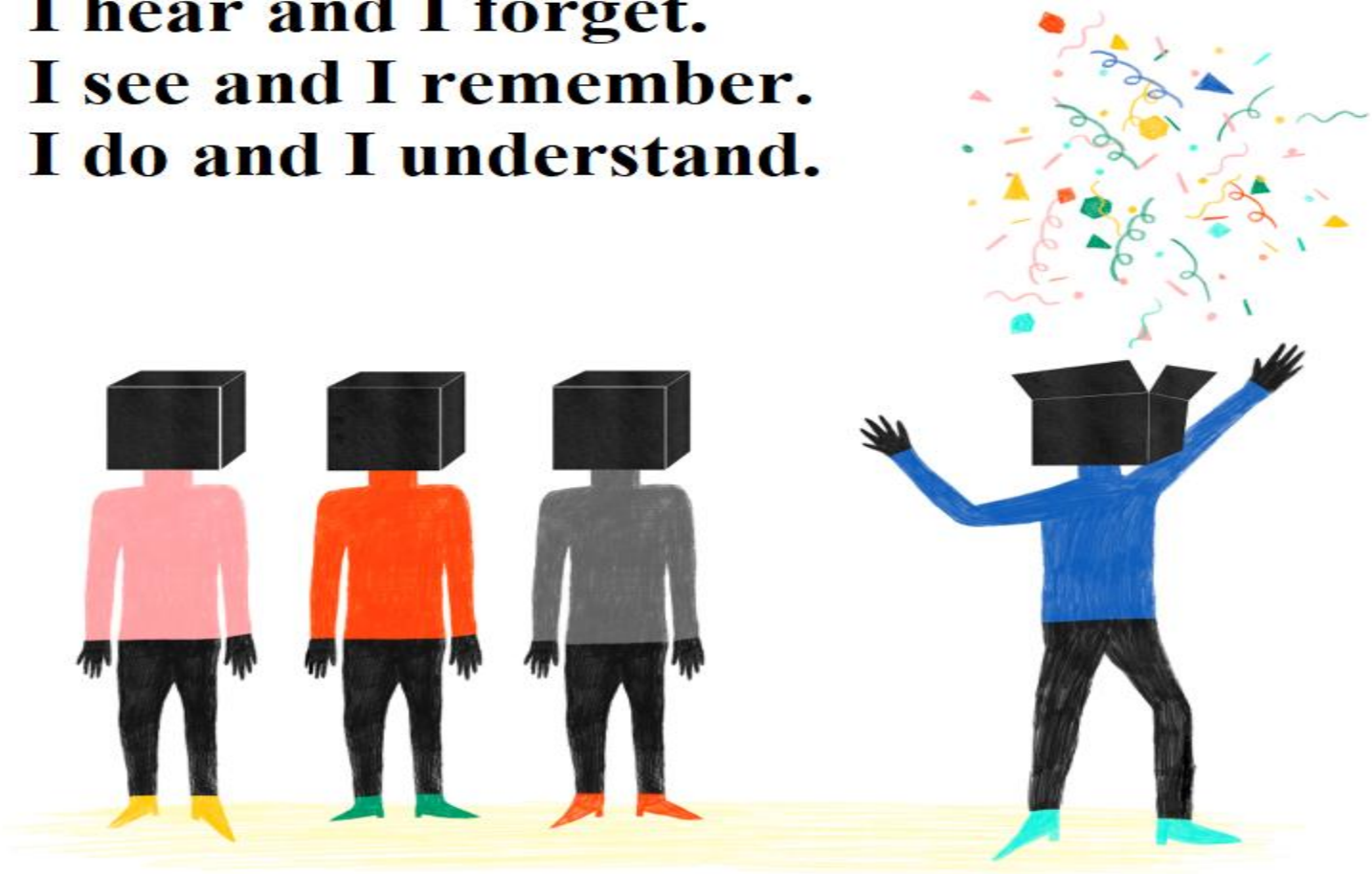
I hear and I forget.
I see and I remember.
I do and I understand.

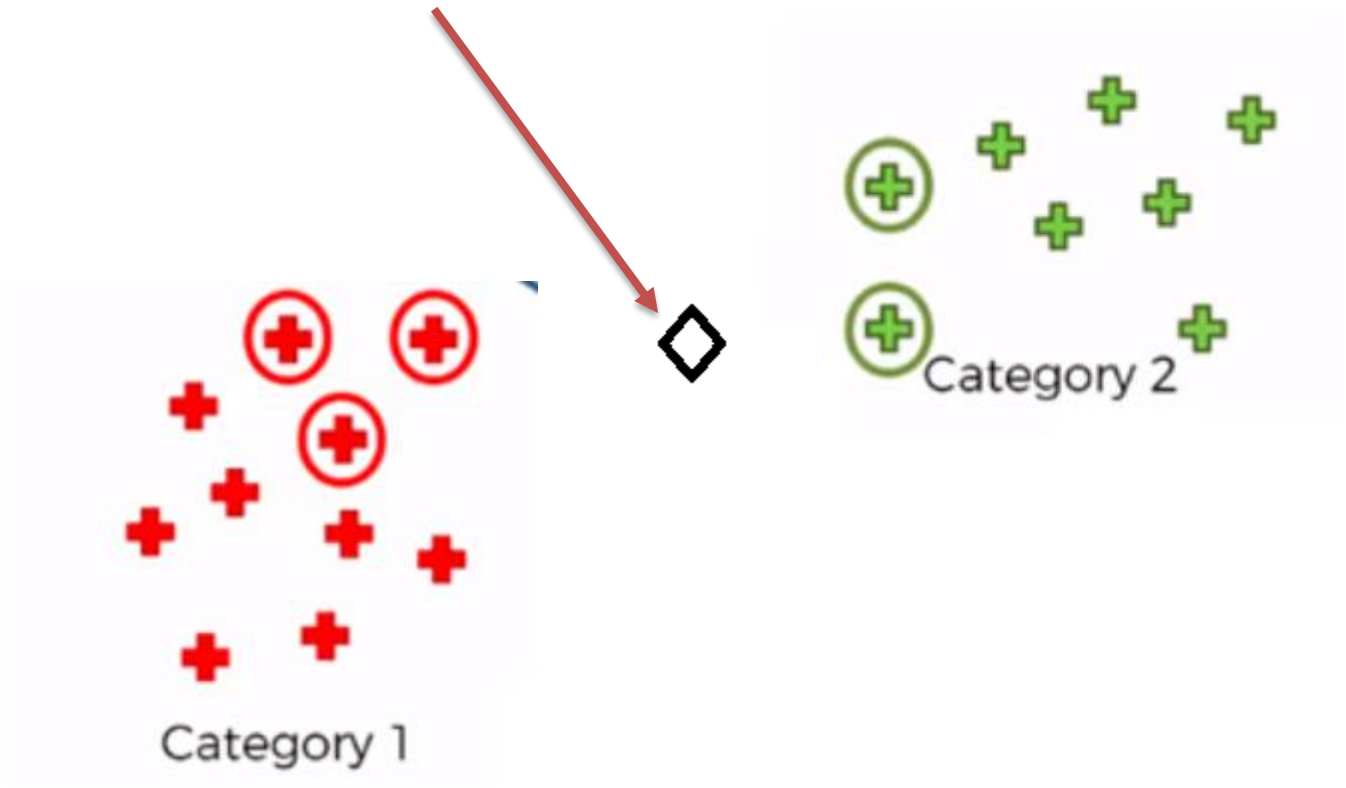


Chandan Verma

Corporate Trainer(Machine Learning,AI,Cloud Computing,IOT)

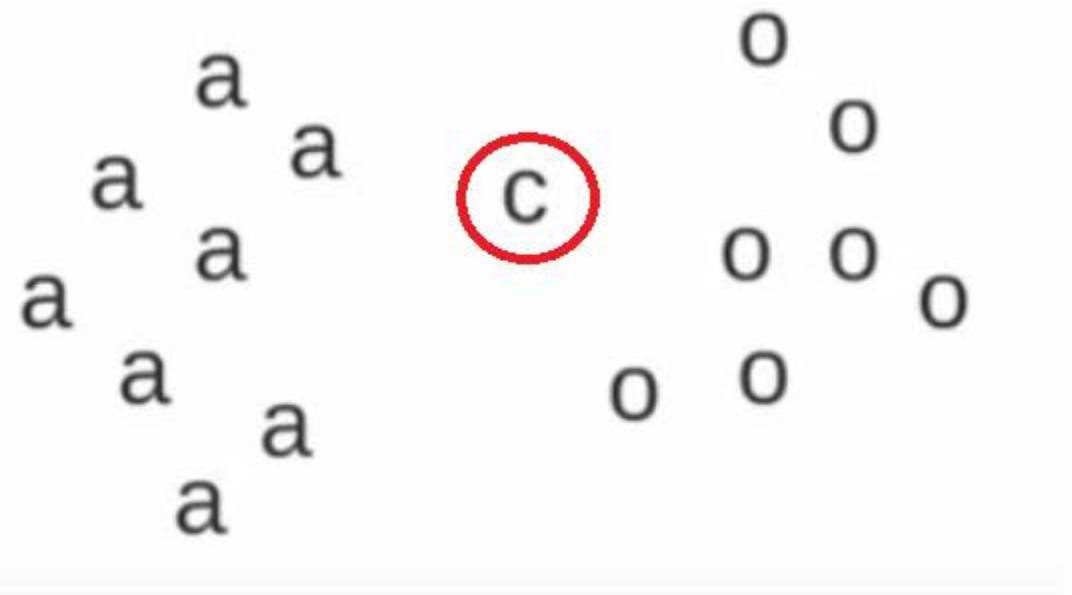
www.facebook.com/verma.chandan.070

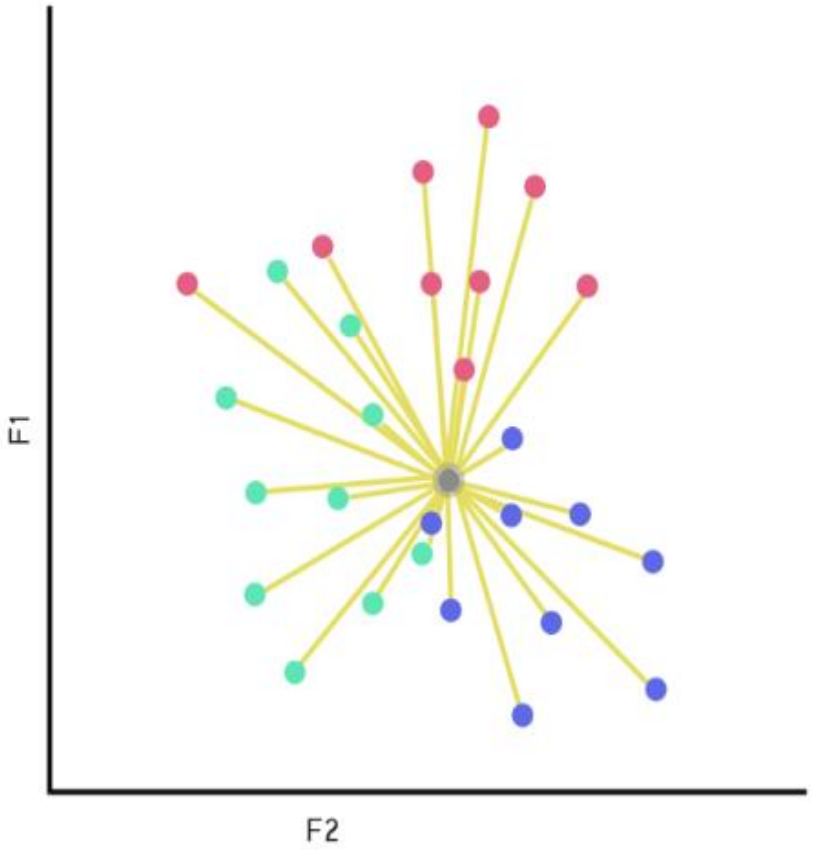
K-nearest neighbors

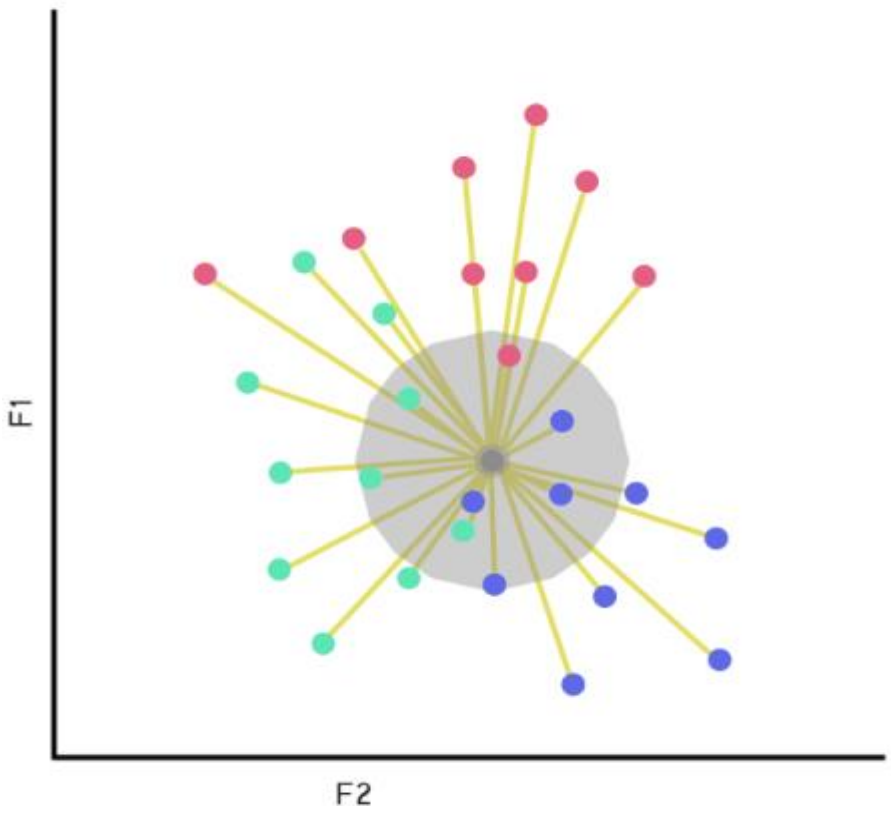


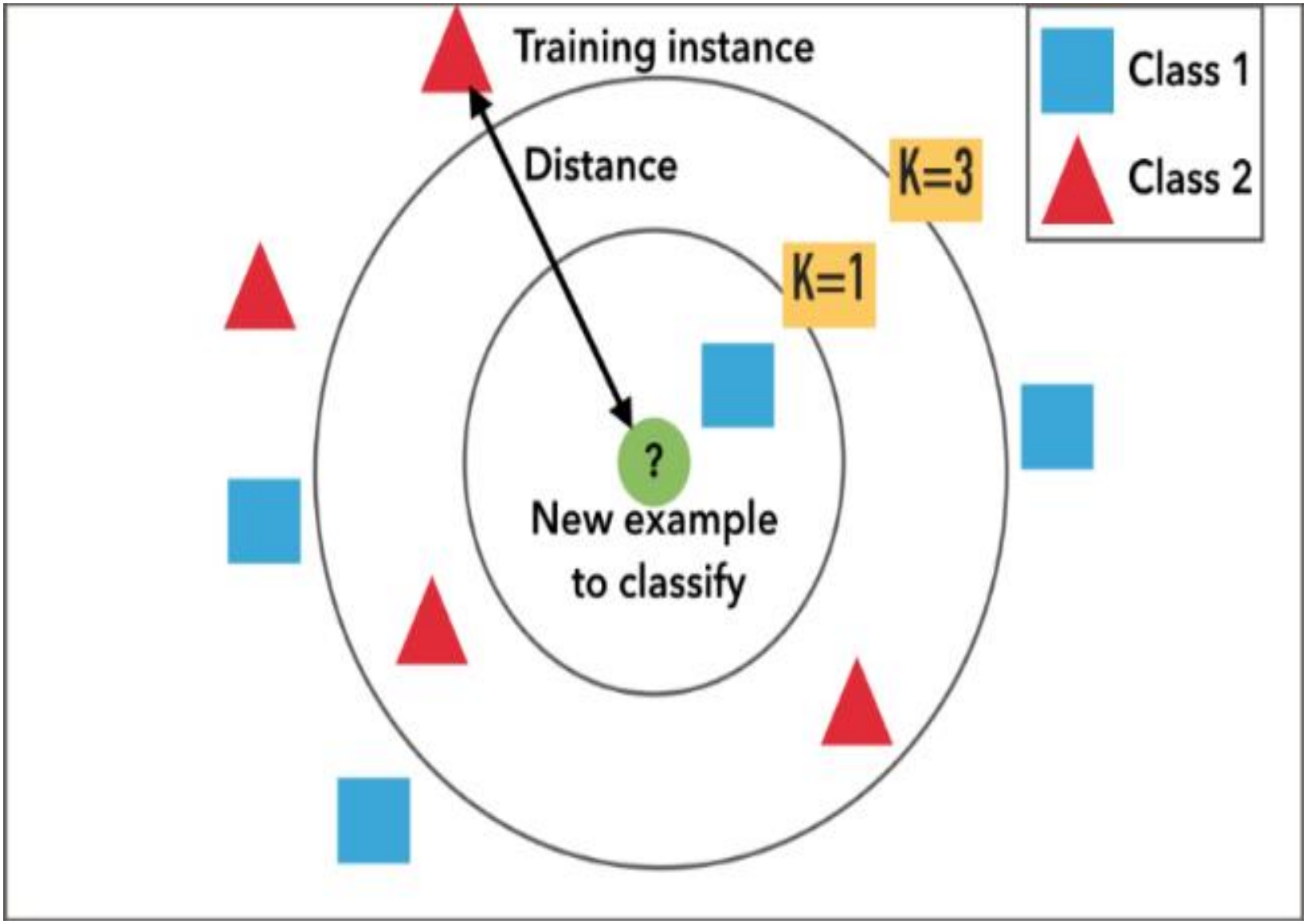
“tell me who your neighbors are, and I’ll tell you who you are

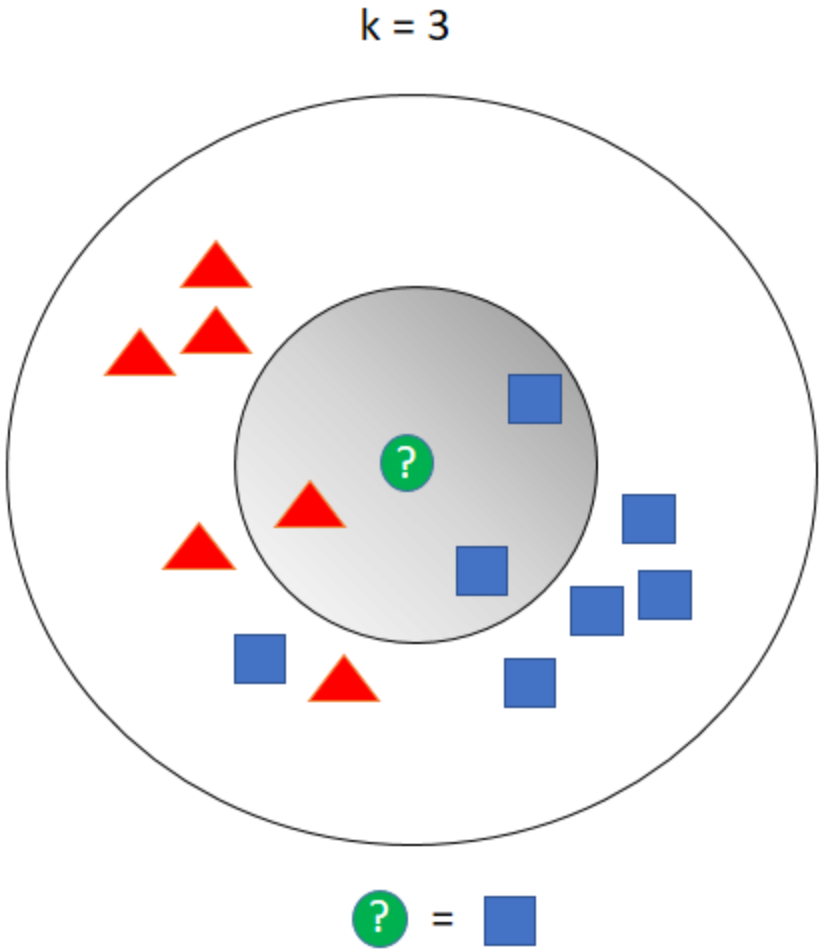
If it walks like a duck, quacks like a duck, and looks like a duck, then it's probably a duck




F1
F2

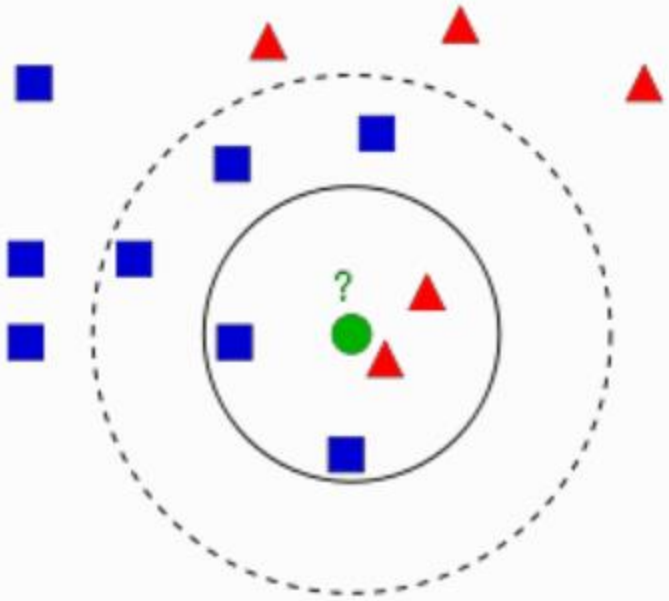

F1
F2

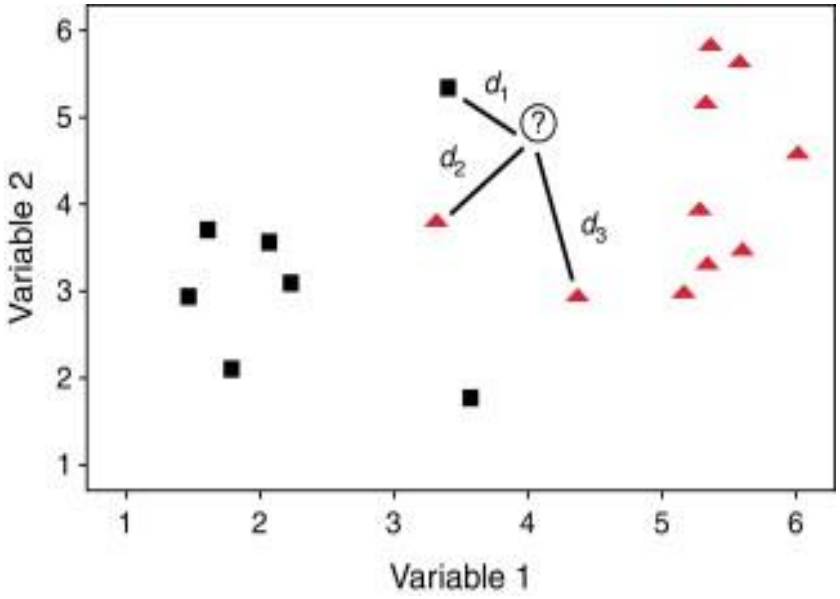

Training instance
Distance
K=3
K=1
?
New example
to classify
Class 1
Class 2


k = 3
?
? =

k-Nearest Neighbour (Knn)

kNN is one of the simplest of classification algorithms available for supervised learning.





Distance functions

Euclidean

$$\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$$

Manhattan

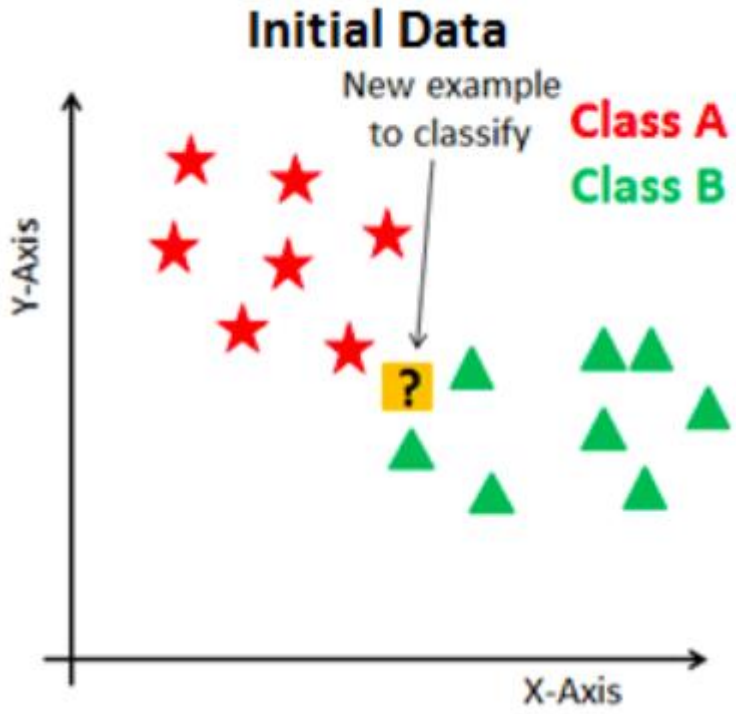
$$\sum_{i=1}^{k}|x_i - y_i|$$

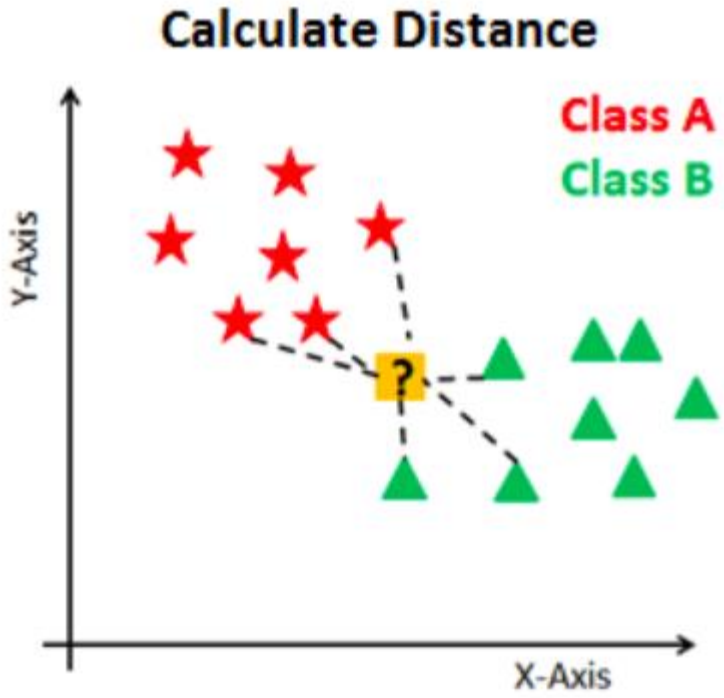
Minkowski

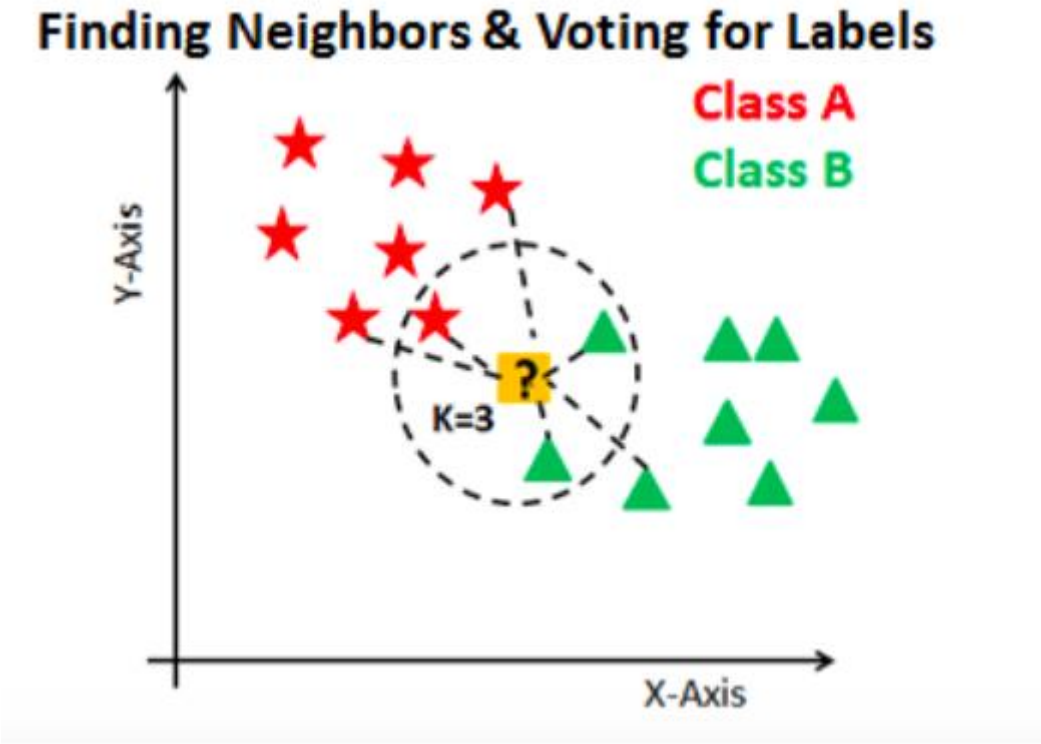
$$\left(\sum_{i=1}^{k}(|x_i - y_i|)^q\right)^{1/q}$$

Steps for KNN

1. Calculate distance (e.g. Euclidean distance, Hamming distance, etc.)
2. Find k closest neighbors
3. Vote for labels or calculate the mea


Initial Data
New example
to classify
Class A
Class B
?
Y-Axis
X-Axis


Calculate Distance
Class A
Class B
?
Y-Axis
X-Axis


Finding Neighbors & Voting for Labels
Class A
Class B
Y-Axis
?
K=3
X-Axis

KNN

K-nearest neighbors is one of the simplest supervised machine learning algorithms. It is also known as an **instance based learning** algorithm or feature similarity algorithm.

It is mainly based on feature similarity. KNN checks how similar a data point is to its neighbor and classifies the data point into the class it is most similar to.

K in kNN is a parameter that refers to number of nearest neighbors. For example k is 5 then a new data point is classified by majority of data points from 5 nearest neighbors. The kNN algorithm can be used in both classification and regression but it is most widely used in classification problem.

KNN is a non-parametric model which means that it does not make any assumptions about the data set. This makes the algorithm more effective since it can handle realistic data.

•KNN is a lazy algorithm, this means that it memorizes the training data set instead of learning a discriminative function from the training data.

- **Non-Parametric**
- **Based on Feature Similarity**
- **Lazy Algorithm**

instance-based learning

instance-based learning where we instead simply store the training data and use it to make new prediction

Store the input data

$$\begin{bmatrix} x_{1,0} & x_{1,1} & \dots & x_{1,n} \\ x_{2,0} & x_{2,1} & \dots & x_{2,n} \\ \vdots & \vdots & \vdots & \vdots \\ x_{m,0} & x_{m,1} & \cdots & x_{m,n} \end{bmatrix} \text{ and } \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_m \end{bmatrix}$$

Store the input data

$$\begin{bmatrix} x_{1,0} & x_{1,1} & \dots & x_{1,n} \\ x_{2,0} & x_{2,1} & \dots & x_{2,n} \\ \vdots & \vdots & \vdots & \vdots \\ x_{m,0} & x_{m,1} & \cdots & x_{m,n} \end{bmatrix} \text{ and } \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_m \end{bmatrix}$$

When asked to predict a new value (a query)

$$y_i = ?$$

Search for similar data points previously stored

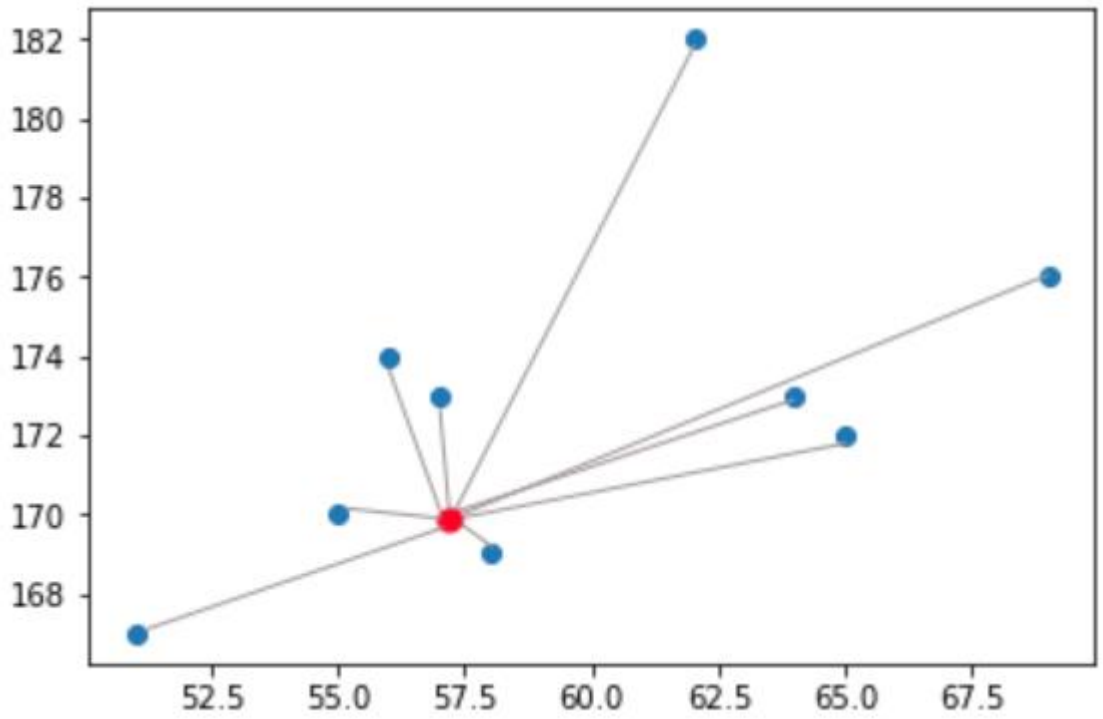
$$\begin{bmatrix} x_{4,1} & x_{4,2} & \cdots & x_{4,n} \\ x_{9,1} & x_{9,1} & \cdots & x_{9,n} \\ x_{15,1} & x_{15,1} & \cdots & x_{15,n} \end{bmatrix} \text{ and } \begin{bmatrix} y_4 \\ y_9 \\ y_{15} \end{bmatrix}$$

And use them to generate your prediction

$$y_i = \frac{y_4 + y_9 + y_{15}}{3}$$

Example-1

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



$$dist(d) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \ldots + (x_i - y_i)^2 + \ldots + (x_n - y_n)^2}$$

euclidean_distances

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

170 57

```
import numpy as np
X=np.array([[167,51],[182,62],[176,69],[173,64],[172,65]
,[174,56],[169,58],[173,57],[170,55]])
from sklearn.metrics import euclidean_distances
xx=np.array([[170,57]])
distance=euclidean_distances(X,xx)
```

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

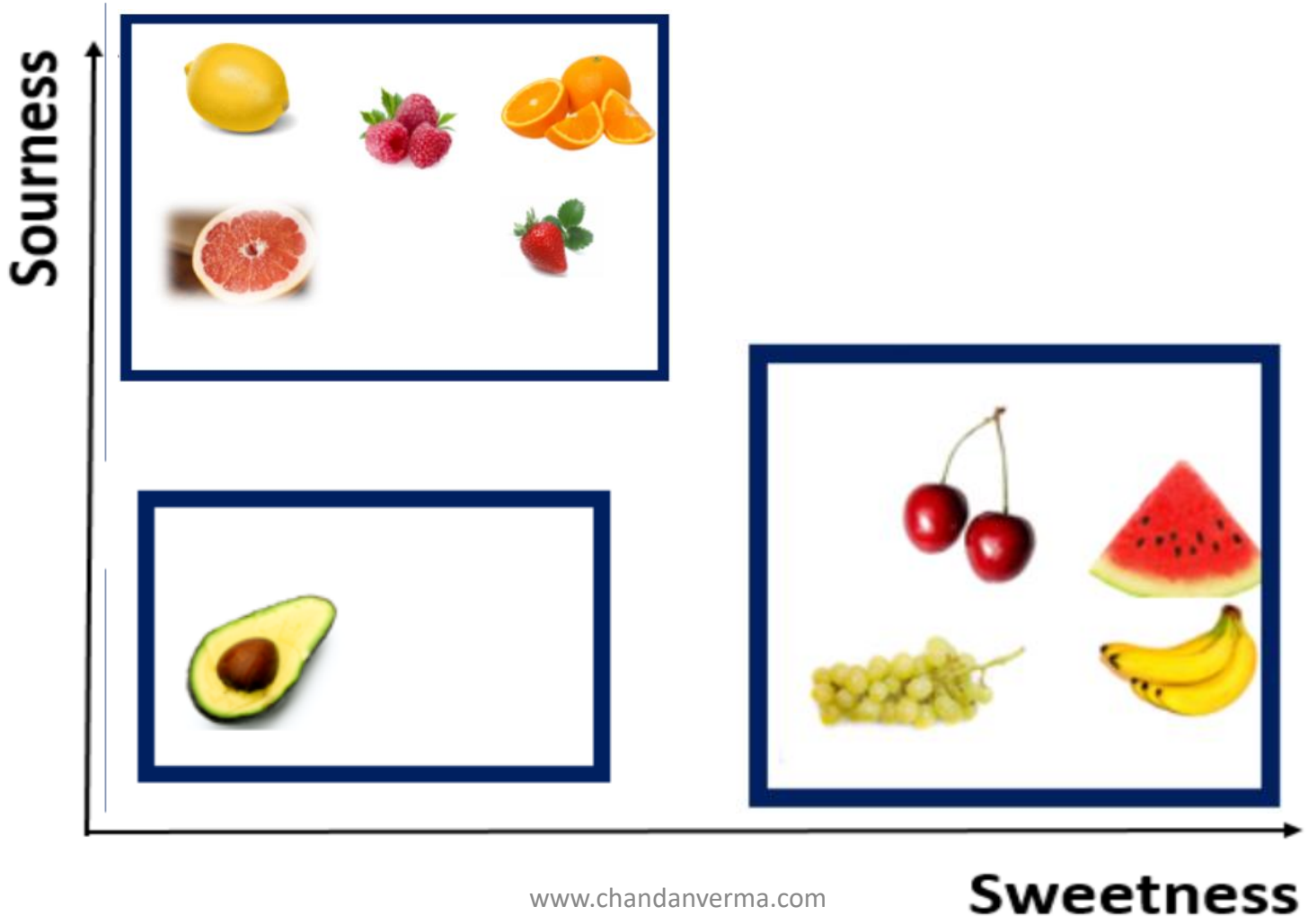
170 57

k=3 then 3 points closest to the data point is

Height (CM)	Weight (KG)	Class	Euclidean Distance
169	58	Normal	1.4
173	57	Normal	3
170	55	Normal	2

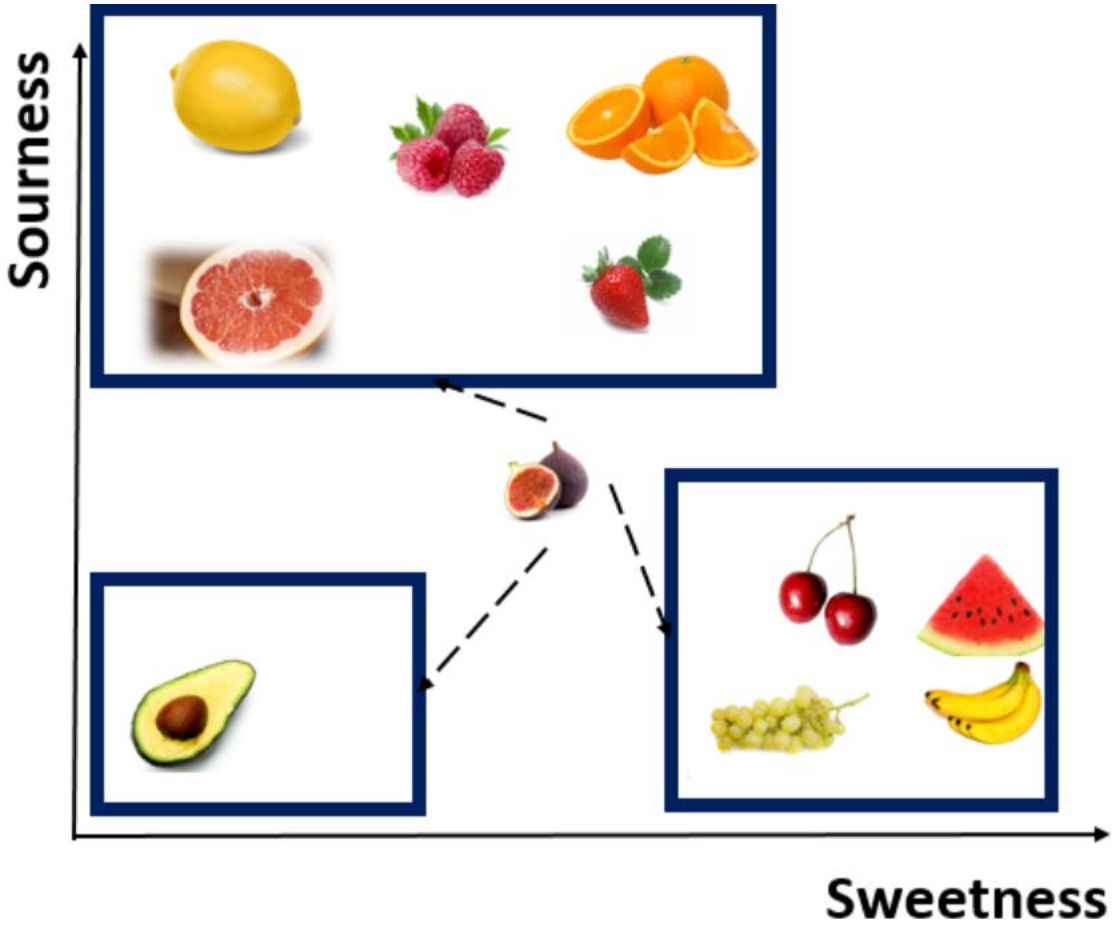
According to above table, we can say that the closest neighbors belong to the class Normal, so our new point HEIGHT = 170 CM and WEIGHT = 57 KG, falls in the category: Normal.

Prediction via KNN

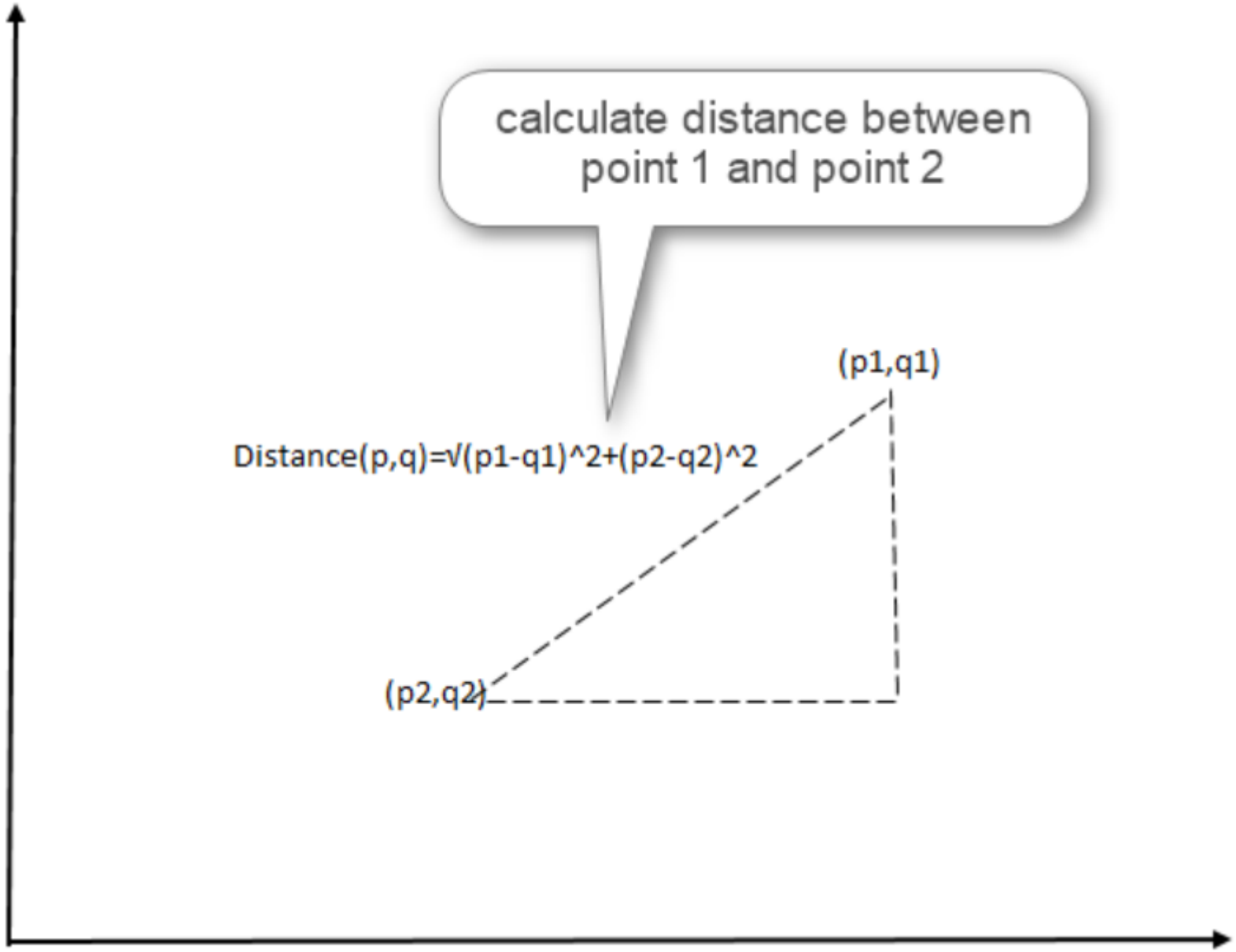


Fruit	Sweetness	sourness	Fruit type
Lemon	**1**	9	Sour
Grapfruite	**2**	8	Sour
Orange	**3**	7	Sour
Rasberry	**2**	8	Sour
Cherry	**6**	4	Sweet
Banana	**9**	1	Sweet
Grapes	**8**	2	Sweet
Watemelon	**9**	1	Sweet
Avacado	**1**	1	None
Strawberry	**5**	5	Sour

Fruit	Sweetness	sourness	Fruit type
Fig	7	3	



Sourness


calculate distance between point 1 and point 2
(p1,q1)
Distance(p,q)=√(p1-q1)^2+(p2-q2)^2
(p2,q2)

Fruite	Sweetnes	Sourness	Fruite Type	Distance to Fig
Lemon	1	9	Sour	sqrt((1-7)^2+(9-3)^2)=8.4
Grapfruite	2	8	Sour	sqrt((2-7)^2+(8-3)^2)=7.1
Orange	3	7	Sour	sqrt((3-7)^2+(7-3)^2)=5.6
Rasberry	2	8	Sour	sqrt((2-7)^2+(8-3)^2)=7.1
Cherry	6	4	Sweet	sqrt((6-7)^2+(4-3)^2)=1.41
banana	9	1	Sweet	sqrt((9-7)^2+(1-3)^2)=2.82
Grapes	8	2	Sweet	sqrt((8-7)^2+(2-3)^2)=1.41
Watremelon	9	1	Sweet	sqrt((9-7)^2+(1-3)^2)=2.44
Avacado	1	1	None	sqrt((1-7)^2+(1-3)^2)=6.3
Strawberry	5	5	Sour	sqrt((5-7)^2+(5-3)^2)=2.82

3 nearest neighbors based on distance as first neighbor(Cherry and Grapes), second neighbor(Watermelon) and third is (Banana and Strawberry).

Because all of these are Sweet fruits, we consider Figs as a sweet one.

k-Nearest Neighbour (Knn)

K-Nearest neighbor is a supervised learning algorithm where the result of new instance query is classified based on majority of K-NN Category. The purpose of this algorithms is to classify a new object based on attributes and training samples.

KNN makes predictions using the training dataset directly

The k-NN algorithm is among the simplest of all machine learning algorithms. It also might surprise many to know that k-NN is one of the top 10 data mining algorithms.

k number decides how many neighbors (where neighbors is defined based on the distance metric) influence the classification. With a given *k* value we can make boundaries of each class

The k-Nearest-Neighbor Classifier (k-NN) works directly on the learned samples, instead of creating rules compared to other classification methods.

KNN-Example

	region	tenure	age	marital	address	income	ed	employ	retire	gender	reside	custcat
0	2	13	44	1	9	64.0	4	5	0.0	0	2	1
1	3	11	33	1	7	136.0	5	5	0.0	0	6	4
2	3	68	52	1	24	116.0	1	29	0.0	1	2	3
3	2	33	33	0	12	33.0	2	0	0.0	1	1	1
4	2	23	30	1	9	30.0	1	2	0.0	0	4	3

Imagine a telecommunications provider has segmented its customer base by service usage patterns, categorizing the customers into four groups. If demographic data can be used to predict group membership, the company can customize offers for individual prospective customers.

The example focuses on using demographic data, such as region, age, and marital, to predict usage patterns.

The target field, called **custcat**, has four possible values that correspond to the four customer groups, as follows:

1- Basic Service

2- E-Service

3- Plus Service

4- Total Service