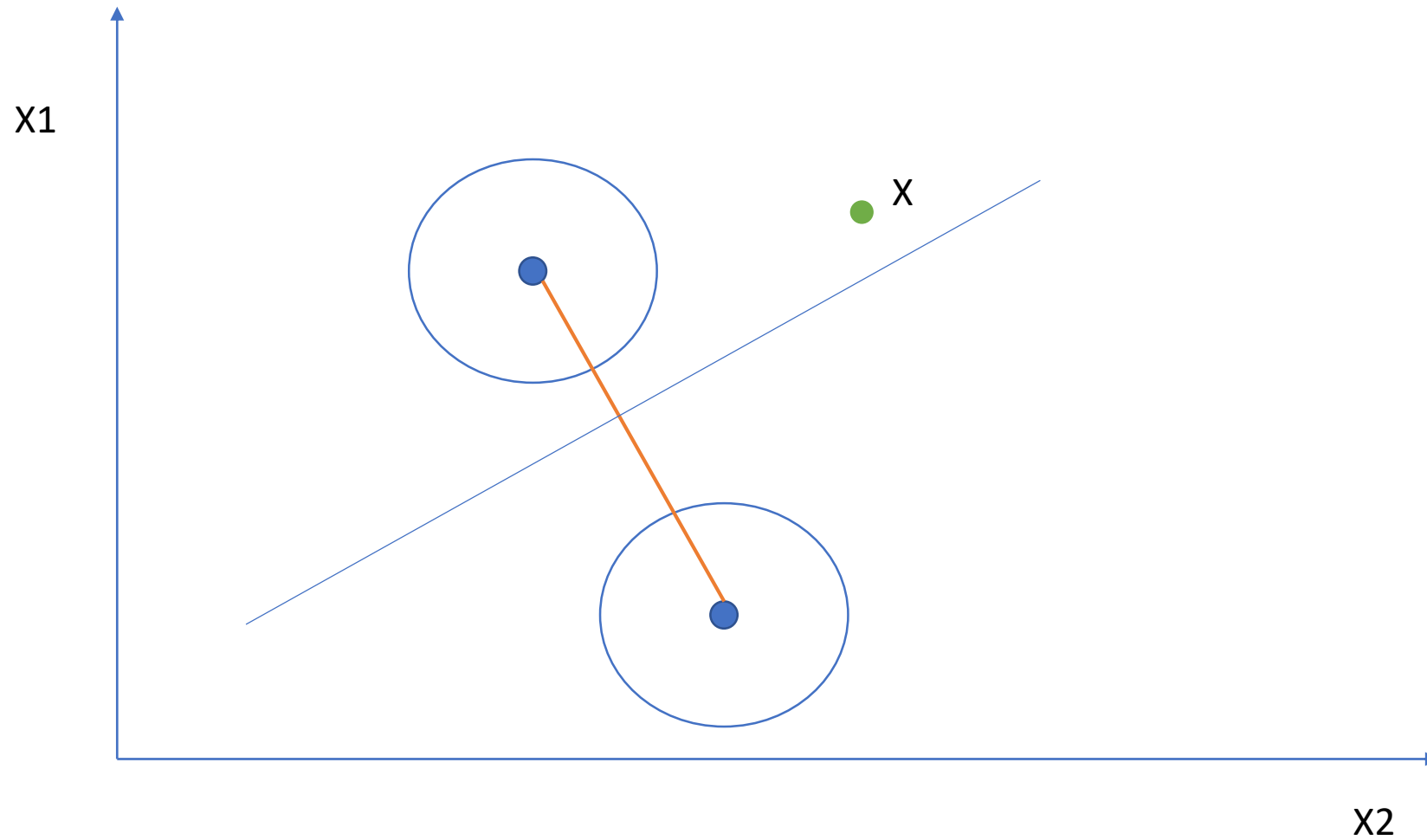


Nearest Neighbours

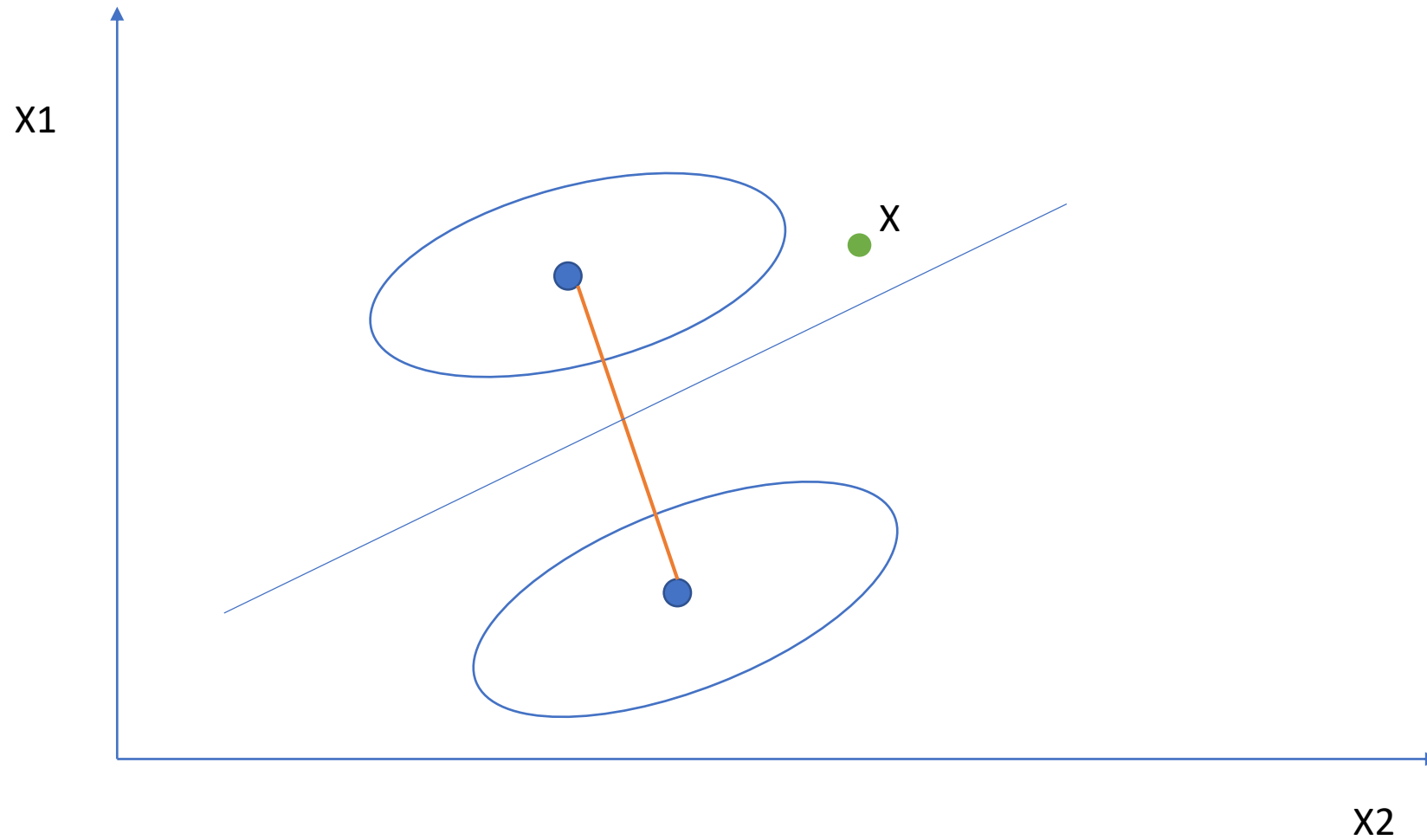
20CP401T

Himanshu K. Gajera
Department of Computer Science & Engineering
Pandit Deendayal Energy University, Gandhinagar

Minimum Distance Classifier



Minimum Distance Classifier



Nearest Neighbor Rule

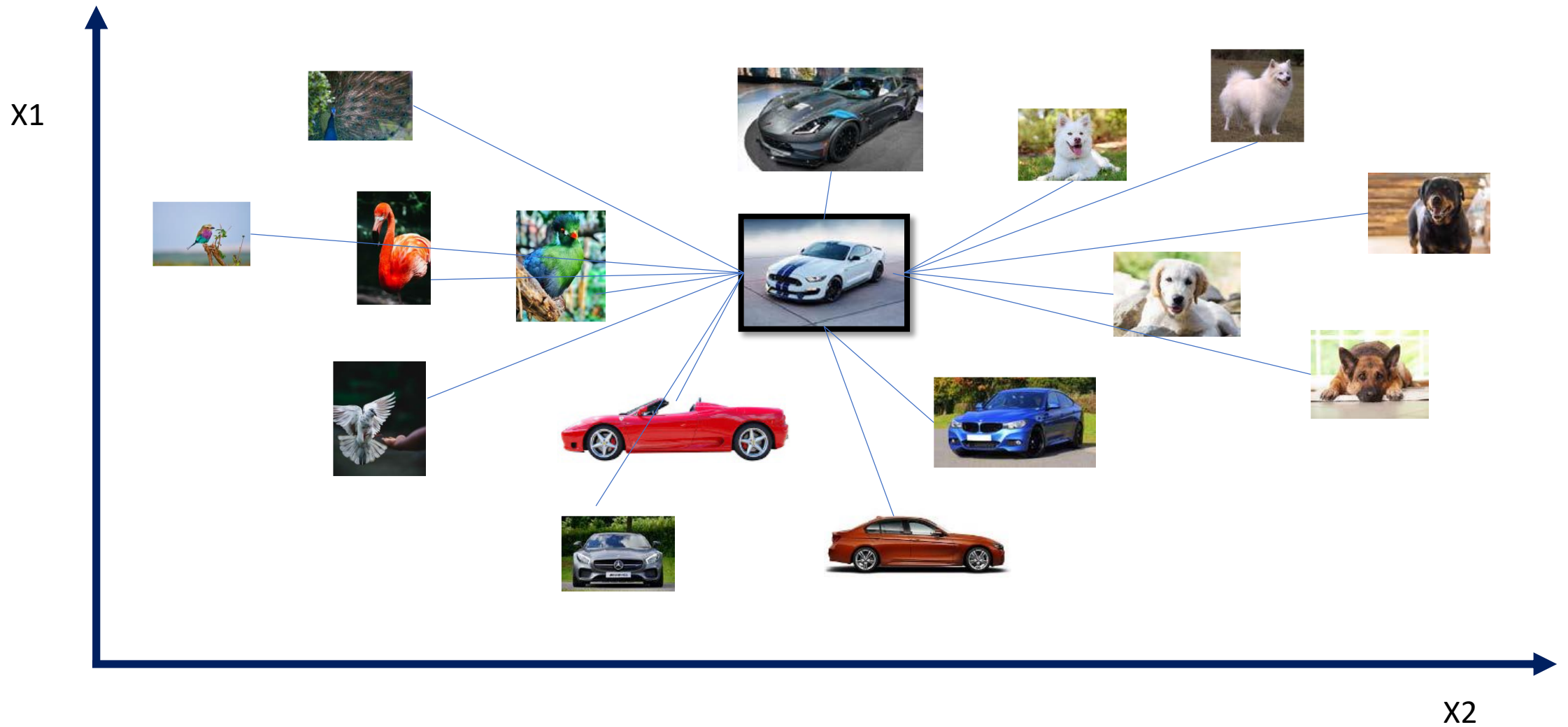


Image Source: Internet

K-Nearest Neighbor Rule

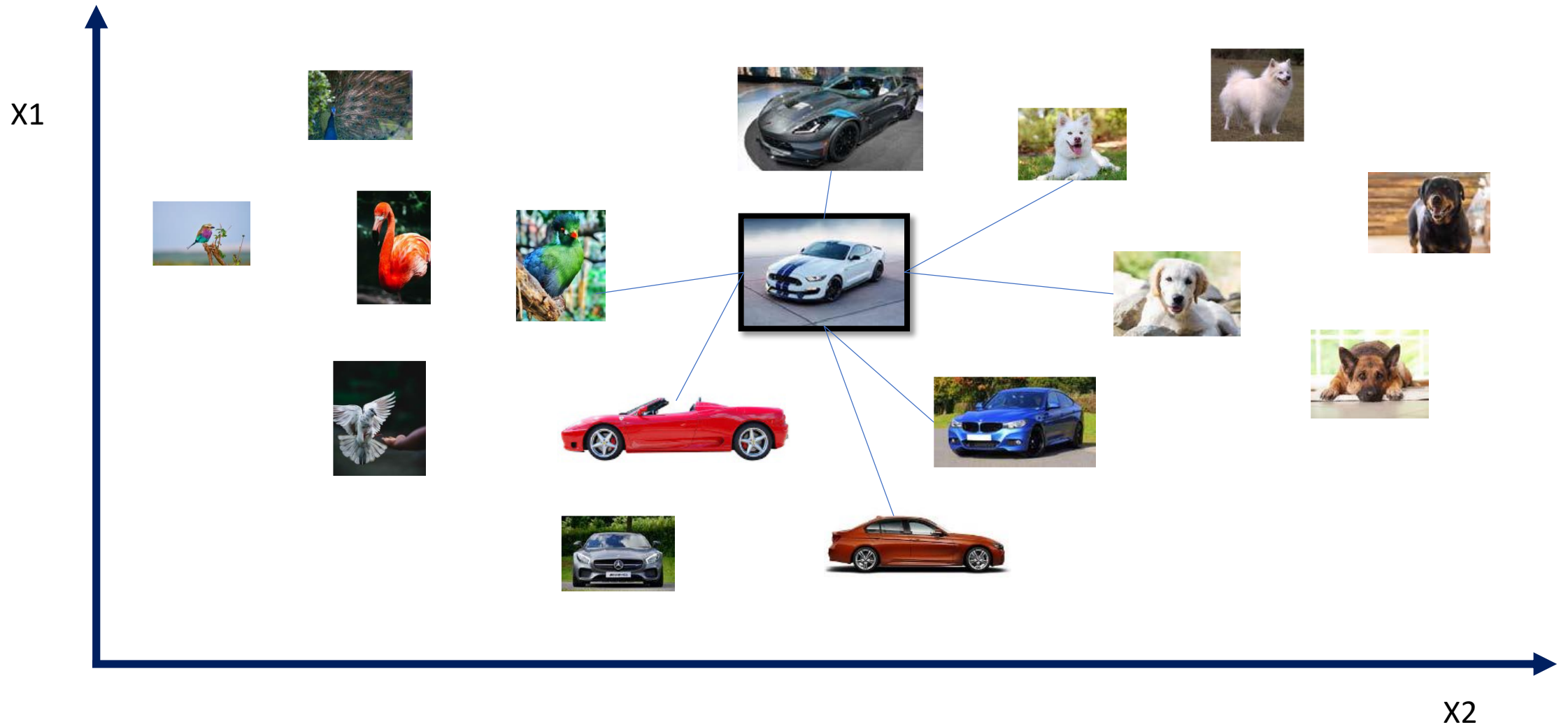


Image Source: Internet

K-Nearest Neighbor

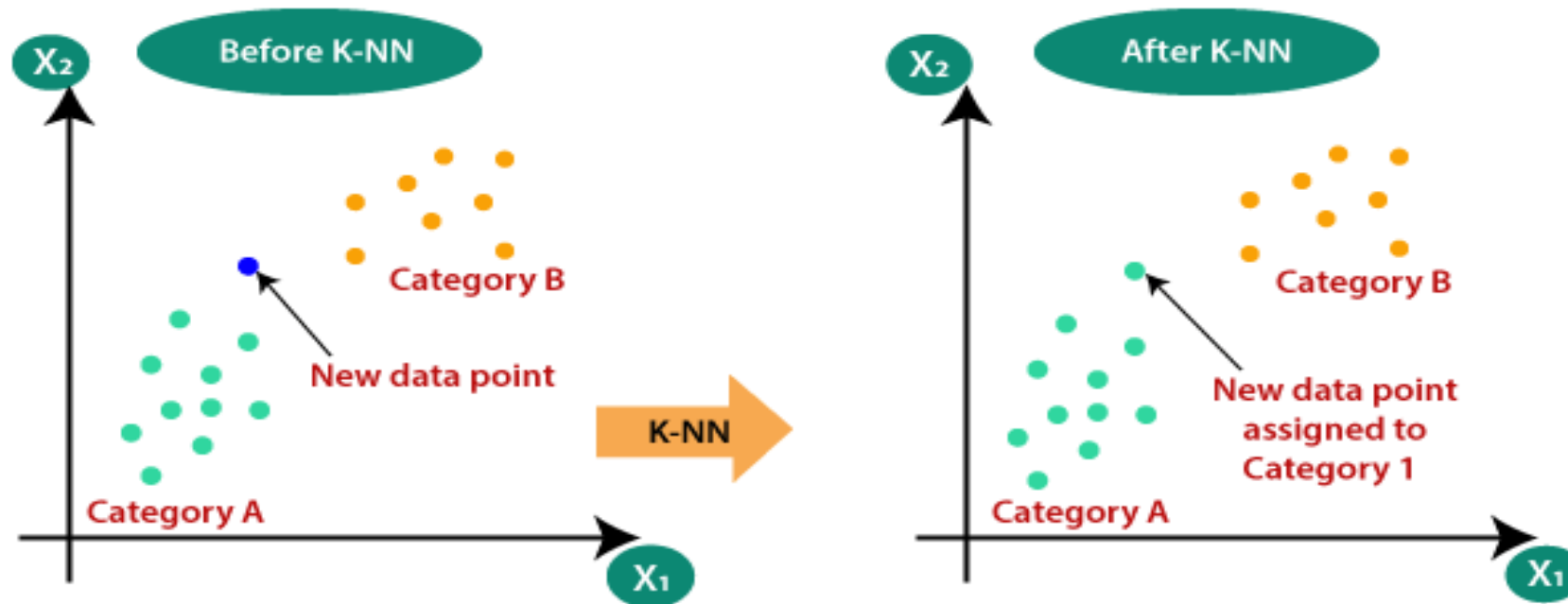
- K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN algorithm 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-Nearest Neighbor

- 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.

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 x_1 , 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:

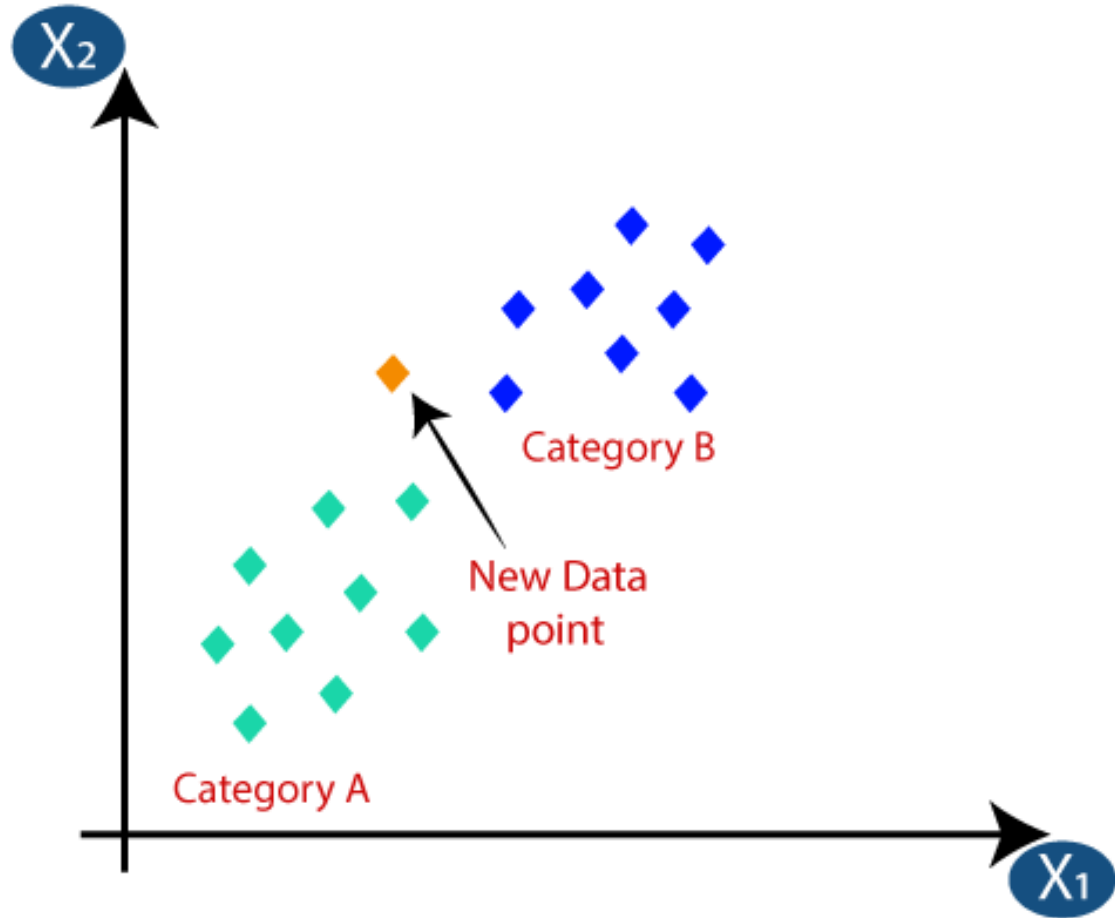


How does K-NN work?

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

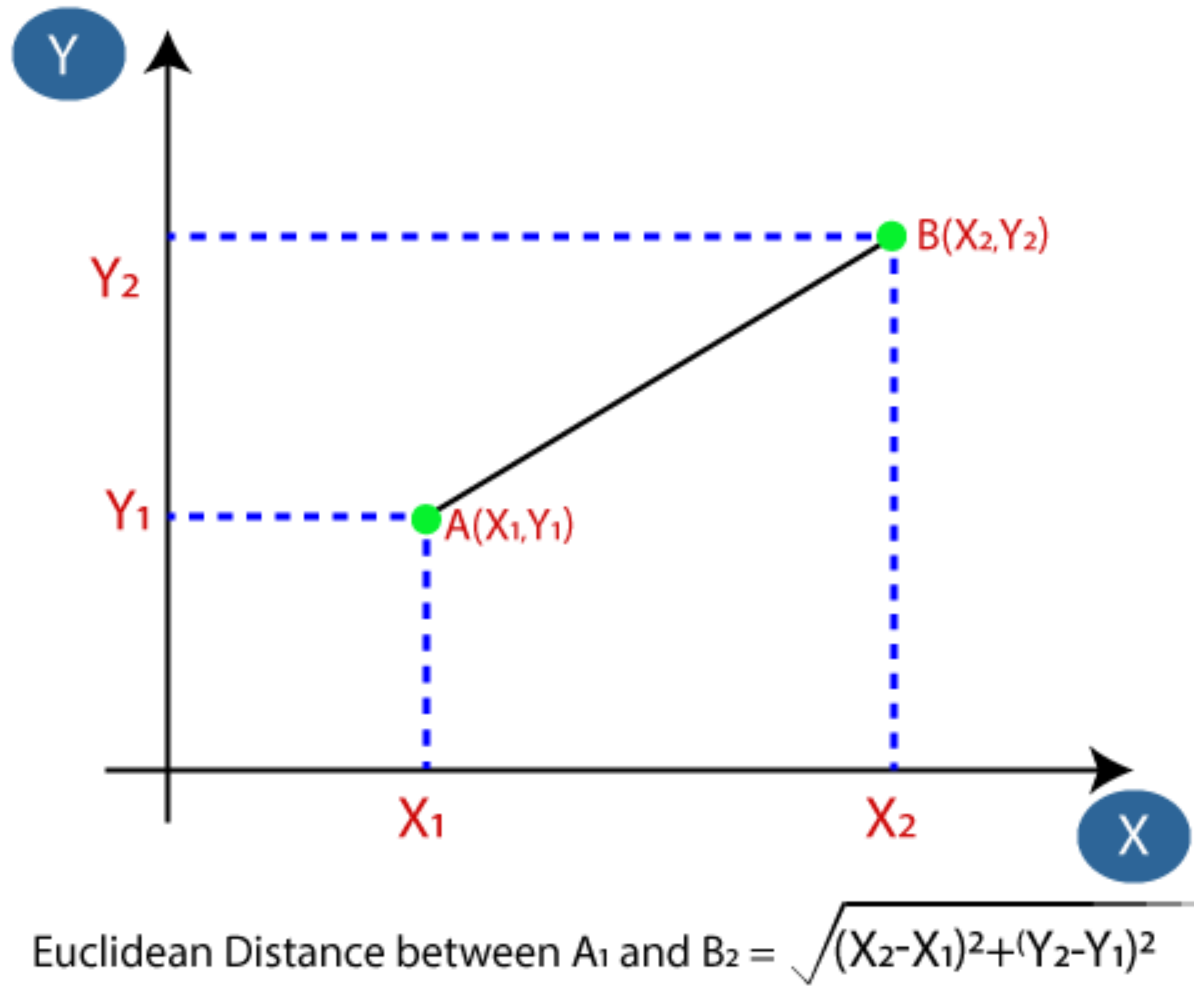
- Step-1: Select the number K of the neighbors
- Step-2: Calculate the Euclidean distance of K number of neighbors
- Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.
- Step-4: Among these k neighbors, count the number of the data points in each category.
- Step-5: Assign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

How does K-NN work?

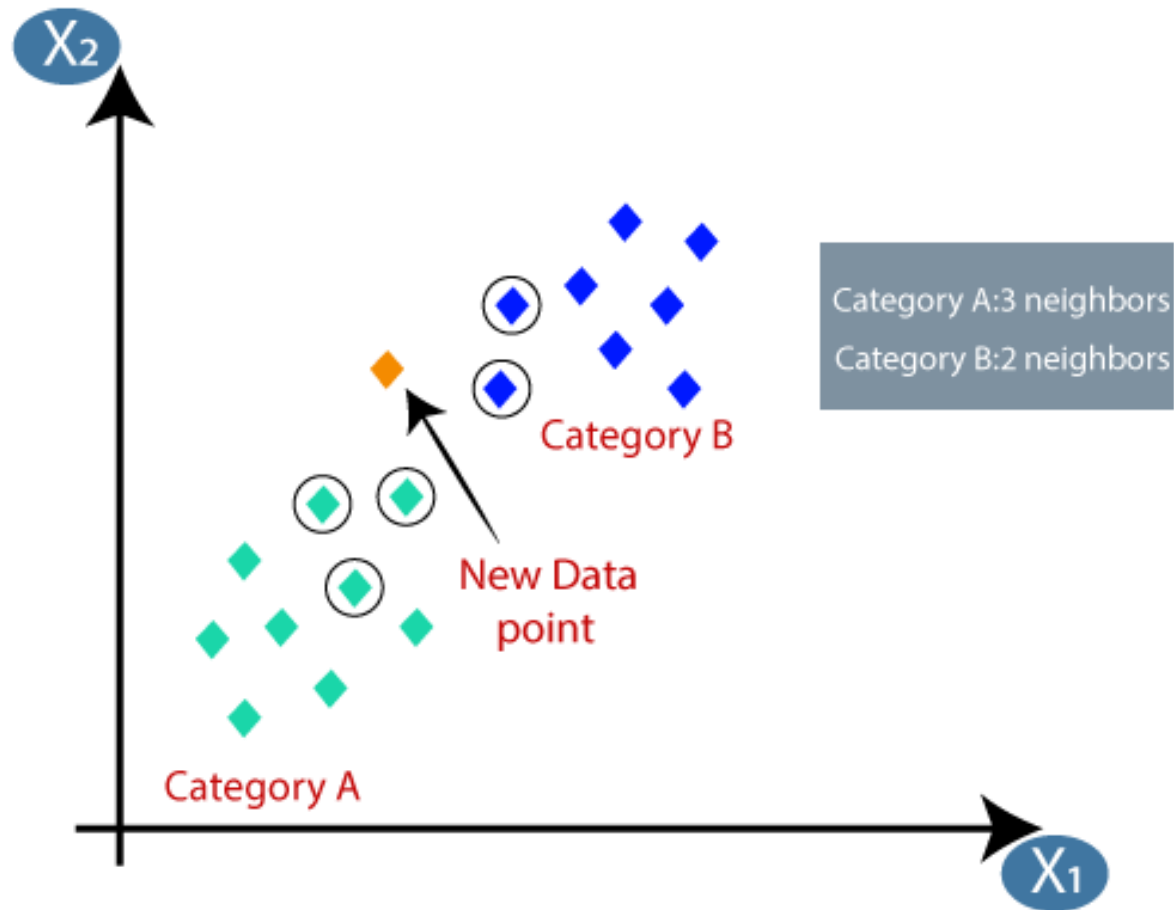


- Firstly, we will choose the number of neighbors, so we will choose the $k=5$.
- Next, we will calculate the Euclidean distance between the data points. The Euclidean distance is the distance between two points, which we have already studied in geometry. It can be calculated as:

How does K-NN work?

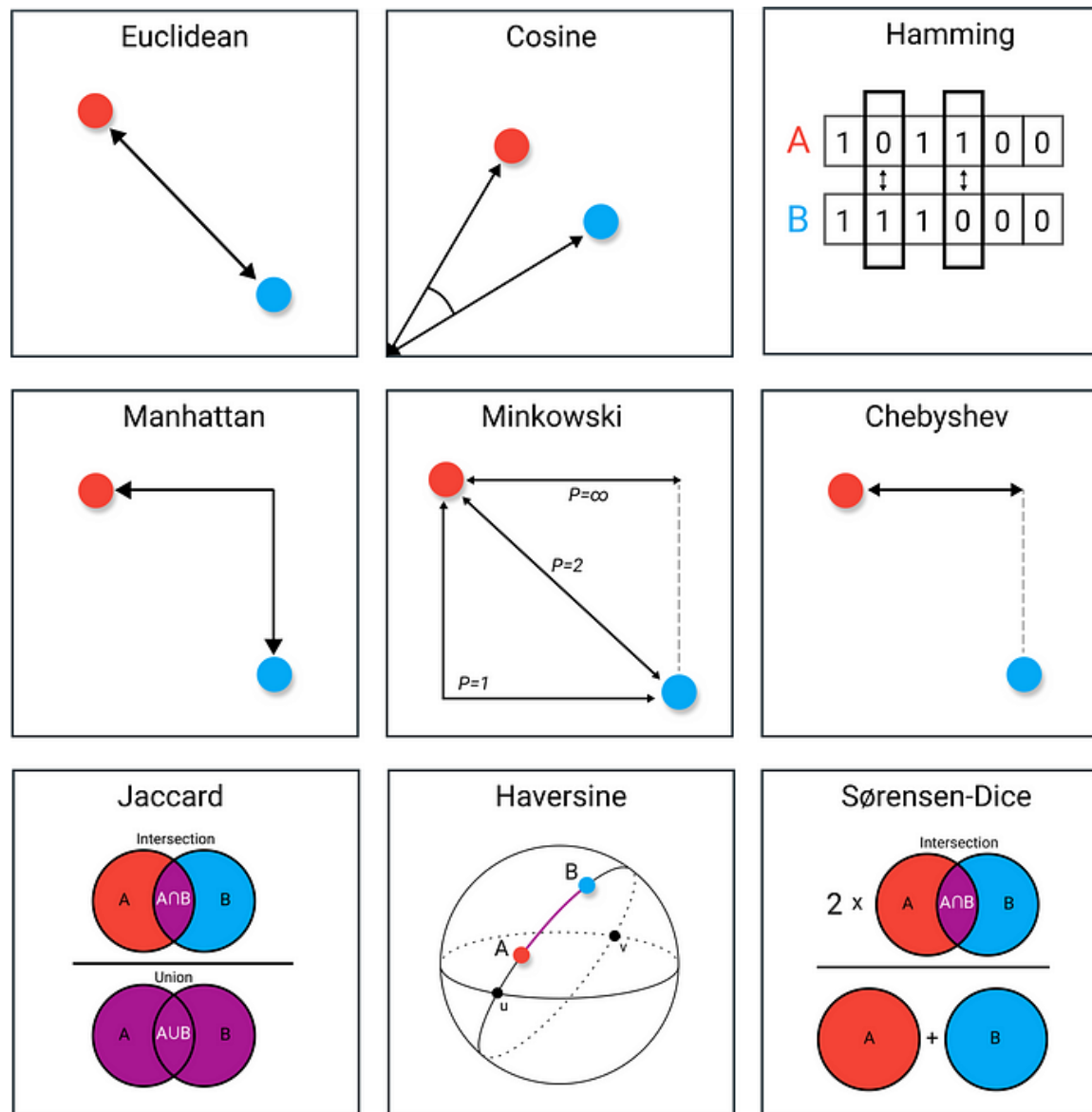


How does K-NN work?



- By calculating the Euclidean distance we got the nearest neighbors, as three nearest neighbors in category A and two nearest neighbors in category B.
- As we can see the 3 nearest neighbors are from category A, hence this new data point must belong to category A.

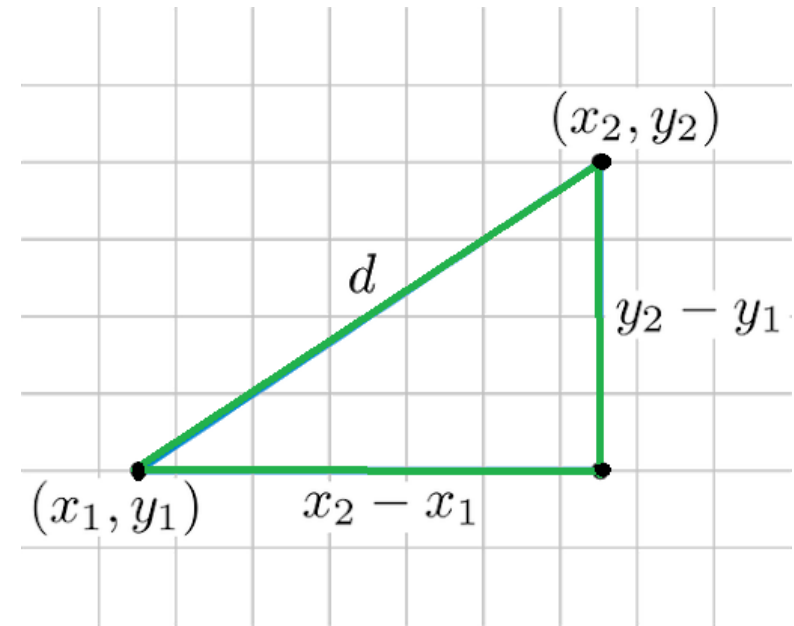
Different Distance in ML



Euclidean Distance

Euclidean distance is considered the traditional metric for problems with geometry. It can be simply explained as the **ordinary distance** between two points. It is one of the most used algorithms in the cluster analysis. One of the algorithms that use this formula would be **K-mean**. Mathematically it computes the **root of squared differences** between the coordinates between two objects.

$$\begin{aligned}d(\mathbf{p}, \mathbf{q}) &= d(\mathbf{q}, \mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2} \\&= \sqrt{\sum_{i=1}^n (q_i - p_i)^2}\end{aligned}$$



These are two points x and y : (5, 1), (9, -2). Using Euclidean distance, here is the distance between these two points:

$$d(x, y) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad \sqrt{(9 - 5)^2 + (-2 - 1)^2} = \sqrt{25} = 5$$

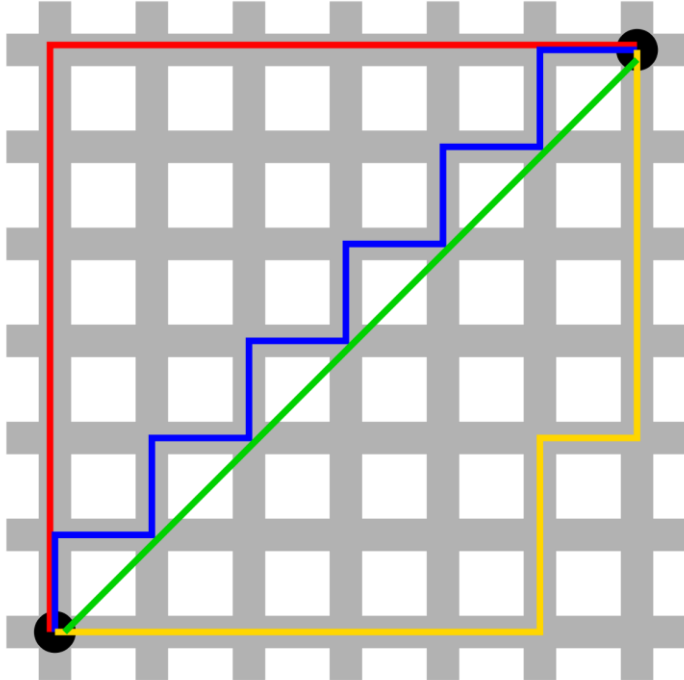
Manhattan Distance

This determines the absolute difference among the pair of the coordinates.

Suppose we have two points P and Q to determine the distance between these points we simply have to calculate the perpendicular distance of the points from X-Axis and Y-Axis.

In a plane with P at coordinate (x1, y1) and Q at (x2, y2).

Manhattan distance between P and Q = $|x1 - x2| + |y1 - y2|$ $d(x, y) = \sum |x_i - y_i|$



Here the total distance of the **Red** line gives the Manhattan distance between both the points.

Here is an x and y:

x = [3, 6, 11, 8]

y = [0, 9, 5, 3]

The Manhattan distance between x and y:

$$d = |3-0| + |6-9| + |11-5| + |8-3| = 3+3+6+5 = 17$$

Manhattan distance works better in higher dimensionality data. But it is less intuitive. Though it's not a big problem.

Minkowski distance

It is the generalized form of the Euclidean and Manhattan Distance Measure. In an N-dimensional space, a point is represented as

$$d(x, y) = \sqrt[h]{|x_1 - y_1|^h + |x_2 - y_2|^h + \dots + |x_p - y_p|^h}$$

Here, x and y are two p-dimensional data objects and 'h' is the order. The distance defined this way is also called the L-h norm.

If '**h**' is **1**, it becomes the **manhattan** distance, and if **h** = **2**, it becomes the **Euclidean** distance.

Here is an example:

x = [9, 2, -1, 12]

y = [-4, 5, 10, 13]

When h = 1, The formula becomes the Manhattan distance formula which is referred to as the L-1 norm:

$$d = \sqrt[1]{|9 - (-4)|^1 + |2 - 5|^1 + |1 - 10|^1 + |12 - 13|^1}$$

When, h= 2, the formula becomes Euclidean distance formula which is also referred to as L-2 norm:

$$d = \sqrt[2]{|9 - (-4)|^2 + |2 - 5|^2 + |1 - 10|^2 + |12 - 13|^2}$$

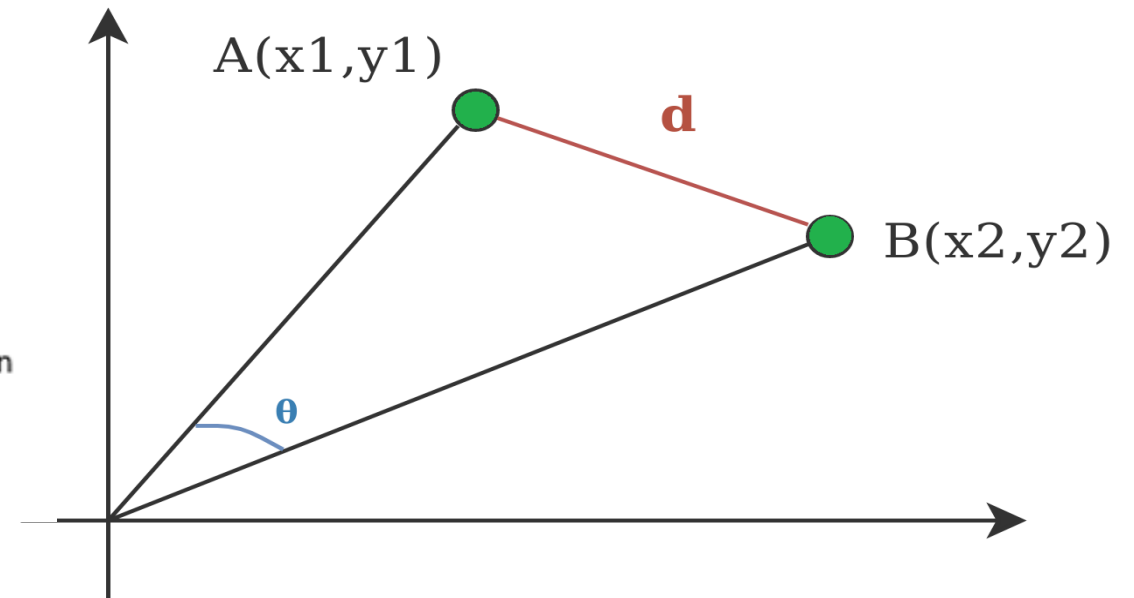
h = infinity, Chebychev Distance

Cosine Index

Cosine distance measure for clustering determines the cosine of the angle between two vectors given by the following formula. $\text{sim}(A, B) = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|}$

Here (theta) gives the angle between two vectors and A, B are n-dimensional vectors.

Documents	team coach	hockey	baseball	penalty	win	loss	season
d1	5	2	1	0	1	3	0
d2	4	0	0	2	2	2	1



Let's find the cosine similarity between d1 and d2

Here is the formula:

$$\cos(d1, d2) = \frac{d1 \cdot d2}{\|d1\| * \|d2\|}$$

$$d1 \cdot d2 = 5*4 + 2*0 + 1*0 + 0*2 + 1*2 + 3*2 + 0*1 = 28$$

$$\|d1\| = (5*5 + 2*2 + 1*1 + 0*0 + 1*1 + 3*3 + 0*0)**0.5 = 6.32$$

$$\|d2\| = (4*4 + 0*0 + 0*0 + 2*2 + 2*2 + 2*2 + 1*1)**0.5 = 5.39$$

$$\cos(d1, d2) = 28 / (6.32*5.39) = 0.82$$

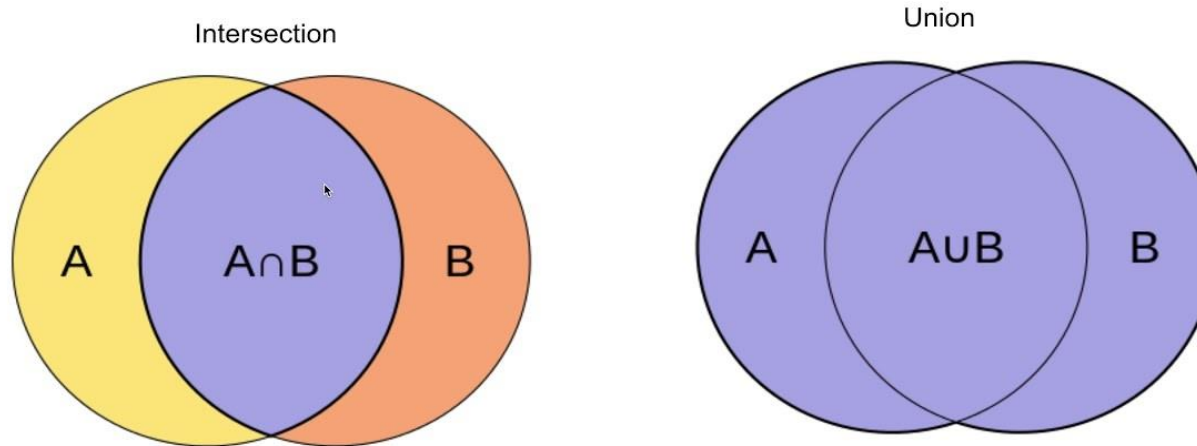
So, this is how we get cosine similarity.

Jaccard distance

The Jaccard distance measures the similarity of the two data set items as the intersection of those items divided by the union of the data items.

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

Jaccard coefficient



$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

Text mining: find the similarity between two text documents using the number of terms used in both documents.

E-Commerce: from a market database of thousands of customers and millions of items, find similar customers via their purchase history.

Hamming Distance

Hamming distance is useful for finding the distance between two binary vectors. In machine learning you will often encounter the one-hot encoded data. Hamming distance would be useful in those cases and many others. **Hamming distance is used in case of categorical variable.**

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

Here n is the length of x or y. They are supposed to have the same length. If x and y are:

x = [0, 1, 0, 1]

y = [1, 1, 0, 0]

Then the distance between x and y is:

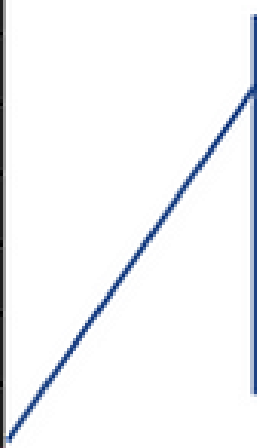
$$d = (|0-1| + |1-1| + |0-0| + |1-0|) / 4 = 1+0+0+1 = 2/4 = 0.5$$

Here 4 is the length of x or y. Both have 4 elements in them.

As you can see from the example above, vectors need to be of the same length.

How does K-NN work?

Customer	Age	Loan	Default
John	25	40000	N
Smith	35	60000	N
Alex	45	80000	N
Jade	20	20000	N
Kate	35	120000	N
Mark	52	18000	N
Anil	23	95000	Y
Pat	40	62000	Y
George	60	100000	Y
Jim	48	220000	Y
Jack	33	150000	Y
Andrew	48	142000	?



We need to predict
Andrew default status
by using Euclidean
distance

We need to predict Andrew default status (Yes or No).

How does K-NN work?

Customer	Age	Loan	Default	Euclidean distance
John	25	40000	N	1,02,000.00
Smith	35	60000	N	82,000.00
Alex	45	80000	N	62,000.00
Jade	20	20000	N	1,22,000.00
Kate	35	120000	N	22,000.00
Mark	52	18000	N	1,24,000.00
Anil	23	95000	Y	47,000.01
Pat	40	62000	Y	80,000.00
George	60	100000	Y	42,000.00
Jim	48	220000	Y	78,000.00
Jack	33	150000	Y	8,000.01
Andrew	48	142000	?	

First Step calculate the Euclidean distance $\text{dist}(d) = \text{Sq.rt}(x_1 - y_1)^2 + (x_2 - y_2)^2$
 $= \text{Sq.rt}(48 - 25)^2 + (142000 - 40000)^2$
 $\text{dist}(d_1) = 1,02,000.$

We need to calculate the distance for all the datapoints

How does K-NN work?

Calculate Euclidean distance for all the data points.

Customer	Age	Loan	Default	Euclidean distance	Minimum Euclidean Distance
John	25	40000	N	1,02,000.00	
Smith	35	60000	N	82,000.00	
Alex	45	80000	N	62,000.00	5
Jade	20	20000	N	1,22,000.00	
Kate	35	120000	N	22,000.00	2
Mark	52	18000	N	1,24,000.00	
Anil	23	95000	Y	47,000.01	4
Pat	40	62000	Y	80,000.00	
George	60	100000	Y	42,000.00	3
Jim	48	220000	Y	78,000.00	
Jack	33	150000	Y	8,000.01	1
Andrew	48	142000	?		

Let assume K = 5

Find minimum euclidean distance and rank in order (ascending)

In this case, 5 minimum euclidean distance. With k=5, there are two Default = N and three Default = Y out of five closest neighbors.

We can say Andrew default status is 'Y' (Yes)

How to select the value of K in the K-NN Algorithm?

- There is no particular way to determine the best value for "K", so we need to try some values to find the best out of them.
- A very low value for K such as $K=1$ or $K=2$, can be noisy and lead to the effects of outliers in the model.
- Large values for K are good, but it may find some difficulties.
- Another way to choose K is through cross-validation. One way to select the cross-validation dataset from the training dataset.
- In general, practice, choosing the value of k is $k = \sqrt{N}$ where N stands for the number of samples in your training dataset.
- Try and keep the value of k odd in order to avoid confusion between two classes of data

Advantages and Disadvantages of K-NN

Advantages of KNN Algorithm:

- It is simple to implement.
- It is robust to the noisy training data
- Flexible to feature/distance choices
- Naturally handles multi-class cases
- Can do well in practice with enough representative data
- It can be more effective if the training data is large.

Disadvantages of KNN Algorithm:

- Always needs to determine the value of K which may be complex some time.
- The computation cost is high because of calculating the distance between the data points for all the training samples.
- Computation cost is quite high because we need to compute the distance of each query instance to all training samples.
- Storage of data
- Must know we have a meaningful distance function.