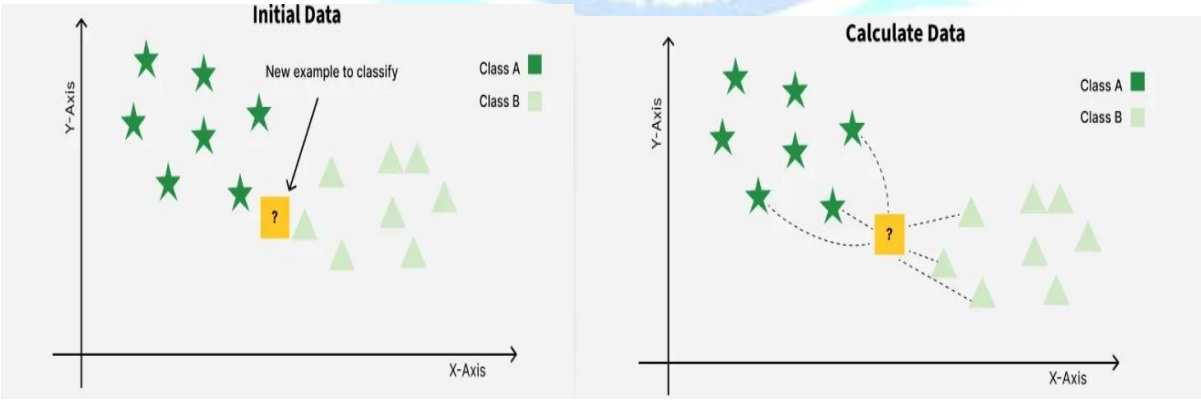
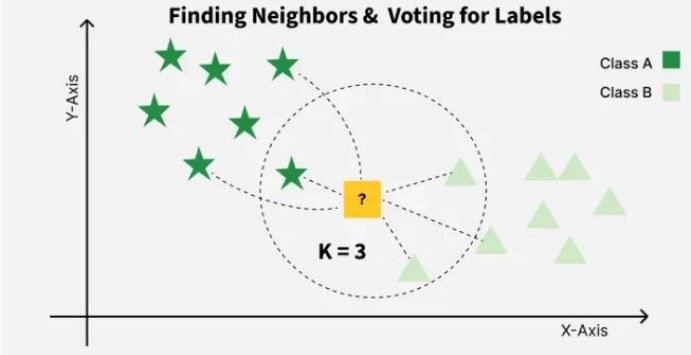
K-Nearest Neighbor (KNN) Algorithm

K-Nearest Neighbor (KNN) is a simple and widely used machine learning technique for classification and regression tasks. It works by identifying the *K* closest data points to a given input and making predictions based on the majority class or average value of those neighbors.

- Classifies data based on similarity with nearby data points
- Uses distance metrics like Euclidean distance to find nearest neighbors
- Since KNN makes no assumptions about the underlying data distribution, it makes it a non-parametric and instance-based learning method.





KNN is also called as a lazy learner algorithm because it does not learn from the training set immediately instead it stores the entire dataset and performs computations only at the time of classification.

For example, consider two features i.e Category 1 and Category 2:

- KNN assigns the category based on the majority of nearby points. The image shows how KNN predicts the category of a new data point based on its closest neighbours.
- The green points represent Category 1 and the red points represent Category 2.
- The new data point checks its closest neighbors (circled points).
- Since the majority of its closest neighbors are red points (Category 2) KNN predicts the new data point belongs to Category 2.

KNN works by using proximity and majority voting to make predictions.

'K' in K Nearest Neighbour

In the KNN algorithm k is just a number that tells the algorithm how many nearby points or neighbors to look at when it makes a decision.

Example: Imagine you're deciding which fruit it is based on its shape and size. You compare it to fruits you already know.

- If k = 3, the algorithm looks at the 3 closest fruits to the new one.

- If 2 of those 3 fruits are apples and 1 is a banana, the algorithm says the new fruit is an apple because most of its neighbors are apples.

How to choose the value of k for KNN Algorithm

- The value of k in KNN decides how many neighbors the algorithm looks at when making a prediction.
- Choosing the right k is important for good results.
- If the data has lots of noise or outliers, using a larger k can make the predictions more stable.
- But if k is too large the model may become too simple and miss important patterns and this is called underfitting.
- So k should be picked carefully based on the data.

Statistical Methods for Selecting k

- **Cross-Validation**: Cross-Validation is a good way to find the best value of k is by using k-fold cross-validation. This means dividing the dataset into k parts. The model is trained on some of these parts and tested on the remaining ones. This process is repeated for each part.
- **Elbow Method**: In Elbow Method we draw a graph showing the error rate or accuracy for different k values. As k increases the error usually drops at first. But after a certain point error stops decreasing quickly. The point where the curve changes direction and looks like an "elbow" is usually the best choice for k.
- **Odd Values for k**: It's a good idea to use an odd number for k especially in classification problems. This helps avoid ties when deciding which class is the most common among the neighbors.

Distance Metrics Used in KNN Algorithm

KNN uses distance metrics to identify nearest neighbor, these neighbors are used for classification and regression task. To identify nearest neighbor we use below distance metrics:

1. Euclidean Distance

Euclidean distance is defined as the straight-line distance between two points in a plane or space. You can think of it like the shortest path you would walk if you were to go directly from one point to another.

$$d(x, X_i) = \sqrt{\sum_{j=1}^{n} (x_j - X_{ij})^2}$$

2. Manhattan Distance

This is the total distance you would travel if you could only move along horizontal and vertical lines like a grid or city streets. It's also called "taxicab distance" because a taxi can only drive along the grid-like streets of a city.

$$d(x, y) = \sum_{i=1}^{n} |x_i - y_i|$$

3. Minkowski Distance

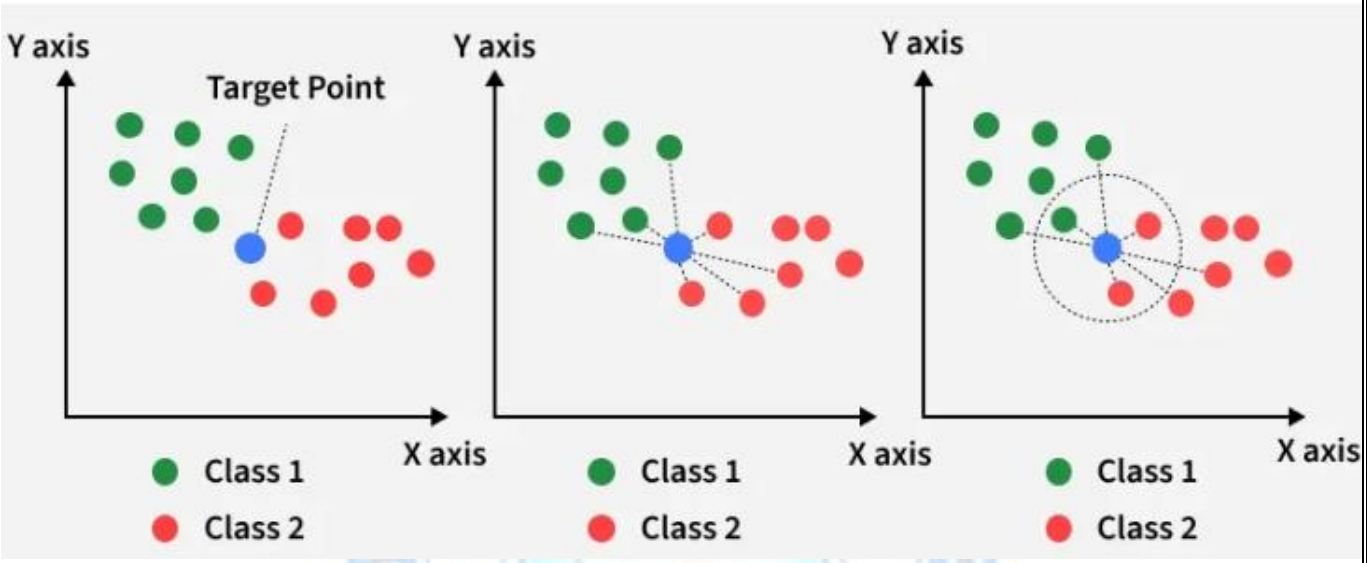
Minkowski distance is like a family of distances, which includes both Euclidean and Manhattan distances as special cases.

$$d(x, y) = \left(\sum_{i=1}^{n} (x_i - y_i)^p\right)^{\frac{1}{p}}$$

From the formula above, when p=2, it becomes the same as the Euclidean distance formula and when p=1, it turns into the Manhattan distance formula. Minkowski distance is essentially a flexible formula that can represent either Euclidean or Manhattan distance depending on the value of p.

Working of KNN algorithm

The KNN algorithm operates on the principle of similarity where it predicts the label or value of a new data point by considering the labels or values of its K nearest neighbors in the training dataset.



Step 1: Selecting the optimal value of K

K is the number of nearest neighbors considered for prediction.

Step 2: Calculating distance

To measure the similarity between target and training data points Euclidean distance is widely used. Distance is calculated between data points in the dataset and target point.

Step 3: Finding Nearest Neighbors

The k data points with the smallest distances to the target point are nearest neighbors.

Step 4: Voting for Classification or Taking Average for Regression

When you want to classify a data point into a category like spam or not spam, the KNN algorithm looks at the K closest points in the dataset. These closest points are called

neighbors. The algorithm then looks at which category the neighbors belong to and picks the one that appears the most. This is called majority voting.

In regression, the algorithm still looks for the K closest points. But instead of voting for a class in classification, it takes the average of the values of those K neighbors. This average is the predicted value for the new point for the algorithm.

It shows how a test point is classified based on its nearest neighbors. As the test point moves the algorithm identifies the closest 'k' data points i.e. 5 in this case and assigns test point the majority class label that is grey label class here.