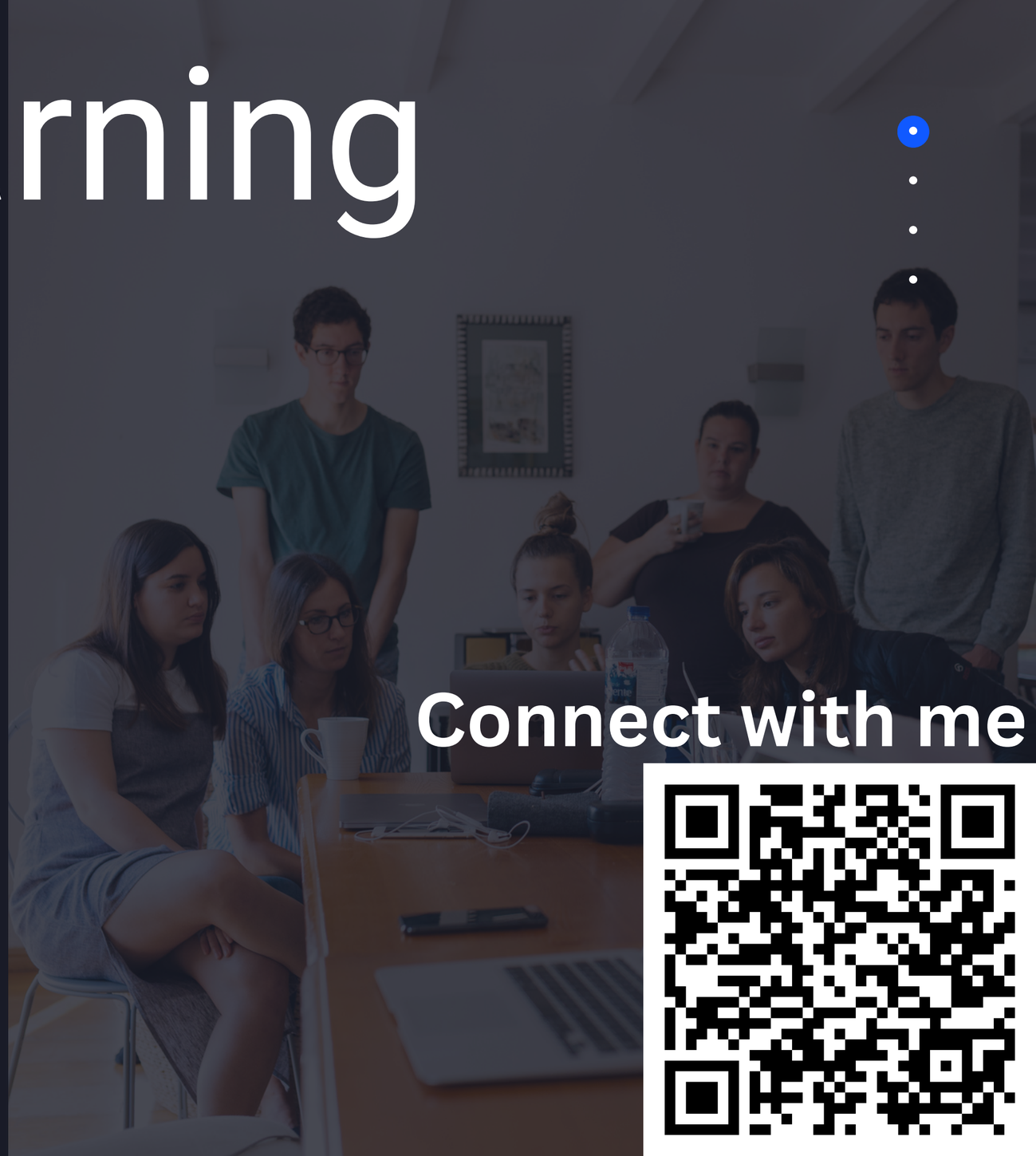
Machine learning Session

Ayush Jain



Connect with me

Table of Contents

DECISION TREE

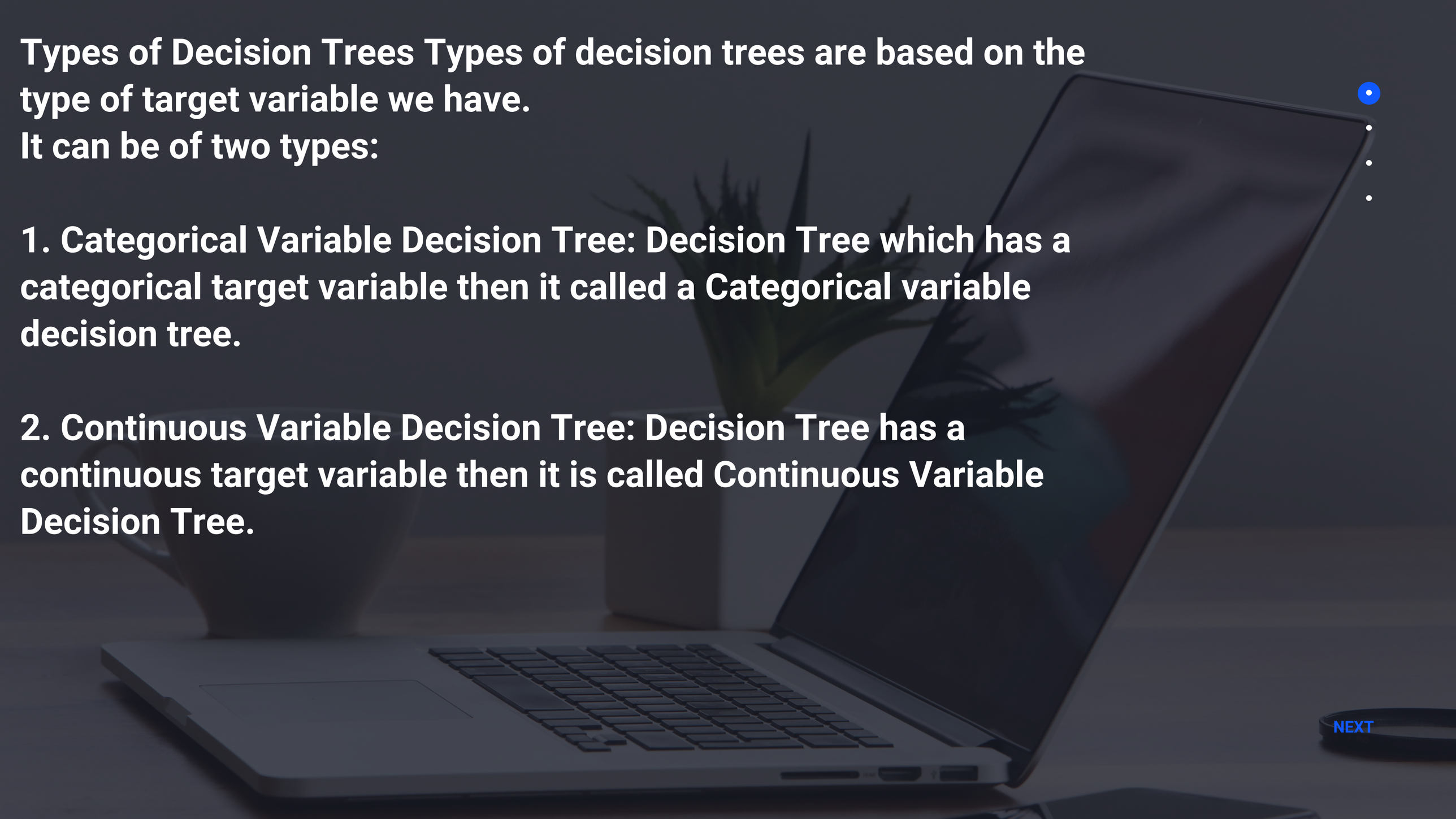
Decision Tree algorithm belongs to the family of supervised learning algorithms. • Unlike other supervised learning algorithms, the decision tree algorithm can be used for solving regression and classification problems too. • The goal of using a Decision Tree is to create a training model that can use to predict the class or value of the target variable by learning simple decision rules inferred from prior data(training data). • In Decision Trees, for predicting a class label for a record we start from the root of the tree. • We compare the values of the root attribute with the record’s attribute. • On the basis of comparison, we follow the branch corresponding to that value and jump to the next node.

NEXT

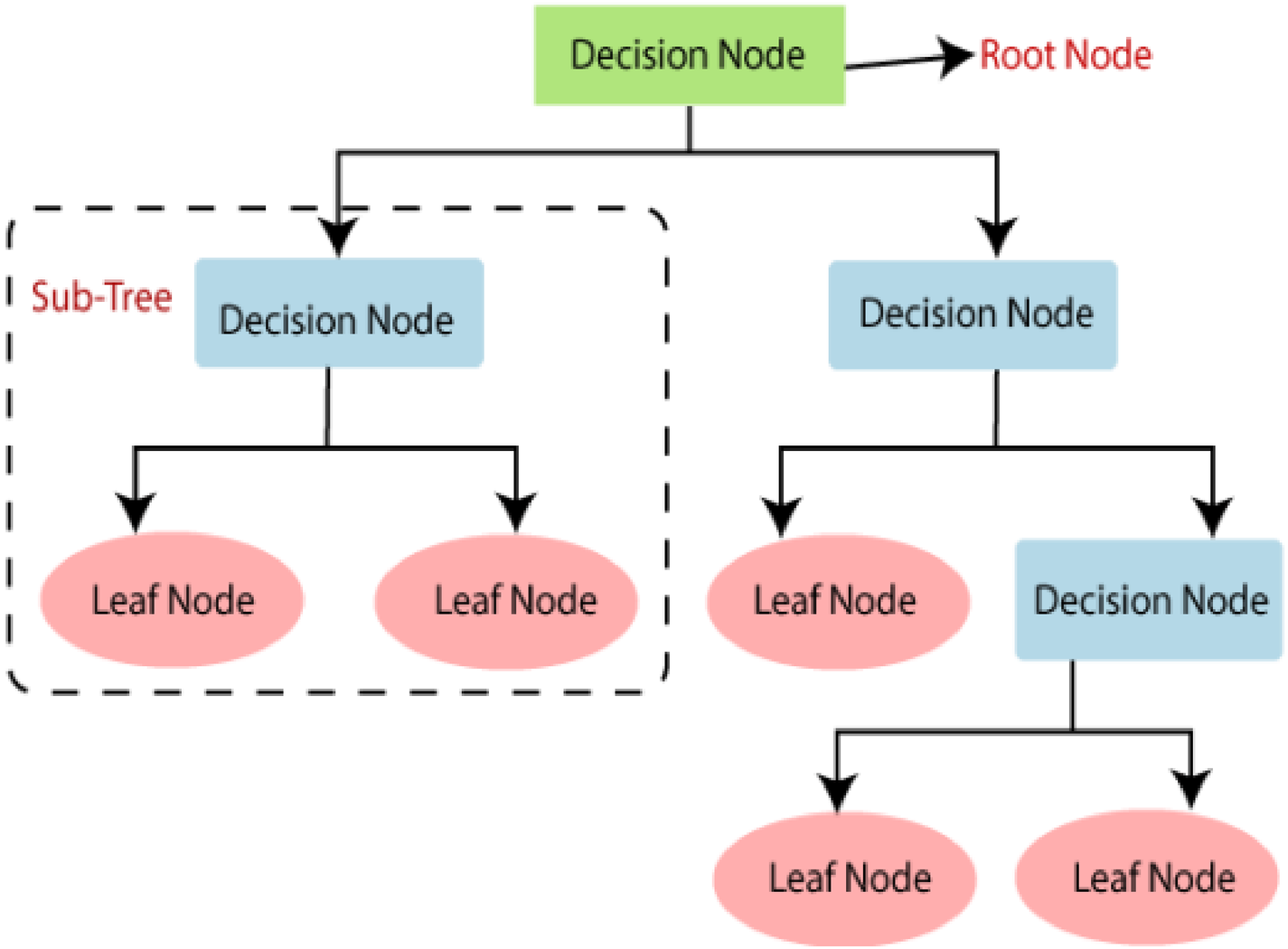
Types of Decision Trees Types of decision trees are based on the type of target variable we have.
It can be of two types:

1. Categorical Variable Decision Tree: Decision Tree which has a categorical target variable then it called a Categorical variable decision tree.

2. Continuous Variable Decision Tree: Decision Tree has a continuous target variable then it is called Continuous Variable Decision Tree.



NEXT


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

NEXT

- Why Decision Trees?
    - Types of Decision Trees
    - Key Terminology
- How To Create a Decision Tree
    - Gini Impurity
    - Chi-Square
    - Information Gain
- Applications of Decision Trees
- Decoding the Hyperparameters
- Coding the Algorithm
- Advantages and Disadvantages
- Summary and Conclusion

NEXT

Why Decision Trees?

Tree-based algorithms are a popular family of related non-parametric and supervised methods for both classification and regression. If you're wondering what supervised learning is, it's the type of machine learning algorithms which involve training models with data that has both input and output labels (in other words, we have data for which we know the true class or values, and can tell the algorithm what these are if it predicts incorrectly).

NEXT

The decision tree looks like a vague upside-down tree with a decision rule at the root, from which subsequent decision rules spread out below. For example, a decision rule can be whether a person exercises. There can also be nodes without any decision rules; these are called **leaf nodes**. Before we move on, let's quickly look into the different types of decision trees.

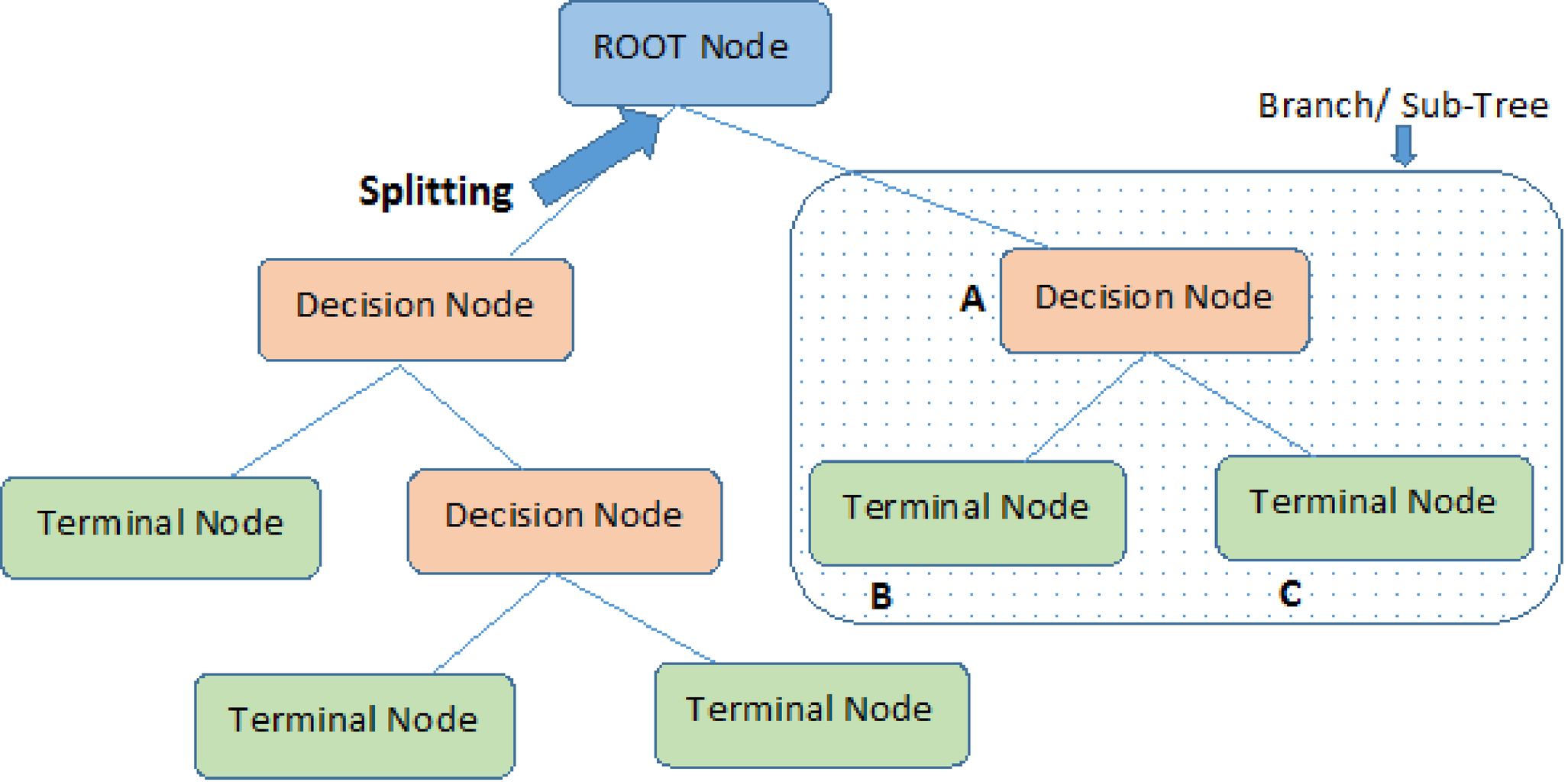
NEXT

Types of Decision Trees

Decision Trees are classified into two types, based on the target variables.

1. **Categorical Variable Decision Trees:** This is where the algorithm has a categorical target variable. For example, consider you are asked to predict the relative price of a computer as one of three categories: *low*, *medium*, or *high*. Features could include *monitor type*, *speaker quality*, *RAM*, and *SSD*. The decision tree will learn from these features and after passing each data point through each node, it will end up at a leaf node of one of the three categorical targets *low*, *medium*, or *high*.
2. **Continuous Variable Decision Trees:** In this case the features input to the decision tree (e.g. qualities of a house) will be used to predict a continuous output (e.g. the price of that house).

NEXT



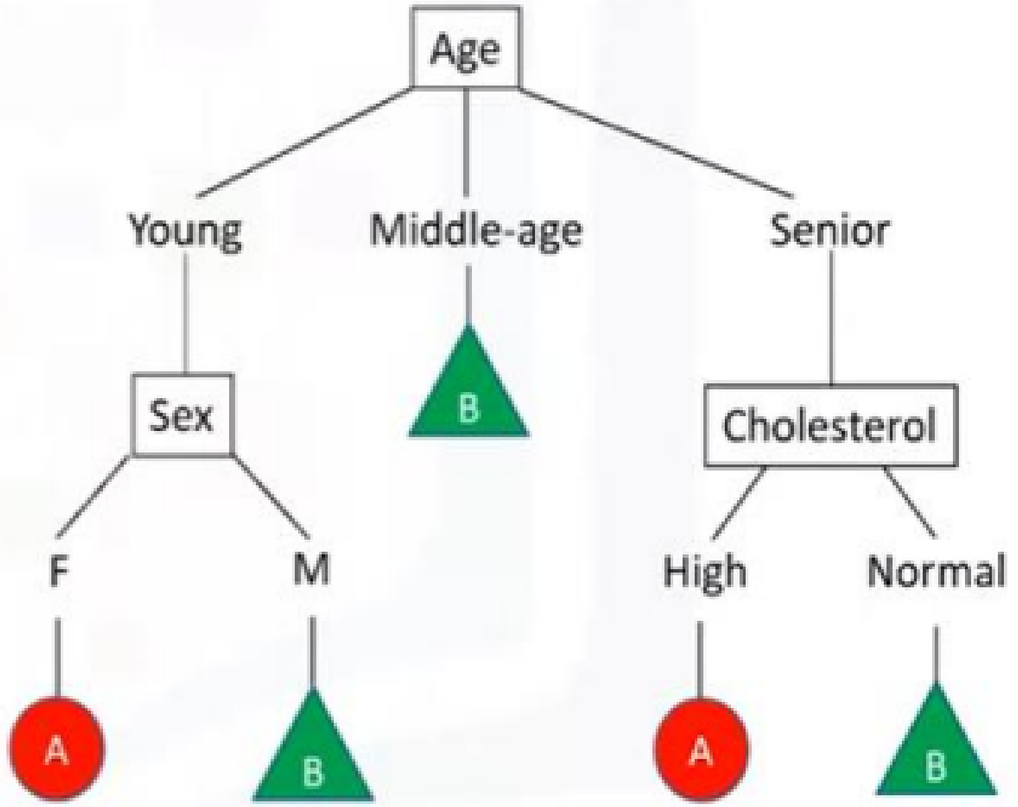
Note:- A is parent node of B and C.

Below is an image explaining the basic structure of the decision tree. Every tree has a root node, where the inputs are passed through. This root node is further divided into sets of decision nodes where results and observations are conditionally based. The process of dividing a single node into multiple nodes is called splitting. If a node doesn't split into further nodes, then it's called a leaf node, or terminal node. A subsection of a decision tree is called a branch or sub-tree (e.g. in the box in the image below).

NEXT

Decision tree learning algorithm

1. Choose an attribute from your dataset.
2. Calculate the significance of attribute in splitting of data.
3. Split data based on the value of the best attribute.
4. Go to step 1.



NEXT

How To Create a Decision Tree

In this section, we shall discuss the core algorithms describing how decision trees are created. These algorithms are completely dependent on the target variable, however, these vary from the algorithms used for classification and regression trees.

There are several techniques that are used to decide how to split the given data. The main goal of decision trees is to make the best splits between nodes which will optimally divide the data into the correct categories. To do this, we need to use he right decision rules.

NEXT

Gini Impurity

If all elements are correctly divided into different classes (an ideal scenario), the division is considered to be **pure**. The Gini impurity (pronounced like "genie") is used to gauge the likelihood that a randomly chosen example would be wrongly classified by a certain node. It is known as an "impurity" measure since it gives us an idea of how the model differs from a pure division.

The degree of the Gini impurity score is always between 0 and 1, where 0 denotes that all elements belong to a certain class (or the division is pure), and 1 denotes that the elements are randomly distributed across various classes. A Gini impurity of 0.5 denotes that the elements are distributed equally into some classes. The mathematical notation of the Gini impurity measure is given by the following formula:

NEXT

$$Gini = 1 - \sum_{i=1}^{n} (p_i)^2$$

Where pi is the probability of a particular element belonging to a specific class.

NEXT

Information gain

Information Gain depicts the amount of information that is gained by an attribute. It tells us how important the attribute is. Since Decision Tree construction is all about finding the right split node that assures high accuracy, Information Gain is all about finding the best nodes that return the highest information gain.

NEXT

c. Chi-Square

The chi-square method works well if the target variables are categorical like success-failure/high-low. The core idea of the algorithm is to find the statistical significance of the variations that exist between the sub-nodes and the parent node. The mathematical equation that is used to calculate the chi-square is,

$$chi - square = \sqrt{\frac{(Actual - Expected)^2}{Expected}}$$

NEXT

Applications of Decision Trees

Decision Tree is one of the basic and widely-used algorithms in the fields of Machine Learning. It’s put into use across different areas in classification and regression modeling. Due to its ability to depict visualized output, one can easily draw insights from the modeling process flow. Here are a few examples wherein Decision Tree could be used,

- Business Management
- Customer Relationship Management
- Fraudulent Statement Detection
- Energy Consumption
- Healthcare Management
- Fault Diagnosis

NEXT

Decoding the Hyperparameters

Scikit-learn provides some functionalities or parameters that are to be used with a Decision Tree classifier to enhance the model's accuracy in accordance with the given data.

- **criterion:** This parameter is used to measure the quality of the split. The default value for this parameter is set to "Gini". If you want the measure to be calculated by entropy gain, you can change this parameter to "entropy".
- **splitter**: This parameter is used to choose the split at each node. If you want the sub-trees to have the best split, you can set this parameter to "best". We can also have a random split for which the value "random" is set.
- **max-depth:** This is an integer parameter through which we can limit the depth of the tree. The default value for this parameter is set to None.
- **min_samples_split:** This parameter is used to define the minimum number of samples required to split an internal node.
- **max_leaf_nodes:** The default value of max_leaf_nodes is set to None. This parameter is used to grow a tree with max_leaf_nodes in best-first fashion.

NEXT

Coding the Algorithm

Step 1: Importing the Modules

Step 2: Exploring the data

Step 3: Create a decision tree classifier object

Step 4: Fitting the Model

Step 5: Making the Predictions

Step 6: Dot Data for the predictions

Step 7: Drawing the Graph

Introduction to Support Vector Machine

Support Vector Machine (SVM) is a supervised machine learning algorithm that can be used for both classification and regression problems. SVM performs very well with even a limited amount of data.

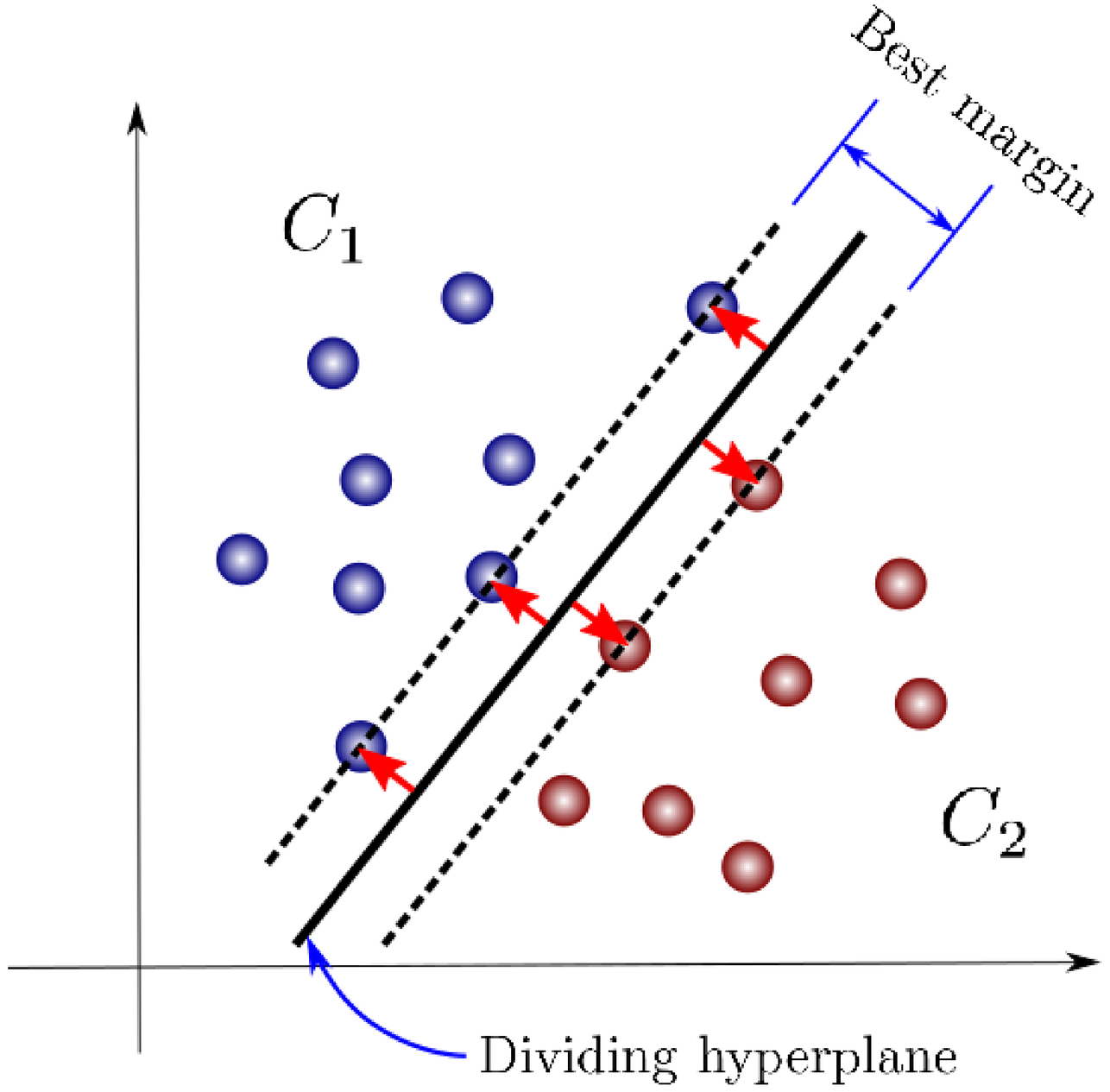
In this we'll learn about support vector machine for classification specifically. Let's first take a look at some of the general use cases of the support vector machine algorithm.

Use Cases

Classification of Disease: For example, if we have data about patients with a disease like diabetes, we can predict whether a new patient is likely to have diabetes or not.

Classification of Text: First we need to transform the text into a vector representation. Then we can apply the support vector machine algorithm to the encoded text in order to assign it to a specific class.

Classification of Images: Images are converted to a vector containing the pixel values, then SVM assigns a class label.


Best margin
C_1
C_2
Dividing hyperplane

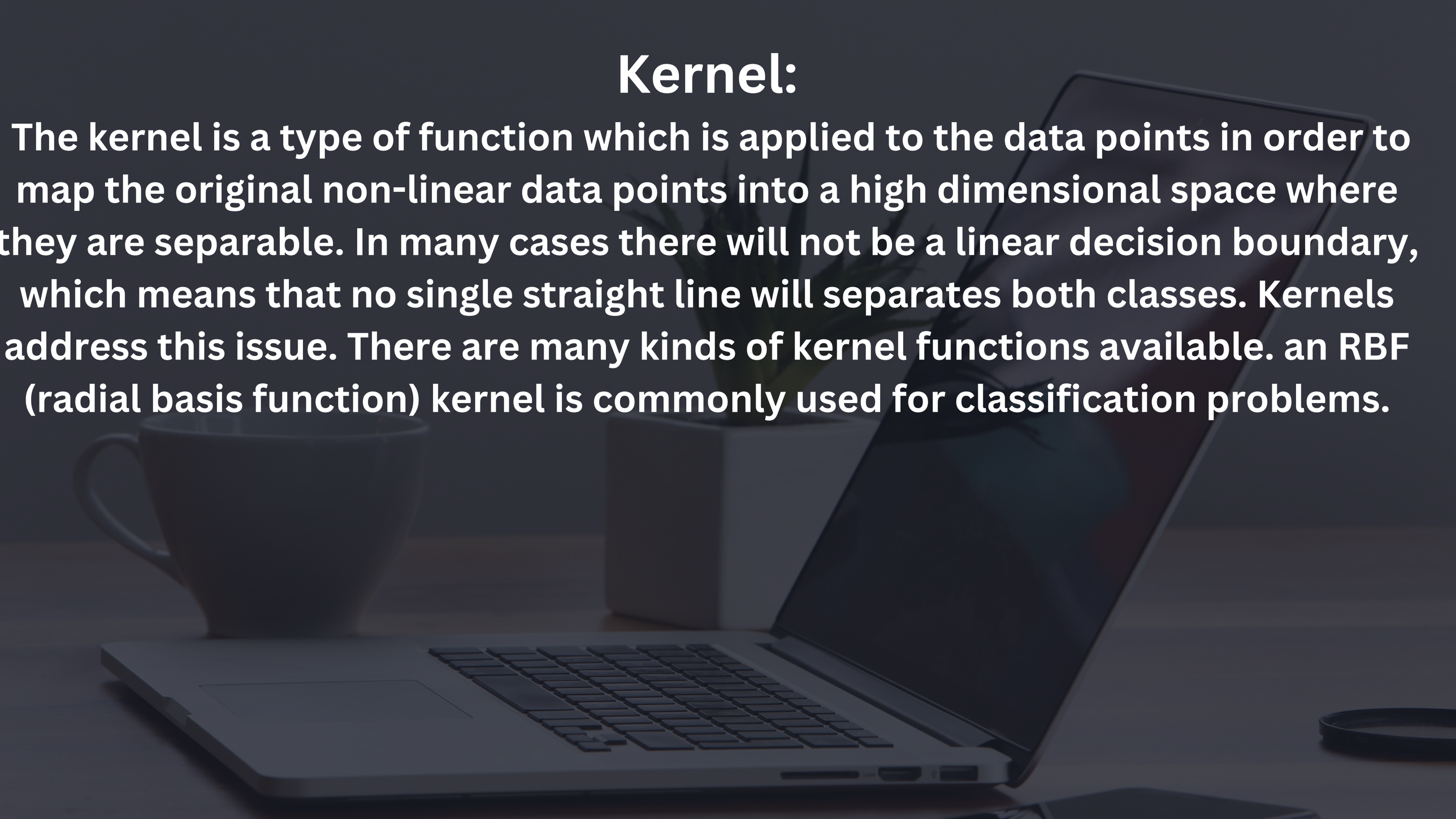
Support Vector: Support vectors are the data points that are closer to a hyperplane, as shown in the image above. Using the support vectors, the algorithm maximizes the margin, or separation, between classes. If the support vectors are changed, the position of the hyperplane will also change.

Hyperplane: A hyperplane in two dimensions is simply a line that best separates the data. This line is the decision boundary between the classes, as shown in the image above. For a three-dimensional data distribution the hyperplane would be a two dimensional surface, not a line.

Margin: This is the distance between each of the support vectors, as shown above. The algorithm aims to maximize the margin. The problem of finding the maximum margin (and hence, the best hyperplane) is an optimization problem, and can be solved by optimization techniques.

Kernel:

The kernel is a type of function which is applied to the data points in order to map the original non-linear data points into a high dimensional space where they are separable. In many cases there will not be a linear decision boundary, which means that no single straight line will separates both classes. Kernels address this issue. There are many kinds of kernel functions available. an RBF (radial basis function) kernel is commonly used for classification problems.



Introduction to KNN

K- Nearest Neighbor: • K-Nearest Neighbor is one of the simplest Machine Learning algorithms based on Supervised Learning technique.

• K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.

• K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.

• K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.

K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.

- It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.

- KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data. • Example: Suppose, we have an image of a creature that looks similar to cat and dog, but we want to know either it is a cat or dog.
- So for this identification, we can use the KNN algorithm, as it works on a similarity measure.

Our KNN model will find the similar features of the new data set to the cats and dogs images and based on the most similar features it will put it in either cat or dog category.

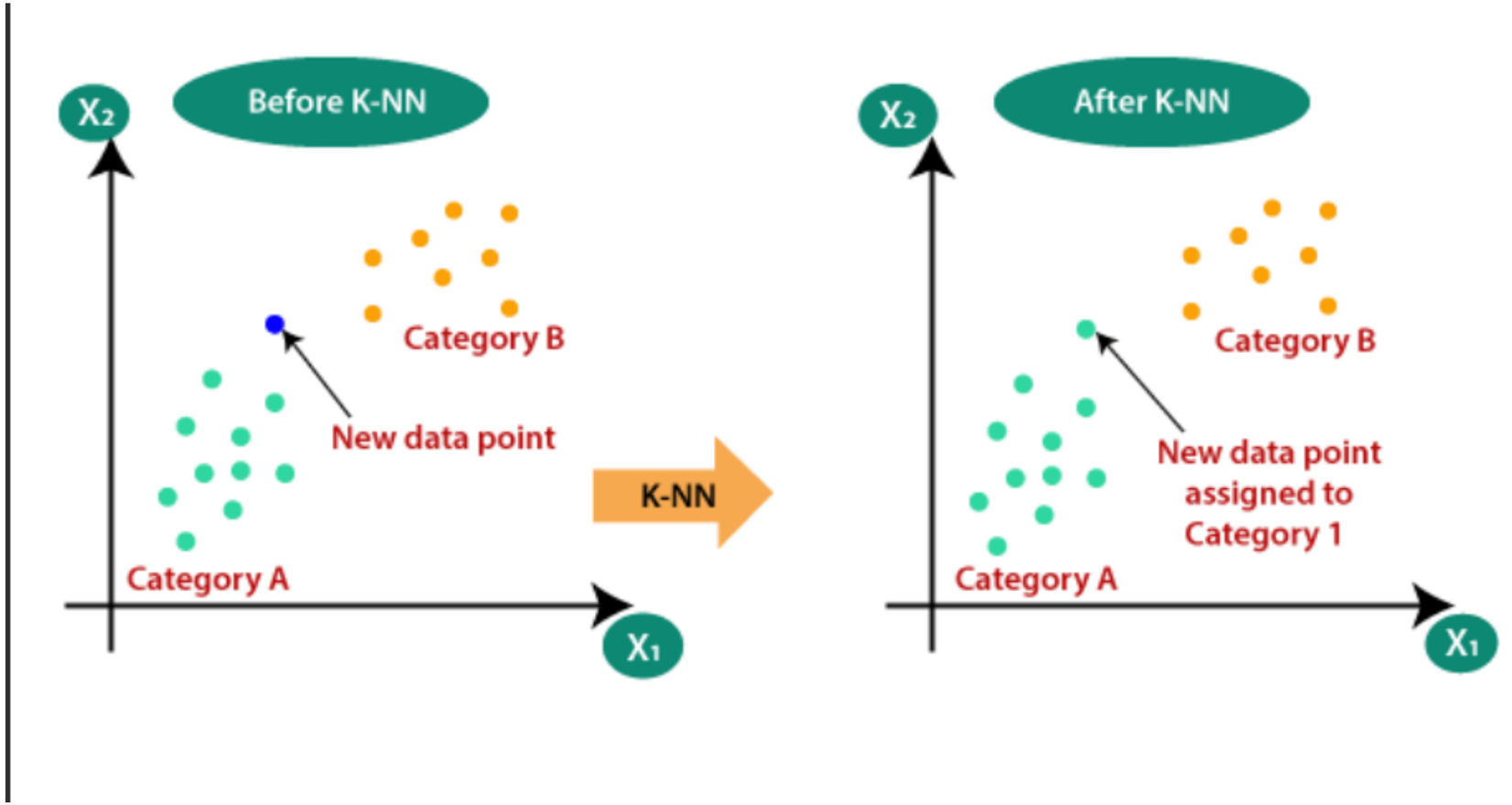

KNN Classifier
Input value
Predicted Output

Why do we need a K-NN Algorithm?

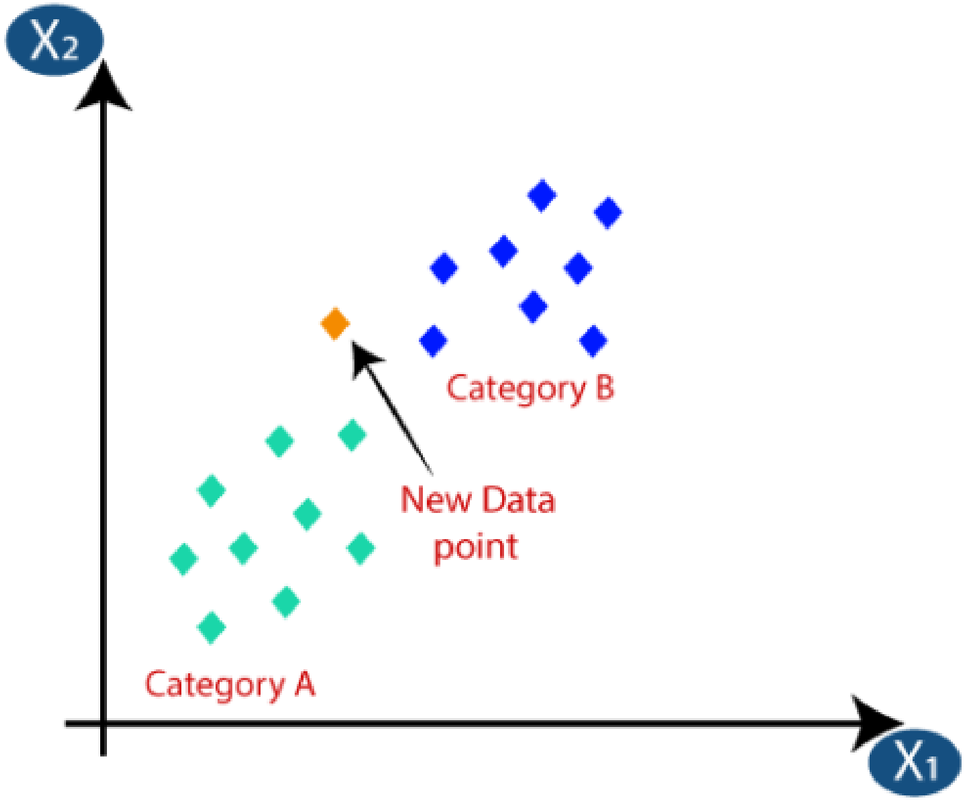
• Suppose there are two categories, i.e., Category A and Category B, and we have a new data point x1, so this data point will lie in which of these categories.

• To solve this type of problem, we need a K-NN algorithm.

• With the help of K-NN, we can easily identify the category or class of a particular dataset. • Consider the below diagram:

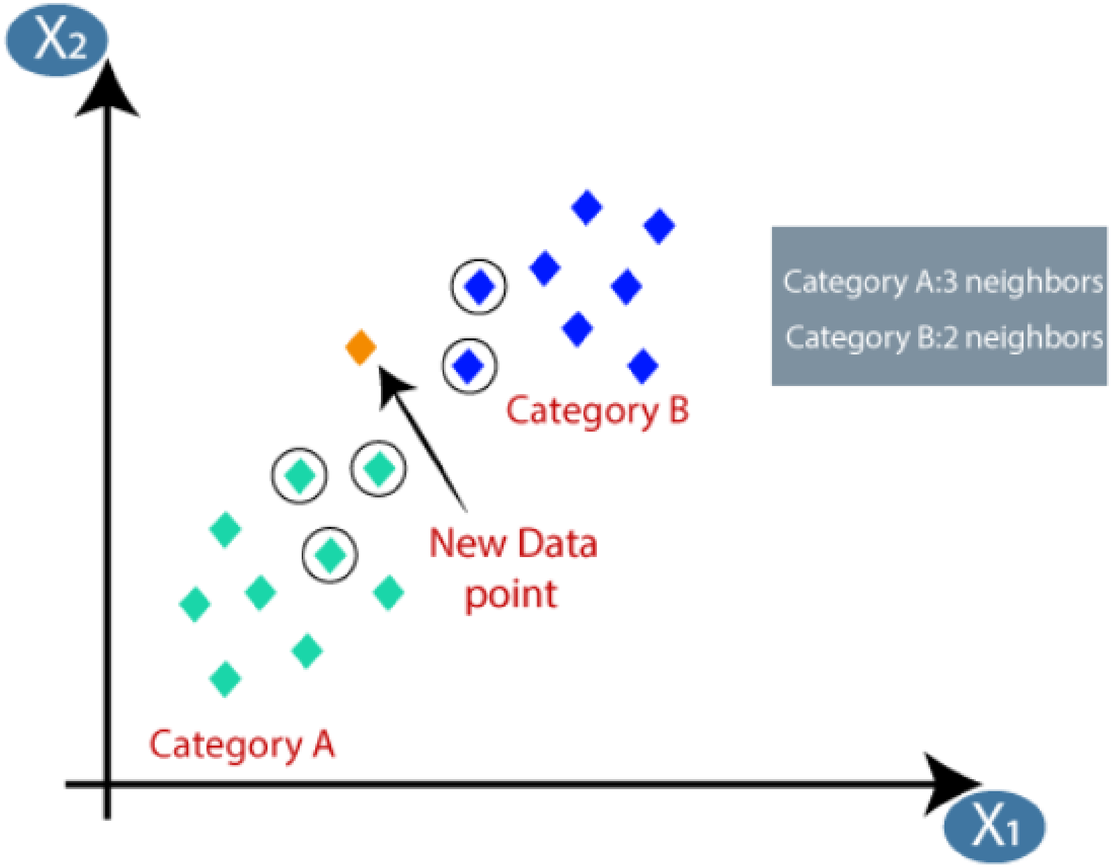

Before K-NN
X2
Category B
New data point
Category A
X1
K-NN
After K-NN
X2
Category B
New data point
assigned to
Category 1
Category A
X1

- The K-NN working can be explained on the basis of the below algorithm:
- **Step-1:** Select the number K of the neighbors
- **Step-2:** Calculate the Euclidean distance of **K number of neighbors**
- **Step-3:** Take the K nearest neighbors as per the calculated Euclidean distance.
- **Step-4:** Among these k neighbors, count the number of the data points in each category.
- **Step-5:** Assign the new data points to that category for which the number of the neighbor is maximum.
- **Step-6:** Our model is ready.



•Firstly, we will choose the number of neighbors, so we will choose the k=5.
•Next, we will calculate the **Euclidean distance** between the data points.
•The Euclidean distance is the distance between two points, which we have already studied in geometry.
• It can be calculated as:

By calculating the Euclidean distance we got the nearest neighbors, as three nearest neighbors in category A and two nearest neighbors in category B.
Consider the below image:



Category A:3 neighbors
Category B:2 neighbors

As we can see the 3 nearest neighbors are from category A, hence this new data point must belong to category A.

Example: The properties of special tissue paper are given below:

Now factory produces new tissue paper with X(acid durability)=3 and X2(Strength) =7, what will be the classification as per KNN algorithm.

Name	Acid Durability	Strength	Class
Type-1	7	7	Bad
Type-2	7	4	Bad
Type-3	3	4	Good
Type-4	1	4	Good

Test-Data ➔ acid durability=3, and strength=7, class=?

Similarity

- Calculated using distance measure like Euclidean

$$\mathrm{d}(\mathbf{p},\mathbf{q}) = \mathrm{d}(\mathbf{q},\mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$$

$$= \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}.$$

Name	Acid Durability	Strength	Class	Distance
Type-1	7	7	Bad	Sqrt((7-3)²+(7-7)²)=4
Type-2	7	4	Bad	$\sqrt{(7-3)^2+(4-7)^2} = 5$
Type-3	3	4	Good	$\sqrt{(3-3)^2+(4-7)^2} = 3$
Type-4	1	4	Good	$\sqrt{(1-3)^2+(4-7)^2} = 3.6$

Rank these attributes

Rank decided based on minimum distance

Name	Acid Durability	Strength	Class	Distance	Rank
Type-1	7	7	Bad	4	3
Type-2	7	4	Bad	5	4
Type-3	3	4	Good	3	1
Type-4	1	4	Good	3.6	2

k=1 Based on immediate Neighbour, Good

Name	Acid Durability	Strength	Class	Distance	Rank
Type-1	7	7	Bad	4	3
Type-2	7	4	Bad	5	4
Type-3	3	4	Good	3	1
Type-4	1	4	Good	3.6	2

Based on immediate neighbour, Good

k=2

Name	Acid Durability	Strength	Class	Distance	Rank
Type-1	7	7	Bad	4	3
Type-2	7	4	Bad	5	4
Type-3	3	4	Good	3	1
Type-4	1	4	Good	3.6	2

Based on two neighbours, Good

k=3

Name	Acid Durability	Strength	Class	Distance	Rank
Type-1	7	7	Bad	4	3
Type-2	7	4	Bad	5	4
Type-3	3	4	Good	3	1
Type-4	1	4	Good	3.6	2

Based on three neighbours, 2 Goods and 1 bad, majority → Good


Thank you!