KNN(K-Nearest Neighbours)Algorithm

KNN (k-Nearest Neighbors) algorithm is a supervised machine learning algorithm that falls under the category of instance-based learning or lazy learning

Instance Based Learning

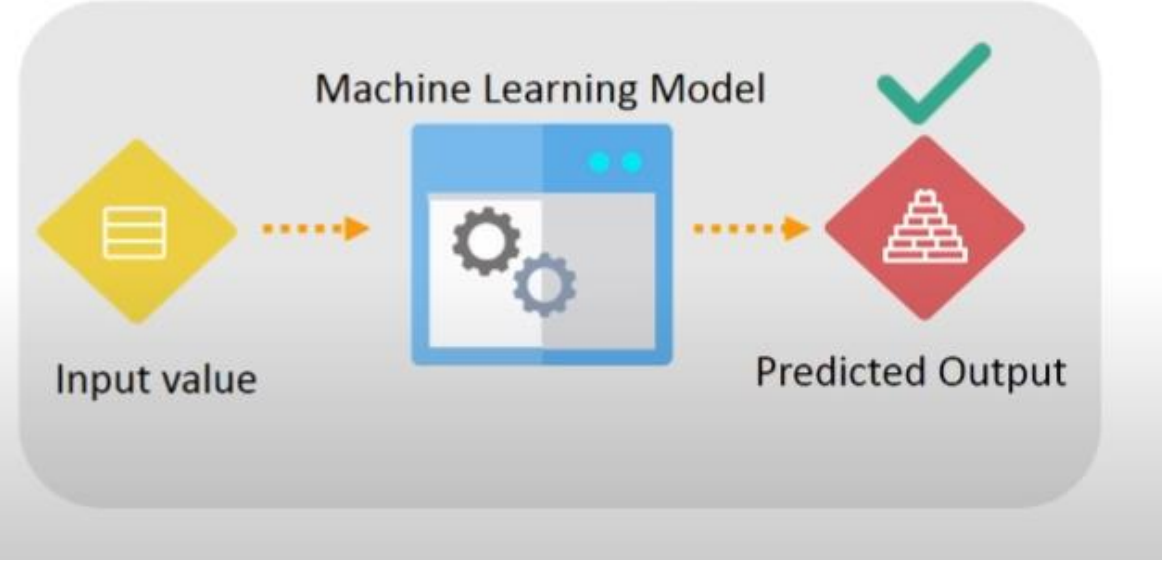
Instance-based learning is a type of machine learning where the algorithm makes predictions based on the similarity between new, unseen data points and the labeled training examples

We will take a closer look at...

- Why do we need KNN?
- What is KNN?
- How do we choose the factor 'K'?
- When do we use KNN?
- How does KNN Algorithm work?
- Use Case: Predict whether a person will have diabetes or not

Why do we need KNN?

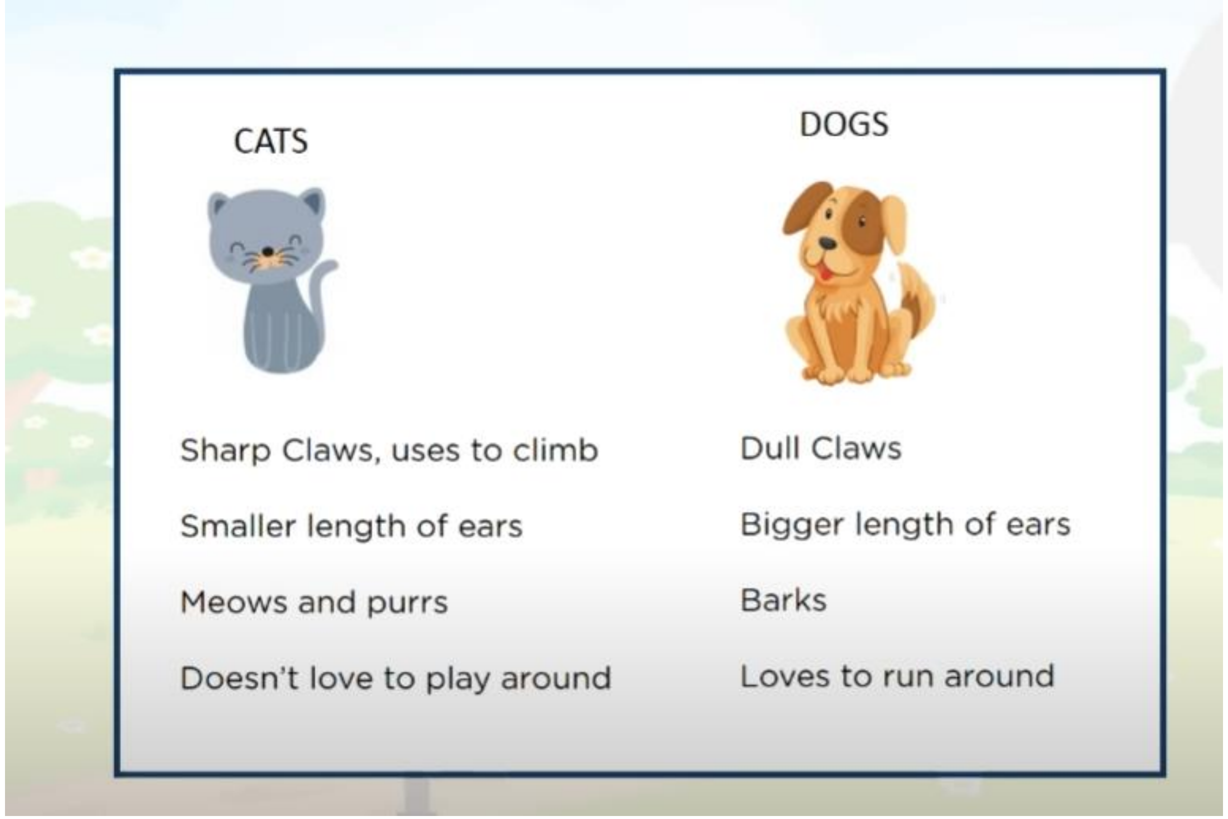
By now, we all know Machine learning models makes predictions by learning from the past data available



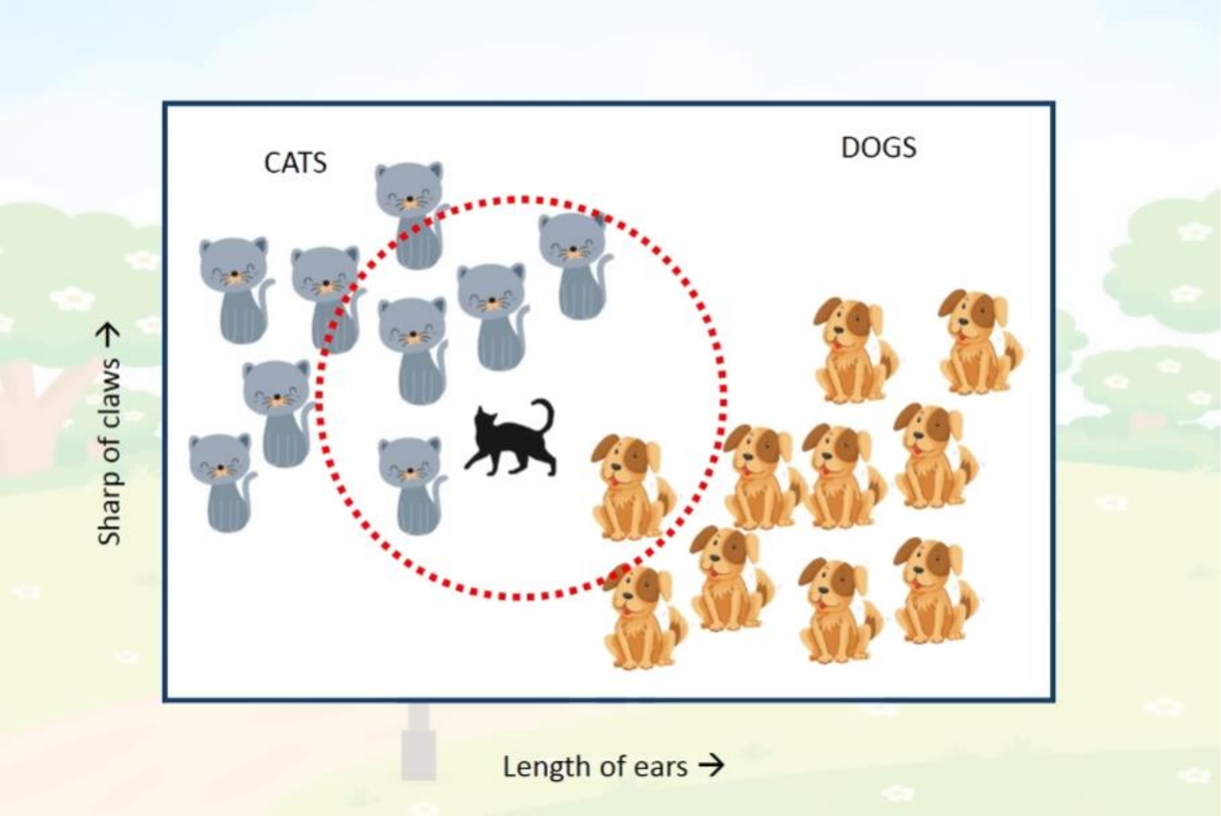
For example

Is that a dog?

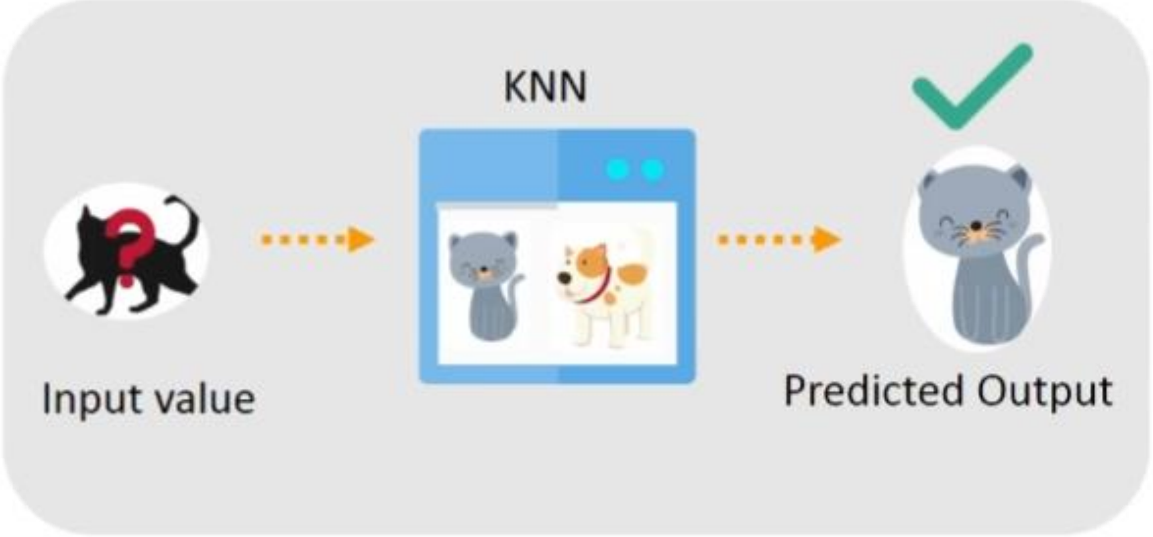
No dear, you can differentiate between a cat and a dog based on their characteristics



CATS	DOGS
Sharp Claws, uses to climb	Dull Claws
Smaller length of ears	Bigger length of ears
Meows and purrs	Barks
Doesn't love to play around	Loves to run around



Because KNN is based on feature similarity, we can do classification using KNN Classifier!



KNN

Input value

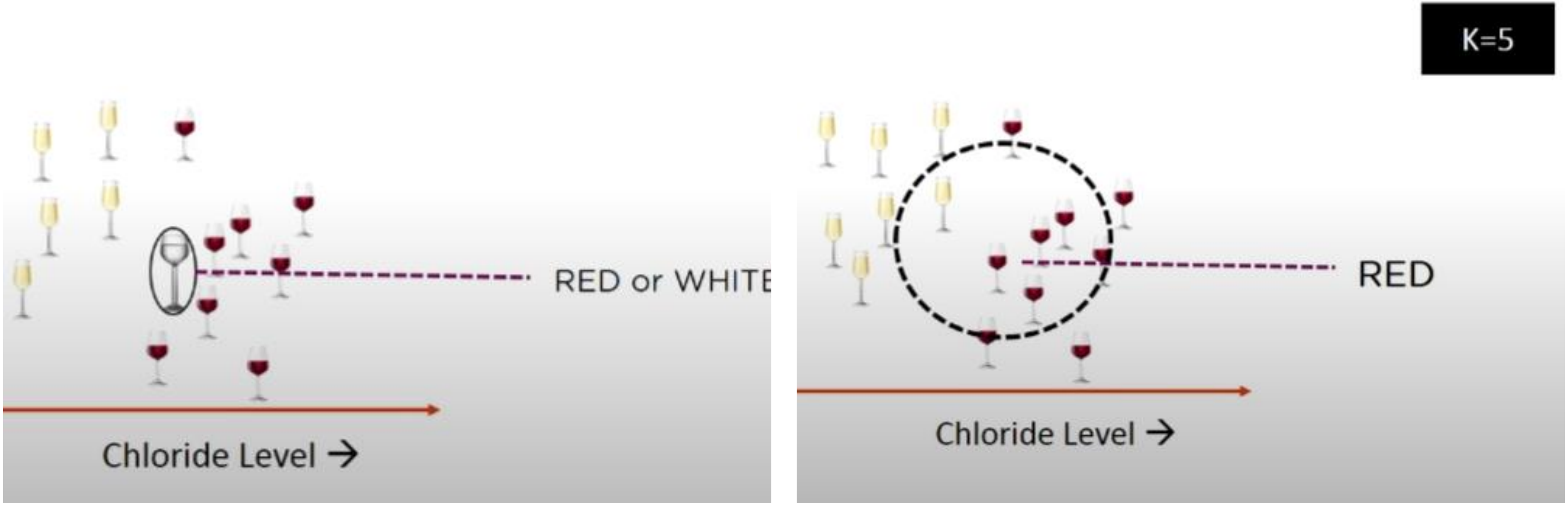
Predicted Output

What is KNN Algorithm?

- KNN -K Nearest Neighbors, is one of the simplest Supervised Machine Learning algorithm mostly used for Classification
- It classifies a data point based on how its neighbors are classified
- For example: Its CAT or not a CAT

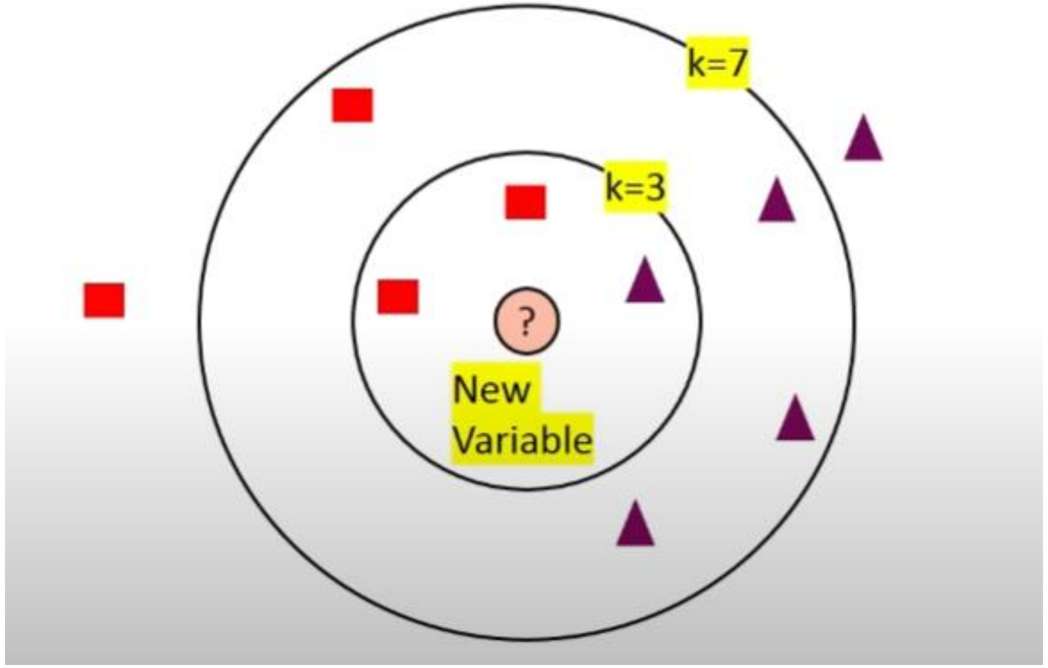
KNN stores all available cases and classifies new cases based on a **similarity** measure

K in KNN is a parameter that refers to the number of nearest neighbors to include in the majority voting process



How do we choose the factor 'k'?

- KNN Algorithm is based on feature similarity: Choosing the right value of k is a process called **parameter tuning**, and is important for better accuracy



For k=3 , we classify as SQUARE and for k=7, we classify as TRIANGLE

To choose a value of k:

- Sqrt(n), where n is the total number of data points
- Odd value of K is selected to avoid confusion between two classes of data

When do we use KNN Algorithm?

Dataset is small

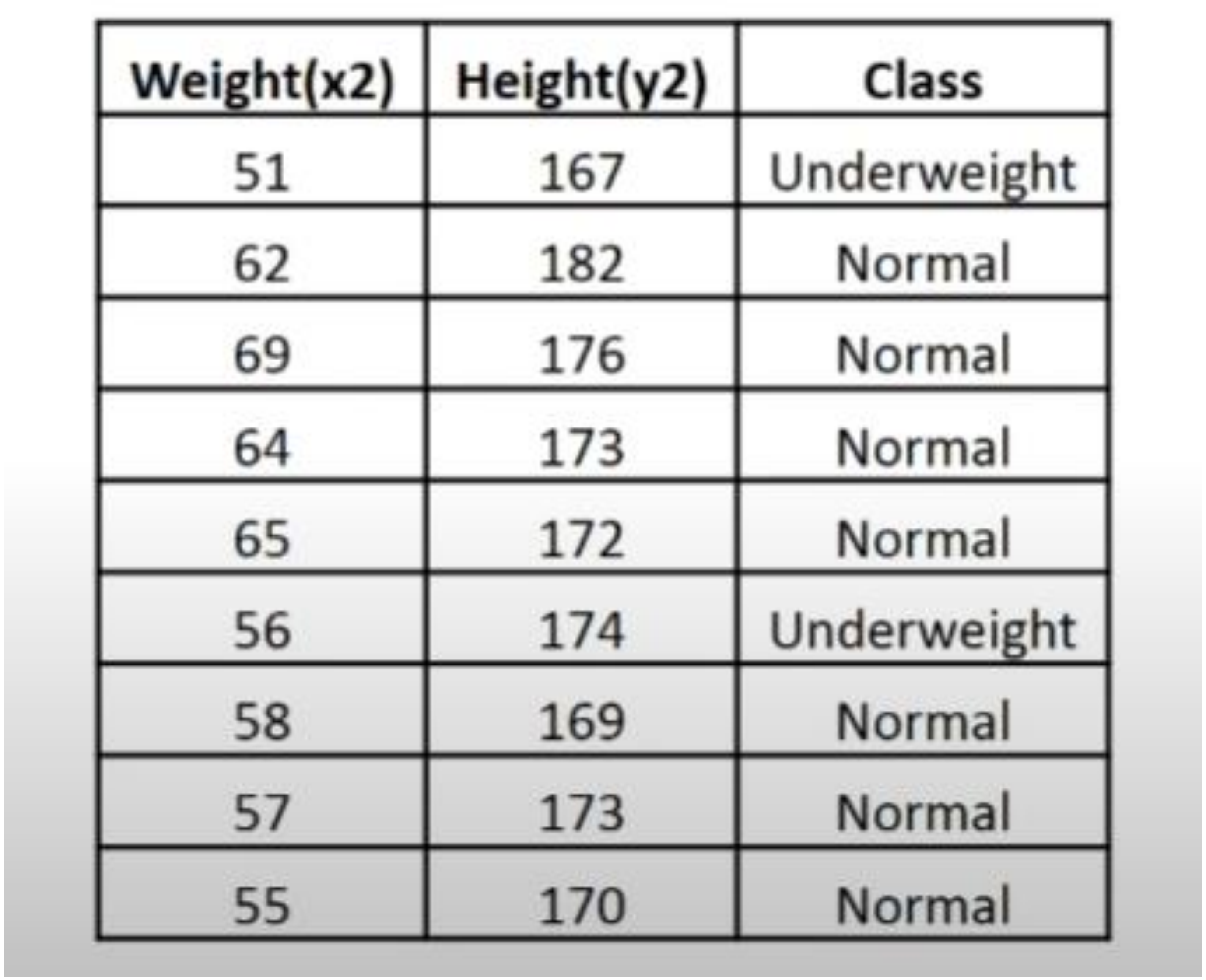
Because KNN is a 'lazy learner' i.e. doesn't learn a discriminative function from the training set

Data is labeled

Data is noise free

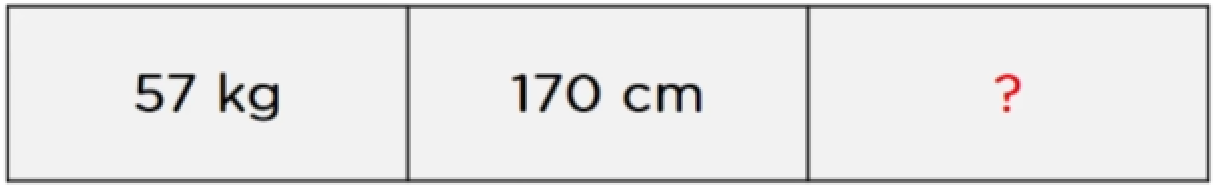
How does KNN Algorithm work?

- Consider a dataset having two variables: height (cm) & weight (kg) and each point is classified as **Normal** or **Underweight**



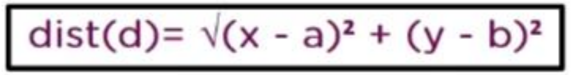
Weight(x2)	Height(y2)	Class
51	167	Underweight
62	182	Normal
69	176	Normal
64	173	Normal
65	172	Normal
56	174	Underweight
58	169	Normal
57	173	Normal
55	170	Normal

On the basis of the given data we have to classify the below set as Normal or Underweight using KNN

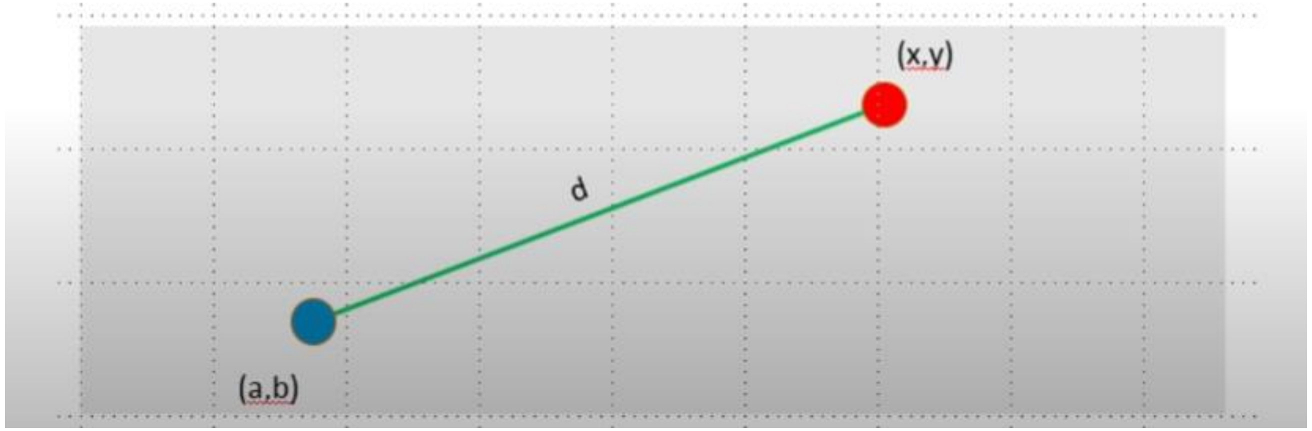


57 kg	170 cm	?

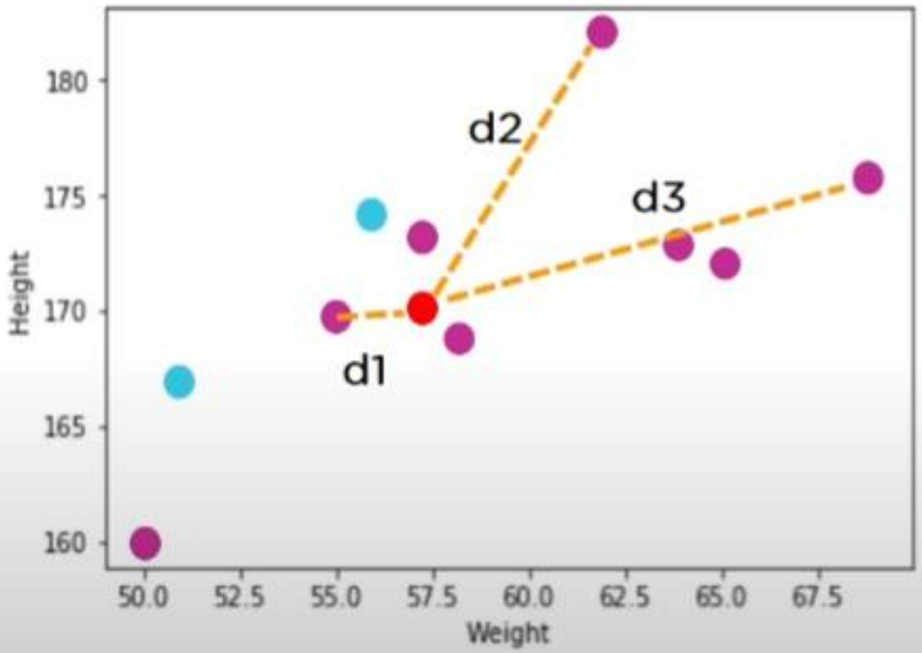
To find the nearest neighbors, we will calculate Euclidean distance



$\text{dist}(d)= \sqrt{(x - a)^2 + (y - b)^2}$



Let's calculate it to understand clearly..



● Unknown data point

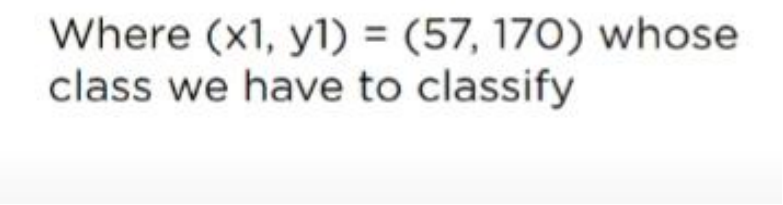
dist(**d1**)= $\sqrt{}(170-167)^2 + (57-51)^2$ ~= 6.7

dist(**d2**)= $\sqrt{}(170-182)^2 + (57-62)^2$ ~= 13

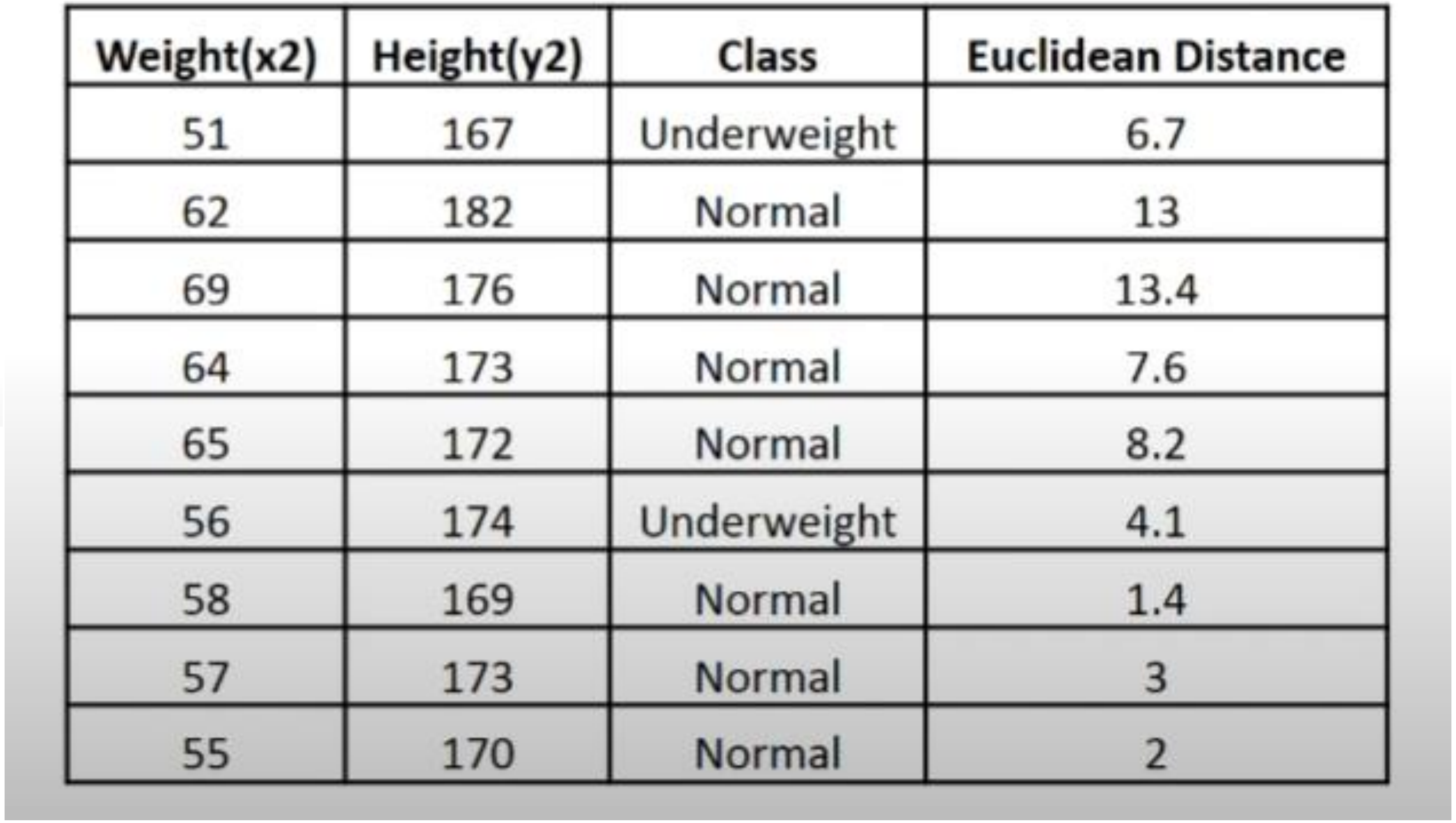
dist(**d3**)= $\sqrt{}(170-176)^2 + (57-69)^2$ ~= 13.4

Similarly, we will calculate Euclidean distance of unknown data point from all the points in the dataset

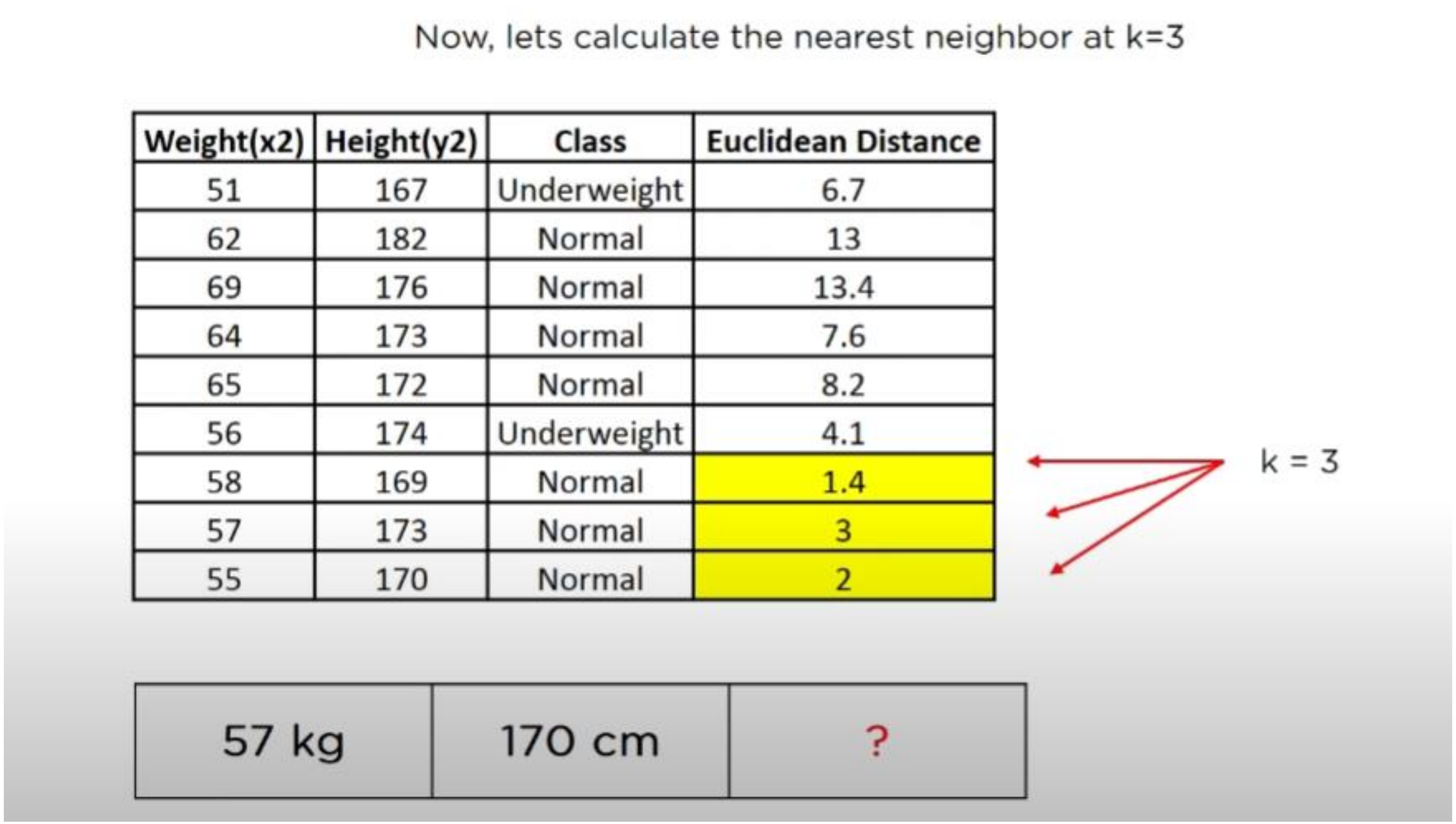
Hence, we have calculated the **Euclidean distance** of unknown data point from all the points as shown:



Where (x1, y1) = (57, 170) whose class we have to classify



Weight(x2)	Height(y2)	Class	Euclidean Distance
51	167	Underweight	6.7
62	182	Normal	13
69	176	Normal	13.4
64	173	Normal	7.6
65	172	Normal	8.2
56	174	Underweight	4.1
58	169	Normal	1.4
57	173	Normal	3
55	170	Normal	2



Now, lets calculate the nearest neighbor at k=3

Weight(x2)	Height(y2)	Class	Euclidean Distance
51	167	Underweight	6.7
62	182	Normal	13
69	176	Normal	13.4
64	173	Normal	7.6
65	172	Normal	8.2
56	174	Underweight	4.1
58	169	Normal	1.4
57	173	Normal	3
55	170	Normal	2

k = 3

57 kg	170 cm	?

Classified as NORMAL

Overview

- **KNN (k-Nearest Neighbors)** algorithm is not introduced to **solve a specific problem** that other algorithms fail to solve, but rather it is a general-purpose supervised learning algorithm used for both classification and regression tasks. However, KNN has some advantages over other algorithms in certain situations. For example:
- KNN can handle **non-linear decision boundaries**, which may be difficult for linear models like logistic regression to handle.
- KNN can easily adapt to changes in the data, as it does not make any assumptions about the underlying data distribution.
- KNN is a **simple and easy-to-implement** algorithm, which makes it a good choice for beginners in machine learning.
- KNN can handle **multiclass classification** problems easily without any modifications.
- Therefore, KNN can be a useful tool in many scenarios where other algorithms may not be the best choice.