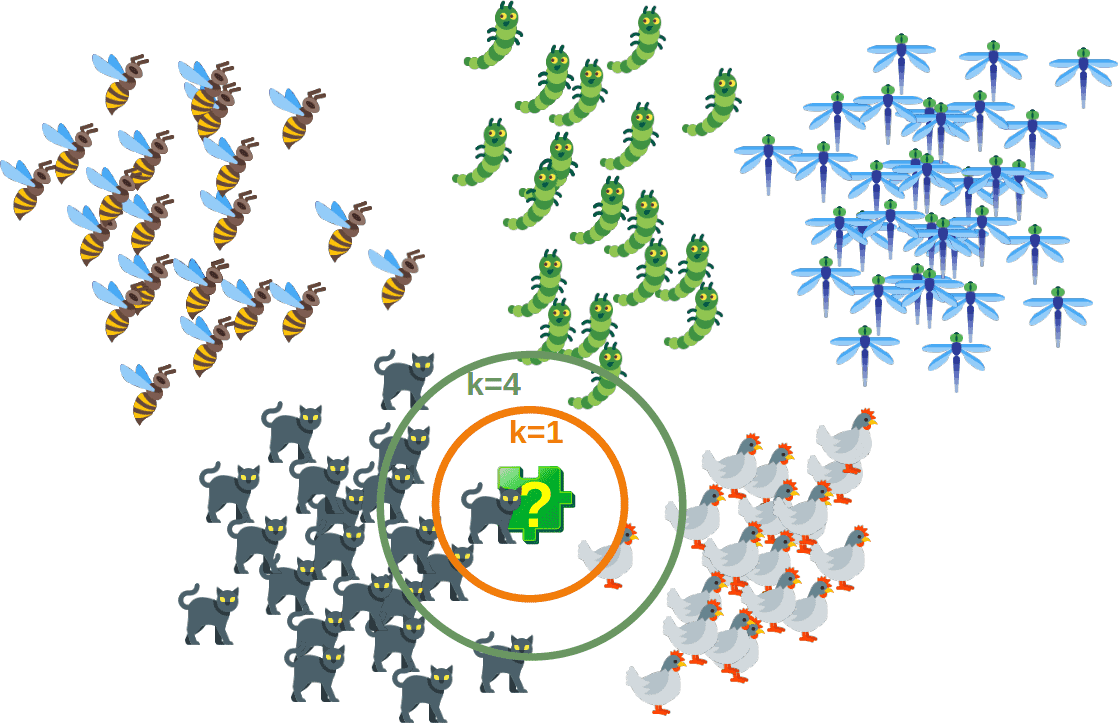
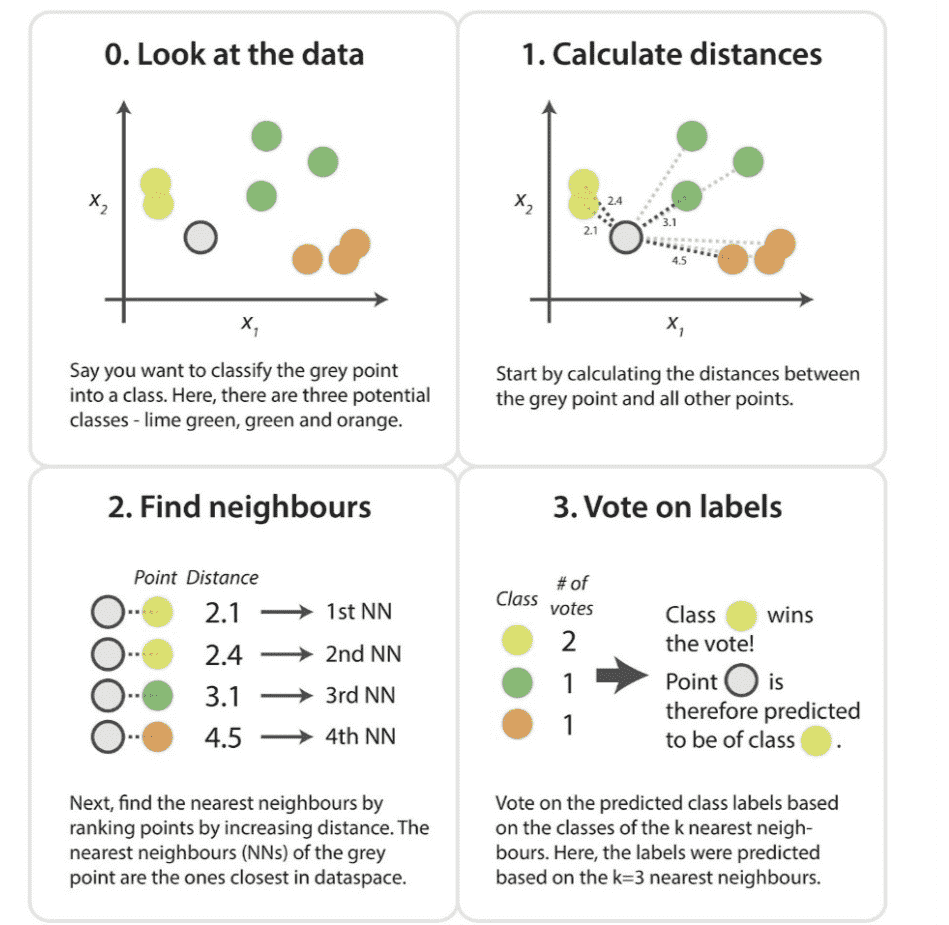
**K-Nearest Neighbor**



**Introduction**

K-nearest neighbors (KNN) is a type of supervised learning algorithm used for both regression and classification. KNN tries to predict the correct class for the test data by **calculating the distance between the test data and all the training points**. Then select the K number of points which is closest to the test data. The KNN algorithm calculates the probability of the test data belonging to the classes of ‘K’ training data and class holds the highest probability will be selected. In the case of regression, the value is the mean of the ‘K’ selected training points.

Let see the below example to make it a better understanding

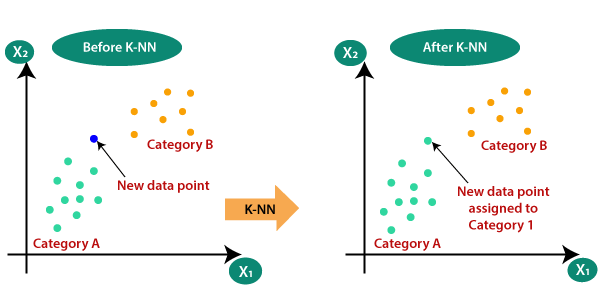


Suppose, we have an image of a creature that looks similar to cat and dog, but we want to know either it is a cat or dog. So for this identification, we can use the KNN algorithm, as it works on a similarity measure. Our KNN model will find the similar features of the new data set to the cats and dogs images and based on the most similar features it will put it in either cat or dog category.



**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:

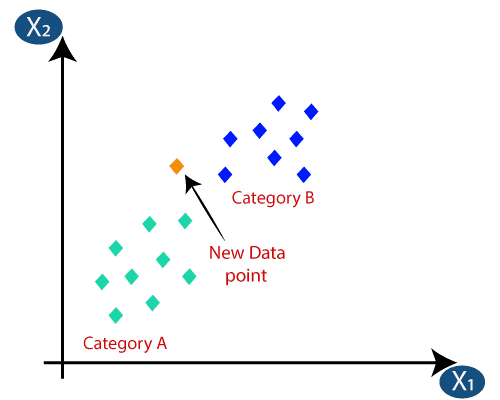


**How does K-NN work?**

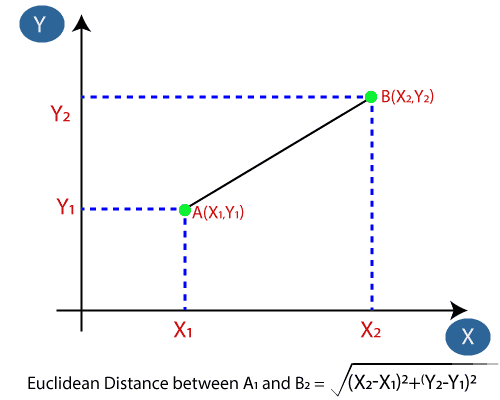
The K-NN working can be explained on the basis of the below algorithm:

* 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.(Majority of voting)
* 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.

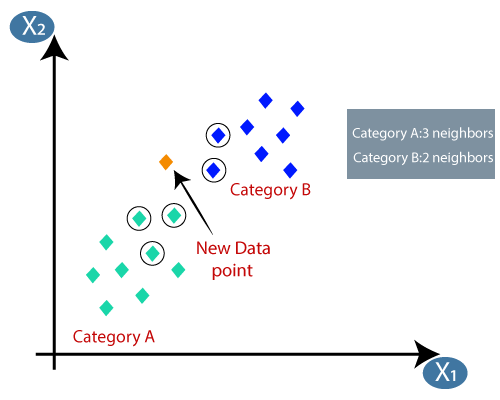
Suppose we have a new data point and we need to put it in the required category. Consider the below image:



* 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:

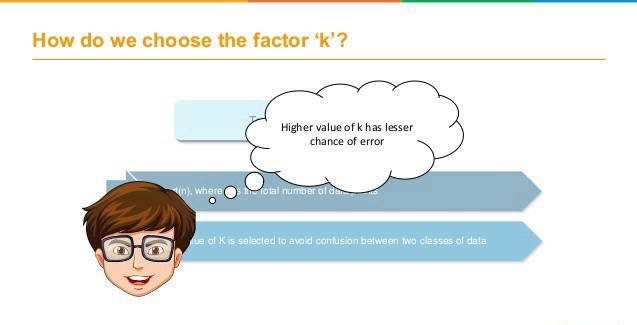


* By calculating the Euclidean distance we got the nearest neighbors, as three nearest neighbors in category A and two nearest neighbors in category B. Consider the below image:

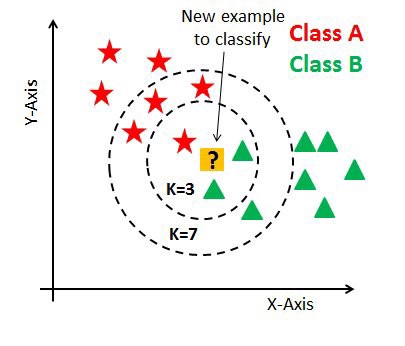


* As we can see the 3 nearest neighbors are from category A, hence this new data point must belong to category A.

**How to choose a K value?**



K value indicates the count of the nearest neighbors. We have to compute distances between test points and trained labels points. Updating distance metrics with every iteration is computationally expensive, and that’s why KNN is a lazy learning algorithm.



* As you can verify from the above image, if we proceed with K=3, then we predict that test input belongs to class B, and if we continue with K=7, then we predict that test input belongs to class A.
* That’s how you can imagine that the K value has a powerful effect on KNN performance.

**Then how to select the optimal K value?**

* **There are no pre-defined statistical methods to find the most favorable value of K.**
* Initialize a random K value and start computing.
* Choosing a small value of K leads to unstable decision boundaries.
* The substantial K value is better for classification as it leads to smoothening the decision boundaries.
* **Derive a plot between error rate and K** denoting values in a defined range. Then choose the K value as having a minimum error rate.

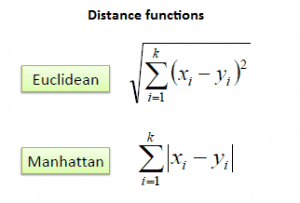
Now you will get the idea of choosing the optimal K value by implementing the model.

**Calculating distance:**

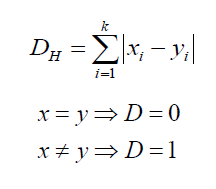
The first step is to calculate the distance between the new point and each training point. There are various methods for calculating this distance, of which the most commonly known methods are — Euclidian, Manhattan (for continuous) and Hamming distance (for categorical).

**Euclidean Distance:** Euclidean distance is calculated as the square root of the sum of the squared differences between a new point (x) and an existing point (y).

**Manhattan Distance:** This is the distance between real vectors using the sum of their absolute difference.



**Hamming Distance:** It is used for categorical variables. If the value (x) and the value (y) are the same, the distance D will be equal to 0 . Otherwise D=1.



[Most Popular Distance Metrics Used in KNN and When to Use Them - KDnuggets](https://www.kdnuggets.com/2020/11/most-popular-distance-metrics-knn.html)