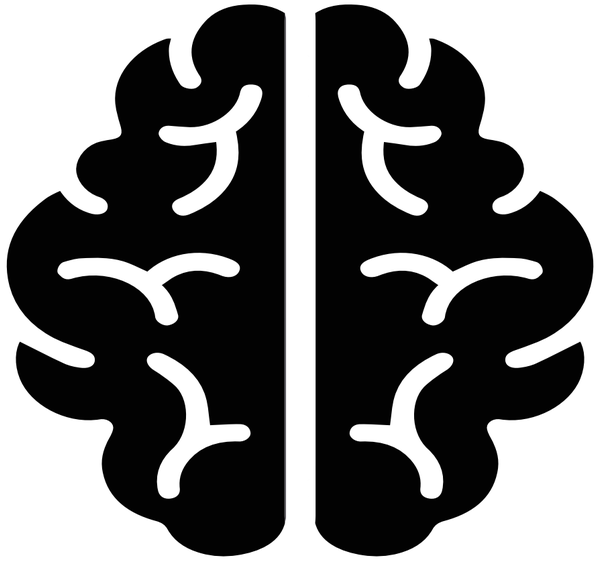
3. **Finally, we consider the high voted predicted target as the final prediction from the random forest algorithm.**

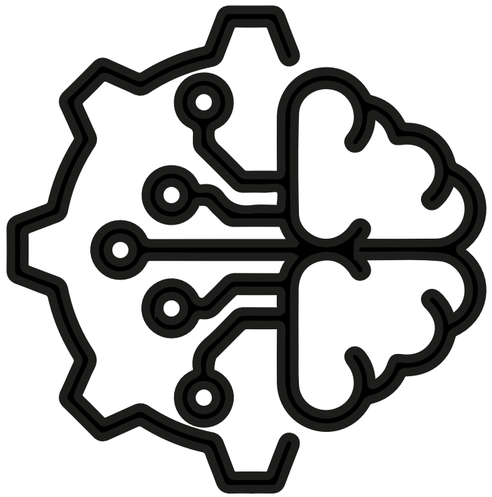
Random Forest Classifier



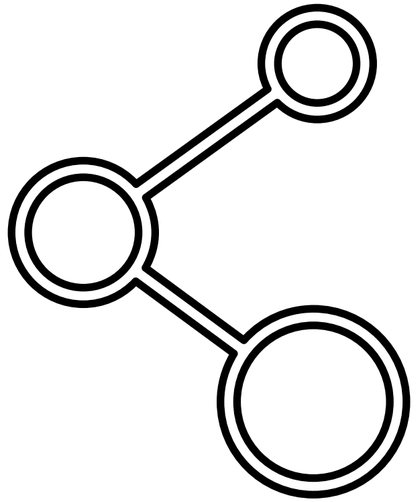

Predict 1
Predict 0
Predict 1
Predict 1
Predict 1
Predict 0
Predict 1
Predict 1
Predict 0
Tally: Six 1s and Three 0s
Prediction: 1

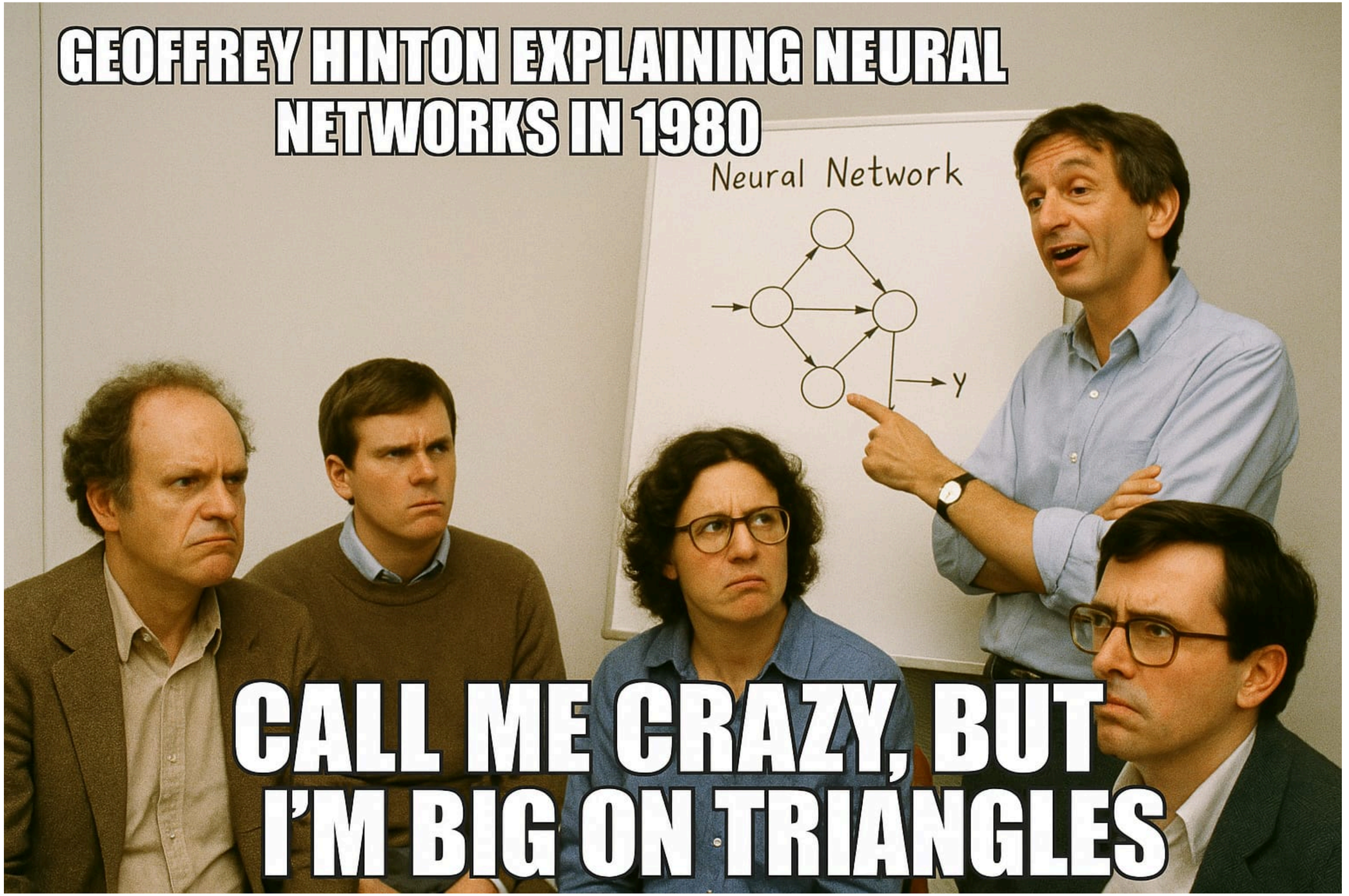

Inputs $D = (X, y)$
Samples $D^*_{b_1}$
Features M_1
Samples $D^*_{b_2}$
Features M_2
Samples $D^*_{b_t}$
Features M_t
Tree 1
Tree 2
Tree t
Regression 1
Regression 2
Regression t
Average
Final Regression





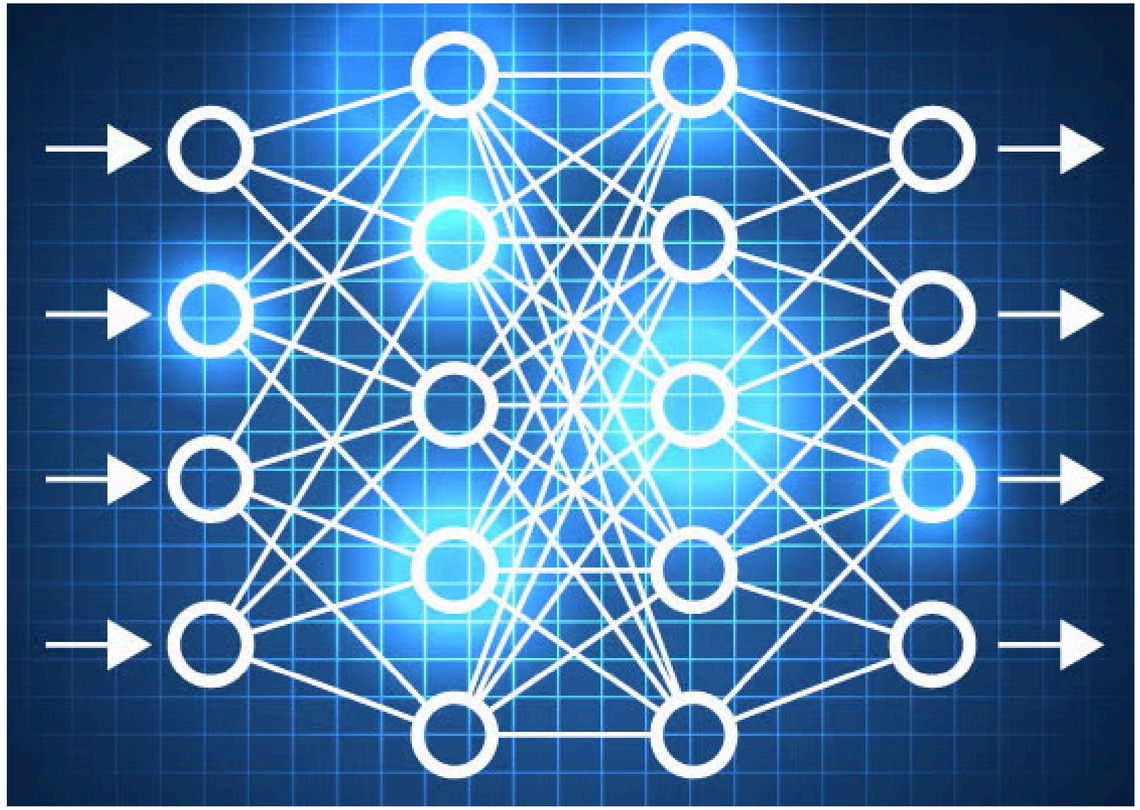
NEURAL NETWORKS



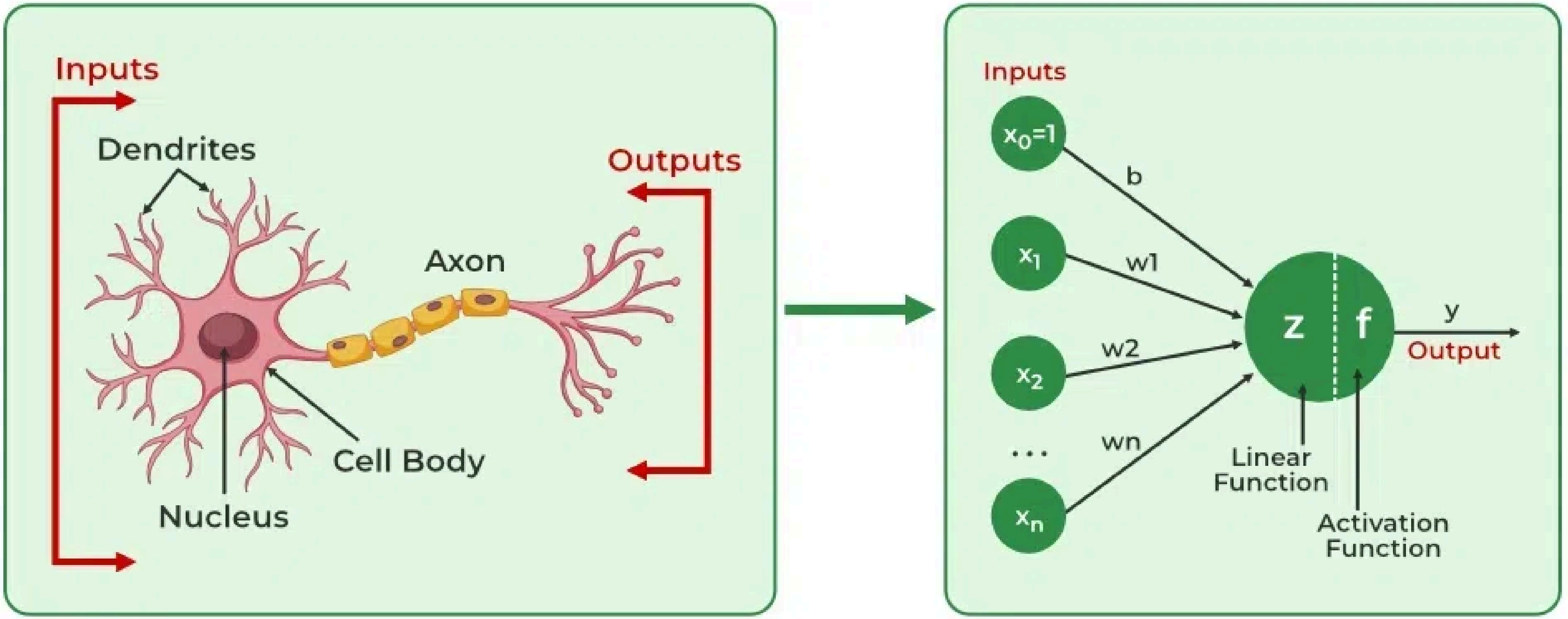

GEOFFREY HINTON EXPLAINING NEURAL NETWORKS IN 1980
Neural Network
y
CALL ME CRAZY, BUT I'M BIG ON TRIANGLES

Neural Networks

- A Neural Network is a machine learning model inspired by the human brain.
- It consists of multiple layers of artificial neurons that process data and learn patterns.
- Used in image recognition, speech processing, recommendation systems, and more.

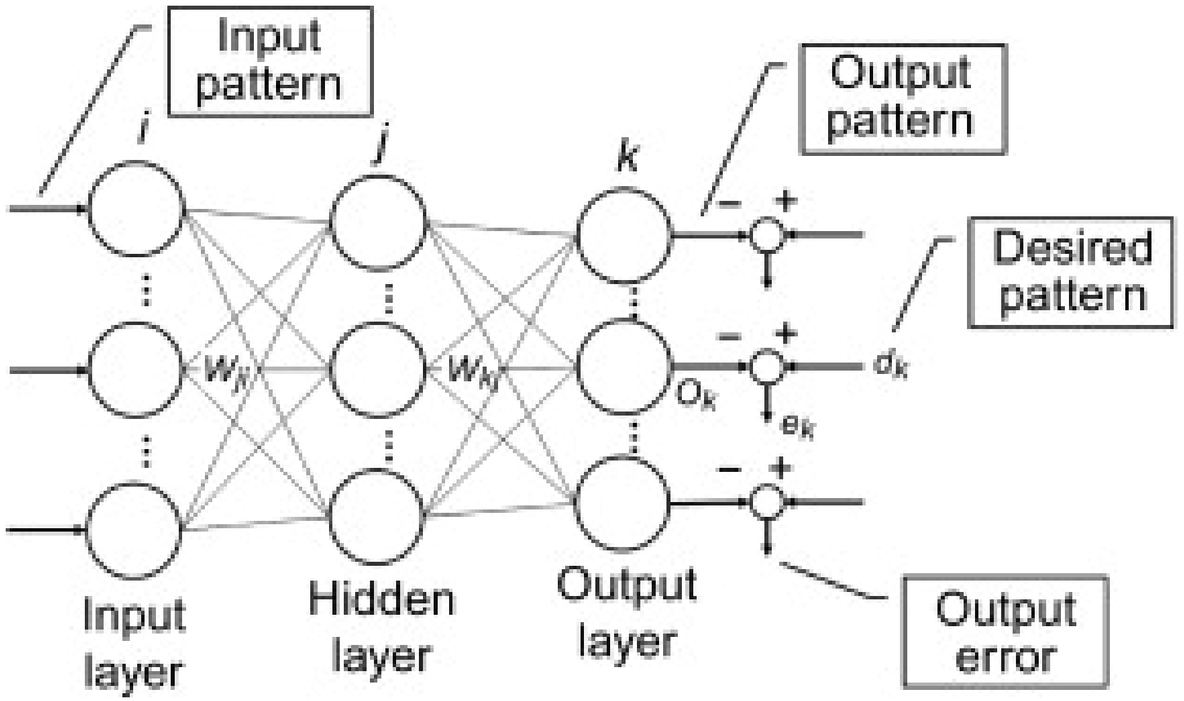


From a neuron to a neural network



Structure of Neural network

- First Layer
  - This layer takes in **raw data.**
  - The **number of neurons here corresponds to the number of input features.**
  - Ex: If an image has 28 × 28 pixels, the input layer will have 784 neurons (one for each pixel).
- Second Layer(Hidden Layer)
  - These are the layers where most of the computation happens.
  - Each connection has an associated **weight** that determines the importance of the input value.
  - A **bias** value is added to improve flexibility.
  - The neuron applies an **activation** function to introduce non-linearity, making the network capable of learning complex patterns.
- Third Layer
  - This layer produces the output.
  - Binary Classification: 1 neuron with **Sigmoid** Activation (outputs a probability).
  - Regression : 1 neuron with **Linear Activation** (outputs a real number).



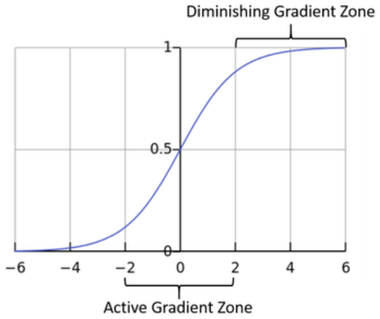
Activation Functions

- Activation functions decide whether a neuron should be activated or not.
- They introduce non-linearity, allowing the network to learn complex patterns.

1. **Sigmoid**

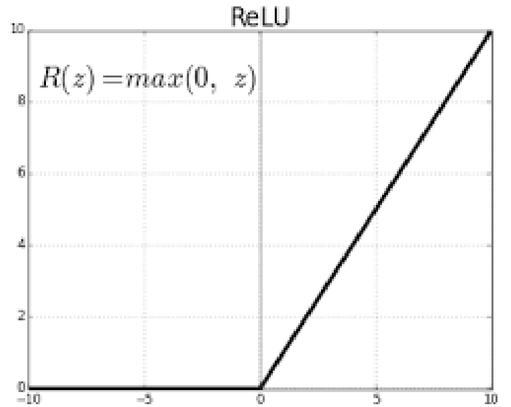
What it does: Turns numbers into a value between 0 and 1.
When to use: If you want to treat the output like a probability (e.g., “Is this a cat? Yes or no?”)
Example: If the output is 0.9, it means “I’m 90% sure it’s a cat.”

$$A = \frac{1}{1+e^{-x}}$$



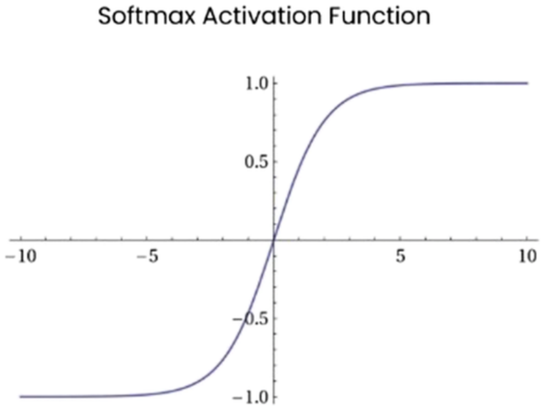
2. **ReLU – Rectified Linear Unit**

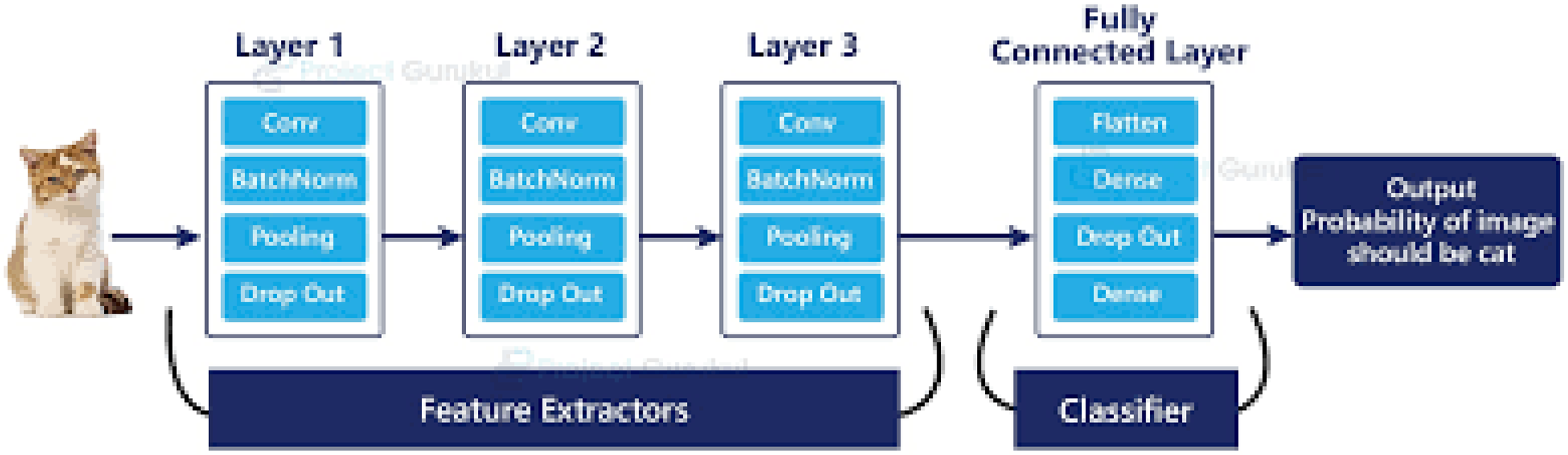
What it does:
If the number is negative, it gives 0.
If the number is positive, it gives it as-is.
When to use: Almost always! It’s super fast and works well in most problems.
Example: Think of it like ignoring all bad (negative) signals and just focusing on the good ones.



3. **Softmax(for classifying purposes)**

What it does: Converts outputs into percentages that all add up to 100%.
When to use: When your model has to choose one thing from many (like picking the right Pokémon from a group).
Example: If you're classifying animals and your model says:
Dog: 10%
Cat: 80%
Rabbit: 10%
Then it's most confident the animal is a Cat



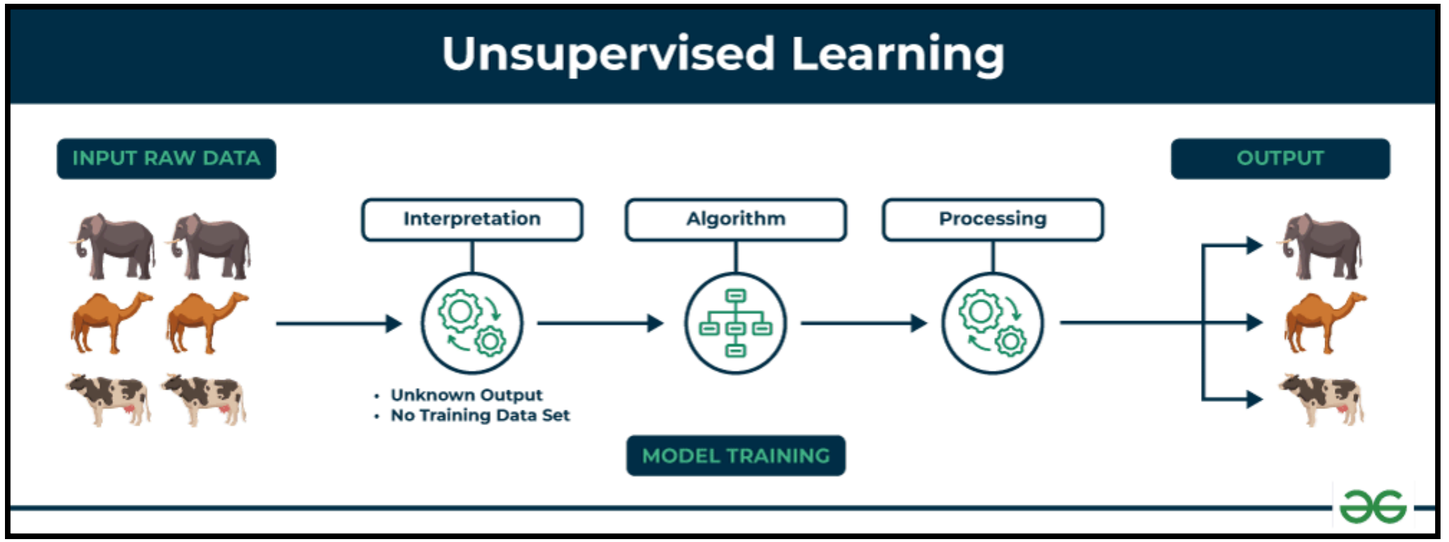


Unsupervised Learning

1.) Unsupervised learning is a branch of machine learning that deals with unlabeled data.

2.) Unlike supervised learning, where the data is labeled with a specific category or outcome, unsupervised learning algorithms are tasked with finding patterns and relationships within the data without any prior knowledge of the data's meaning.

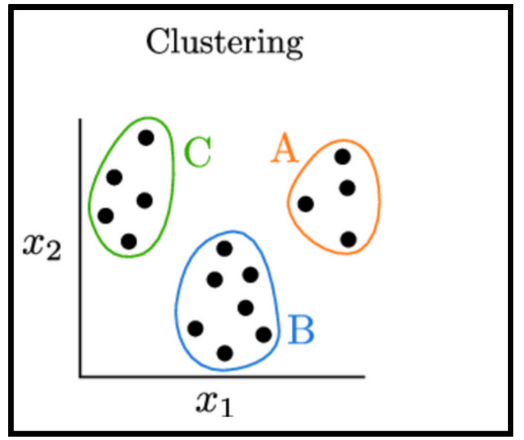
3.) Unsupervised machine learning algorithms find hidden patterns and data without any human intervention, i.e., we don't give output to our model.



Unsupervised Learning Algorithms

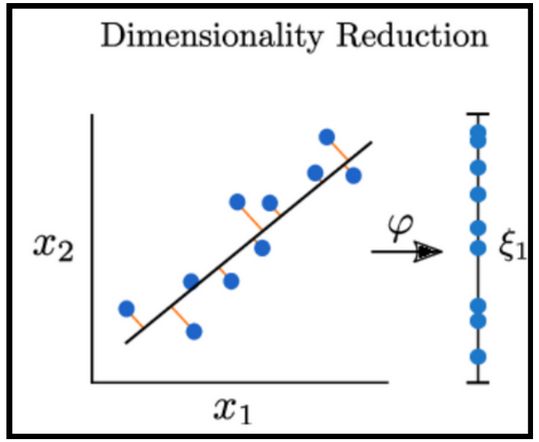
There are mainly 3 types of Algorithms which are used for Unsupervised dataset.

1) Clustering : Clustering in unsupervised machine learning is the process of grouping unlabeled data into clusters based on their similarities.



2.) Association Rule Learning : This technique is a rule-based ML technique that finds out some very useful relations between parameters of a large data set.

3.) Dimensionality Reduction :Dimensionality reduction is the process of reducing the number of features in a dataset while preserving as much information as possible.

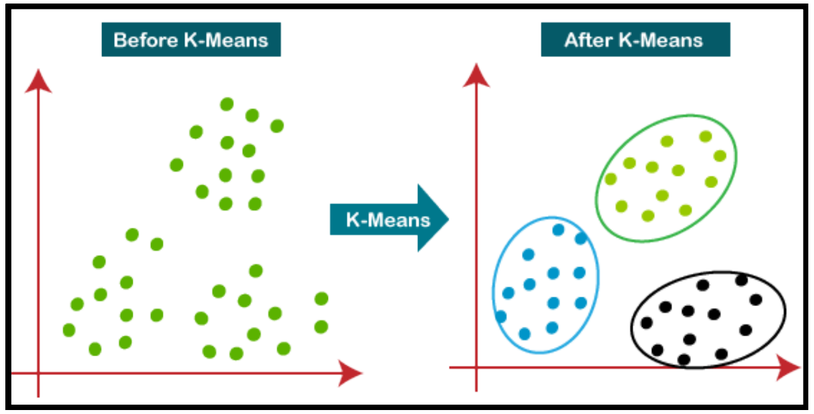


K means Clustering

1.) K-Means Clustering is an Unsupervised Machine Learning algorithm which groups the unlabeled dataset into different clusters.

2.) K-means clustering is a technique used to organize data into groups based on their similarity.

3.) A movie recommendation system like Netflix can use K-Means Clustering to group users based on parameters like watch time, preferred movie ratings, genre preferences, and viewing frequency, forming clusters such as Casual Viewers, Regular Movie Lovers, Binge Watchers, and Genre Specialists, enabling personalized recommendations for each group.



How does the K-Means Algorithm Work?

PIacmT STUDENT CHAPTER

Step-1: Select the number K to decide the number of clusters.

Step-2: Select random K points or centroids. (It can be other from the input dataset).

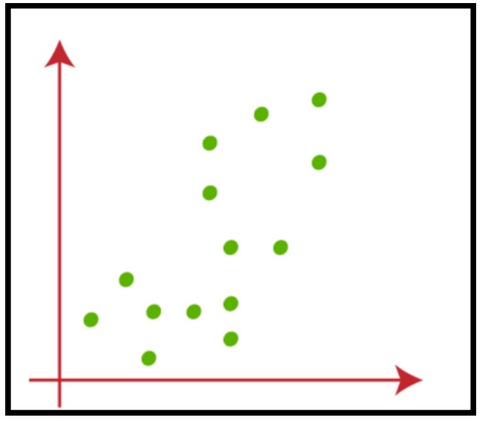
Step-3: Assign each data point to their closest centroid, which will form the predefined K clusters.

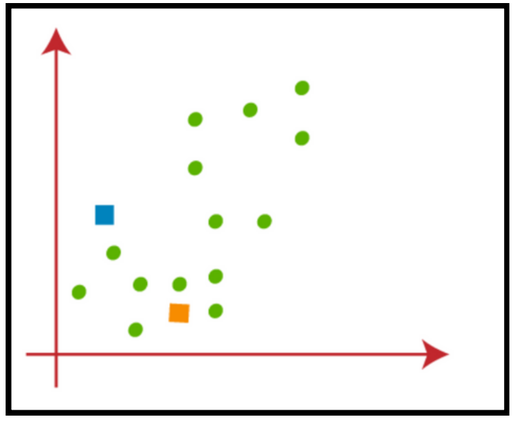
Step-4: Calculate the variance and place a new centroid of each cluster.

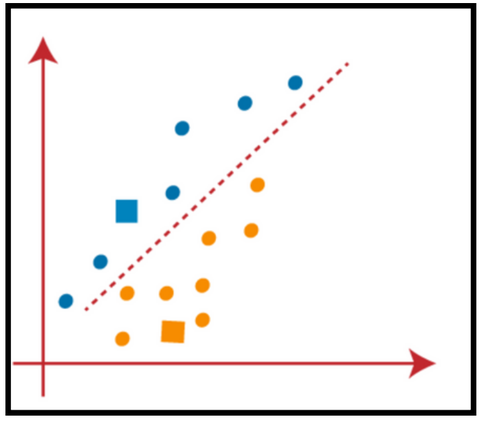
Step-5: Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.

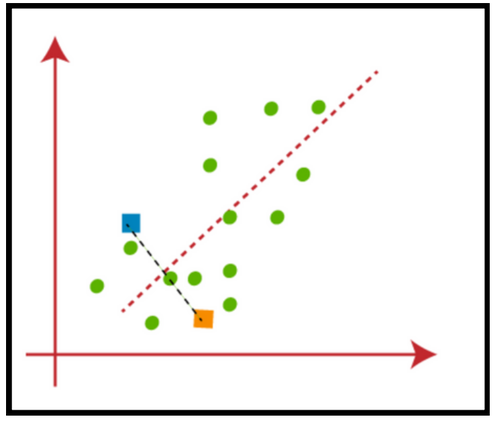
Step-6: If any reassignment occurs, then go to step-4 else go to FINISH.

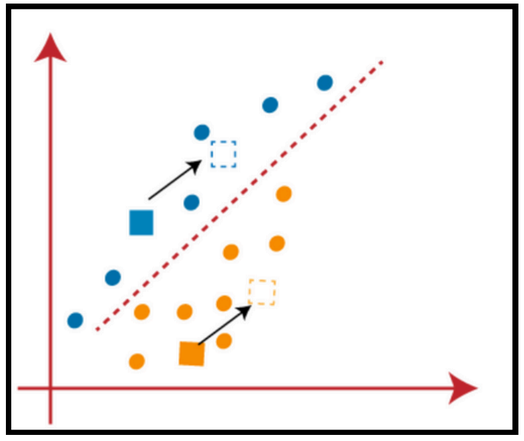
Step-7: The model is ready.

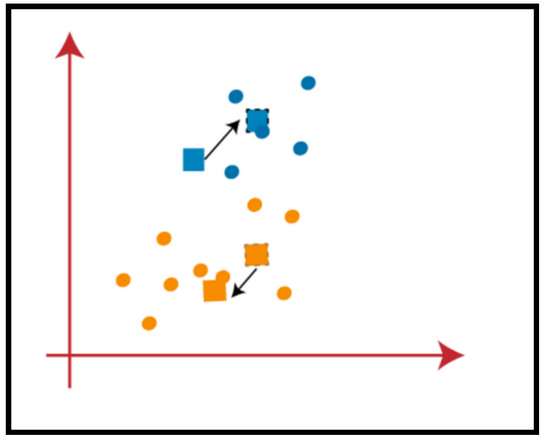


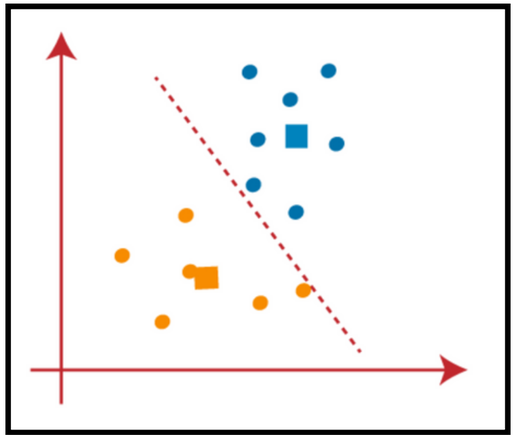


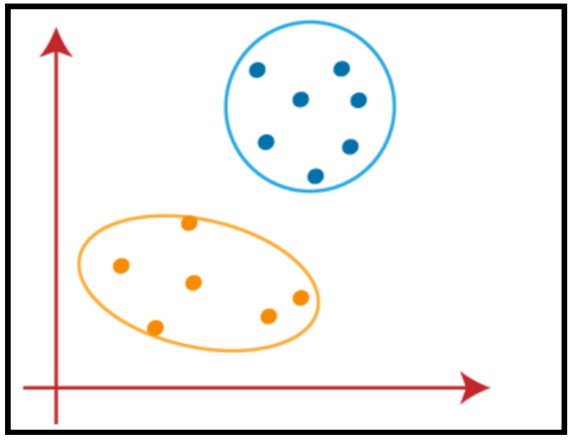










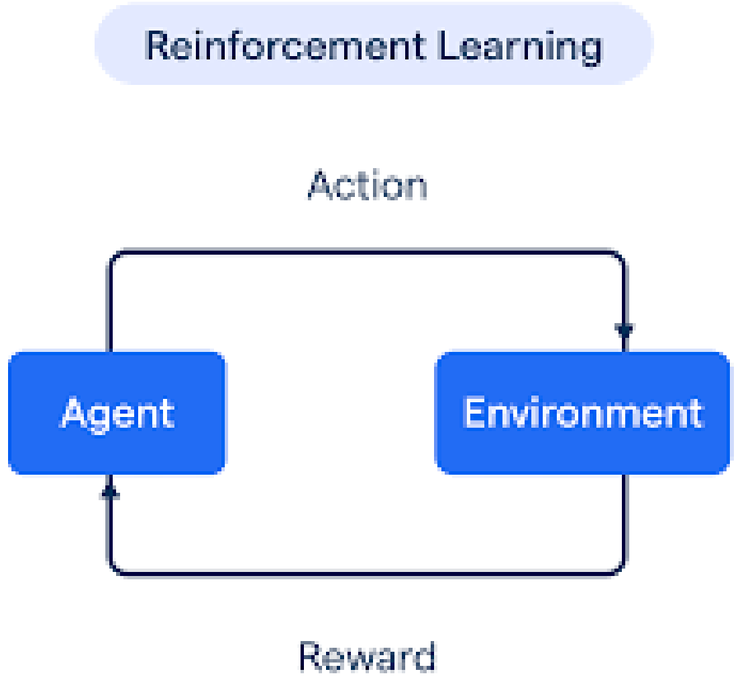


Reinforcement Learning

Reinforcement Learning (RL) is a type of machine learning where an agent learns to make decisions by interacting with its environment. The agent receives rewards or penalties based on its actions and uses this feedback to improve its performance over time, aiming to maximize the total reward.

Diference from Supervised and Unsupervised Learning

- Supervised Learning deals with labeled data and maps inputs to outputs.
- Unsupervised Learning works with unlabeled data to find patterns or groups
- Reinforcement Learning (RL), as mentioned earlier, differs because it relies on interaction and reward systems rather than labeled or unlabeled datasets. It's like trial-and-error learning for optimal decisions!.



Here’s a breakdown of each component in Reinforcement Learning:

- **Agent: The decision-maker that takes actions to achieve its goals.**
- **Environment: The external system or world that the agent interacts with.**
- **Rewards: The feedback (positive or negative) the agent gets from the environment based on its actions.**

These components form the foundation of Reinforcement Learning!

Real Life Examples

- A real-life example is AlphaGo, where RL enabled an AI agent to master the board game Go. It used deep reinforcement learning to analyze millions of game scenarios and develop strategies, ultimately defeating world champions

- **In robotics, RL is used for teaching robotic arms how to grasp objects effectively, such as in warehouse automation for sorting and picking items. For automation, a great example is self-driving cars, where RL helps vehicles make real-time decisions like navigation, obstacle avoidance, and traffic management. These showcase RL's powerful applications in real-world tasks!**

PIEACM STUDENT CHAPTER

Youtube video here just showing how RL Works

https://youtu.be/C2zw2H1c5Fk?si=-i1PXUCvXPdbFbWK

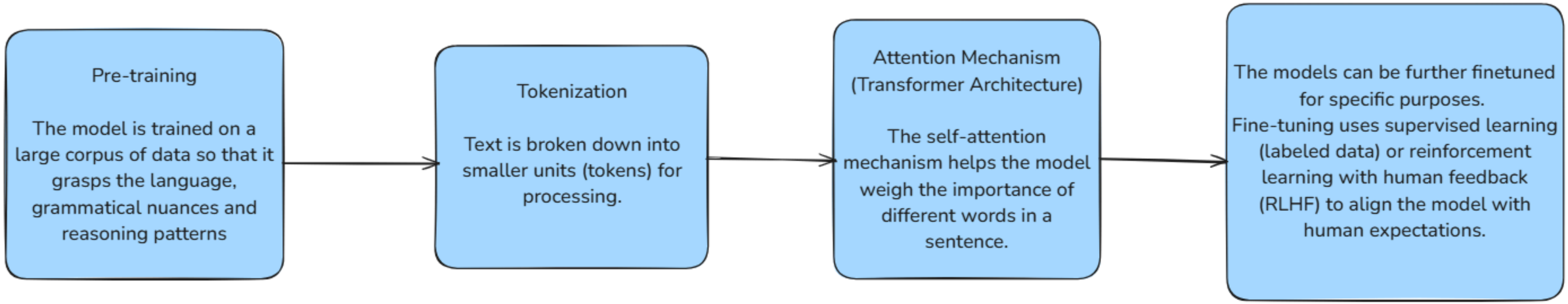
-

Large Language Models(LLMs)

PIEacmT STUDENT CHAPTER

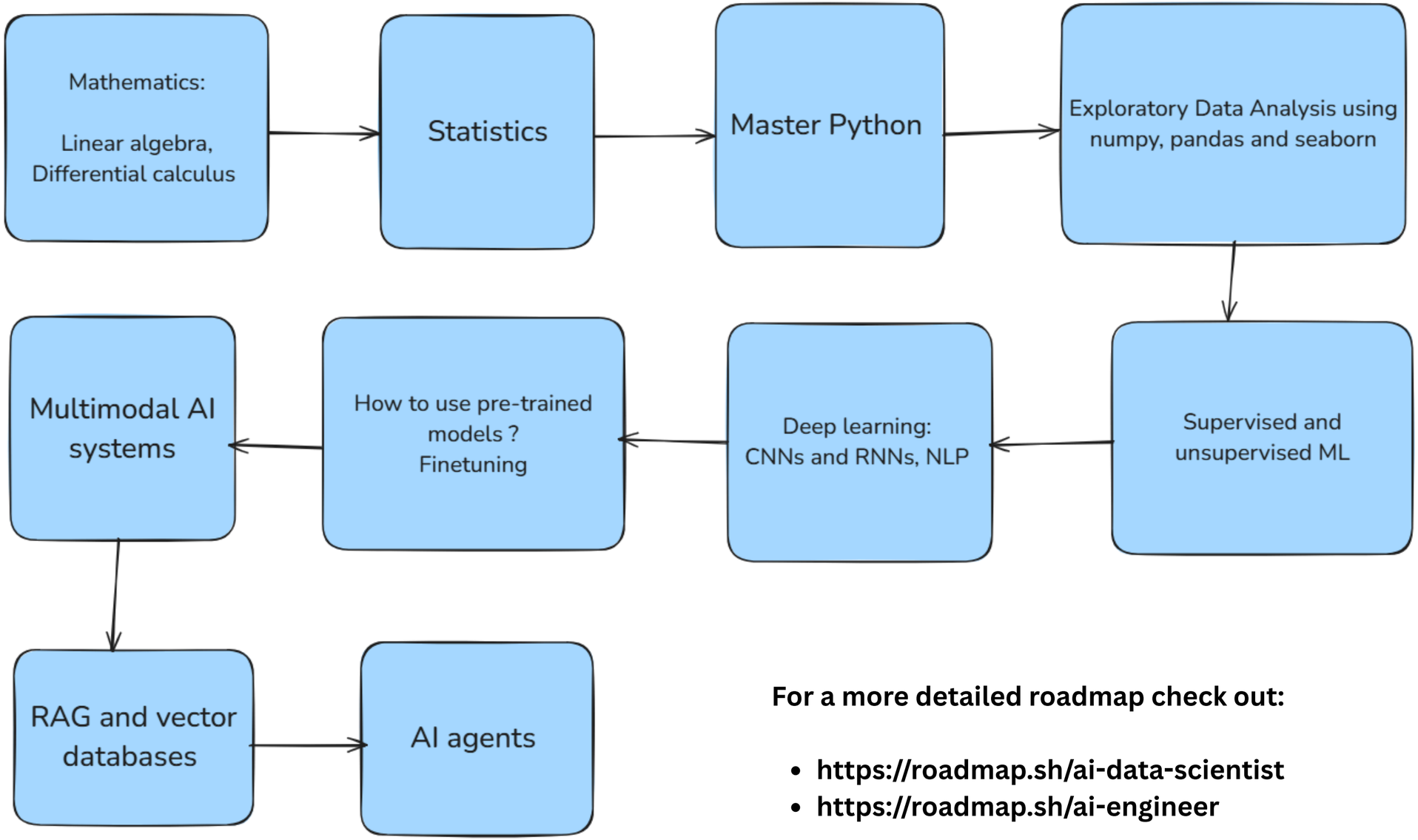
Large Language Models (LLMs) are AI models trained on text data. They understand and generate human language. LLMs are used in applications with billions or trillions of parameters.

How does ChatGPT work ?



Real life applications of LLMs

ROADMAP



For a more detailed roadmap check out:

- **https://roadmap.sh/ai-data-scientist**
- **https://roadmap.sh/ai-engineer**

THANK YOU