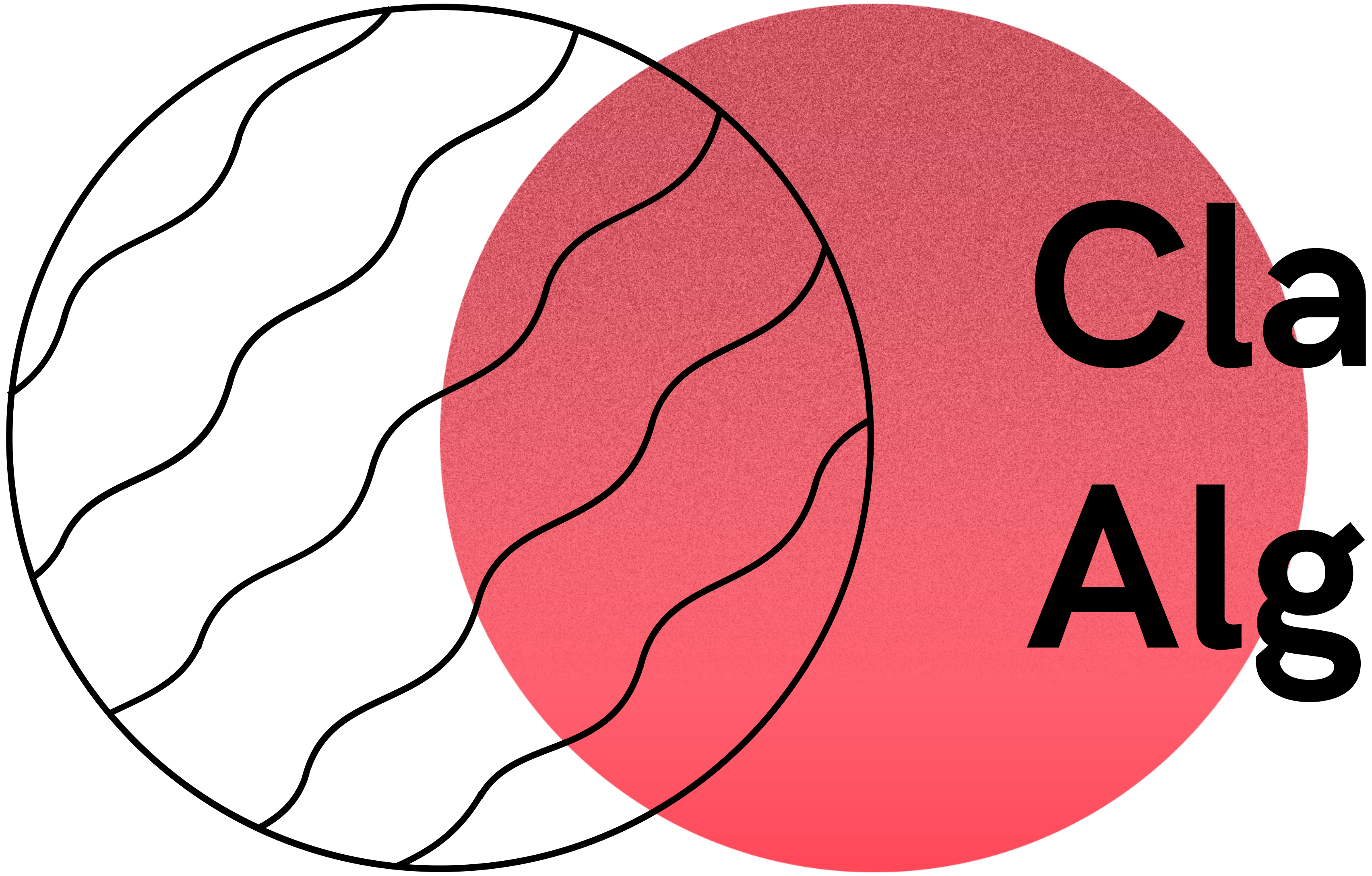


Classification Algos

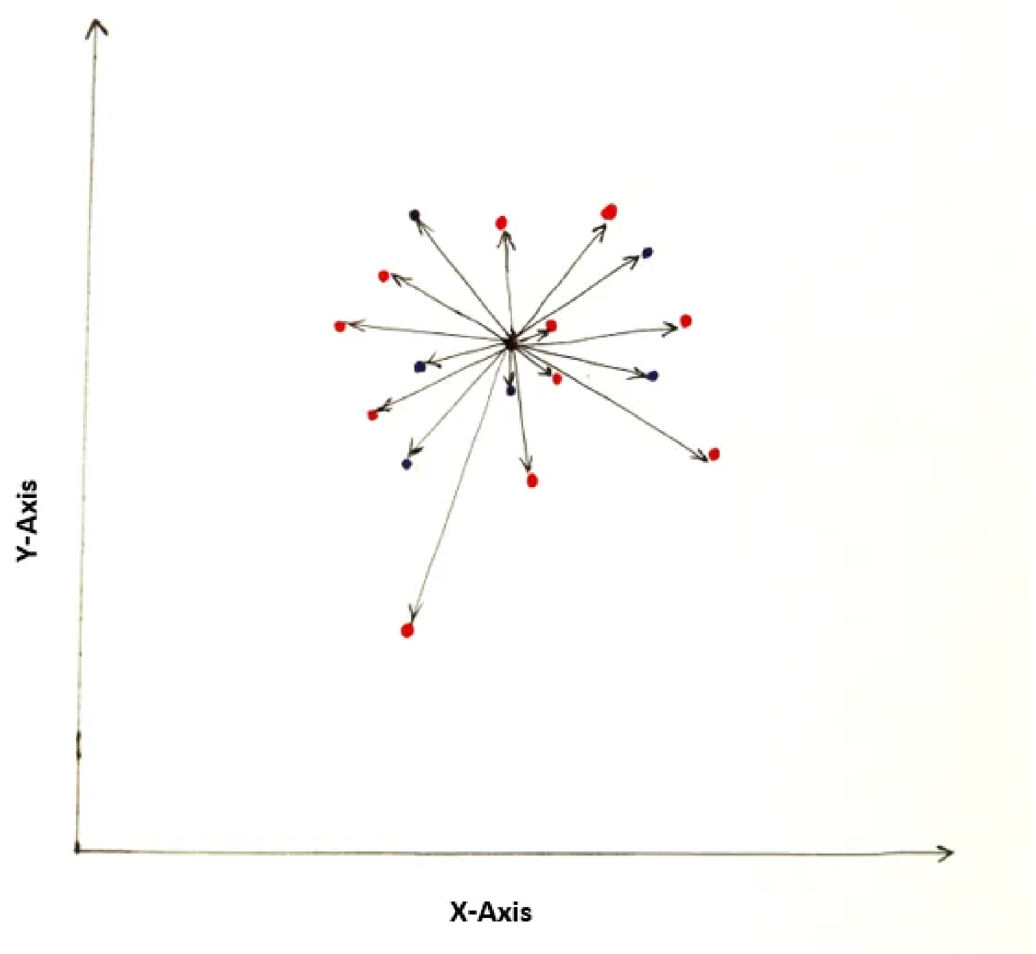
KNN Algorithm

KNN stands for K nearest neighbour. The name itself suggests that it considers the nearest neighbour. It is one of the supervised machine learning algorithms. Interestingly we can solve both classification and regression problems with the algorithm. It is one of the simplest Machine Learning models. Though it is a simple model, sometimes it plays a significant role, basically when our dataset is small, and the problem is simple.

Overview of the KNN Algorithm

Step 1: Calculating the Distance

First of all, we need to load the labelled dataset as the KNN algorithm is a supervised learning algorithm

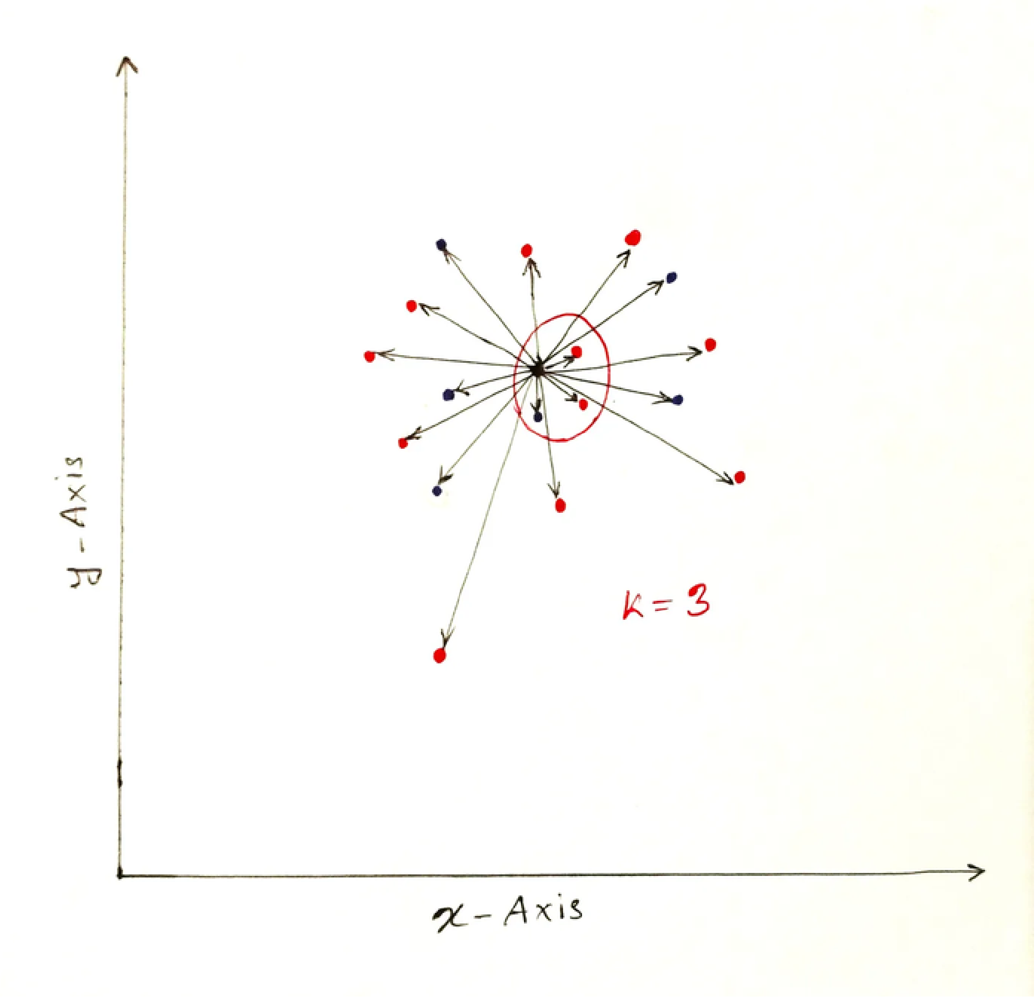


Suppose our dataset has only two features, and we plotted the data as shown in the image. Blue and Red points indicate two different categories. Let’s have new unlabelled data that requires classification based on the given dataset.

In the image, the central point needs to be classified. Now, we will calculate the distance of all the data from the unlabelled data. The arrow from the central point represents the distances.

Step 2: Selecting K-nearest neighbour

In the previous step, we calculated the distances of the new point from all other data. We will sort the data points in ascending order according to the distance. Finally, we will consider the K number of nearest points from the unlabelled data.



In the above image, I have considered the 3 nearest data points (K=3). Observe the image; among 3 nearest points, 2 data belong to the red category, and 1 to the blue category. So, red is the majority class. According to the KNN algorithm, new data points will be classified as red.

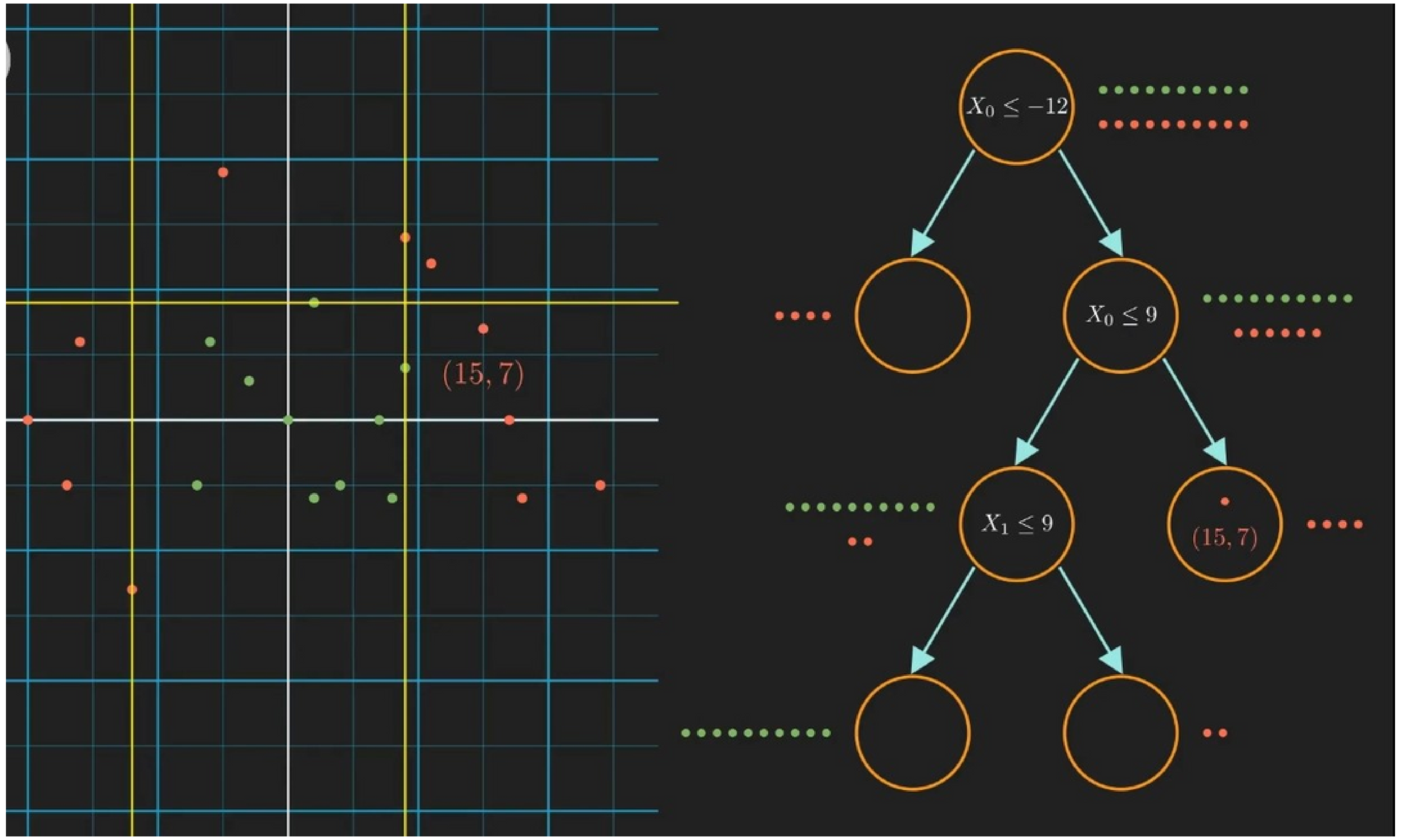
In case of a regression problem, we will consider the average value of K nearest data points.

Code:

Notebook Link

Decision Tree

The **decision tree algorithm** is a supervised machine learning algorithm for both regression and classification. The decision tree algorithm aims to create a model that can predict the class or value of a target variable by learning simple decision rules inferred from the input features. The decision rules are represented in a tree structure, where each internal node corresponds to a decision based on a feature, and each leaf node corresponds to a class or value prediction.

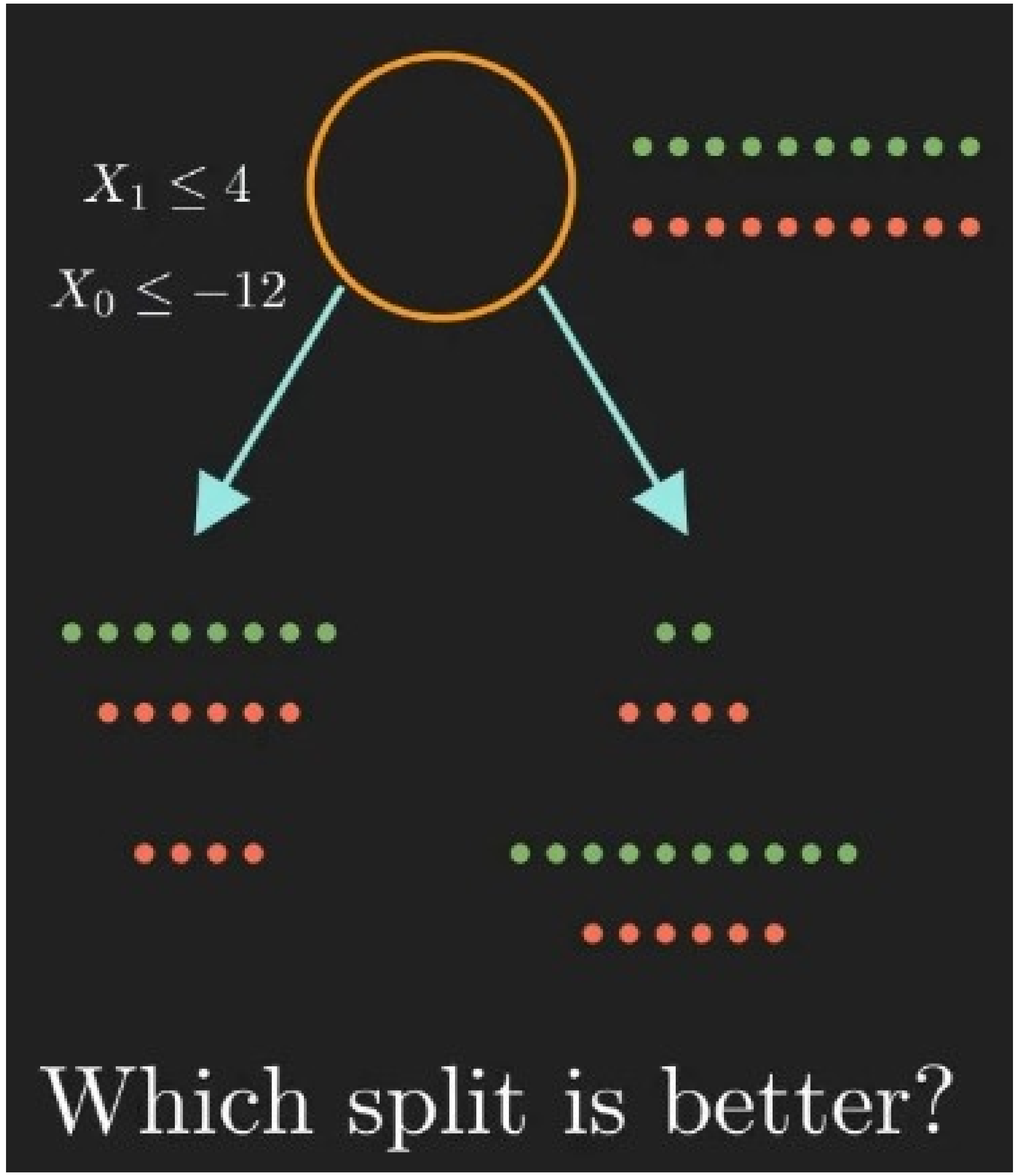


Algorithm 1 : Entropy and Information Gain

$$Entropy = \sum - p_i \ \log(p_i)$$

p_i = probability of class i

Information Gain =
Entropy(S) - (Weight *
Entropy(each feature)



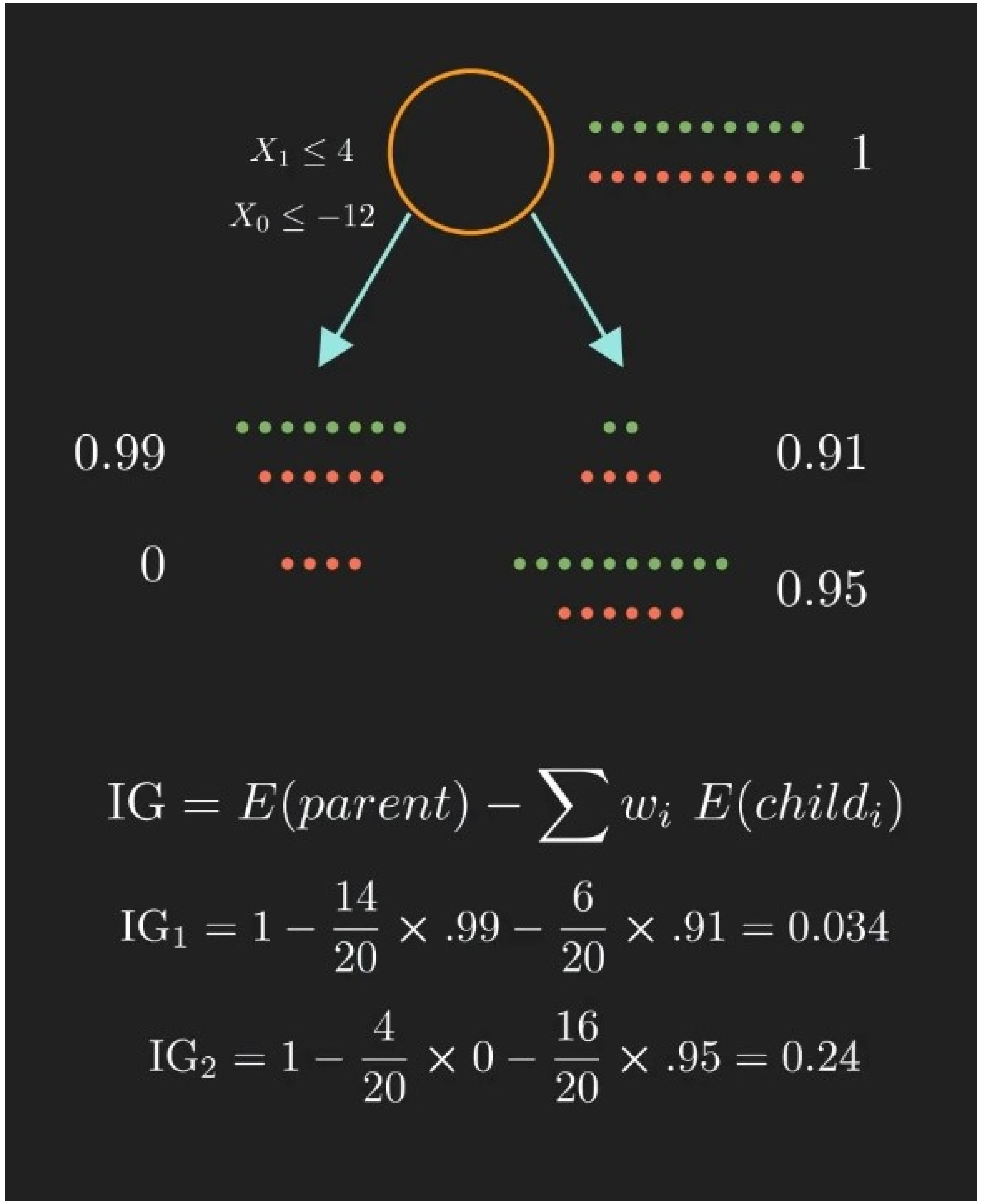
Which split is better?

Algorithm 2 : Gini Impurity Index and Gini Gain

Gini Impurity Index= 1- ∑P(i)^2

Gini Gain =
Gini impurity before split
- sum (Weight*Gini impurity of each subset)

Weight of a feature = Number of samples in the feature/ Total samples before split

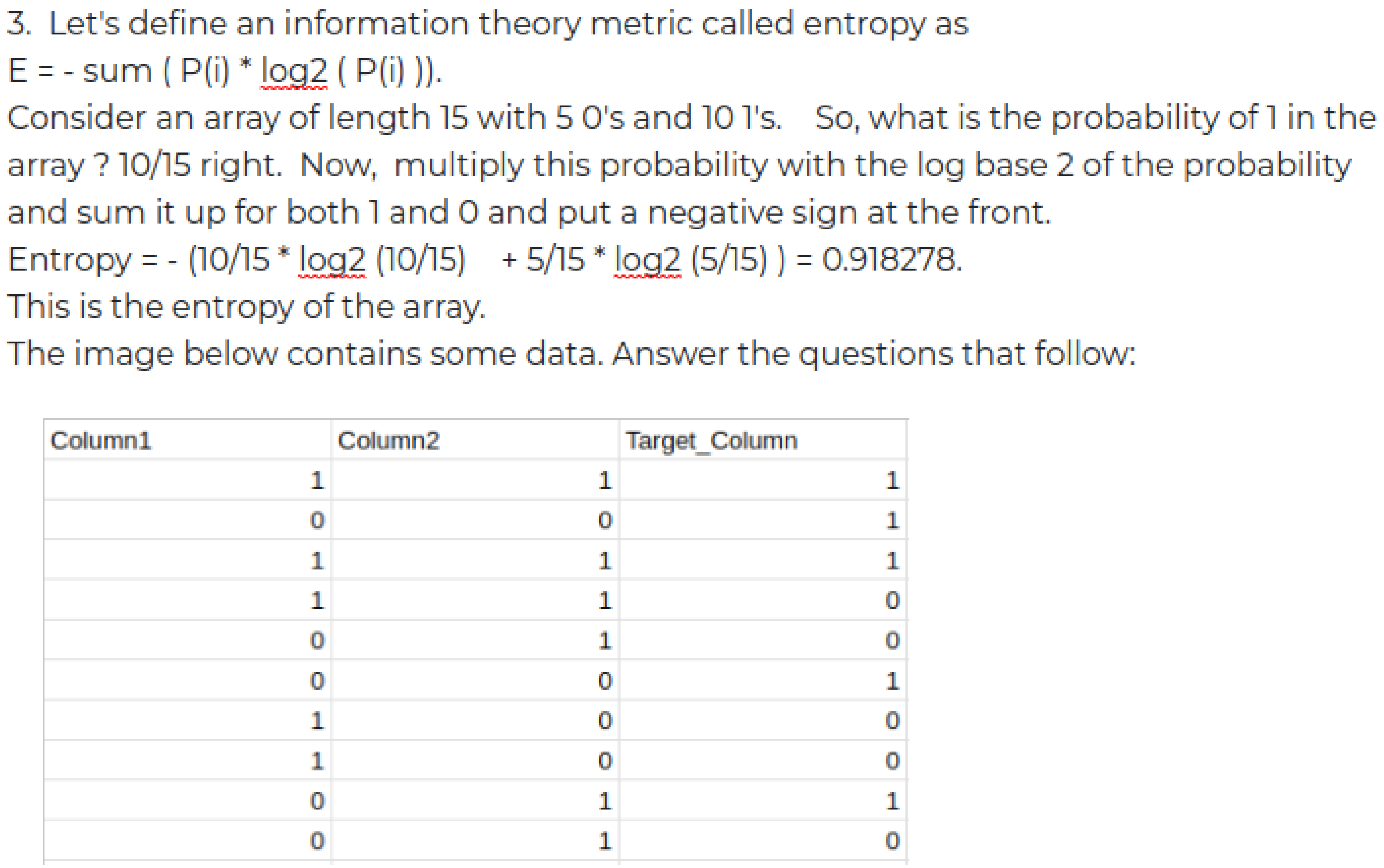


$$\text{IG} = E(parent) - \sum w_i \, E(child_i)$$

$$\text{IG}_1 = 1 - \frac{14}{20} \times .99 - \frac{6}{20} \times .91 = 0.034$$

$$\text{IG}_2 = 1 - \frac{4}{20} \times 0 - \frac{16}{20} \times .95 = 0.24$$

Remember this question from the selection test !!!



3. Let's define an information theory metric called entropy as
E = - sum (P(i) * log2 (P(i))).
Consider an array of length 15 with 5 0's and 10 1's. So, what is the probability of 1 in the array ? 10/15 right. Now, multiply this probability with the log base 2 of the probability and sum it up for both 1 and 0 and put a negative sign at the front.
Entropy = - (10/15 * log2 (10/15) + 5/15 * log2 (5/15)) = 0.918278.
This is the entropy of the array.
The image below contains some data. Answer the questions that follow:

Column1	Column2	Target_Column
1	1	1
0	0	1
1	1	1
1	1	0
0	1	0
0	0	1
1	0	0
1	0	0
0	1	1
0	1	0

Current entropy of "Target COlumn" is 1.00 . If we split it on the basis of Column1, the information gain is 0.94 , on the basis of Column2, IG is 1.00 . So, in the root node, we will split the split the dataset on the Column2.

The decision tree algorithm

- Select the best feature: The first step is to select the feature that best splits the data into classes or reduces the variance of the target variable for regression tasks. The most common measure for selecting the best feature is the information gain or the Gini impurity.
- Split the data: Once the best feature is selected, the data is split into subsets based on the values of the selected feature. Each subset corresponds to a child node of the current node in the tree.
- Recursively repeat steps 1 and 2: The above steps are repeated recursively for each child node until a stopping criterion is met. This stopping criterion can be a predefined maximum depth of the tree, a minimum number of samples required to split an internal node or a minimum number of samples required to be at a leaf node.
- Predict the target variable: Once the decision tree is built, the target variable can be predicted by traversing the tree from the root node to a leaf node based on the decision rules at each node.
- Prune the tree: To avoid overfitting, the tree can be pruned by removing unnecessary nodes or subtrees that do not improve the performance on a validation set.

Code:

Notebook Link