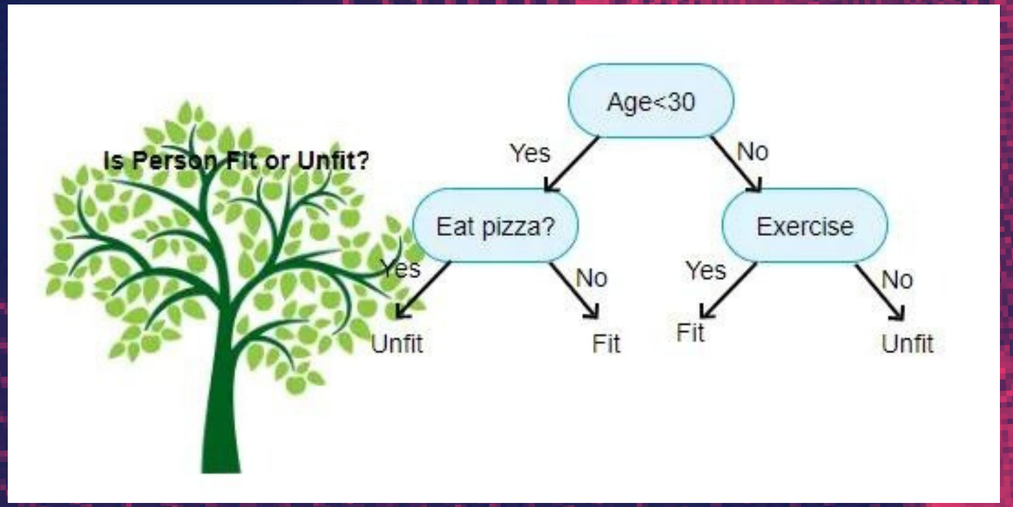
AI CLUB

Decision

TREES



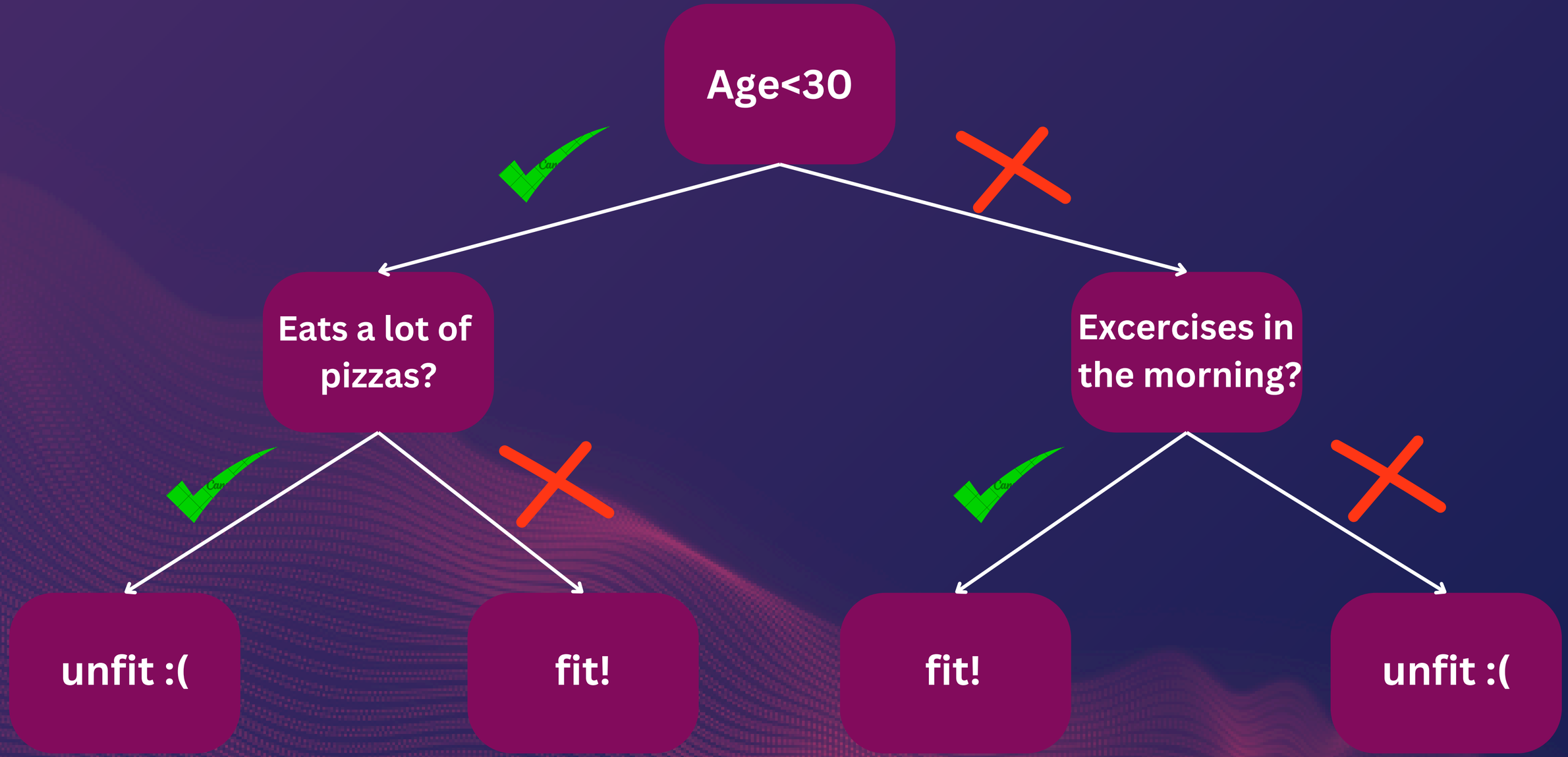
WHAT IS A DECISION TREE?

A decision tree is a supervised learning algorithm, which is utilized for both classification and regression tasks. It has a hierarchical, tree structure, which consists of a root node, branches, internal nodes and leaf nodes.

OKAY?...BUT WHAT IS A DECISION TREE?

A decision tree basically consists of a bunch of nested if-else statements. In general, it makes a statement and then makes a decision based on whether or not the statement is True or False.

For better understanding, let’s take an example of a simple decision tree.

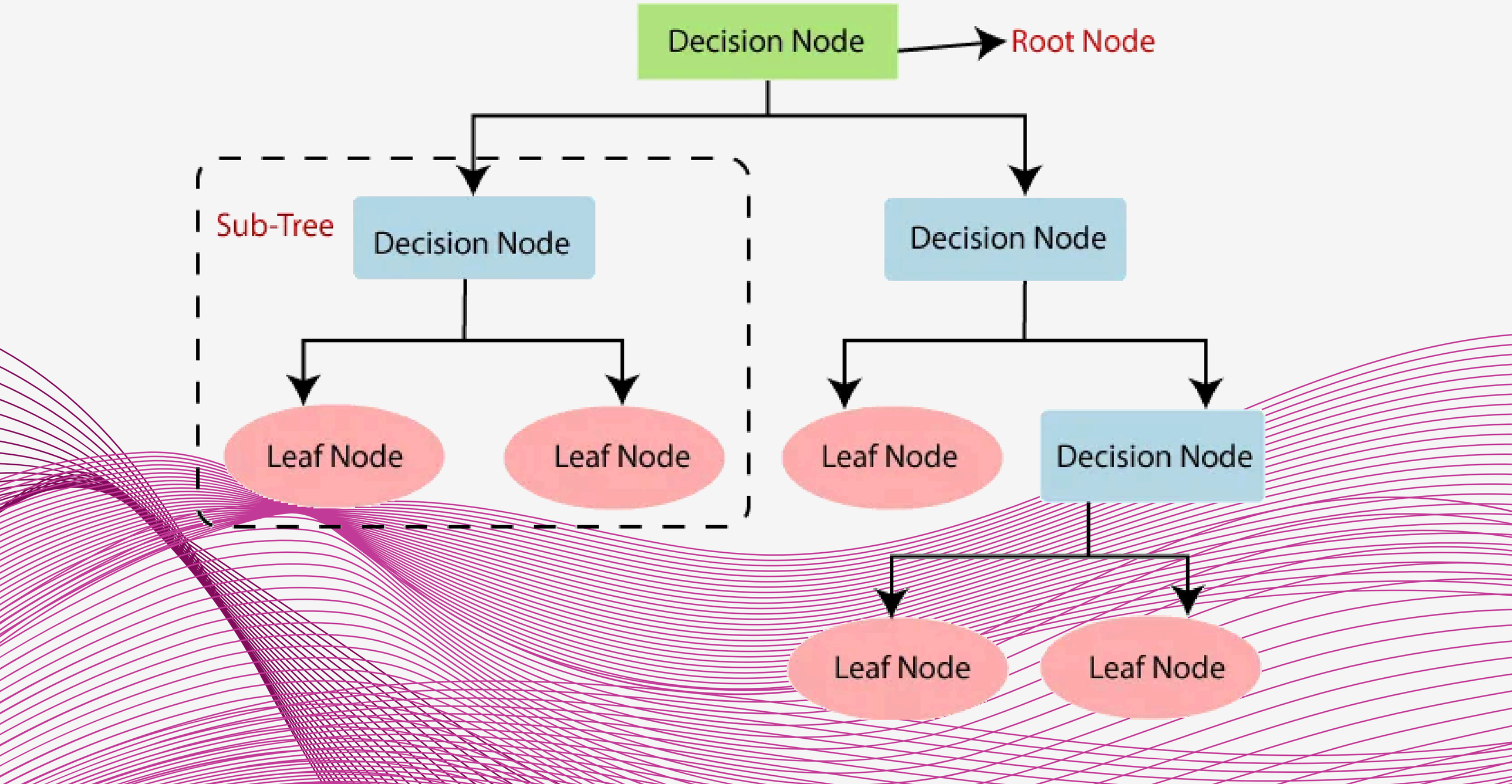

Age<30
Eats a lot of pizzas?
Excercises in the morning?
unfit :(
fit!
fit!
unfit :(

The example given is called a ‘Classification Tree’ where the tree classifies things into categories. Decision trees can also be used to predict numeric values and are known as ‘Regression Trees’.

Here, we will mainly focus on Classification Trees.

NOTE: By convention, when a statement is true, we go to the left as we traverse down a tree.

IMPORTANT TERMINOLOGIES


Decision Node
Root Node
Sub-Tree
Decision Node
Decision Node
Leaf Node
Leaf Node
Leaf Node
Decision Node
Leaf Node
Leaf Node

. The very top of the tree is called the ‘Root’ or ‘Root Node’.

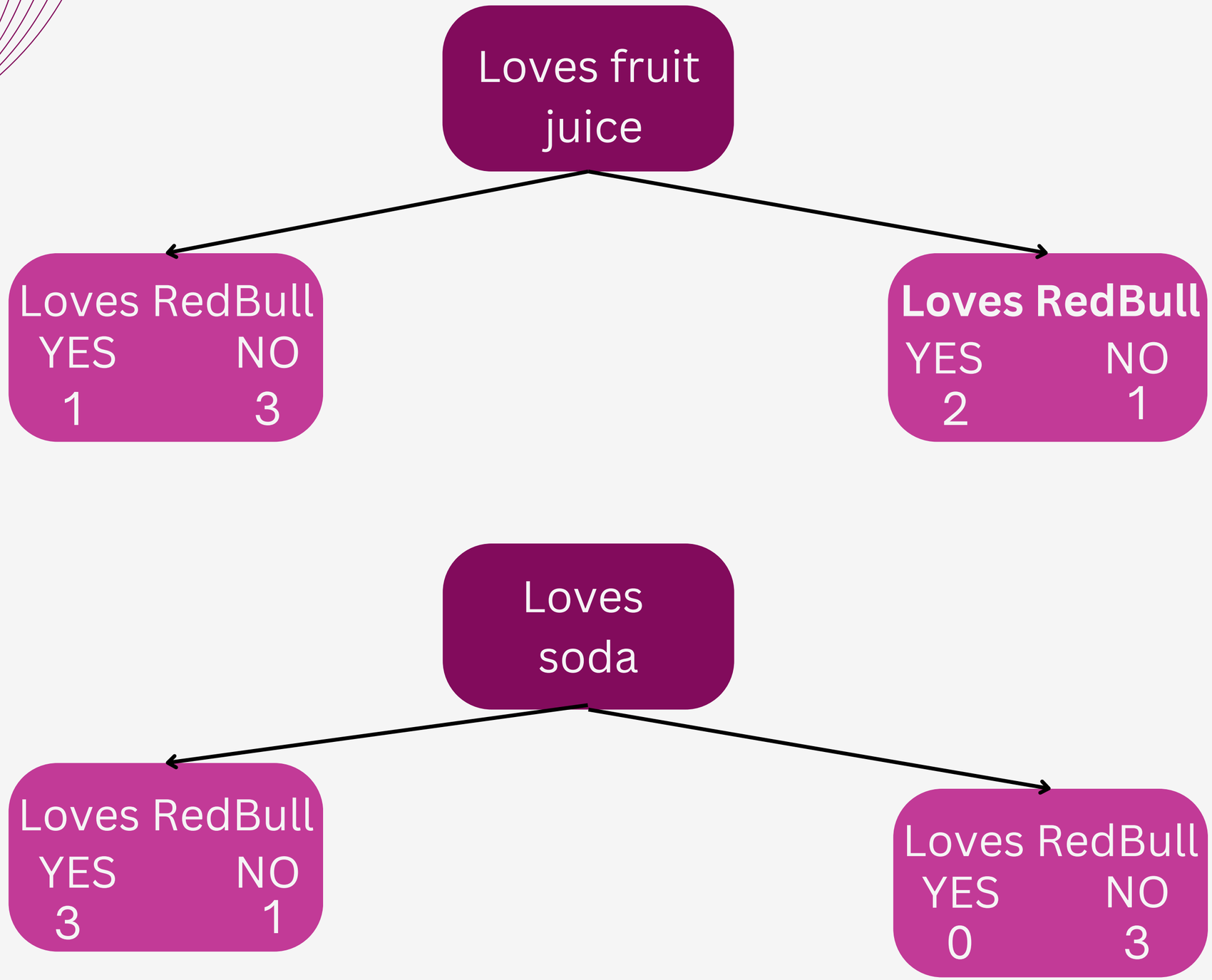
. The subsequent nodes are just known as ‘Internal Nodes’ or ‘Decision Nodes’.

. Finally, we have the ‘leaf nodes’, that do not have any further arrows pointing away from them.

Now, that we are familiar with some basic aspects of Decision trees, let’s learn how to build one.

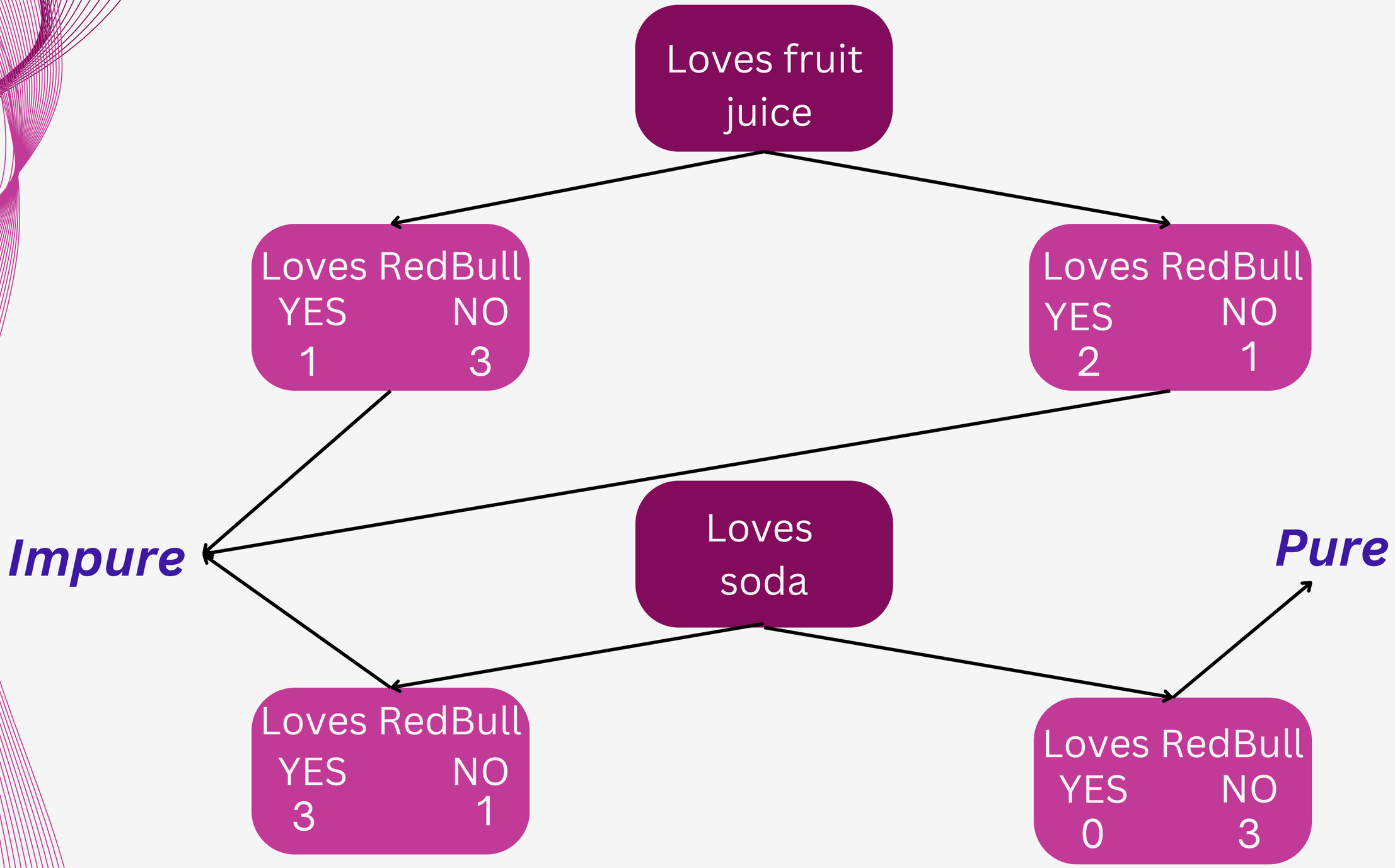
Consider the data set given below which tells us if a person likes the drink RedBull or not based on some of their other likes and dislikes and age.

LOVES FRUIT JUICE	LOVES SODA	AGE	LOVES RED BULL
YES	YES	7	NO
YES	NO	12	NO
NO	YES	18	YES
NO	YES	35	YES
YES	YES	38	YES
YES	NO	50	NO
NO	NO	83	NO


Loves fruit juice
Loves RedBull
YES NO
1 3
Loves RedBull
YES NO
2 1
Loves soda
Loves RedBull
YES NO
3 1
Loves RedBull
YES NO
0 3

From the looks of it, neither of the two trees do a perfect job in predicting if a person likes RedBull or not, but we can see that 3 of the 4 leaves contain mixtures of people that do or do not like RedBull.

Such leaves, that contain a mixture of people are known as ‘Impure’ leaves and the leaf that contains only those people that either love or do not love RedBull are called ‘Pure’ leaves.


Loves fruit juice
Loves RedBull
YES NO
1 3
Loves RedBull
YES NO
2 1
Impure
Loves soda
Pure
Loves RedBull
YES NO
3 1
Loves RedBull
YES NO
0 3

Since the second tree had one pure leaf, we can say that it performs better in predicting if a person likes RedBull. But, this alone isn't satisfactory, and it would make more sense if we were able to quantify this impurity . This is done using :-

1) Gini Impurity
2)Entropy and information gain

Gini impurity is the more popular one and we will be using that to build our tree as well.

Since the second tree had one pure leaf, we can say that it performs better in predicting if a person likes RedBull. But, this alone isn't satisfactory, and it would make more sense if we were able to quantify this impurity . This is done using :-

1) Gini Impurity
2)Entropy and information gain

Gini impurity is the more popular one and we will be using that to build our tree as well.

QR code for Q1

GINI IMPURITY

$$Gini\ Index\ (GI) = \sum_{i=1}^{K} p_i^2$$

where K is the number of class labels,

p_i is the proportion of i^{th} class label

$$Gini\ Impurity = 1 - \sum_{i=1}^{K} p_i^2$$

$$= 1 - Gini\ Index$$

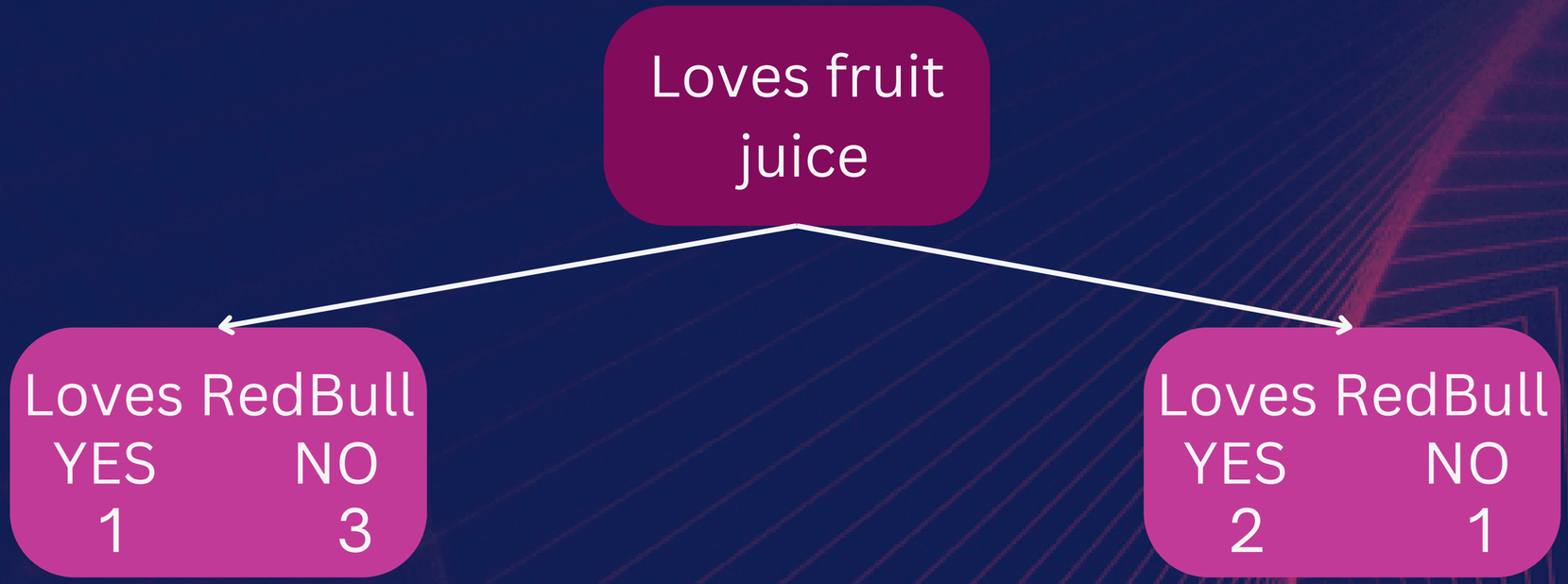
where K is the number of class labels,

p_i is the proportion of i^{th} class label

In our case, Gini impurity of leaf =

1 - (probability of yes)^2 - (probability of no)^2

This calculation is done for the two leaves arising from a node and then to find the total Gini impurity of the Decision node, we take the weighted average of the two individual Gini impurity values



Let’s now calculate the Gini impurity for the two leaves of this decision tree.

Right leaf

Gini Impurity for a Leaf = 1 - (the probability of "Yes")^2 - (the probability of "No")^2

$$= 1 - \left(\frac{2}{2+1}\right)^2 - \left(\frac{1}{2+1}\right)^2$$

$$= 0.444$$

And when we do the math we get **0.444**.

Left leaf

Gini Impurity for a Leaf = 1 - (the probability of "Yes")^2 - (the probability of "No")^2

$$= 1 - \left(\frac{1}{1+3}\right)^2 - \left(\frac{3}{1+3}\right)^2$$

$$= 0.375$$

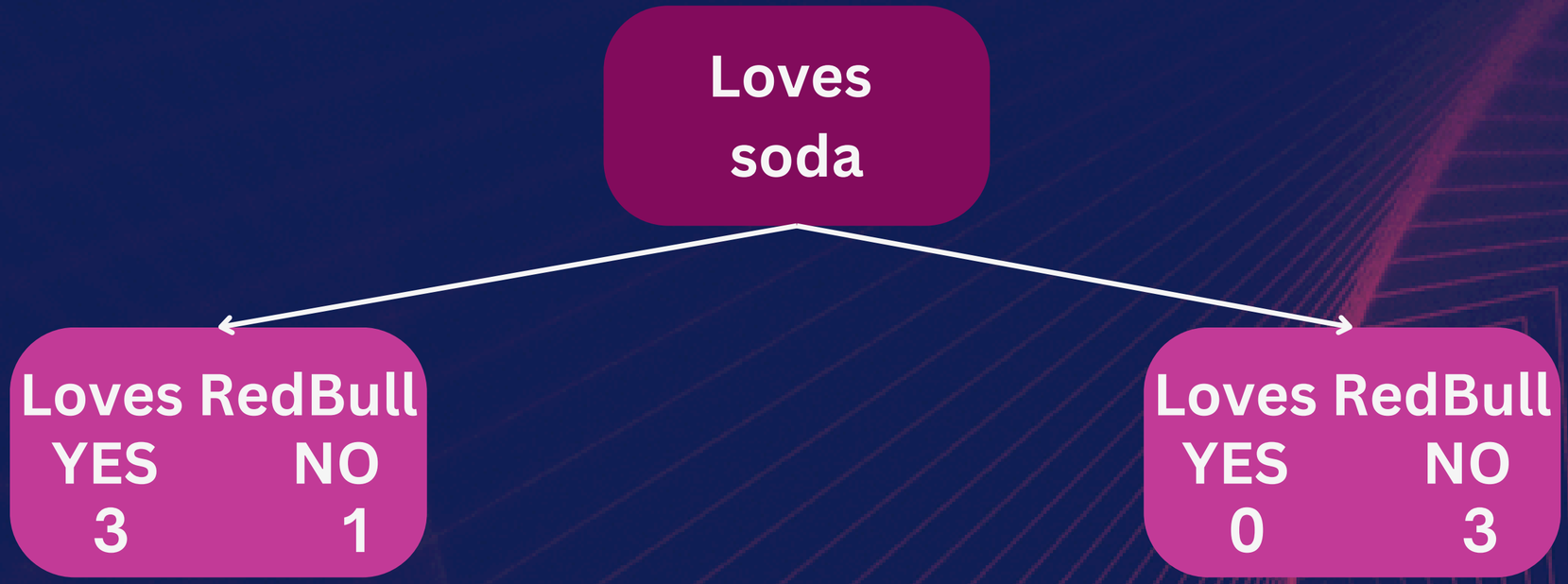
And when we do the math, we get **0.375**.

Total **Gini Impurity** = weighted average of **Gini Impurities** for the **Leaves**

$$= \left(\frac{4}{4+3}\right) 0.375 + \left(\frac{3}{4+3}\right) 0.444$$

And when we do the math, we get **0.405**.

So the Gini impurity for ‘Loves fruit juice’ node is 0.405



Similarly, Gini impurity for this node is 0.214

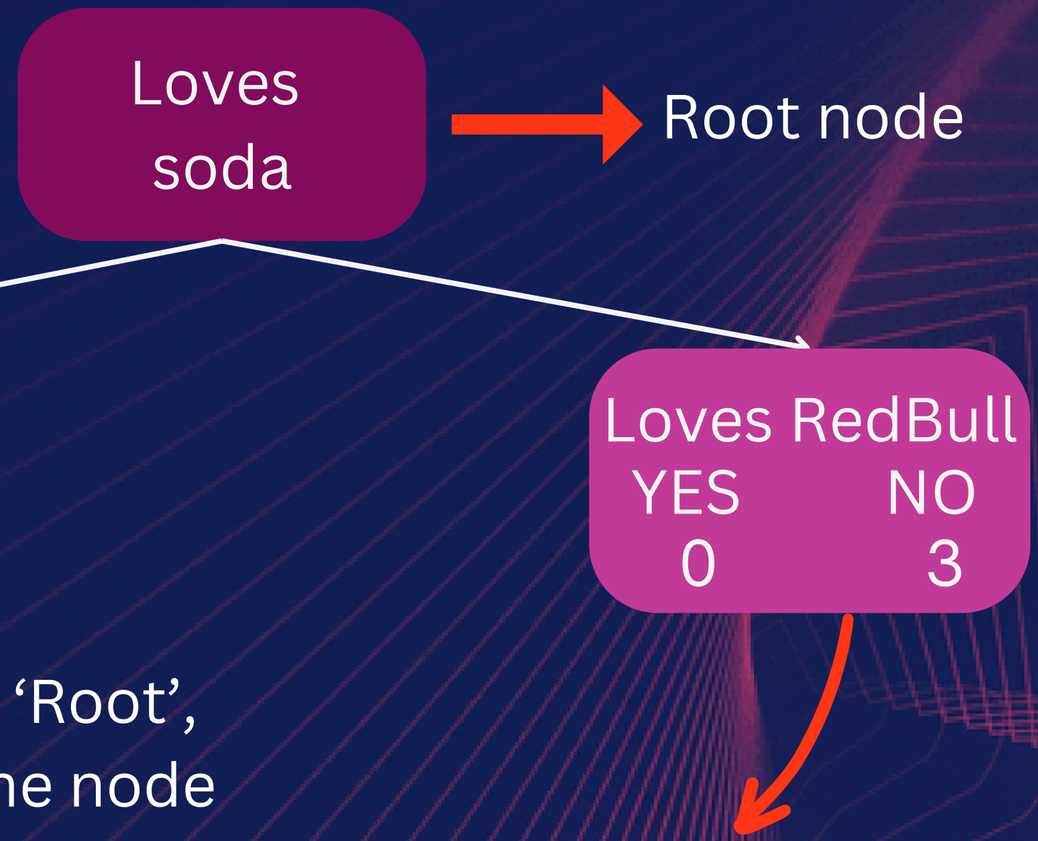
Now, we need to calculate Gini impurity for the Age column. This is less straightforward as compared to the other two as we now have numeric data. To proceed, we find the average of age value of consecutive entries and then calculate the Gini impurity for those averages.

Clearly, the leaf on the left is pure and has gini impurity = 0 and the leaf on the right has gini impurity 0.5.

On taking the weighted mean, we get the value of Gini impurity to be (1/7)*0 + (6/7)*0.5 which is *0.429*

AVG AGE	GINI IMPURITY
9.5	0.429
15	0.343
26.5	0.476
36.5	0.476
44	0.343
66.5	0.429

When we perform similar calculations for the remaining age values, we get the following table Here, the lowest Gini impurity value is 0.343 which is still higher than the one we got for ‘Loves soda’. Thus we choose ‘Loves soda’ as the root.



Now that we have our ‘Root’, we shift our focus to the node on the left here (Since it isn’t pure). This node contains all those people who ‘love soda’. We can now restrict ourselves to this smaller dataset (group of people in this case)

Adding Branches

Loves fruit juice	Loves soda	Age	Loves RedBull
Yes	Yes	7	No
No	Yes	18	Yes
No	Yes	35	Yes
Yes	Yes	38	Yes

We can split the left node further based on whether they love fruit juice or based on their age. To choose between these two options, we use Gini impurity again.

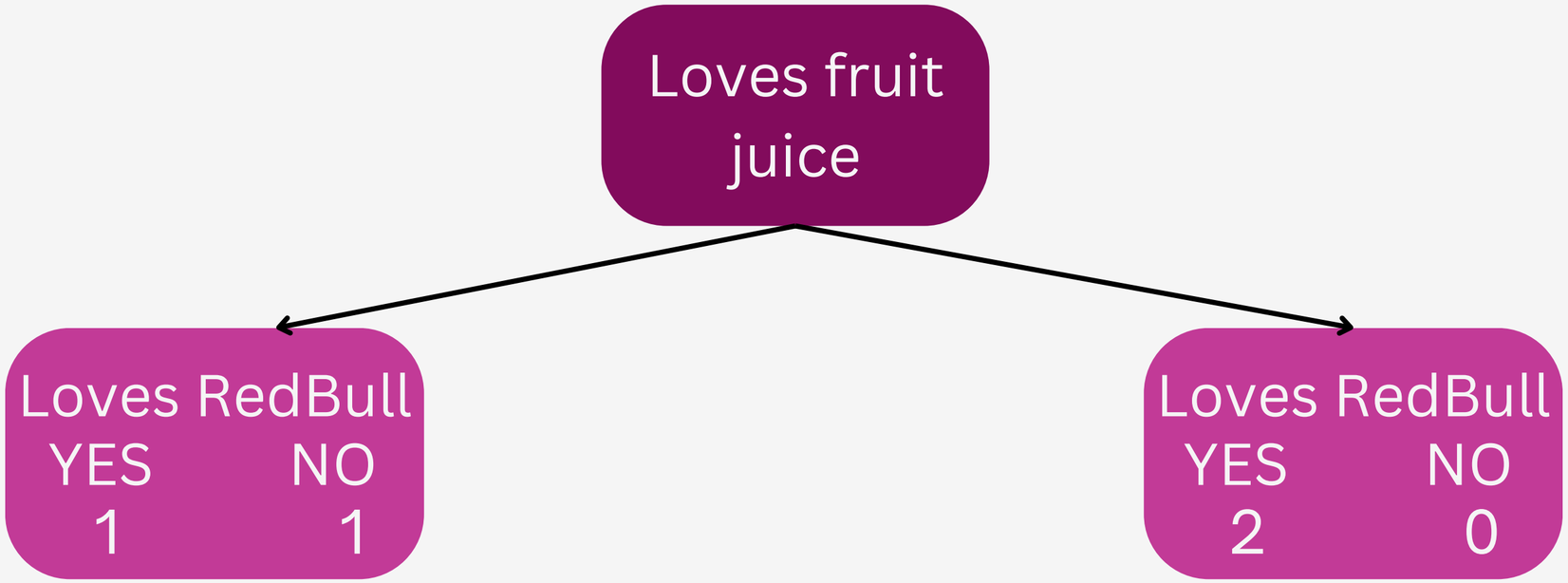
Loves fruit juice	Loves soda	Age	Loves RedBull
Yes	Yes	7	No
No	Yes	18	Yes
No	Yes	35	Yes
Yes	Yes	38	Yes

12.5

26.5

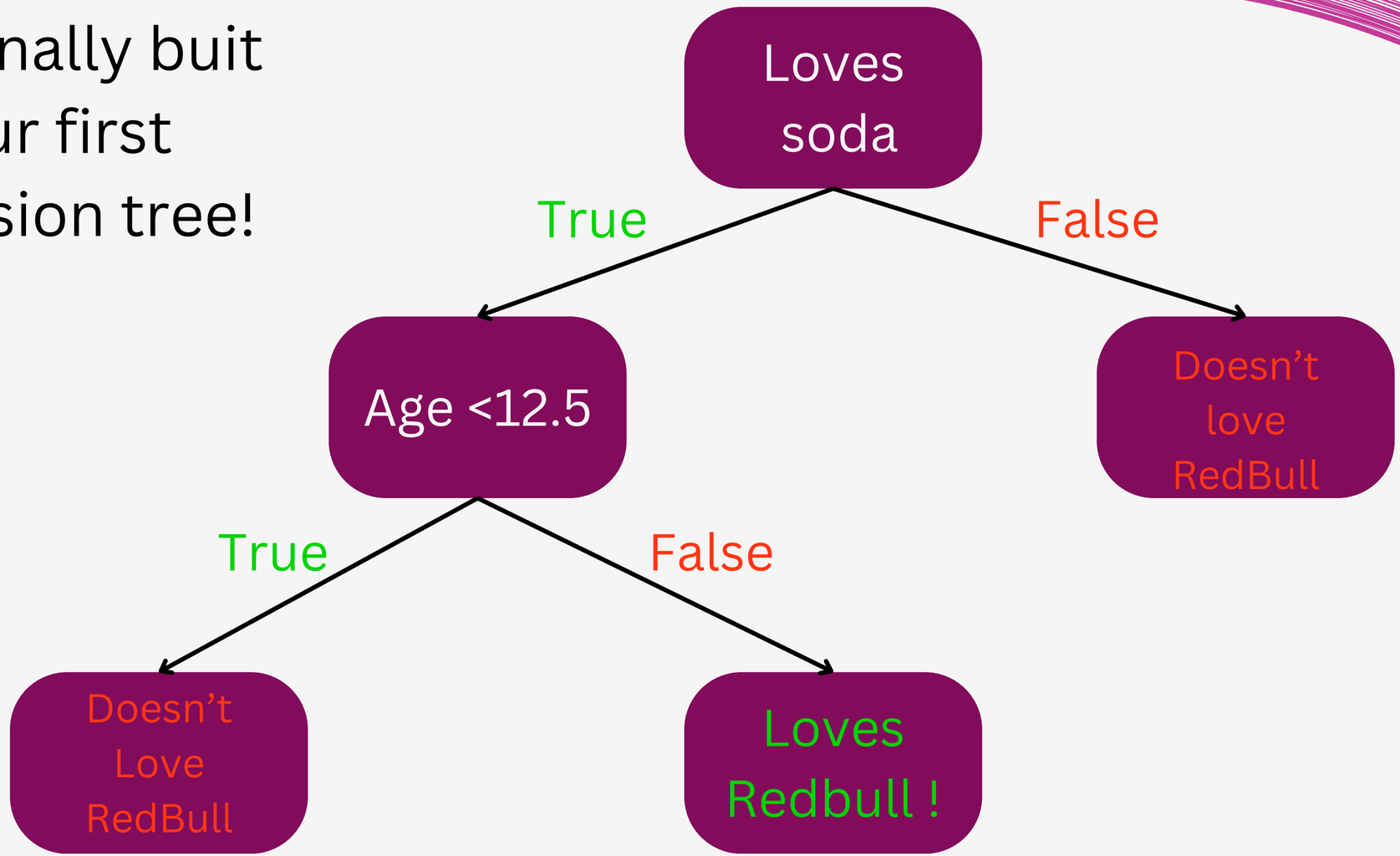
36.5

We take the average ages again and when we choose 12.5, we get a perfect split leading to Gini impurity =0. Now we need to compare this with ‘Loves Fruit juice’.



The Gini impurity of the left node is 0.5 and right node is 0. This then gives the total Gini impurity of this split to be (1/2)*0.5 + (1/2)*0.5 which is 0.25 (This is more than 0). Thus, we choose ‘Age<12.5’ to make the next and final split.

We finally buit our first Decision tree!



It's clear that there are no other splits possible and we are left with only the 'Pure' leaf nodes. That's when we realise that the tree has been constructed.

This tree can now be used to predict if a person likes RedBull or not. While building the tree, to measure impurity, we used ‘Gini impurity’. Now we will look at the alternate methods to measure impurity which we had mentioned earlier.

ENTROPY AND INFORMATION GAIN

Entropy

Entropy is measure of disorder or impurity in a dataset. It is given by the formula :-

$$Entropy = -\sum_{i} P(i) \cdot log_2 P(i)$$

The higher the entropy, the harder it is to draw any conclusion. When the tree finally reaches the terminal or leaf node maximum purity is added and entropy is zero.

Information gain

The Information Gain measures the expected reduction in entropy. Entropy measures impurity in the data and information gain measures reduction in impurity in the data

It is used to decide which feature to split on at each step in building the tree. It is given by :-

Information gain = E(S) - weighted average

Information gain of a parent node can be calculated as the entropy of the parent node subtracted entropy of the weighted average of the child node. This can be understood with a help of an example.

Observations	COLOR	Outcome
1	Red	Yes
2	Red	No
3	Yellow	Yes
4	Yellow	Yes
5	Red	Yes
6	Yellow	Yes
7	Red	No
8	Red	No
9	Red	Yes
10	Yellow	No

Consider this dataset with feature being the colours with two possible outcomes- Yes or No. The entropy for this dataset is given by -

$$E(S) = -(P_{yes}\log_2 P_{yes} + P_{no}\log_2 P_{no})$$

And the value is -

$$E(S) = -(6/10 * \log_2 * 6/10 + 4/10 * \log_2 * 4/10) \approx 0.971$$

Now, if we split it based on colour, we get two nodes whose entropy can again be calculated.

$$E(S_{Yellow}) = -\left(\frac{3}{4} * \log_2 \frac{3}{4} + \frac{1}{4} * \log_2 \frac{1}{4}\right) \approx 0.811$$

$$E(S_{Red}) = -\left(\frac{3}{6} * \log_2 \frac{3}{6} + \frac{3}{6} * \log_2 \frac{3}{6}\right) = 1$$

$$\text{Weighted average} = \frac{6}{10} * E(S_{Red}) + \frac{4}{10} * E(S_{Yellow})$$

$$= \frac{6}{10} * 1 + \frac{4}{10} * 0.811$$

$$= 0.924$$

Information Gain (S, Color) = E(S) – Weighted Average

= 0.971 - 0.924 ≈ - 0.047

Clearly, since the information gain is negative, the split up based on colour is not favourable.

QR code for Q2

CODE IMPLEMENTATION