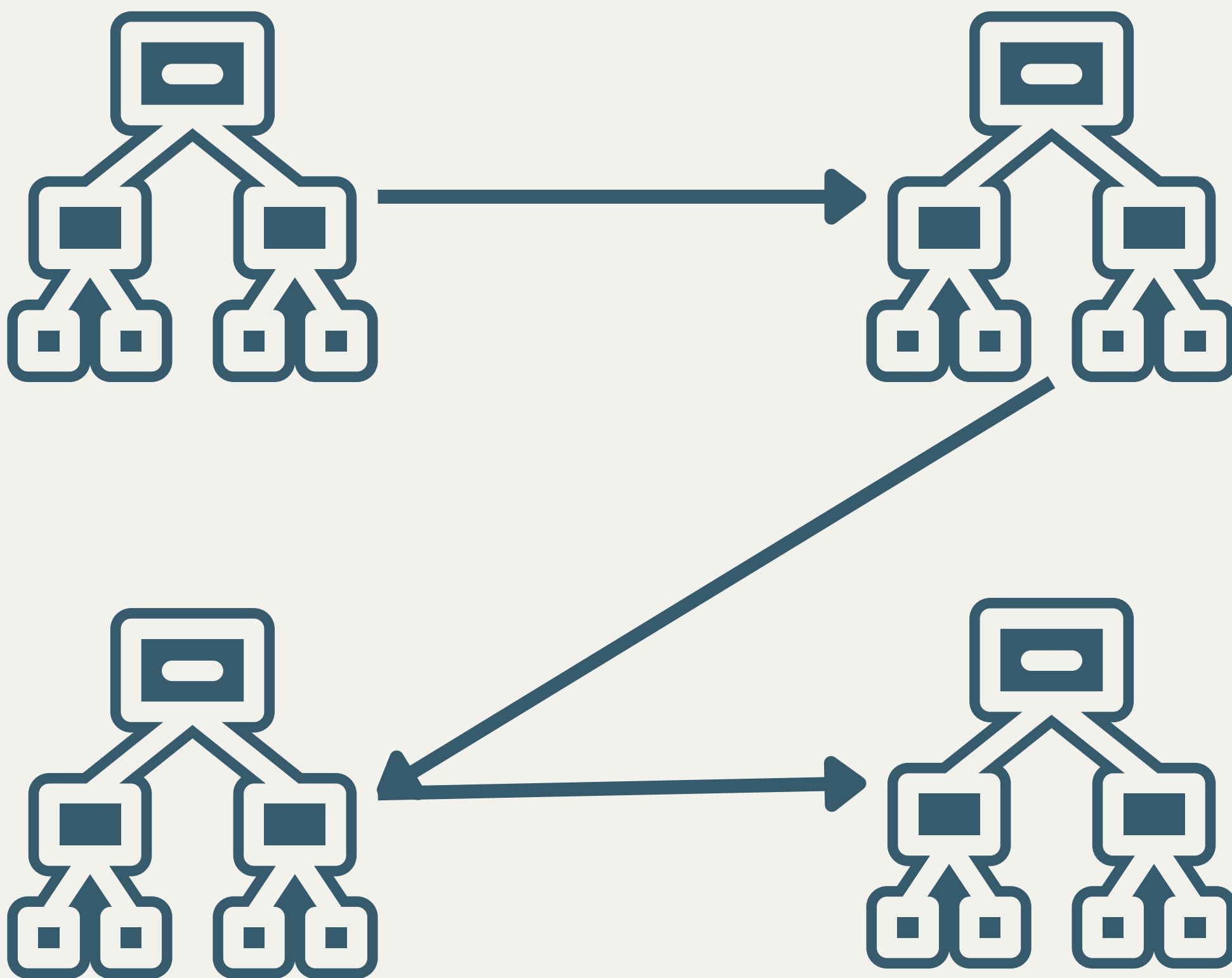




[linkedin.com/in/vikrantkumar95](https://www.linkedin.com/in/vikrantkumar95)

AdaBoost

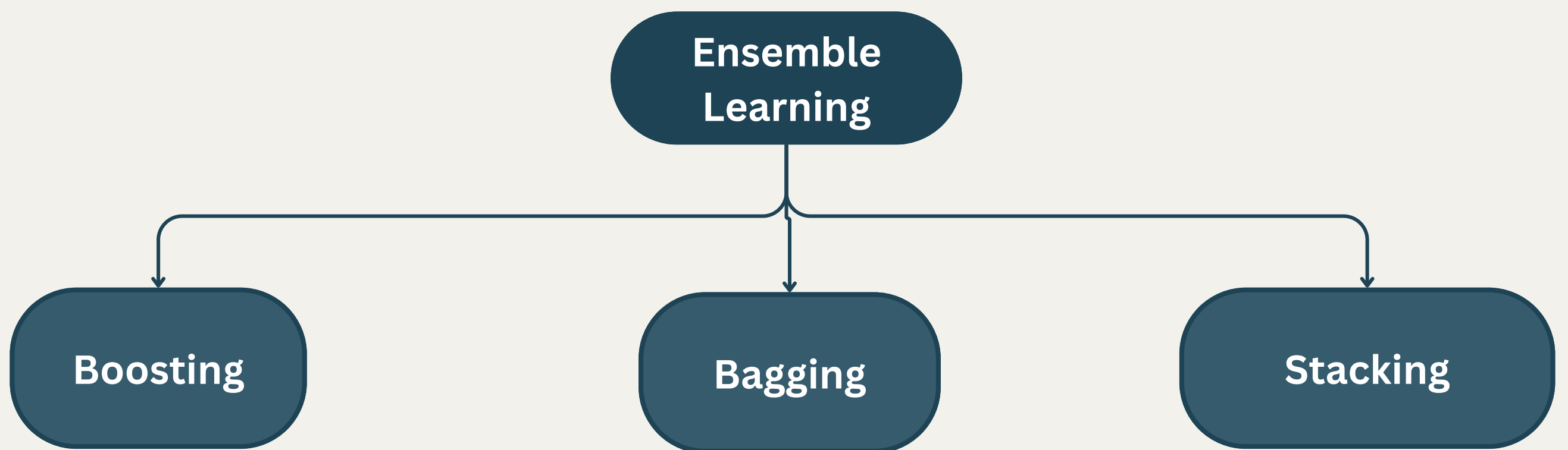
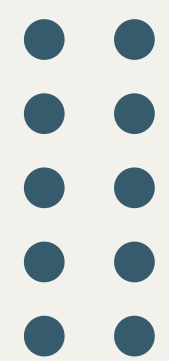
Clearly Explained



First, What is Ensemble Learning?

In simple terms, **Ensemble learning** is a machine learning method that **combines multiple models** in-order to achieve better performance than a single model.

There's three main types of Ensemble Learning methods:



A **sequential** approach that trains models to correct the errors of previous ones, increasing the weight of misclassified instances

Ex: AdaBoost, Gradient Boosting, and XGBoost

A method that trains multiple models in **parallel** on different subsets of the data and **aggregates** their predictions to improve stability and accuracy

Ex: Random Forest

Combines multiple models by training a **meta-model** to make a final prediction based on the predictions of the base models

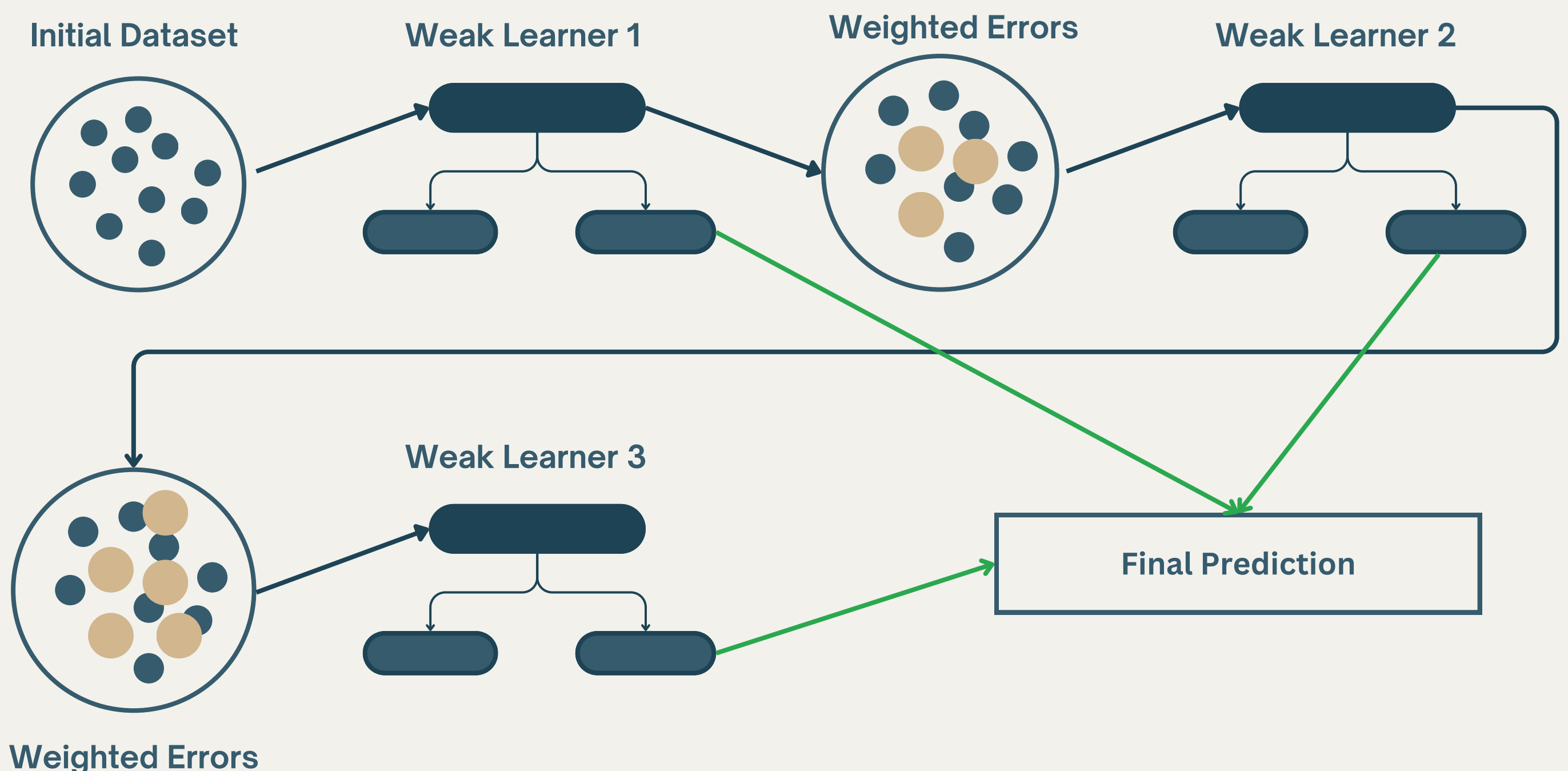
Ex: Combination of base models such as SVM, Decision Trees & Neural Networks

In this guide, we'll take a look at **Adaboost** which, as the name suggests, is a **Boosting** Ensemble Method.

What is Boosting?

As we saw in the previous page, Boosting is a sequential algorithm that uses multiple weak learners to construct a model that performs better than its components.

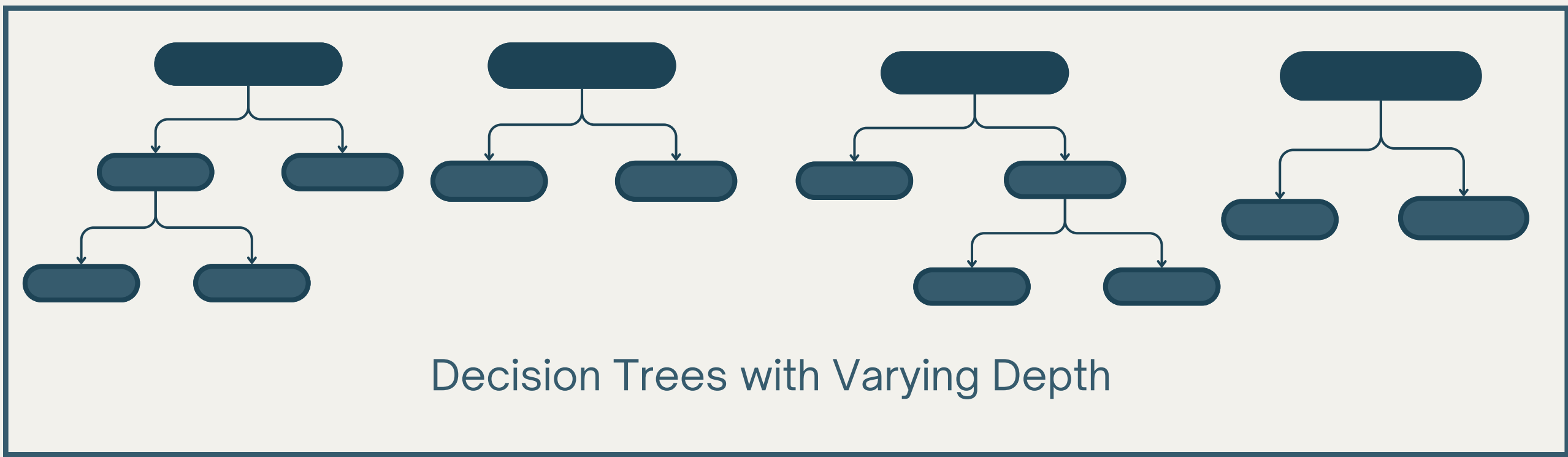
1. **Initial Model Training:** A weak learner is trained on the dataset, attempting to predict as accurately as possible.
2. **Weight Adjustment:** Data points misclassified by this initial model are given higher importance.
3. **Sequential Learning:** The next model focuses on these weighted misclassifications, aiming to correct them.
4. **Iterative Process:** Steps 2 and 3 repeat, with each subsequent weak learner adjusting based on the errors of its predecessor, until the ensemble comprises the intended number of models.
5. **Aggregate Prediction:** The final model's prediction is the combined outcome of all weak learners, significantly improving accuracy over any individual learner.



AdaBoost vs Random Forest (1/3)

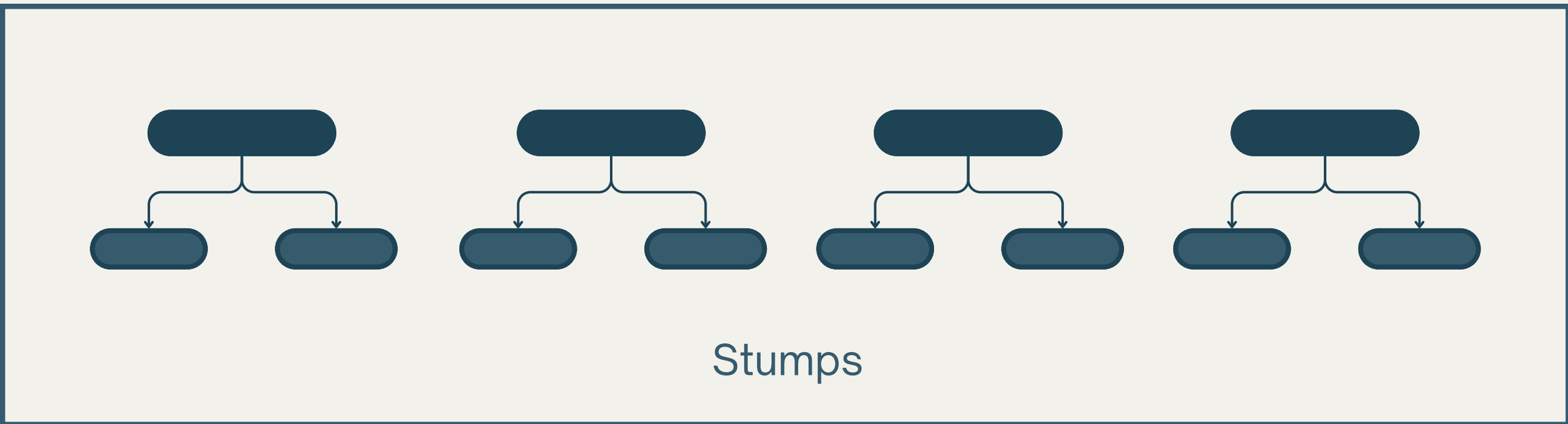
We'll look at how exactly the **AdaBoost** algorithm works. However, before that, let's see how it differs from **Random Forest**, a Bagging algorithm.

As we saw in our Random Forest Chapter (check out my profile if you've not seen it), a Random Forest is made up of multiple Decision Trees of **varying depth**. There is no pre-determined maximum depth.



Random Forest

AdaBoost on the other hand is usually comprised of multiple Decision Trees with only **two leaf nodes**. These Decision Trees are called **Stumps** and are what we earlier referred to as Weak Learners

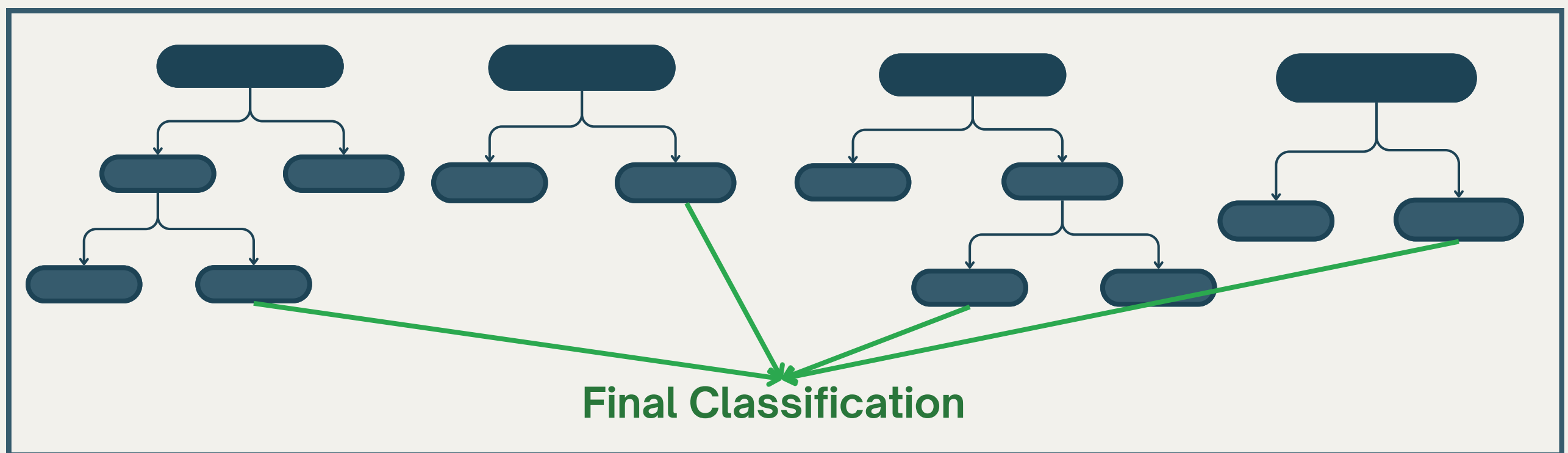


AdaBoost



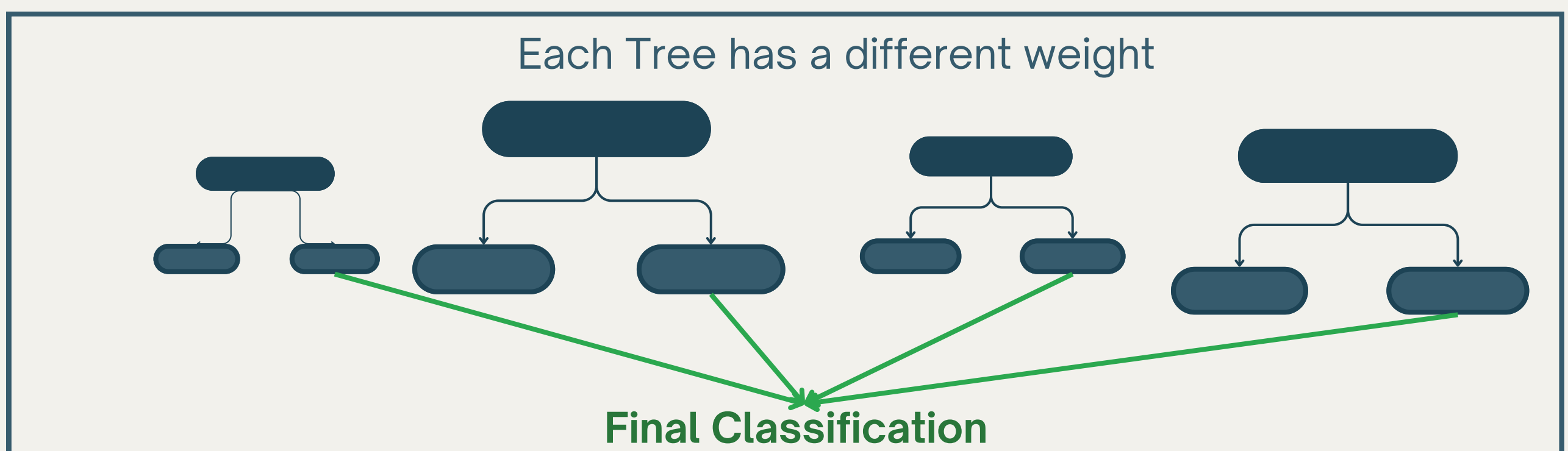
AdaBoost vs Random Forest (2/3)

In a Random Forest, the final classification is made by counting the classification of each Tree and picking up the class with the majority vote. Hence, each Tree in the forest has **equal vote** in the final classification.



Random Forest

With AdaBoost on the other hand, some Trees (i.e Stumps) have a **higher vote** in the final classification as compared to the others. We'll see later how exactly is each Stump is constructed and weighted.



AdaBoost



[linkedin.com/in/vikrantkumar95](https://www.linkedin.com/in/vikrantkumar95)



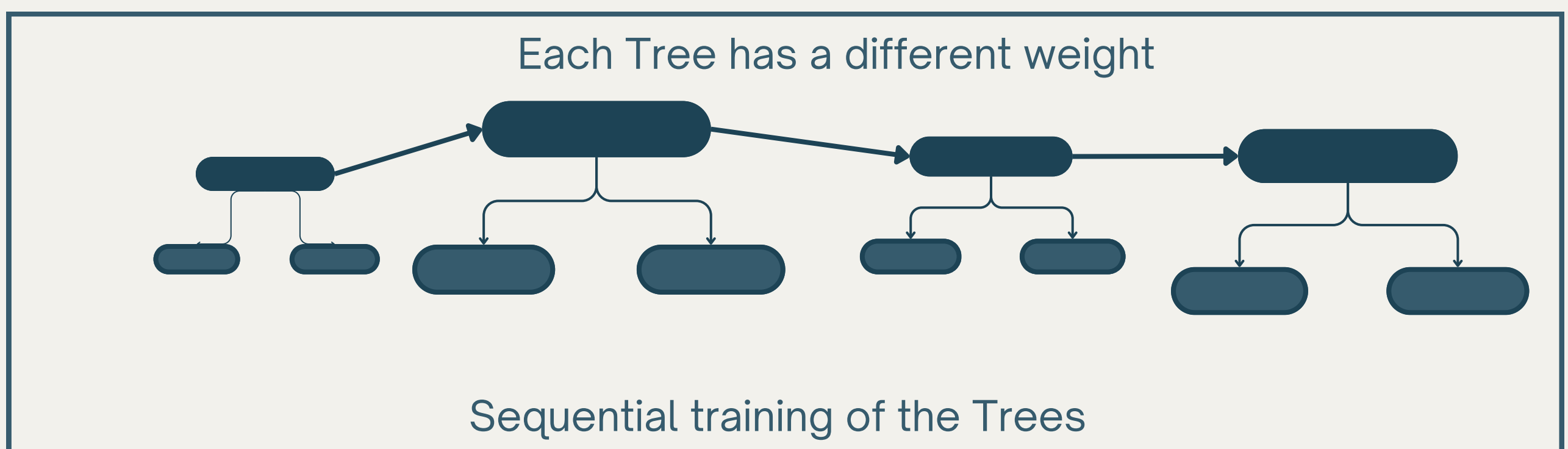
AdaBoost vs Random Forest (3/3)

In a Random Forest, each Tree is **trained independently**. It doesn't matter which Tree is trained first. Hence, the Tree can be trained in parallel to speed up the training time.



Random Forest

In AdaBoost, the Trees are trained sequentially. The **order** of training of the Trees matters. The errors in classification from the first Tree influence the training of the second tree. The errors from the second Tree influence the training of the third Tree and so on.



AdaBoost



[linkedin.com/in/vikrantkumar95](https://www.linkedin.com/in/vikrantkumar95)



AdaBoost Explained

To understand how AdaBoost truly works, let us train a model using a small sample dataset. The goal is to predict if a person reaches their **fitness goal** (target variable) based on their **age**, **diet**, and the number of **hours they exercise in a week** (predictor variables).

Person	Diet	Gym Membership	Age	Achieved Goal?
1	Vegan	Yes	24	0
2	Not Vegan	No	27	0
3	Vegan	No	29	1
4	Vegan	Yes	31	1
5	Not Vegan	Yes	35	1
6	Vegan	Yes	36	1
7	Not Vegan	Yes	42	0

Above is a dataset comprising 7 people. We'll go through each step in training the AdaBoost Model. As we saw earlier, this involves training Decision Trees sequentially.

Let's get to the training!

AdaBoost Explained

Let’s start working towards creating the first stump of our AdaBoost model. Here is the data with one additional column, **Sample Weight**:



Person	Diet	Gym Membership	Age	Achieved Goal?	Sample Weight
1	Vegan	Yes	24	0	1/7
2	Not Vegan	No	27	0	1/7
3	Vegan	No	29	1	1/7
4	Vegan	Yes	31	1	1/7
5	Not Vegan	Yes	35	1	1/7
6	Vegan	Yes	36	1	1/7
7	Not Vegan	Yes	42	0	1/7

As the name suggests, it’s the weight of each Data Point in the Dataset. We’ll see how these weights come into play soon. Initially, each of the data points is assigned an equal weight (we have 7 data points, so each gets 1/7 weight).

Since initially all data points have the same weight, let us ignore that column when we train our first Decision Tree.



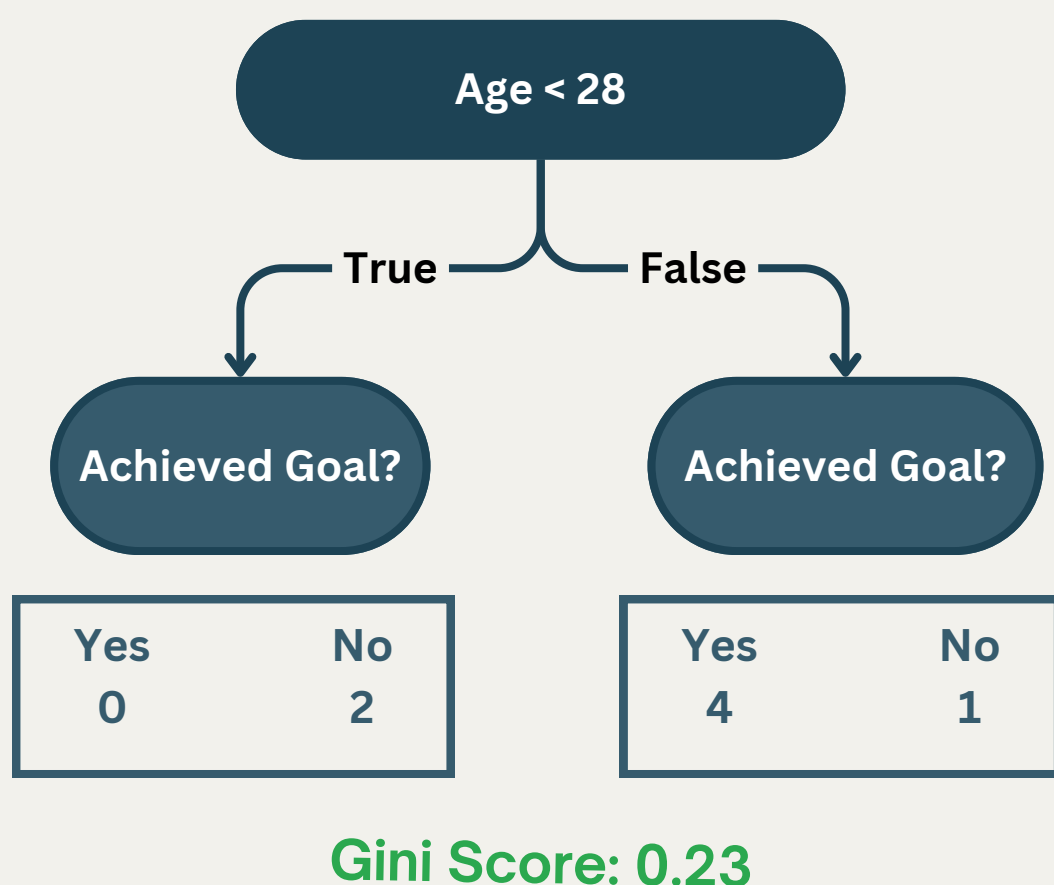
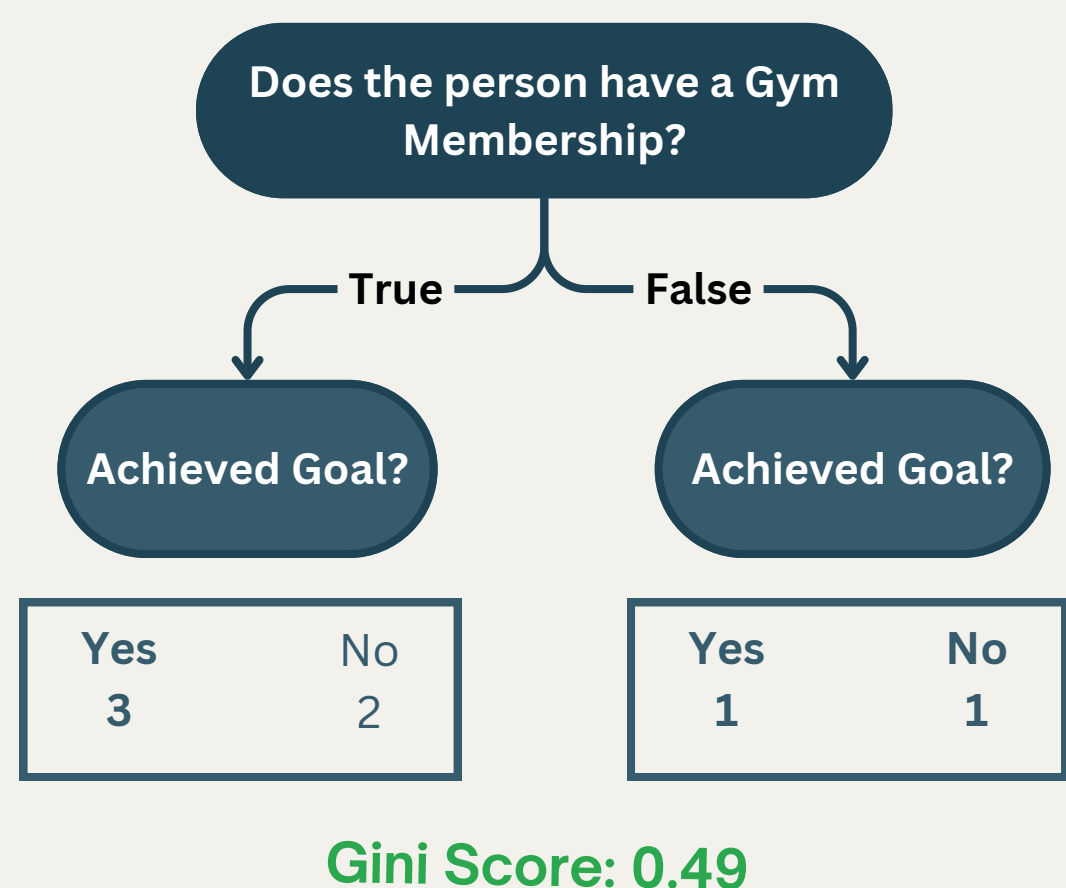
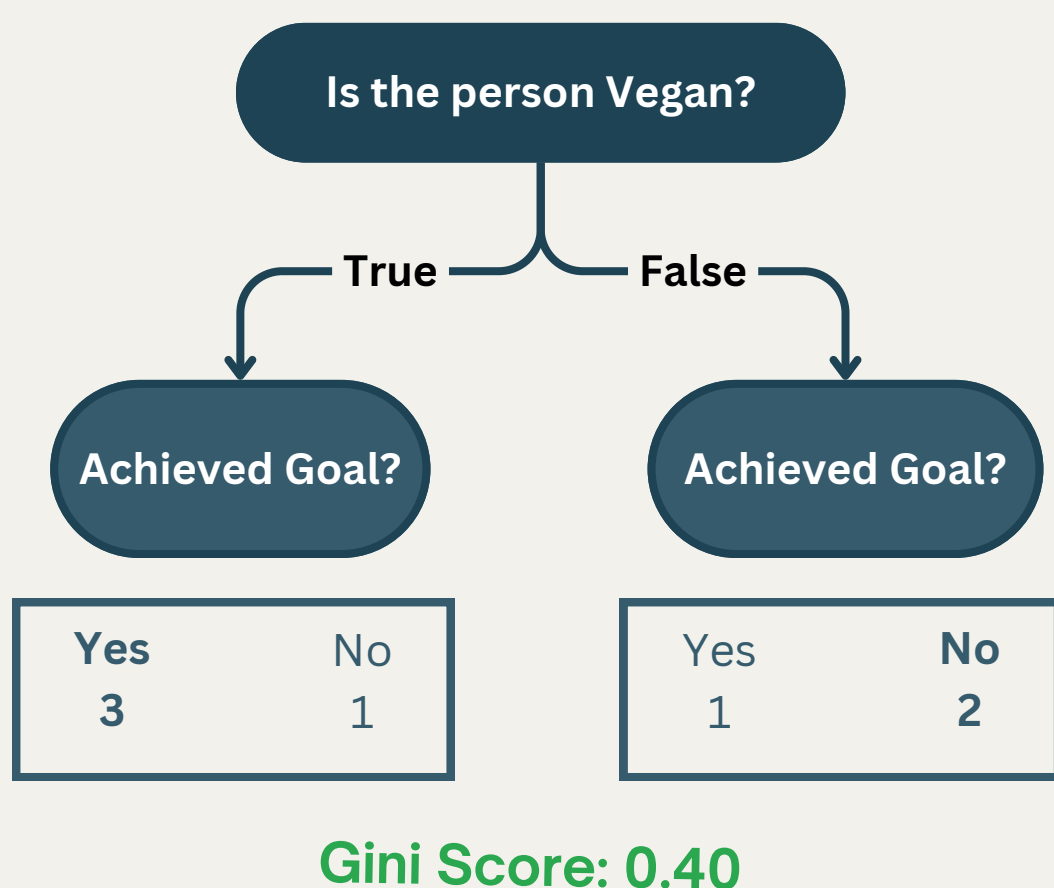
[linkedin.com/in/vikrantkumar95](https://www.linkedin.com/in/vikrantkumar95)



AdaBoost Explained

To build the first stump of our AdaBoost forest, we create one stump split on each of the three predictor variables we have.

1. First, we calculate the Gini Impurity for each stump (to see how we decided the split and calculated the Gini Impurity, check out the Decision Trees guide on my profile):

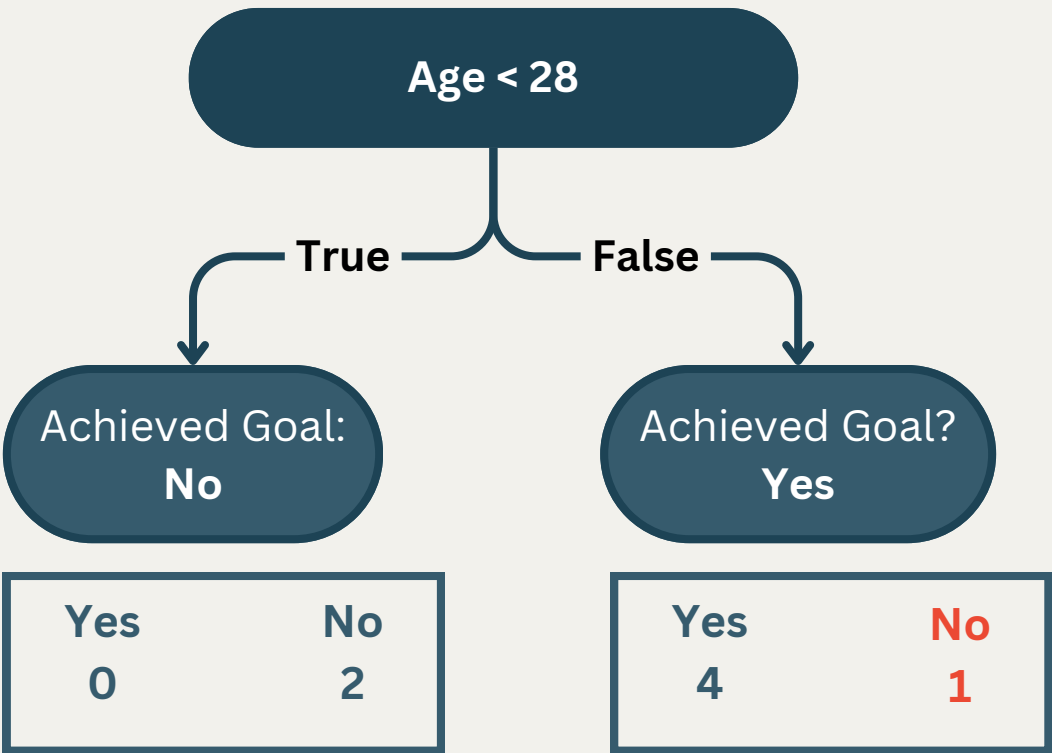


2. We pick the the tree with the lowest Gini Impurity score. Of the 3 trees, we see that the tree split on **Age** has the lowest Gini Impurity score of 0.23. This will be the first stump in our model. Also as a reminder, the classification of each the leaves is based on the majority class. So **left Leaf** will be **No** and **right leaf** will be **Yes**

As we saw before, in the Adaboost algorithm, the errors from each Stump influence the creation of the next Stump. We also saw that each stump has a weighted vote in the final classification. So first let's see how we calculate the Error and the weight each stump has.

AdaBoost Explained

As we saw earlier, we picked the stump split on Age. The stump says that people aged less than 28 do not achieve their fitness goal (ironically) while those aged 28 and above do reach their fitness goal



We see above that **one sample** on the right leaf is classified **incorrectly**. The sample was the one below. One sample person above the age of 28 did not achieve their fitness goal.

Person	Diet	Gym Membership	Age	Achieved Goal?	Sample Weight
7	Not Vegan	Yes	42	0	1/7

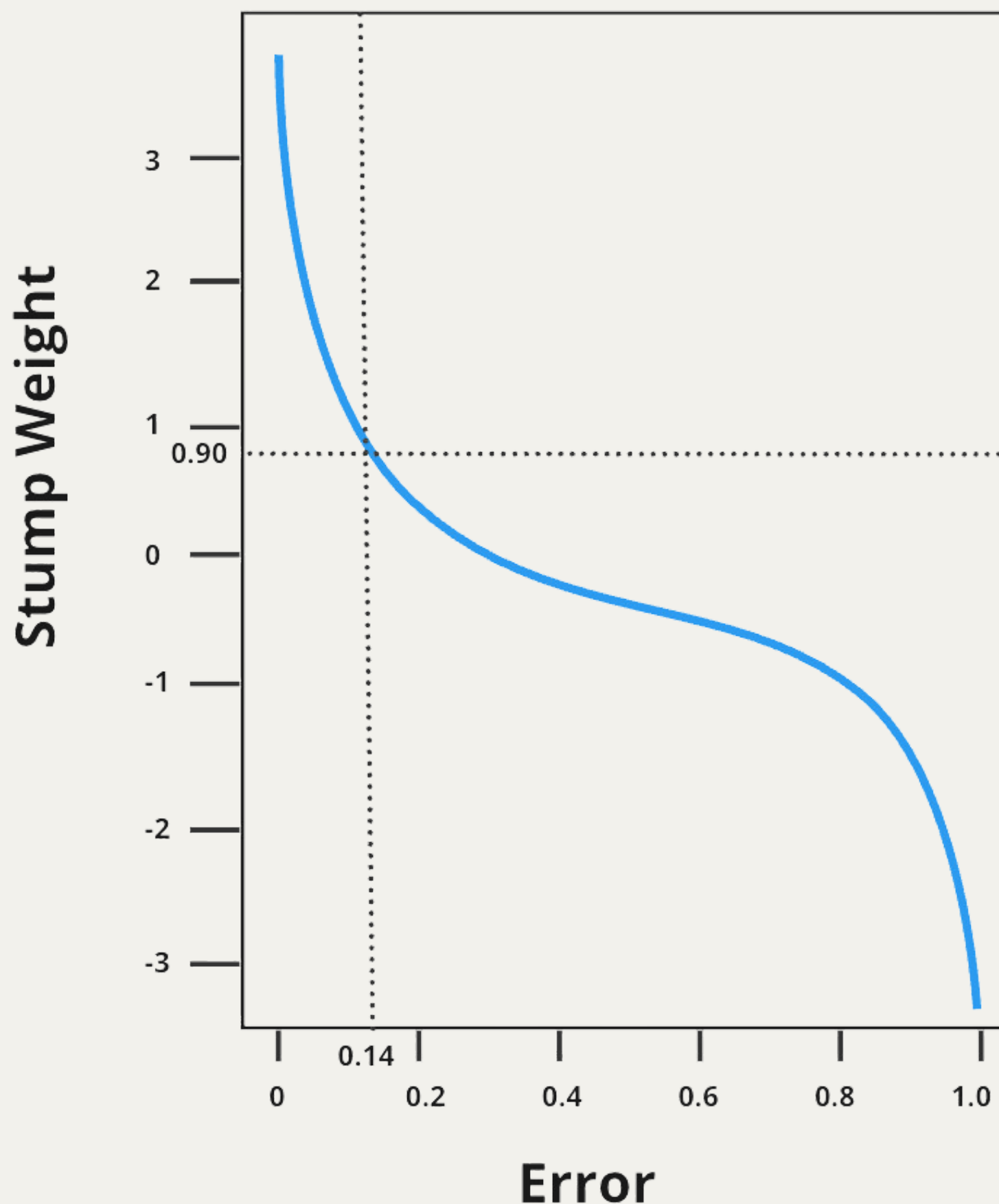
Error = Sum of the Sample Weights of the Incorrectly Classified Samples

This is where the Sample Weights come into play. The **sum of the Sample Weights** of all the **incorrectly classified samples** is the total **Error**. Since only 1 sample was incorrectly classified, the error for the first stump comes out to be **1/7**.

AdaBoost Explained

Below is the formula to calculate the weight of each Stump:

$$Weight = \frac{1}{2} \log \left(\frac{1 - Error}{Error} \right)$$



Above is the graph for the equation of the Weight. Let's plug in **1/7** (0.14) as the Error and see what we get.

$$Weight = \frac{1}{2} \log \left(\frac{1 - \left(\frac{1}{7}\right)}{\frac{1}{7}} \right) = \frac{1}{2} \log(6) = 0.90$$

As we see the in the graph, an error of 1/7 (0.14) gives the Stump a **weight** of **0.90**. This is what this tree will be weighed by for the final prediction.

AdaBoost Explained

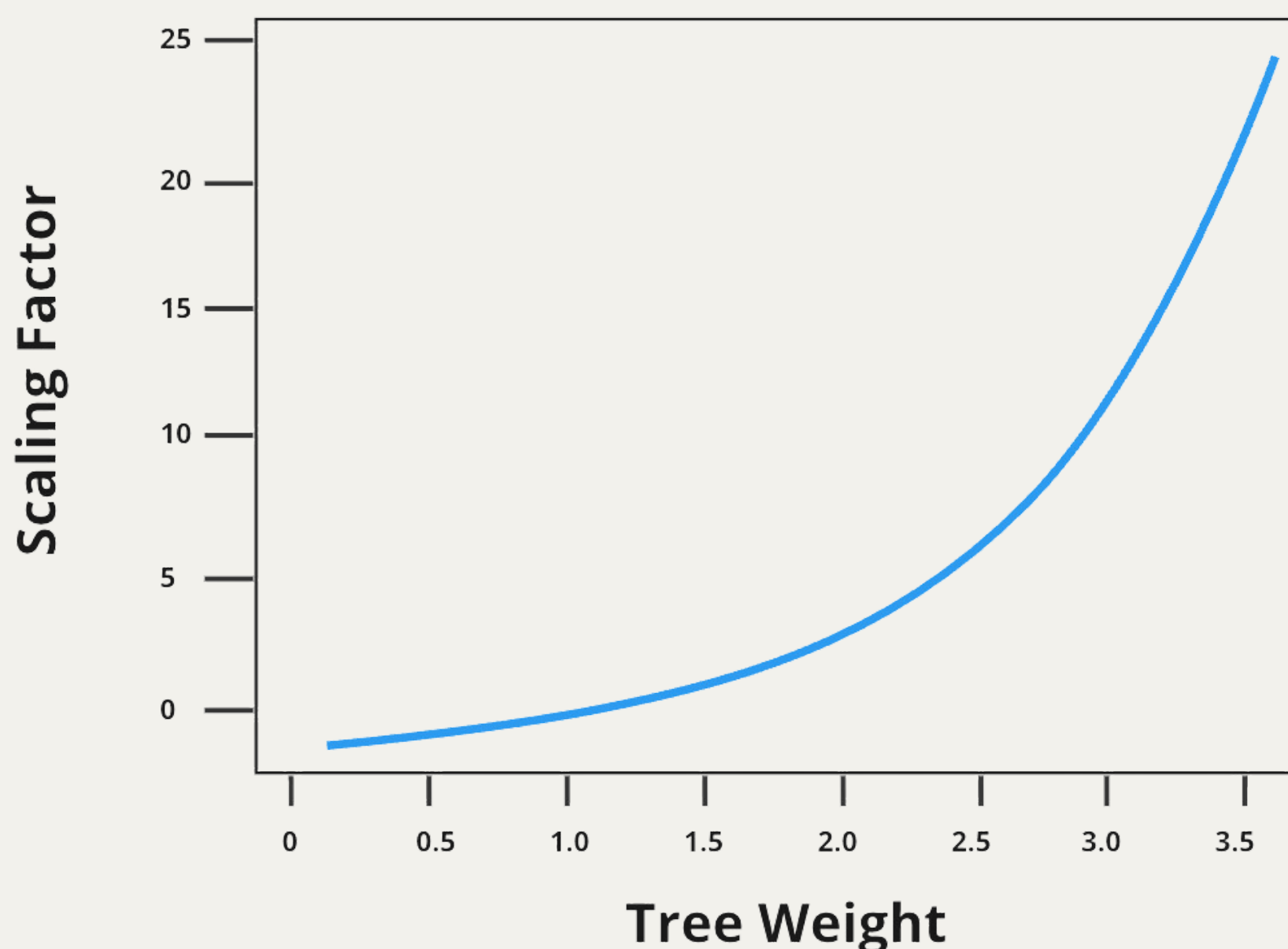
Now we also saw earlier that the **Errors** from the first Stump are given **higher importance** when we train the next Stump. This is how the AdaBoost algorithm works sequentially. We train the next Stump on the Errors in the hope that it learns to correctly predict them.

So we **increase** the sample **weight** of the **incorrectly classified samples** (only 1 sample in this case) and **decrease** the **weight** of the **correctly classified** samples (6 samples).

First let's increase the sample weight of the incorrectly classified sample, using the formula below:

$$NewWeight = SampleWeight \times e^{TreeWeight} = \frac{1}{7} \times e^{0.9} = 0.35$$

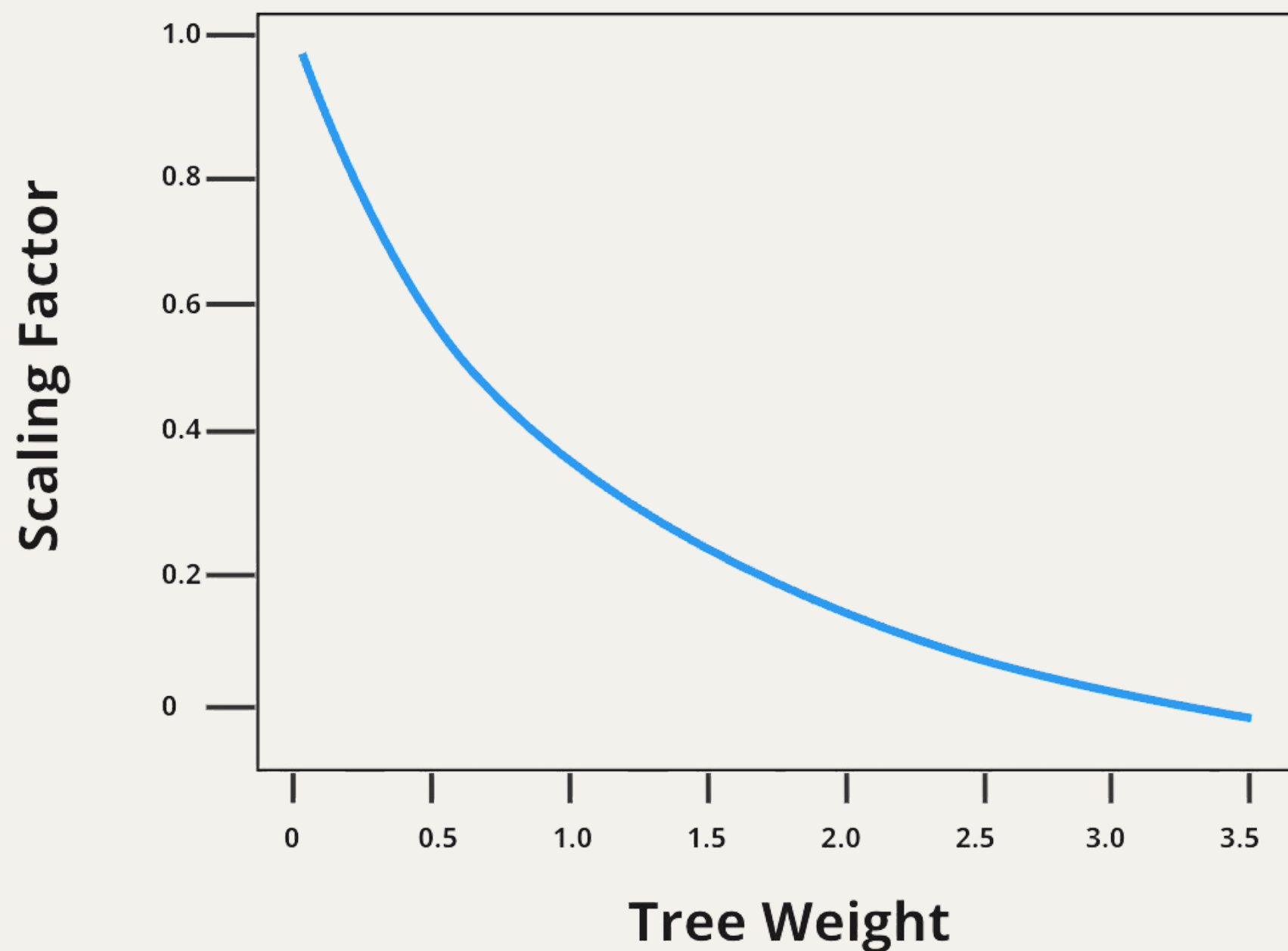
$e^{TreeWeight}$ is the **factor** by which we **scale** the **Sample Weight** by to arrive at the **new Sample Weight**. If we were to plot the graph of the Scaling Factor, it would look like below. We can see clearly that, the higher the Tree Weight (i.e the better the Stump classified the samples) the higher the New Sample weight of the incorrectly classified sample:



AdaBoost Explained

Now let's decrease the sample weight of the correctly classified samples, using the formula above

$$NewWeight = SampleWeight \times e^{-TreeWeight} = \frac{1}{7} \times e^{-0.9} = 0.058$$



$e^{TreeWeight}$ is the **factor** by which we **scale** the **Sample Weight** by to arrive at the **new Sample Weight**. If we were to plot the graph of the Scaling Factor, it would look like above. Now we see that the **higher** the **Tree Weight** (i.e the better the Stump classified the samples) the **lower** the New Sample weight of the **correctly** classified sample.

AdaBoost Explained

Now let’s see what the new Sample Weights look like:

Person	Diet	Gym Membership	Age	Achieved Goal?	Sample Weight	New Sample Weight	Norm. Weight
1	Vegan	Yes	24	0	1/7	0.058	0.083
2	Not Vegan	No	27	0	1/7	0.058	0.083
3	Vegan	No	29	1	1/7	0.058	0.083
4	Vegan	Yes	31	1	1/7	0.058	0.083
5	Not Vegan	Yes	35	1	1/7	0.058	0.083
6	Vegan	Yes	36	1	1/7	0.058	0.083
7	Not Vegan	Yes	42	0	1/7	0.35	0.501

- 1.All the correctly classified samples (Person 1 - 6) were given the lowered weight of **0.058** while the incorrectly classified sample (Person 7) was given the weight of **0.35**.
- 2.Now, we need the **sum** of all the **weights** to be **1**, so we **normalize** the weights by dividing each weight by the sum of all the weights (**0.058*6 + 0.35) = 0.698**.
- 3.The **final new weights** (Norm. Weight) then are $0.058 / 0.698 = \mathbf{0.083}$ for the incorrectly classified samples and $0.35 / 0.698 = \mathbf{0.501}$

AdaBoost Explained

Now we have to train the **second Stump**. We create a new dataset by going through the following steps:

- 1. Choose a number at **random** between **0** and **1**.
- 2. Use the **New Sample Weights as a distribution** and see where the number lies. For example if the number we choose randomly turns out to be between 0 and 0.083, we take Person 1 in the new dataset. If the number we choose is between 0.083 and 0.166 ($0.083 + 0.083$), then we pick Person 2.
- 3. Now for all the numbers that come out to be between 0.501 and 1 we choose Person 7. Since this is a much bigger interval, we are likely to end up with more duplicate Person 7 samples in the new dataset.
- 4. Having **more Person 7 samples** in the training dataset for Stump 2 ensures that these duplicate samples will be correctly classified by the new Stump.

Person	Diet	Gym Membership	Age	Achieved Goal?	New Sample Weight
1	Vegan	Yes	24	0	0.083
2	Not Vegan	No	27	0	0.083
3	Vegan	No	29	1	0.083
4	Vegan	Yes	31	1	0.083
5	Not Vegan	Yes	35	1	0.083
6	Vegan	Yes	36	1	0.083
7	Not Vegan	Yes	42	0	0.501

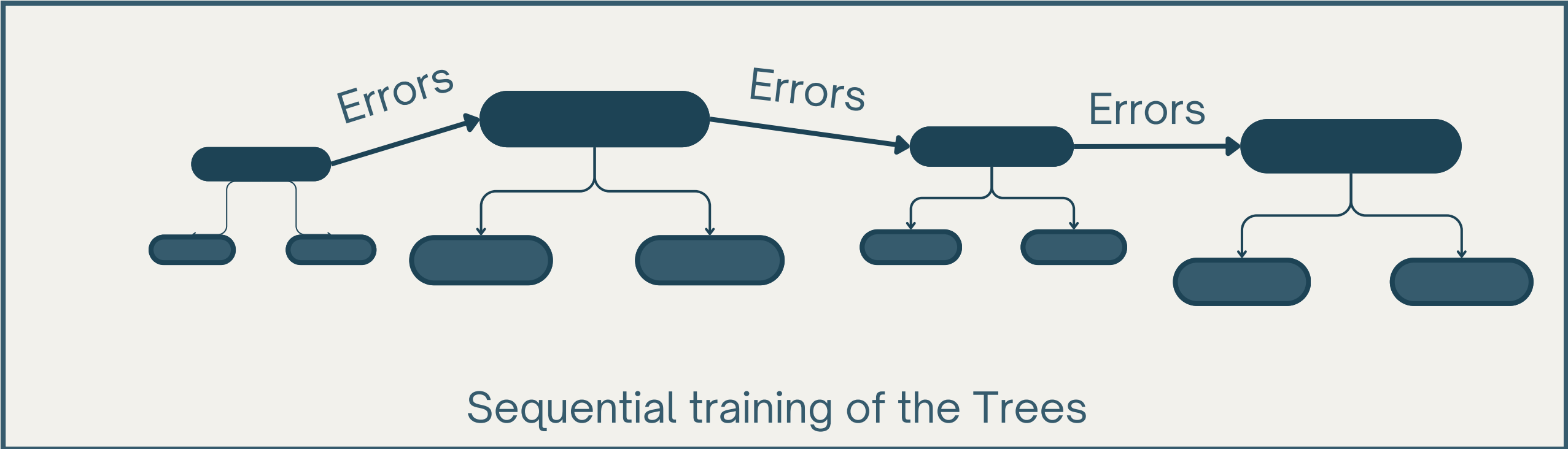
AdaBoost Explained

After going through the process of choosing 7 random numbers between 0 & 1, we arrive at the following **new Dataset** to train the second Stump on:

Person	Diet	Gym Membership	Age	Achieved Goal?	Sample Weight
7	Not Vegan	Yes	42	0	1/7
3	Vegan	No	29	1	1/7
7	Not Vegan	Yes	42	0	1/7
6	Vegan	Yes	36	1	1/7
7	Not Vegan	Yes	42	0	1/7
5	Not Vegan	Yes	35	1	1/7
7	Not Vegan	Yes	42	0	1/7



We see that Person 7 ended up coming 4 times in the new Dataset. We also **reset the weights** back to 1/7 for each sample and **repeat the process** we followed for first Stump to train the second Stump. This is how we **sequentially train stumps using errors from the previous Stump**.



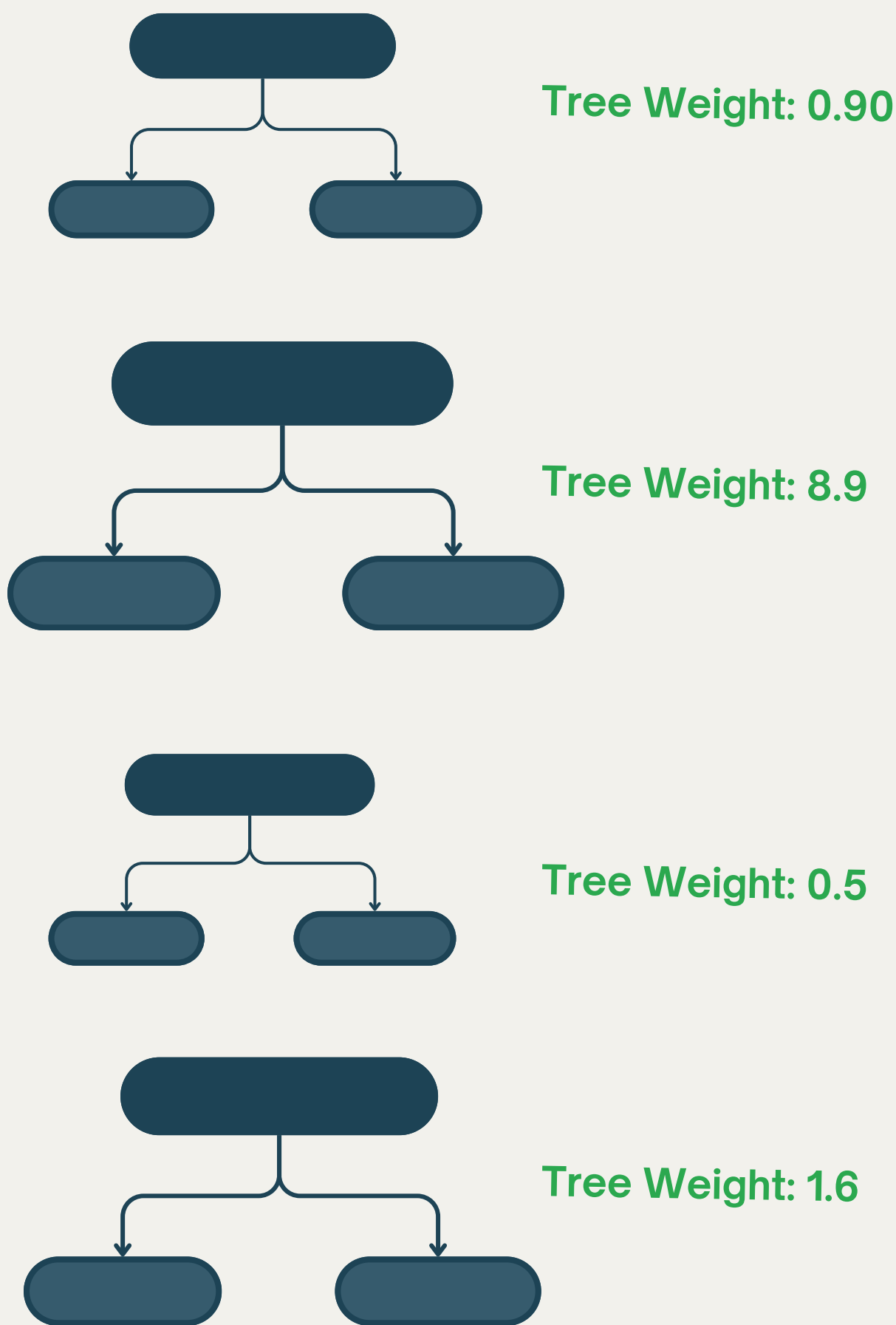
AdaBoost Explained

Now imagine our AdaBoost model has 6 trees. Below are the Stumps that classified a new Person as having achieved their Fitness Goal while the ones on the right are the Stumps that classified the Person as not having achieved their Fitness Goal. Let's see what the final classification is, based on the Tree Weight of each Stump:



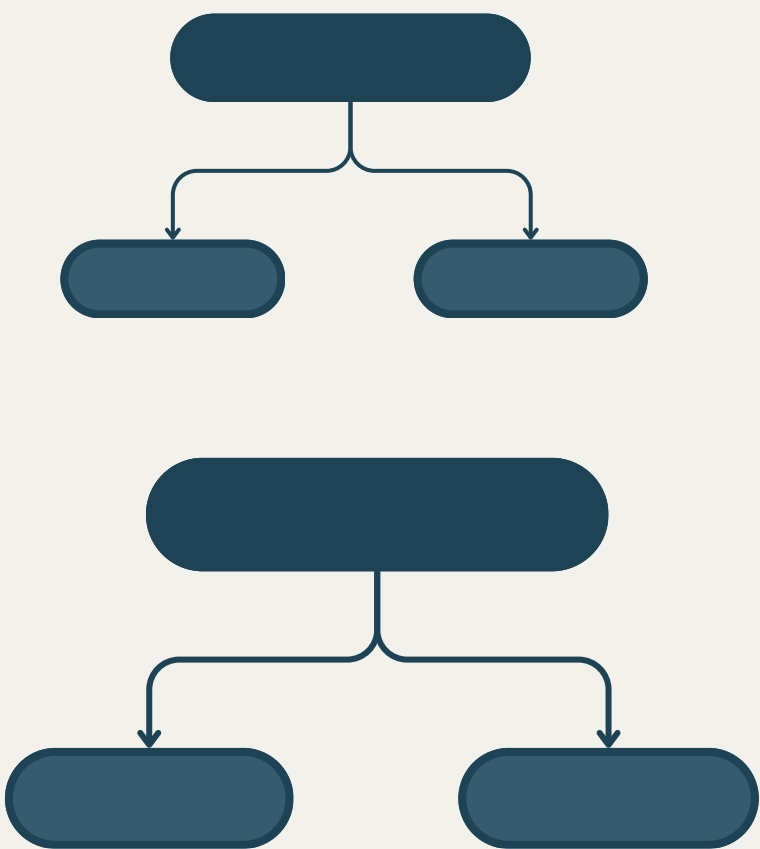
Achieved Goal:
Yes

Achieved Goal:
Yes



Tree Weight: 1.7

Tree Weight: 3.8



Total Weight = 11.9

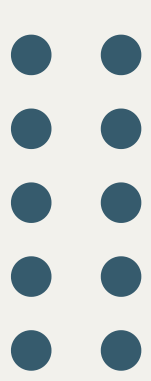
Total Weight = 5.5

Since the trees on the **left** have a **higher total weight**, we will classify the Person as having **achieved** their **Fitness Goal**!



Let's summarise

Let's summarise the steps we followed to create our AdaBoost Model:

- 
1. **Initial Model Training:** We trained the first Stump (weak learner) with equal Sample Weights given to every sample
 2. **Weight Adjustment:** The data points incorrectly classified are given a higher weight.
 3. **Sequential Learning:** The next model is trained on a new dataset that is sampled with replacement using the new sample weights. This results in more duplicates for the incorrect samples in the new dataset and allows the second Stump to learn to correctly classify these samples.
 4. **Iterative Process:** Steps 2 and 3 repeat, with each subsequent weak learner adjusting based on the errors of its predecessor, until the ensemble comprises the intended number of models.
 5. **Aggregate Prediction:** The final model's prediction is based on the prediction from the trees with the higher sum of weights.

Hope that helped explain clearly how the AdaBoost algorithm works. We'll look at other important Boosting algorithms in the future guides!



Enjoyed
reading?

Follow for
everything Data
and AI! 😊



[linkedin.com/in/vikrantkumar95](https://www.linkedin.com/in/vikrantkumar95)

