K-NEAREST NEIGHBORS

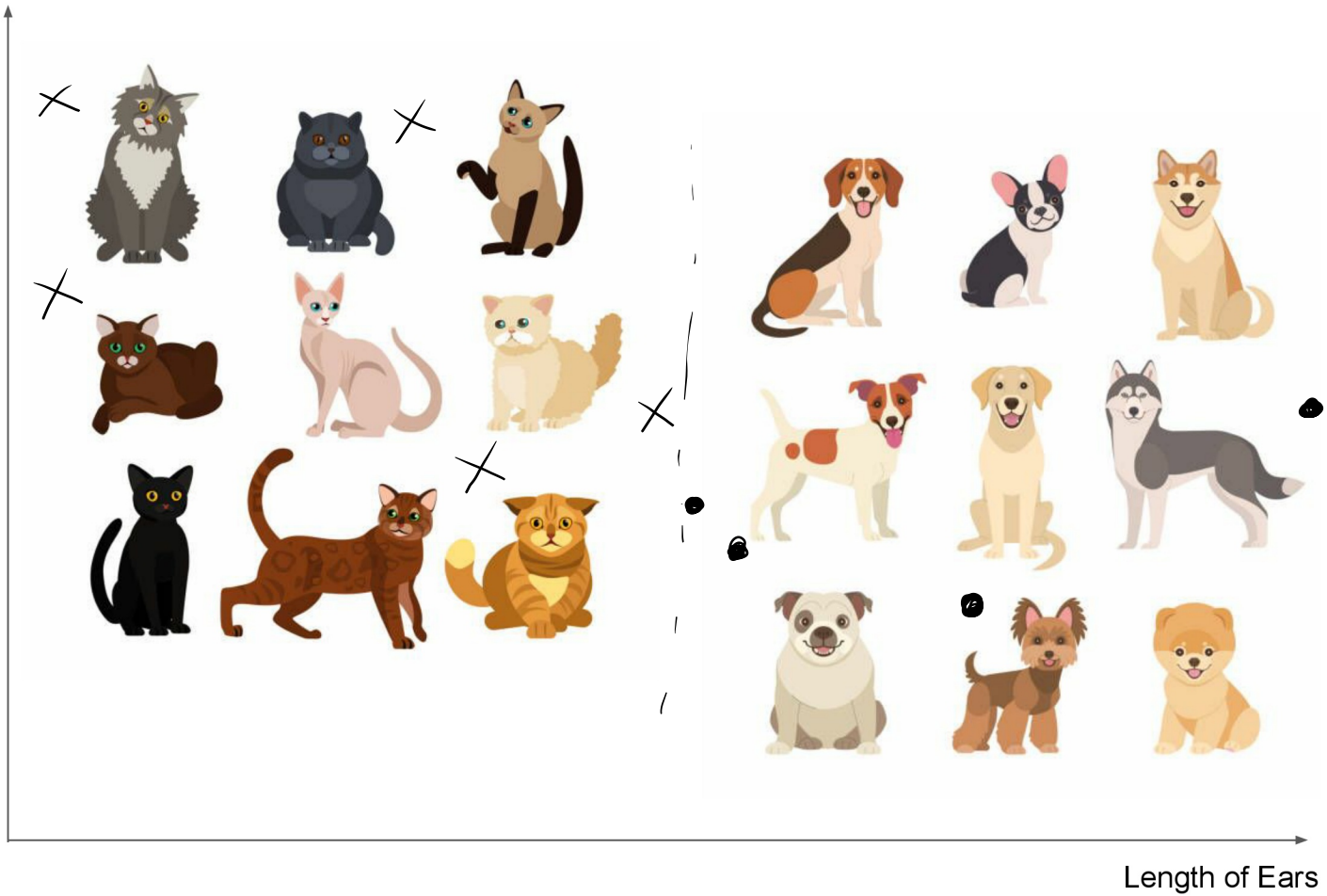
INNOMATICS
RESEARCH LABS

WHAT IS MACHINE LEARNING

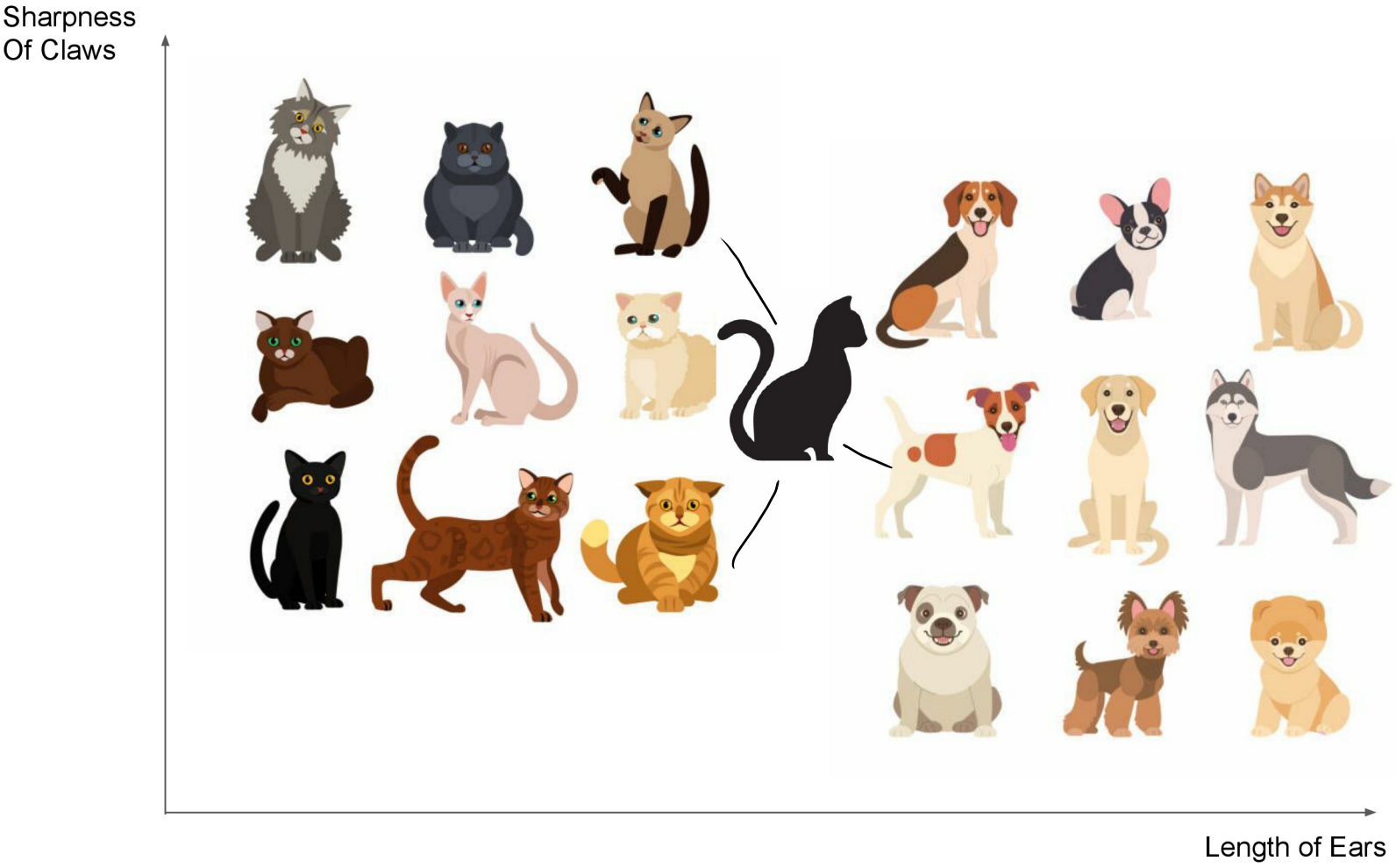
> “A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.”

- Tom Mitchell

Sharpness
Of Claws

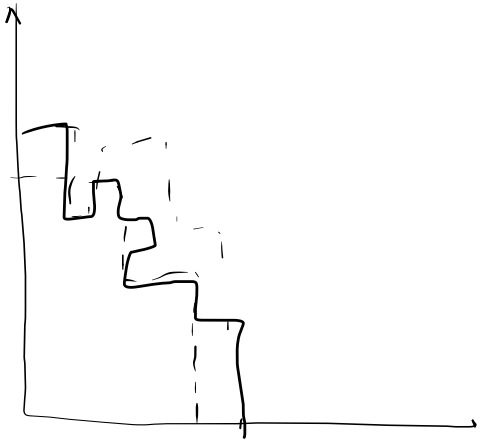


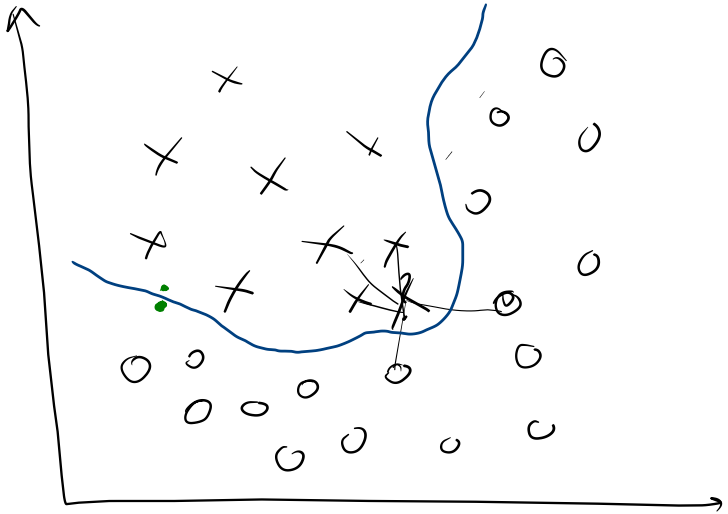
Length of Ears

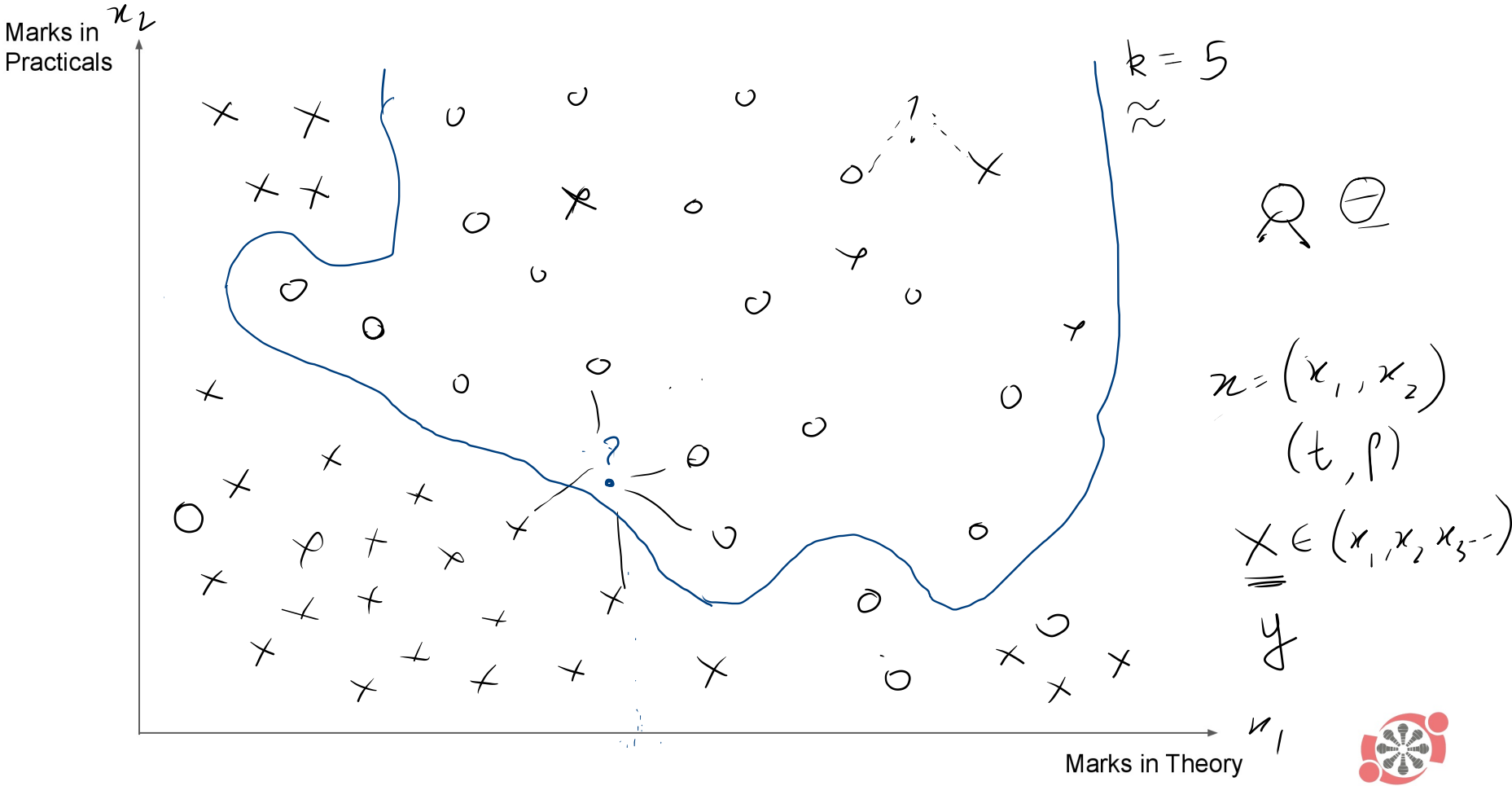

Sharpness
Of Claws
Length of Ears

COMPARISON OF FEATURES

If the features like a cat, it must be a cat.






Marks in Practicals
x_2
Marks in Theory
x_1
$k = 5$
$\approx$
$x = (x_1, x_2)$
(t, p)
$X \in (x_1, x_2, x_3 \cdots)$
y

K NEAREST NEIGHBORS

K-nearest neighbors algorithm (k-NN) is a non-parametric method proposed by Thomas Cover used for classification and regression. In both cases, the input consists of the k closest training examples in the feature space.

In k-NN classification, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbor.

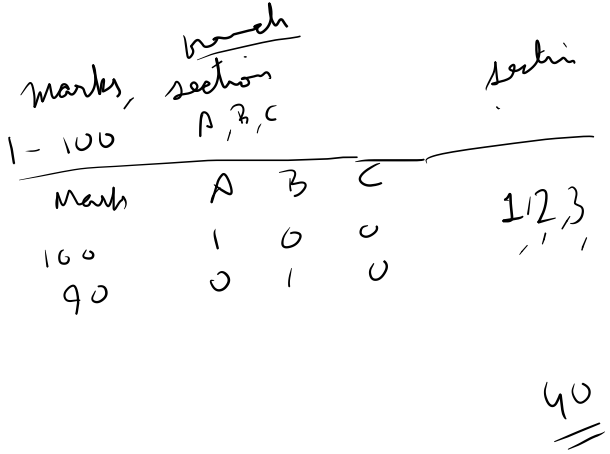
In k-NN regression, the output is the property value for the object. This value is the average of the values of k nearest neighbors.

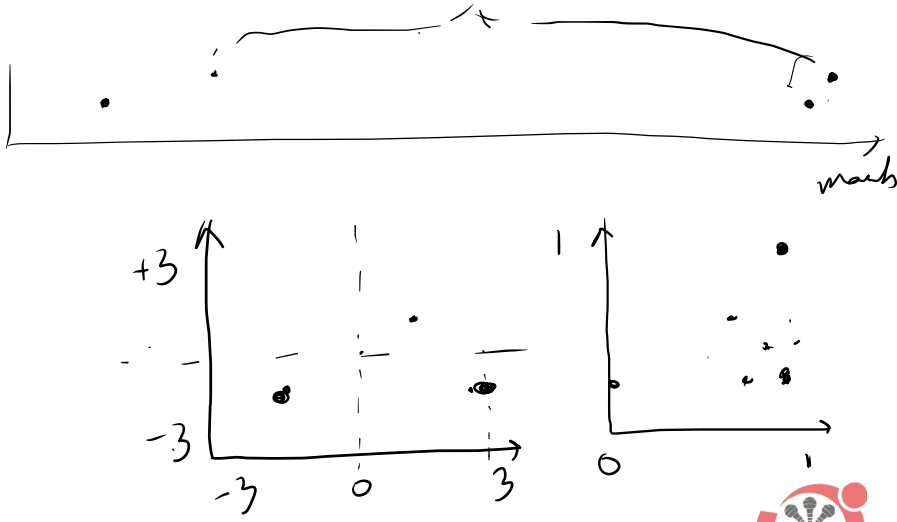
Cover, Thomas M.; Hart, Peter E. (1967). "Nearest neighbor pattern classification"(PDF). *IEEE Transactions on Information Theory*. 13 (1): 21–27.

Altman, Naomi S. (1992). "An introduction to kernel and nearest-neighbor nonparametric regression" (PDF). *The American Statistician*. 46 (3): 175–185.

K NEAREST NEIGHBORS

k-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until function evaluation. Since this algorithm relies on distance for classification, normalizing the training data can improve its accuracy dramatically





WHAT IS A NEIGHBOR?

Distance between points

$$D(A,B) = |y_2 - y_1| + |x_2 - x_1|$$

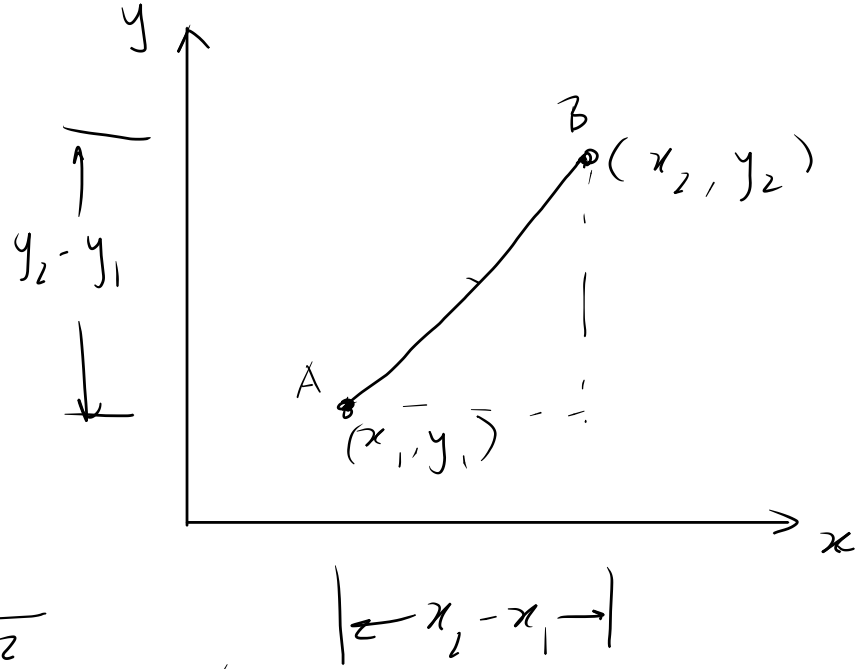
Manhattan Distance

$$D(A,B) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Euclidian Distance

$$= \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$

$$= \sqrt{d_{dim_1}^2 + d_{d_2}^2 + d_{dim_3}^3 + \cdots + d_{d_n}^2}$$



WHAT IS A NEIGHBOR?

	x_1	x_2	y	d
→	2	3	1	d_1
→	2	4	0	d_2 ←
	3	4	1	d_3 —
→	3	5	1	d_4 —
→	1	2	0	d_5
→	1	3	0	d_6
→	2	3	0	d_7

$k = 3$

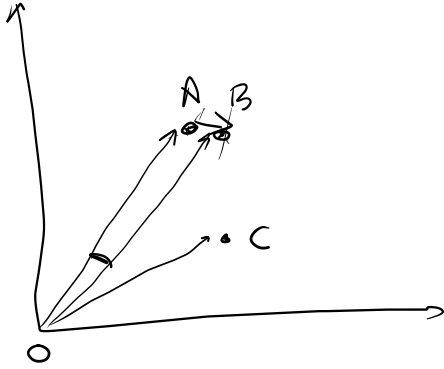
dimensions

N

(1)

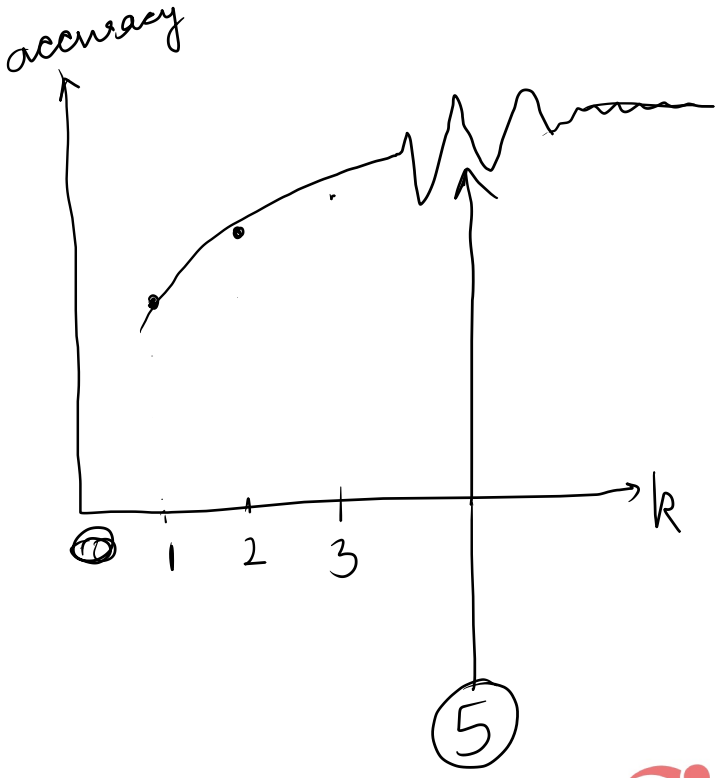
$$d(\mathbf{p}, \mathbf{q}) = d(\mathbf{q}, \mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$$
$$= \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}.$$

def qe1. dist(p1, p2)
retur



N (3, 5, 4)

CHOOSING THE RIGHT K



DISADVANTAGES

1. Entire training dataset needs to be stored
2. Slower to Predict
3. Computationally Expensive

ADVANTAGES

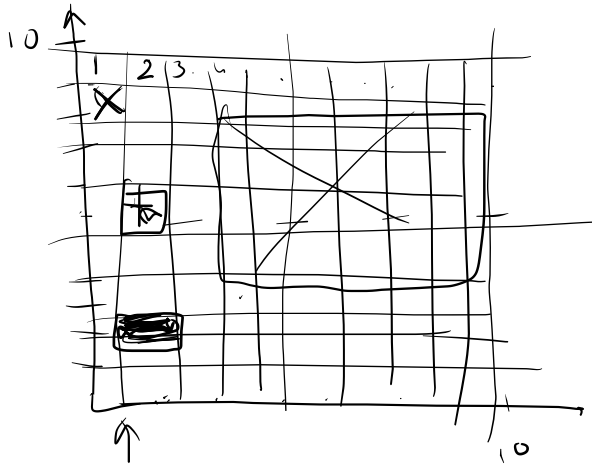
1. Seamless learning with new datapoints
2. Easy to implement
3. Easy to understand and interpret

MAKING KNN SEARCH FASTER

Limit the region of search

{ (1,9), (2,3), (4,1), (3,7), (5,4), (6,8), (7,2), (8,8), (7,9), (9,6) }

1 0 1 1 0



(2.5, 3.5) → 54

(4, 5) → 60

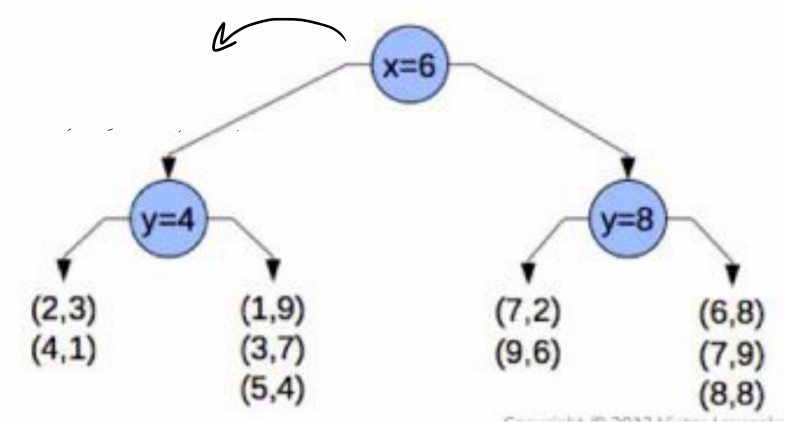
KNN

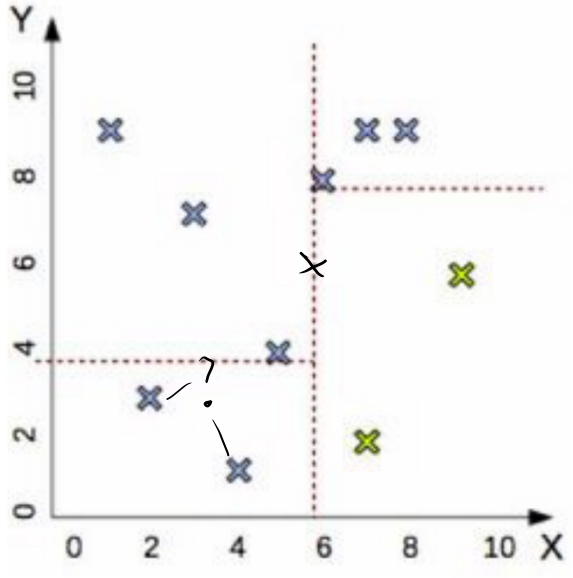
MAKING KNN SEARCH FASTER

Limit the region of search

{ (1,9), (2,3), (4,1), (3,7), (5,4), (6,8), (7,2), (8,8), (7,9), (9,6) }

KD TREE





WHERE IS KNN USED

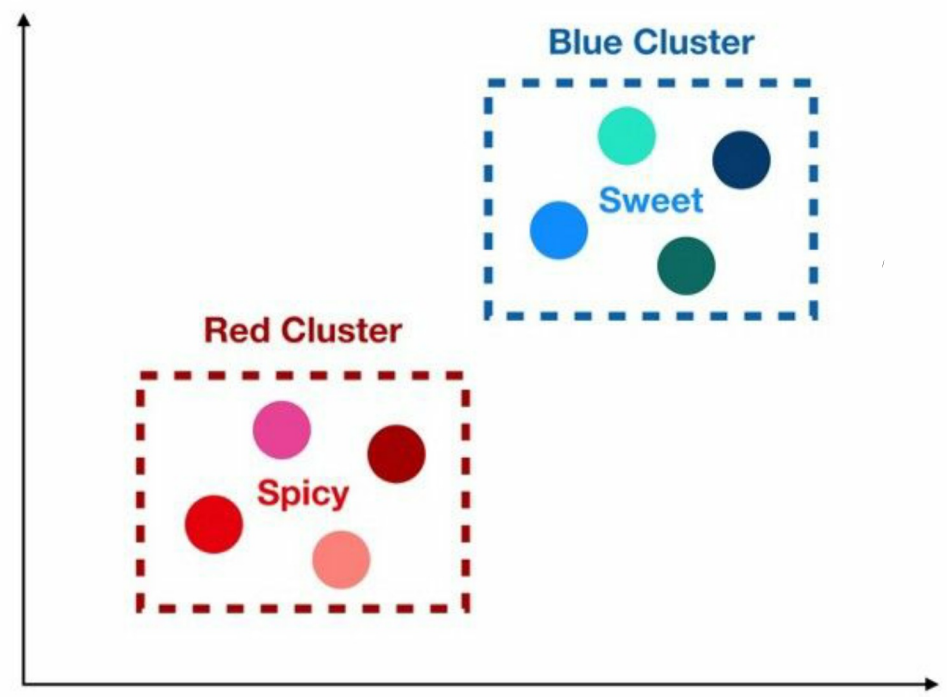
KNN is highly used where a problem can be resolved in too many classes. E.g. Digit Classification or Face Recognition.

Amazon uses KNN based methods in Product Recommendation

Herta Security uses deep learning algorithms to generate feature vectors representing people's faces. They then use k-NN to identify a person by compare the face to their watchlist.

CODE DEMO

CURSE OF DIMENSIONALITY



Overly simplified example, courtsey https://towardsdatascience.com/the-curse-of-dimensionality-50dc6e49aa1e

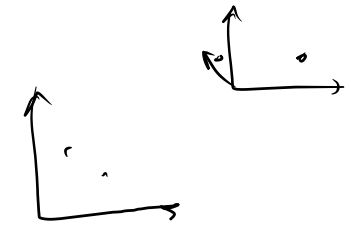
CURSE OF DIMENSIONALITY

	Reddish	Bluish
	1	0
	1	0
	1	0
	1	0
	0	1
	0	1
	0	1
	0	1

1
0
1
1
0
0
1
0

CURSE OF DIMENSIONALITY

	Red	Maroon	Pink	Flamingo	Blue	Turquoise	Seaweed	Ocean
	1	0	0	0	0	0	0	0
	0	1	0	0	0	0	0	0
	0	0	1	0	0	0	0	0
	0	0	0	1	0	0	0	0
	0	0	0	0	1	0	0	0
	0	0	0	0	0	1	0	0
	0	0	0	0	0	0	1	0
	0	0	0	0	0	0	0	1



CURSE OF DIMENSIONALITY

Every machine learning algorithm needs a dense data set in order to accurately predict over the entire data space. Errors arise in all algorithms if there are gaps between the data.

If we have more features than observations than we run the risk of massively overfitting our model — this would generally result in terrible out of sample performance.

When we have too many features, observations become harder to cluster — too many dimensions causes every observation in your dataset to appear equidistant from all the others.

The problem is fundamentally that there isn't enough data available for the number of dimensions. As the number of dimensions increases the size of the data space increases, and the amount of data needed to maintain density also increases. Without dramatic increases in the size of the data set, k-nearest neighbors loses all predictive power. And this makes one possible solution for the issue straightforward: Add more data. It's entirely possible to add more and more data to ensure that you have enough data density even as you add more dimensions.

CURSE OF DIMENSIONALITY

The problem is fundamentally that there isn't enough data available for the number of dimensions. As the number of dimensions increases the size of the data space increases, and the amount of data needed to maintain density also increases. Without dramatic increases in the size of the data set, k-nearest neighbors loses all predictive power. And this makes one possible solution for the issue straightforward: Add more data. It's entirely possible to add more and more data to ensure that you have enough data density even as you add more dimensions.

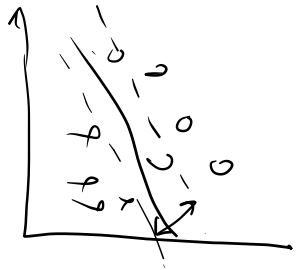
In some cases, we will use unsupervised techniques for Dimensionality Reduction. You will study about them in the next module.

BETTER DECISION BOUNDARY

If we have a situation where the data is linearly separable, we can create a hyperplane that maximizes the margin between the two classes.

The margin can pass through the nearest representative points of the two classes.

Rather than simply creating a boundary to minimize the cost, we maximize the width of the margin.



THAT’S ALL FOLKS