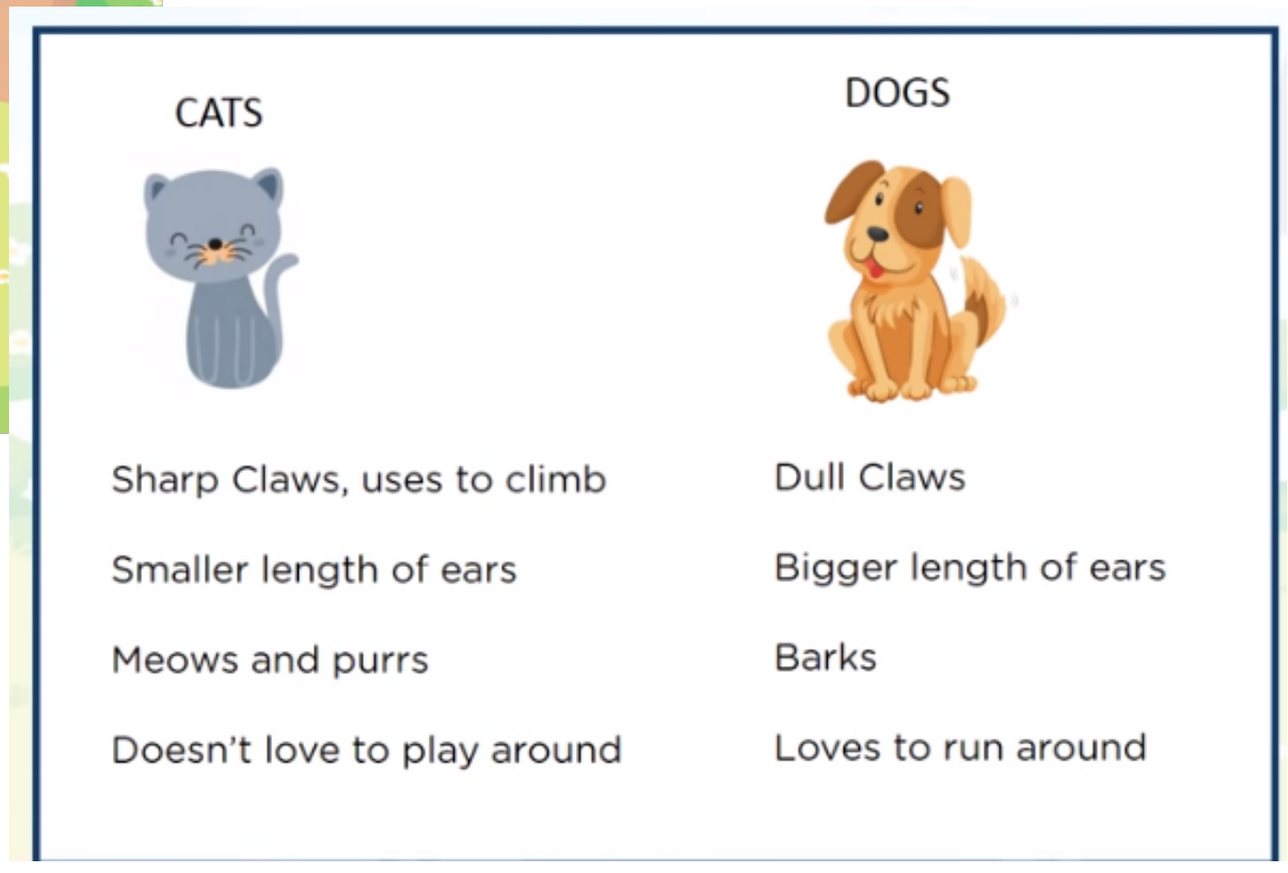
K-Nearest Neighbor

Roadmap

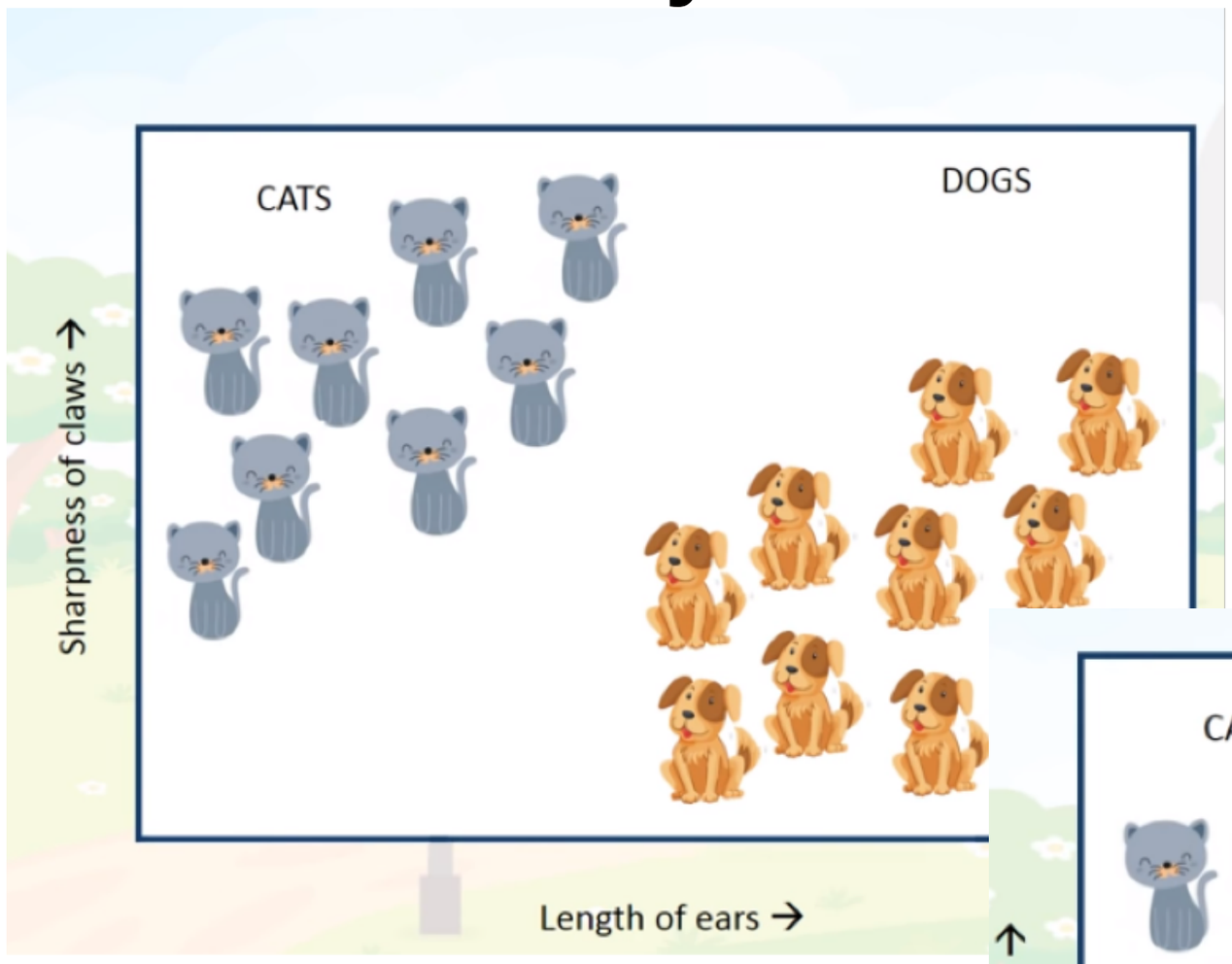
- Why do we need KNN?
- What is KNN?
- How to do we choose the factor ‘k’?
- When do we use KNN?
- Example

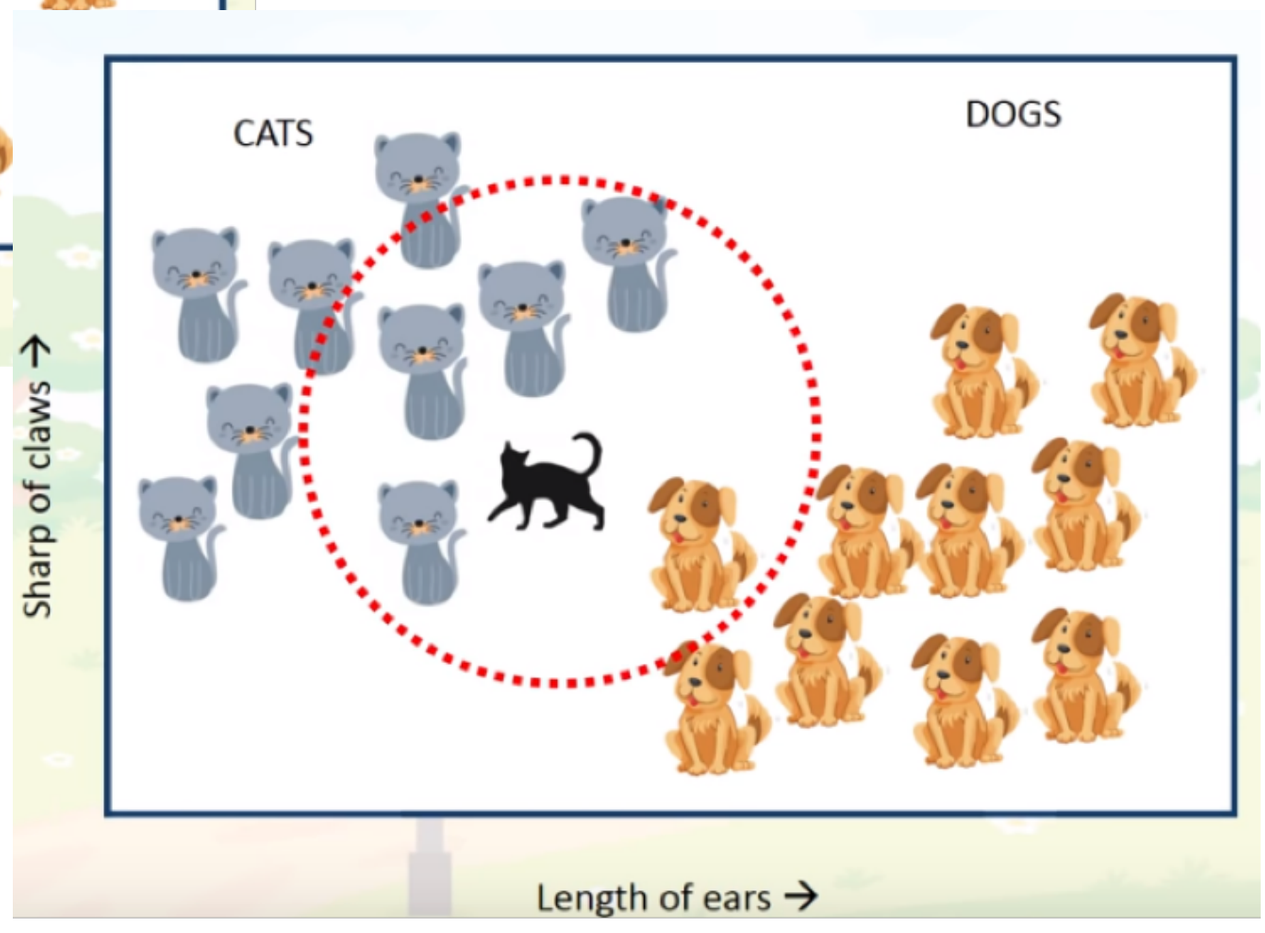
Why do we need KNN?

Why do we need KNN?

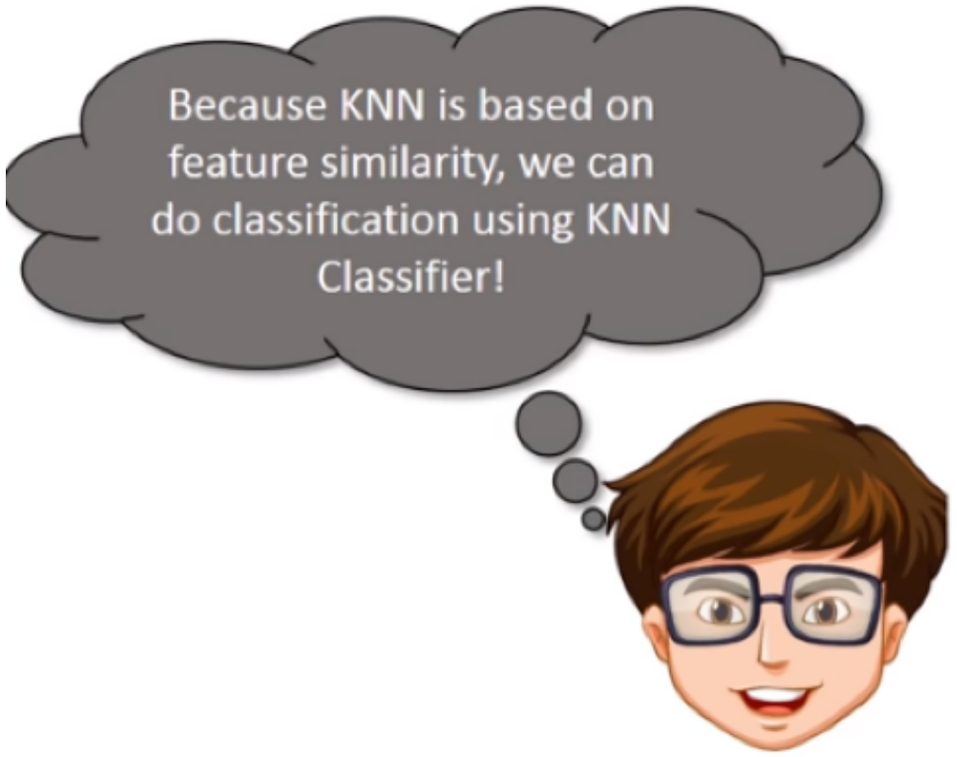


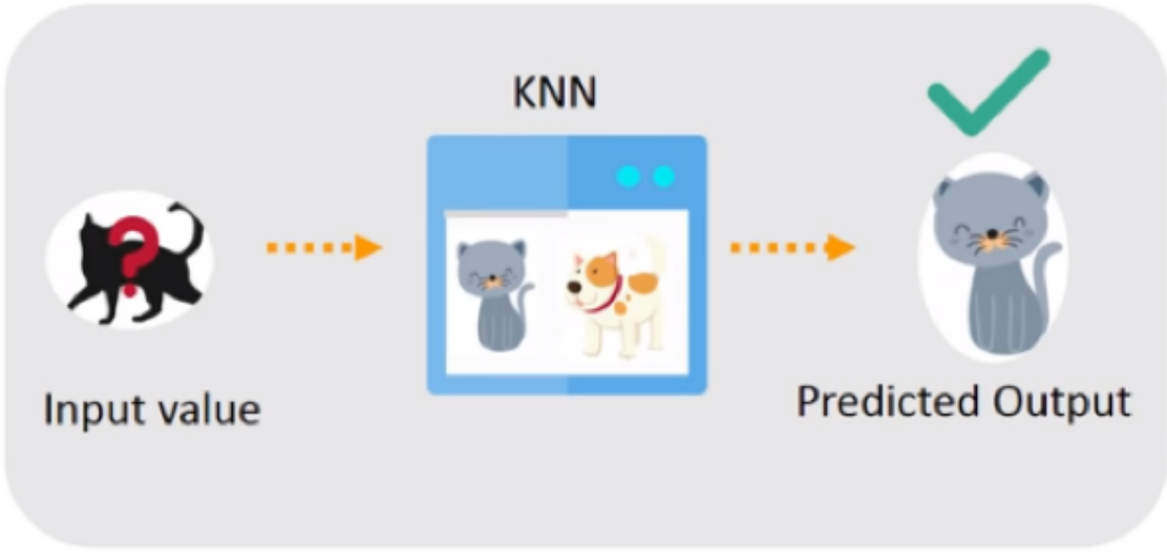
Why do we need KNN?





Why KNN?


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


KNN
Input value
Predicted Output

What is KNN algorithm?

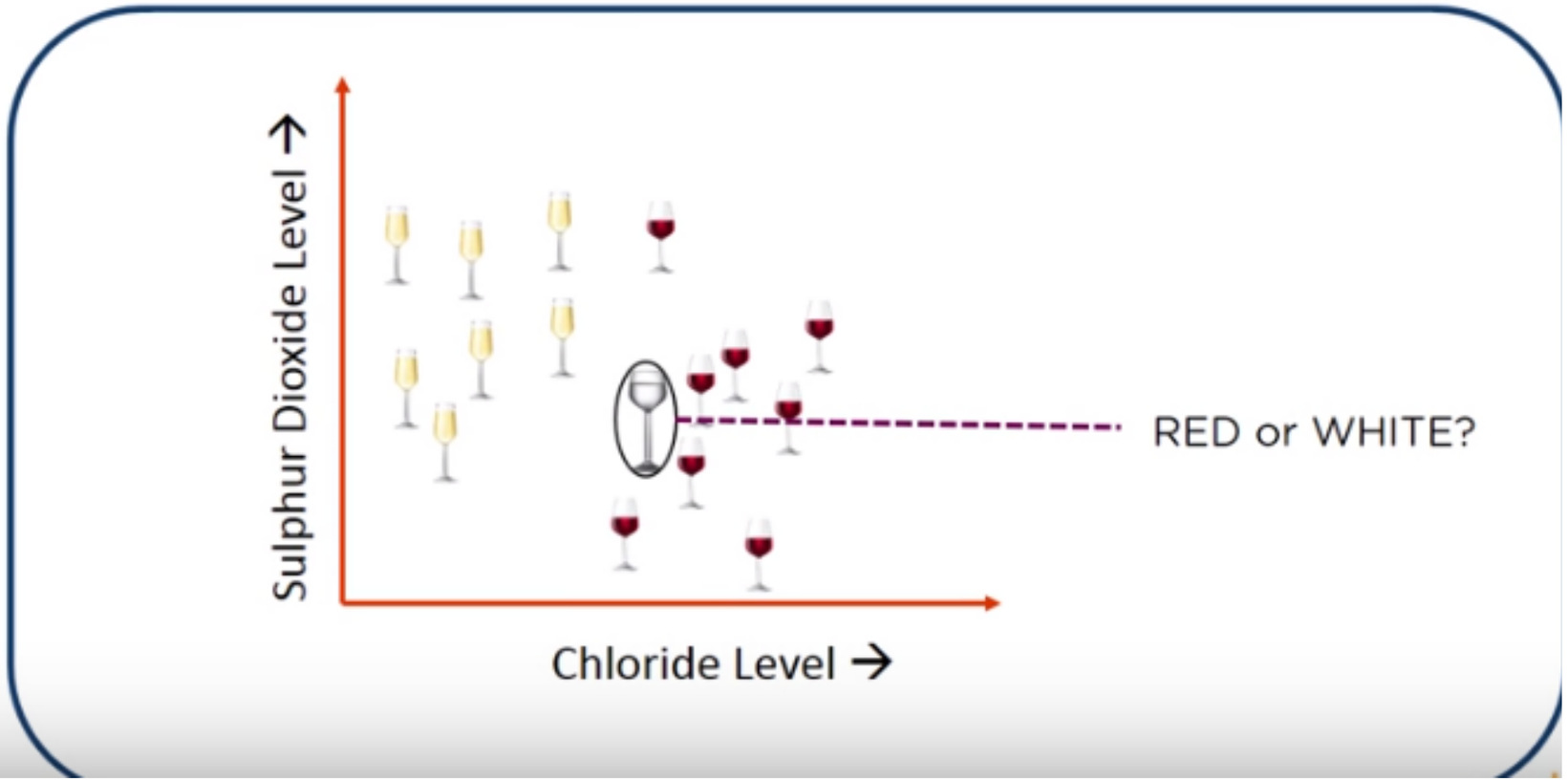
KNN – K Nearest Neighbors, is one of the simplest **Supervised** Machine Learning algorithm mostly used for

Classification

It classifies a data point based on how its neighbors are classified

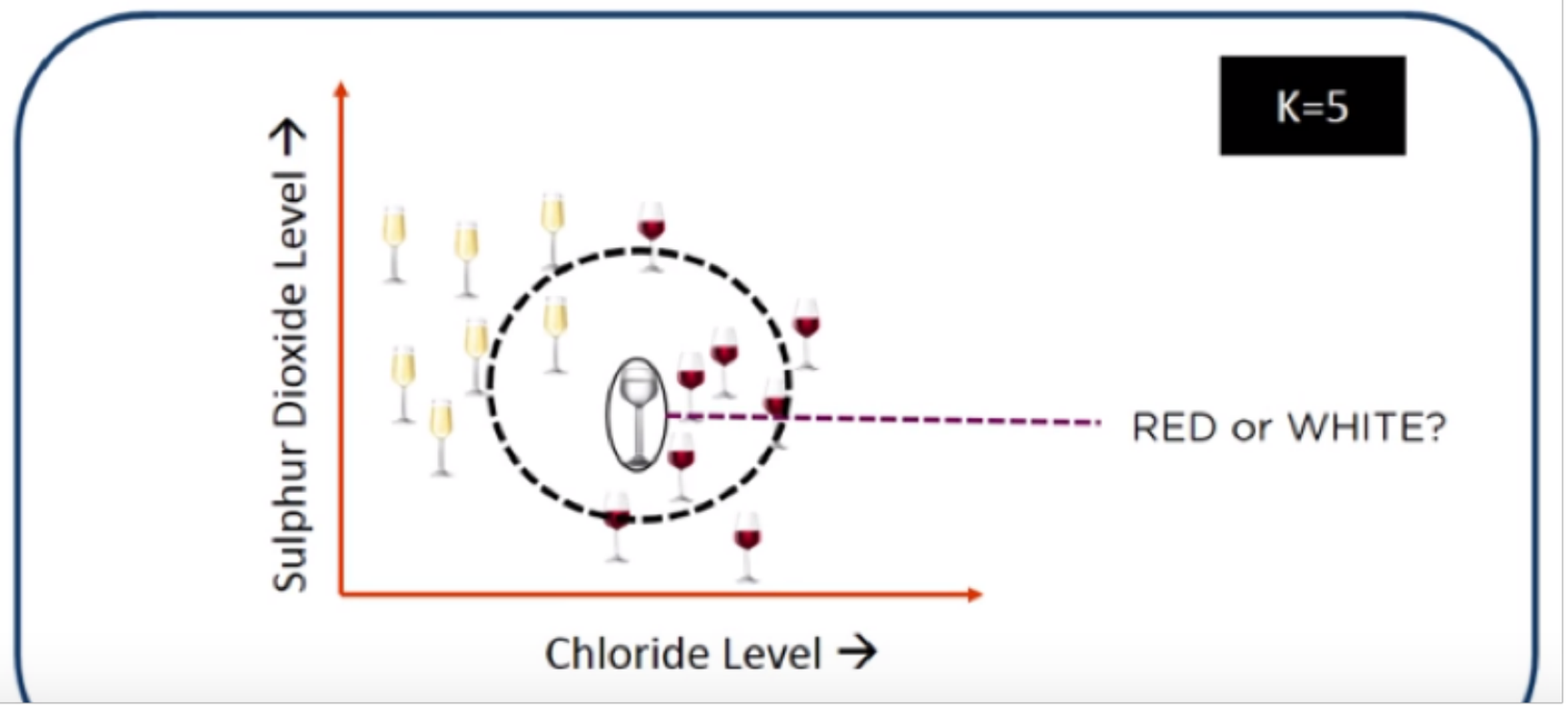
What is KNN algorithm?

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



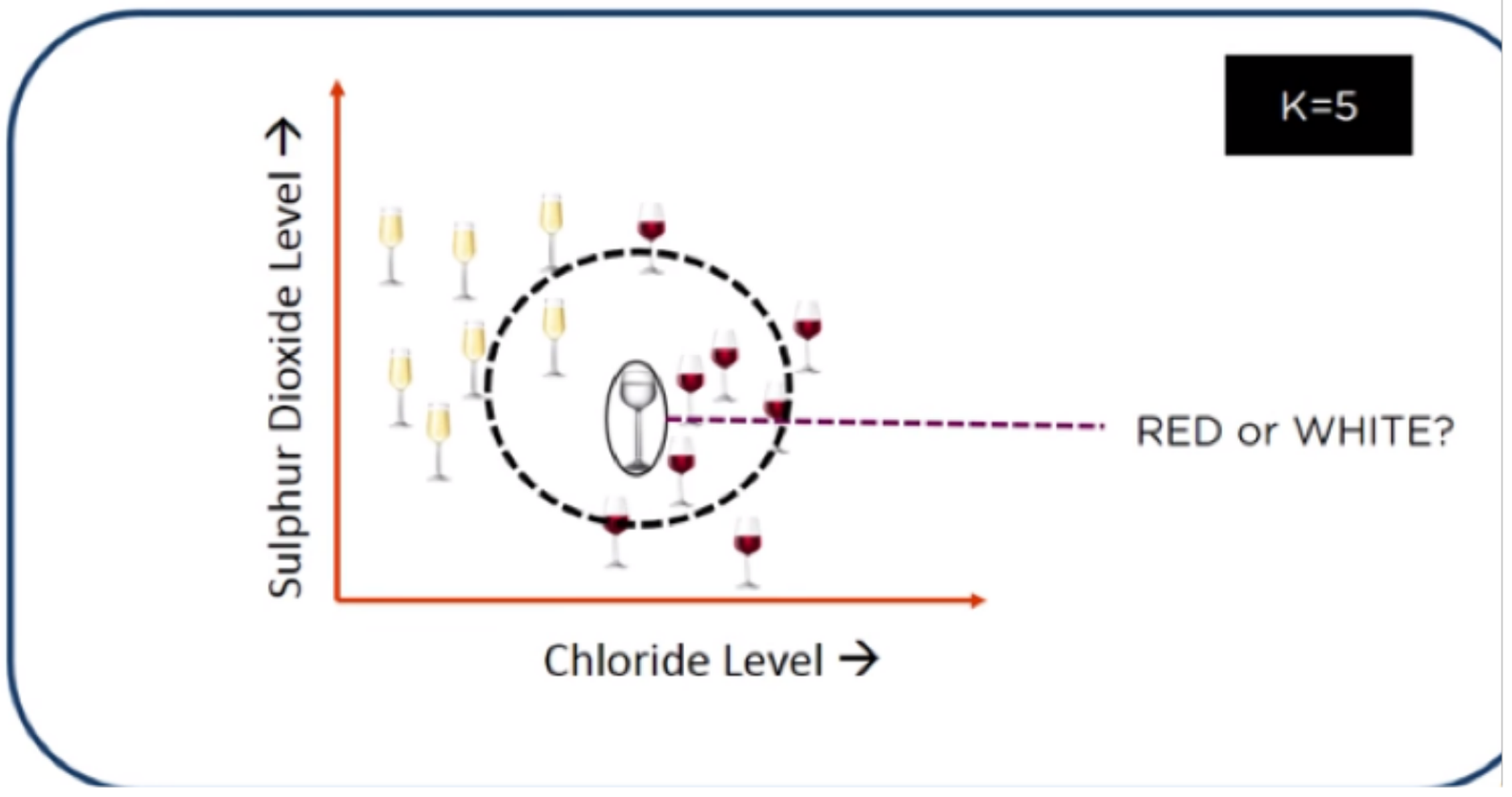
What is KNN algorithm?

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



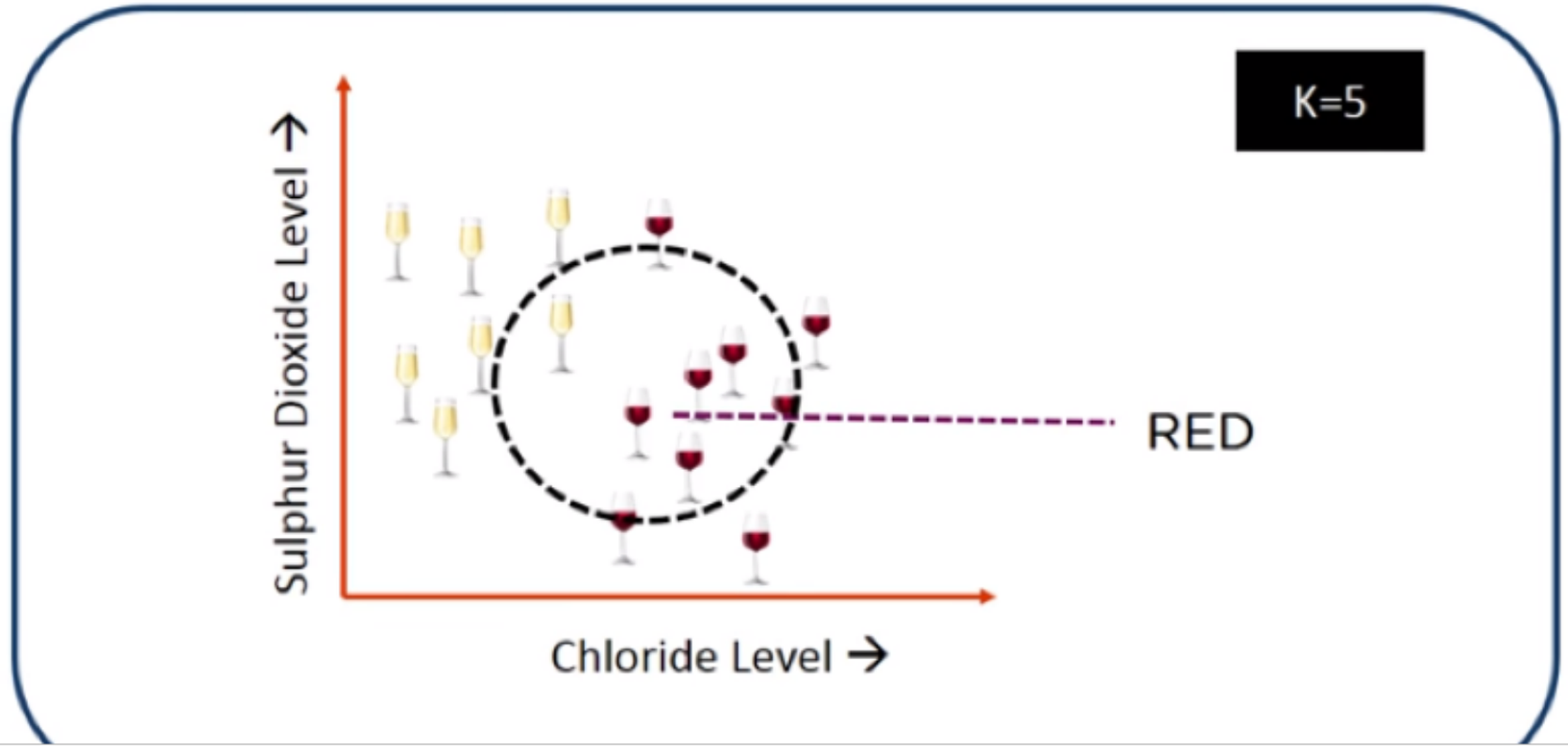
What is KNN algorithm?

A data point is classified by majority votes from its 5 nearest neighbors



What is KNN algorithm?

Here, the unknown point would be classified as red, since 4 out of 5 neighbors are red



KNN Algorithm

1. Load the data

2. Initialize K to your chosen number of neighbors

3. For each example in the data
   - Calculate the distance between the query example and the current example from the data.
   - Add the distance and the index of the example to an ordered collection

4. Sort the ordered collection of distances and indices from smallest to largest (in ascending order) by the distances

5. Pick the first K entries from the sorted collection

6. Get the labels of the selected K entries

7. If regression, return the mean of the K labels

8. If classification, return the mode of the K labels

How does KNN Works?

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

How does KNN Works?

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

To find the nearest neighbors, we will calculate Euclidean distance

According to the **Euclidean distance** formula, the **distance** between two points in the plane with coordinates (x, y) and (a, b) is given by:

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

How does KNN Works?

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

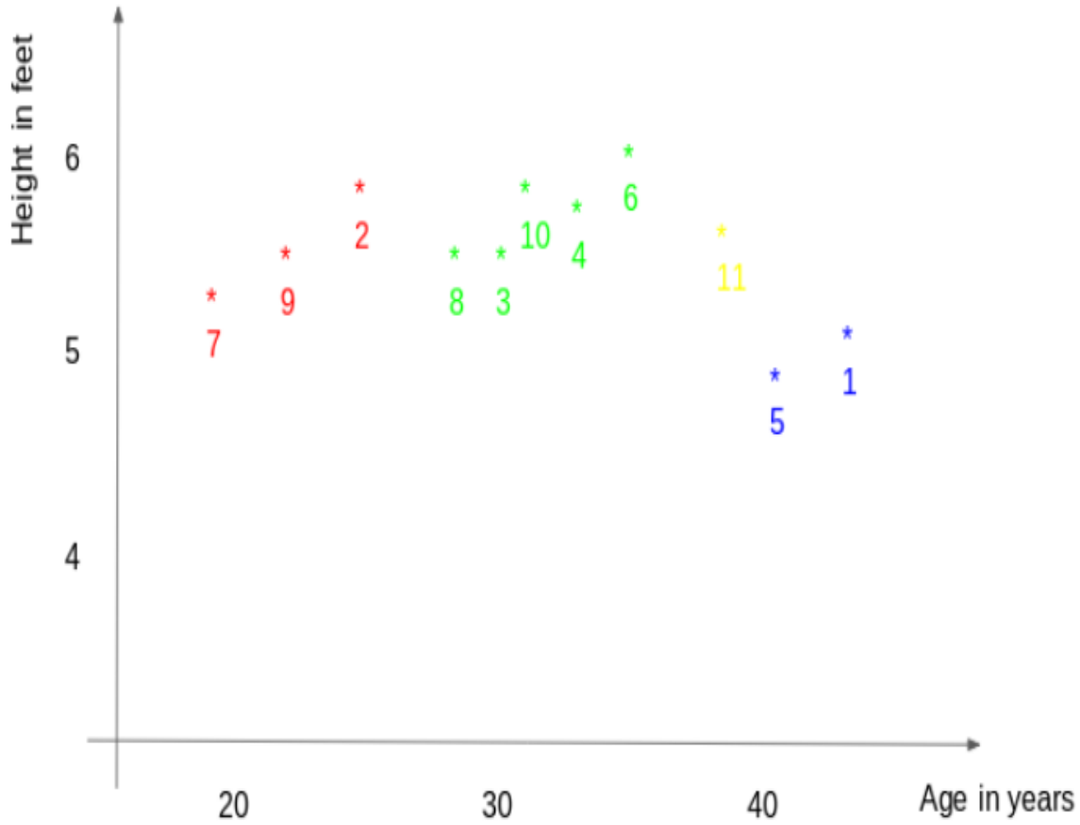
57 kg	170 cm	?

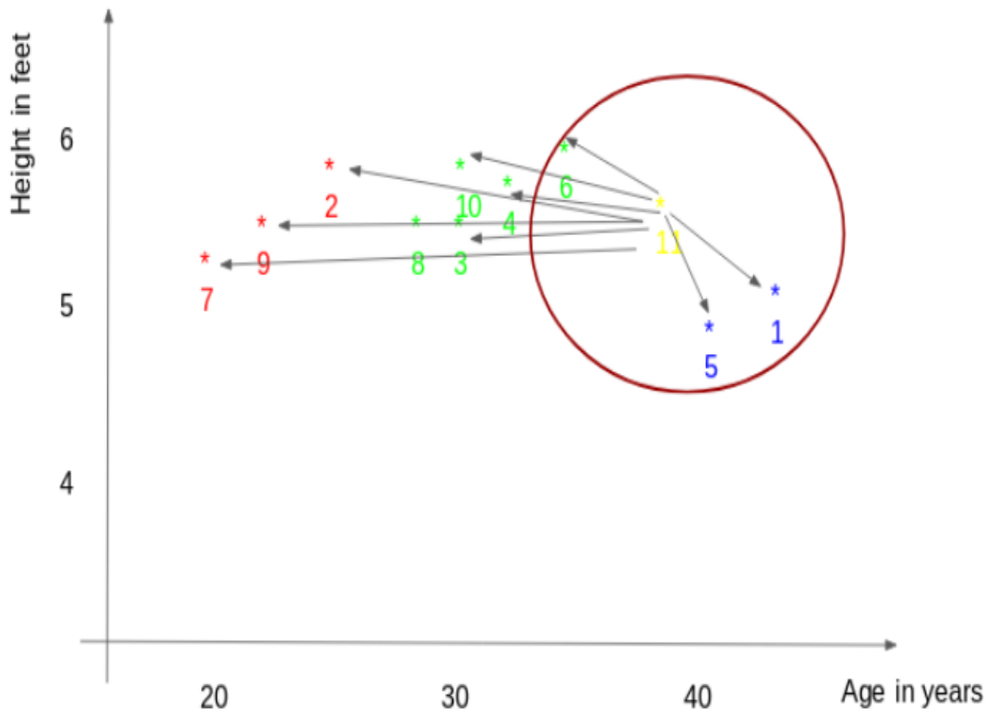
Example-2 Prediction

Consider the following table – it consists of the height, age and weight (target) value for 10 people. As you can see, the weight value of ID-11 is missing. Predict the weight of this person based on their height and age.

ID	Height	Age	Weight
1	5	45	77
2	5.11	26	47
3	5.6	30	55
4	5.9	34	59
5	4.8	40	72
6	5.8	36	60
7	5.3	19	40
8	5.8	28	60
9	5.5	23	45
10	5.6	32	58
11	5.5	38	?

Example2 - Prediction





Solution

ID	Height	Age	Weight
1	5	45	77
5	4.8	40	72
6	5.8	36	60

he prediction of weight for ID11 will be:

```
ID11 = (77+72+60)/3


ID11 = 69.66 kg
```

ID	Height	Age	Weight
1	5	45	77
4	5.9	34	59
5	4.8	40	72
6	5.8	36	60
10	5.6	32	58

The prediction for ID11 will be :

```
ID 11 =  (77+59+72+60+58)/5


ID 11 = 65.2 kg
```

Distance Measure - Minkowski

- Minkowski distance is a metric in Normed vector space

$$\left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{1/p}$$

- ***p = 1, Manhattan Distance***
- ***p = 2, Euclidean Distance***
- ***p = ∞, Chebychev Distance***

Distance Measure - Manhattan distance

- the distance between two points is the sum of the absolute differences.

- City block distance, L_1 distance

- The distance between p and q in n-dimensional space,

$$d_1(\mathbf{p}, \mathbf{q}) = \|\mathbf{p} - \mathbf{q}\|_1 = \sum_{i=1}^{n} |p_i - q_i|,$$

where $(\mathbf{p}, \mathbf{q})$ are vectors

$$\mathbf{p} = (p_1, p_2, \ldots, p_n) \text{ and } \mathbf{q} = (q_1, q_2, \ldots, q_n)$$

For example, in the plane, the taxicab distance between (p_1, p_2) and (q_1, q_2) is $|p_1 - q_1| + |p_2 - q_2|$.

Euclidean Vs Manhattan

- Euclidean is a good distance measure to use if the input variables are similar in type

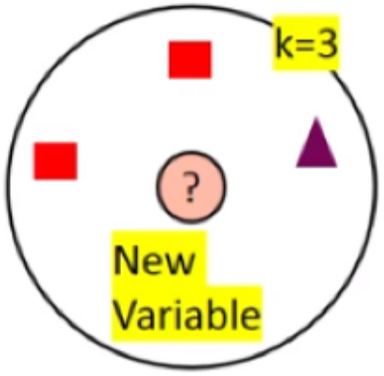
- All measured widths and heights

- Manhattan distance is a good measure to use if the input variables are not similar in type

- Example such as age, gender, height, etc.

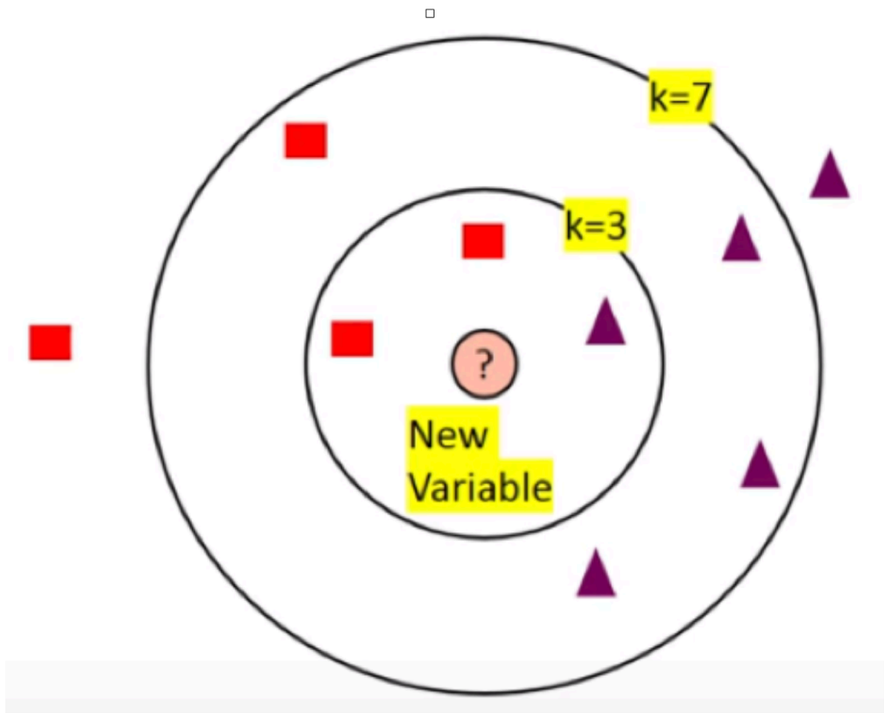
How do we choose ‘k’?

KNN Algorithms is based on feature similarly: Choosing the right value of k is a process called parameter tuning, and is important for better accuracy.



So at k=3, we can classify ‘?’ As “Red”

But at k=7, we can classify ‘?’ As “Purple”



How do we choose ‘k’?

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?

We can use KNN when

Data is labeled

Dog

Dataset is small

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

Data is noise free

Weight(x2)	Height(y2)	Class
51	167	Underweight
62	182	one-fourty
69	176	23
64	173	hello kitty
65	172	Normal

Noise

KNN variants in names

- **Instance-Based Learning**: The raw training instances are used to make predictions. As such KNN is often referred to as instance-based learning or a case-based learning (where each training instance is a case from the problem domain).

- **Instance-based learning** (sometimes called **memory-based learning**) is a family of learning algorithms that, instead of performing explicit generalization, compares new problem instances with instances seen in training, which have been stored in memory.

KNN variants in names

- **Lazy Learning**: No learning of the model is required and all of the work happens at the time a prediction is requested. As such, KNN is often referred to as a lazy learning algorithm.
- The generalization beyond the **training data is delayed until a query** is made to the system
- Example,
  - used by **online recommendation systems**, is that the data set is continuously updated with new entries.
  - Because of the continuous update, the "training data" would be rendered obsolete in a relatively short time especially in areas like books and movies, where new best-sellers or hit movies/music are published/ released continuously.
  - Therefore, one cannot really talk of a "training phase".

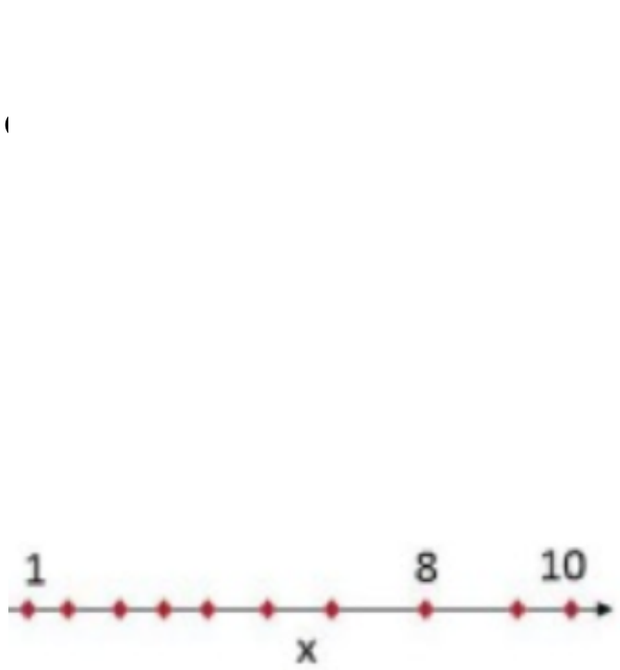
KNN variants in names

- **Non-Parametric**: KNN makes no assumptions about the functional form of the problem being solved. As such KNN is referred to as a non-parametric machine learning algorithm.
- Not based solely on parametrized families of probability distributions such as the mean and variance
- Example
  - A histogram is a simple nonparametric estimate of a probability distribution.
  - KNNs classify the unseen instance based on the K points in the training set which are nearest to it.

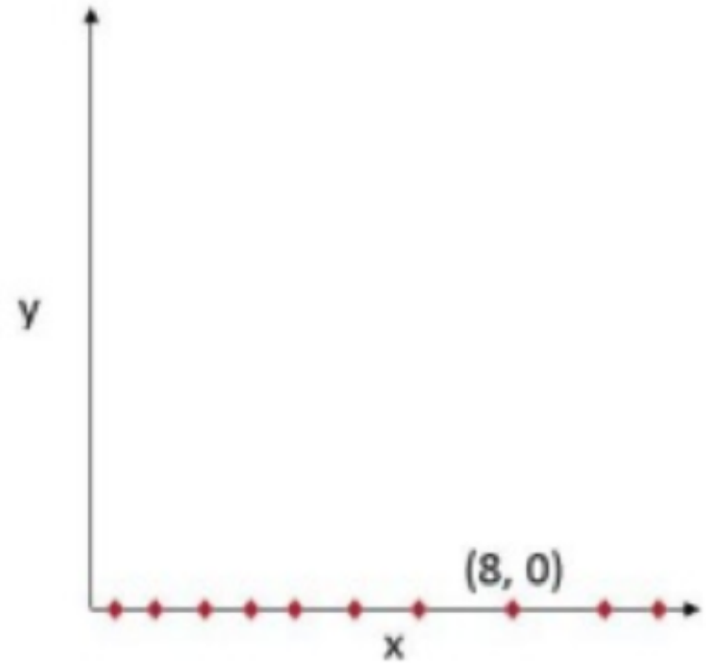
KNN a lazy learner?

- Eager learner
  - Generalized model from training data set is constructed
  - Using the model the class of test data set is predicted
  - Example – decision tree
- Lazy learner
  - Training dataset is stored
  - On querying similarity between test data and training set records is calculated to predict the class of test data
  - Example – k-Nearest Neighbour

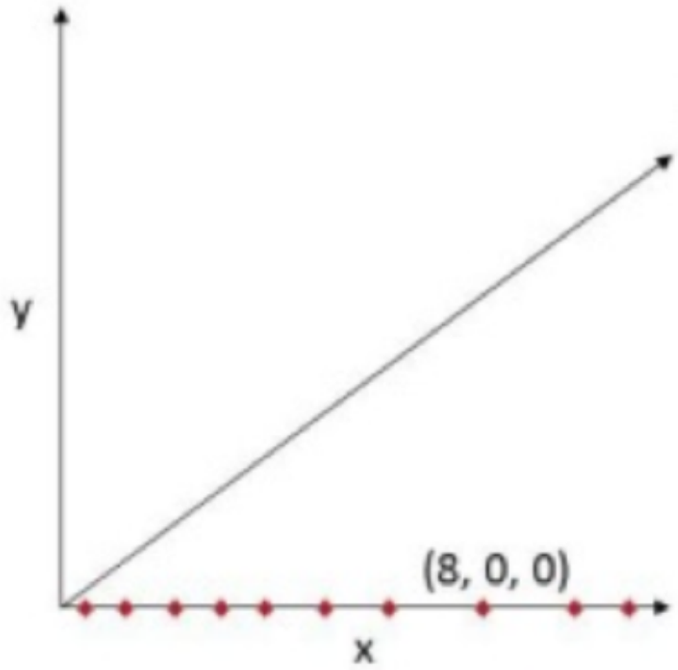
Curse of Dimensionality



Dimension Space = 10 Units

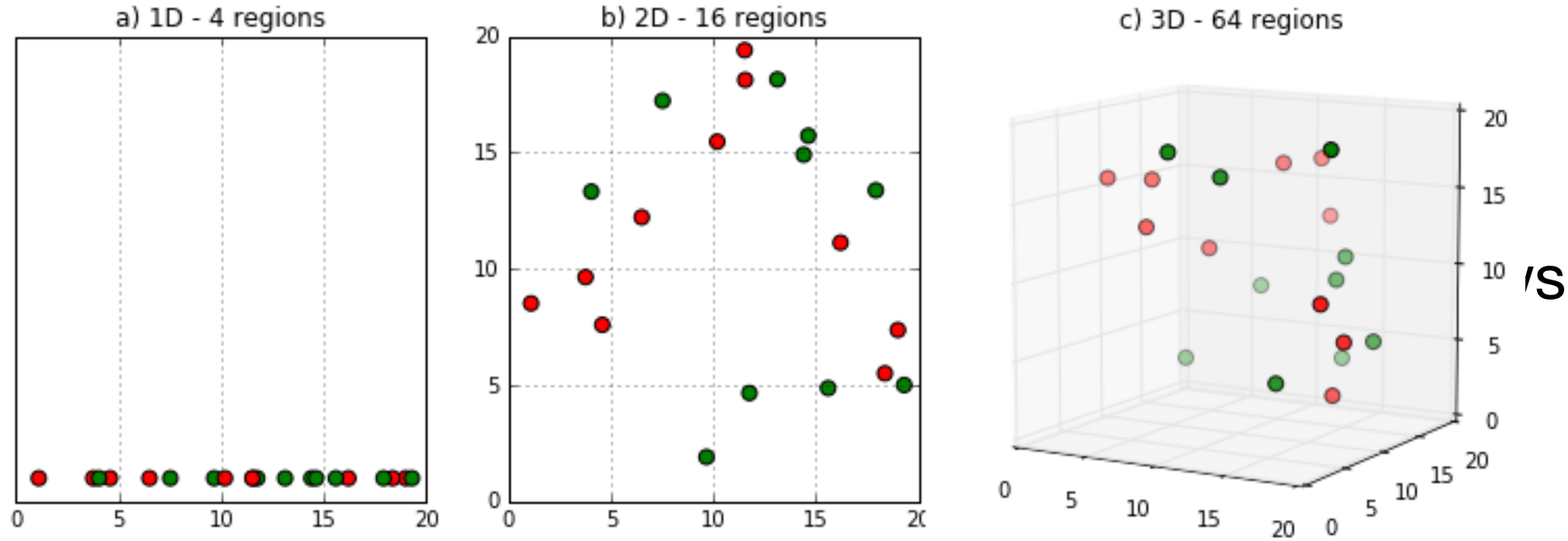


Dimension Space = 100 Units



Dimension Space = 1000 Units

Curse of Dimensionality

- 

- ...s

This exponential growth in data causes high sparsity in the data set and unnecessarily increases storage space and processing time for the particular modelling algorithm. Think of image recognition problem of high resolution images 1280 × 720 = 921,600 pixels i.e. 921600 dimensions

Curse of Dimensionality

Value added by additional dimension is much smaller compared to overhead it adds to the algorithm.

The data that can be represented using 10 space units of one **true dimension**, needs 1000 space units after adding 2 more dimensions just because we **observed these dimensions** during the experiment.

The **true dimension means** the dimension which accurately generalize the data and **observed dimensions means** whatever other dimensions we consider in dataset which may or may not contribute to accurately generalize the data.

Best Prepare Data for KNN

- **Rescale Data**: KNN performs much better if all of the data has the same scale. Normalizing your data to the range [0, 1] is a good idea. It may also be a good idea to standardize your data if it has a Gaussian distribution.
- **Address Missing Data**: Missing data will mean that the distance between samples can not be calculated. These samples could be excluded or the missing values could be imputed.
- **Lower Dimensionality**: KNN is suited for lower dimensional data.
  - You can try it on high dimensional data (hundreds or thousands of input variables) but be aware that it may not perform as well as other techniques.
  - KNN can benefit from feature selection that reduces the dimensionality of the input feature space.

Review Questions

- **k-NN algorithm does more computation on test time rather than train time ??**
  - TRUE or FALSE
- **Which of the following statement is true about k-NN algorithm?**
  - k-NN performs much better if all of the data have the same scale
  - k-NN works well with a small number of input variables (p), but struggles when the number of inputs is very large
  - k-NN makes no assumptions about the functional form of the problem being solved

Review Questions

x	y	Class
-1	1	$-$
0	1	$+$
0	2	$-$
1	-1	$-$
1	0	$+$
1	2	$+$
2	2	$-$
2	3	$+$

- Suppose, you have given the following data where x and y are the 2 input variables and Class is the dependent variable.

Suppose, you want to predict the class of new data point x=1 and y=1 using eucludian distance in 3-NN. In which class this data point belong to?

In the previous question, you are now want use 7-NN instead of 3-KNN which of the following x=1 and y=1 will belong to?