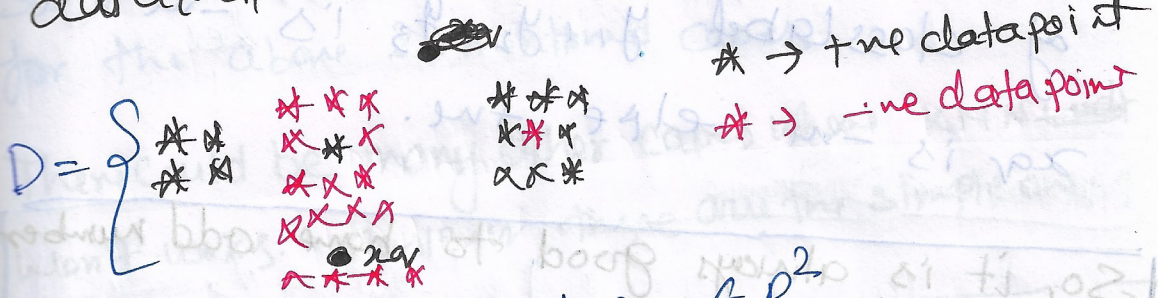
K- Nearest Neighbour (KNN)

Let's take an example to understand the concept. I'm having 2D- toy dataset



$$D = \{(x_i, y_i) \mid x_i \in R^2,\ y_i = \{0, 1\}\}$$

(y_i: 0 → -ve, 1 → +ve)

Let's I'm having a query point x_q, now I want to know, weather the x_q is +ve or -ve.

i.e. $x_q \longrightarrow y_q$

→ If we see the neighbourhood point of x_q, we will see mostly * point (-ve). So, by just uping the proximity, I can say that x_q is also -ve.

This is what the KNN is.

How KNN works

#1 Find K - "Nearest" Points to x_q in D.

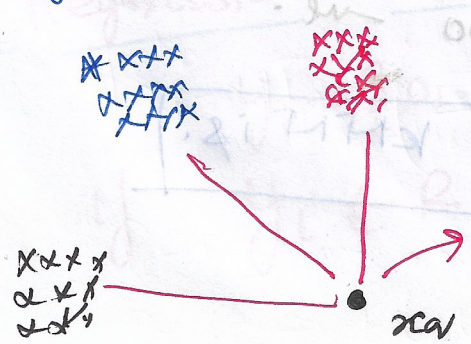
#2 Let K = 3 (x_1, x_2, x_3) (How to select the value of K, we will see later)

#3 If the majority of ~~point~~ ~~belo~~ of vote of class label y_1, y_2, y_3 is -ve the x_q is -ve else +ve.

So, it is always good to have odd number of K.

Now, Lets understand where ~~the~~ K-NN fails:

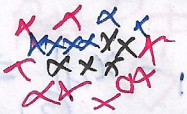
Case I My query point is very far away from my dataset, in that case it doesn't make any sense to calculate the behaviour base on the neighbour point.



However KNN based on the K value predict the behaviour, that is untrue.

In this case we can't say ~~what this is~~ anything about this point.

Case II The query point is so much jumbled across each other.

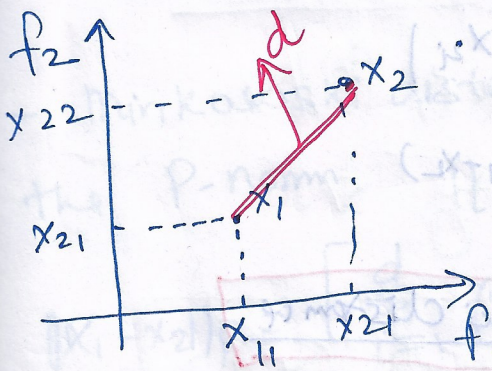


Whatever the KNN predict most probably wrong or No Useful information.

So KNN is not the algorithm we should select for the above 2 mentioned cases.

There could be many other cases where KNN won't works. However these are the simple one.

→ The base of the KNN is the distance b/w the points. Now, here we are going to understand how to measure the distance b/w the points.



$x_1 = (x_{11}, x_{12})$

$x_2 = (x_{21}, x_{22})$

$$d = \sqrt{(x_{21} - x_{11})^2 + (x_{22} - x_{21})^2}$$

→ Pythagorous Theorm

↳ Euclidean Distance ⇒ distance b/w shortest line b/w 2 data point.

Similarly $x_i \in R^d$,

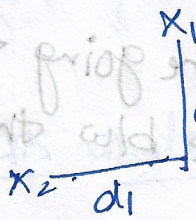
$$\|x_1 - x_2\|_2 = \left(\sum_{i=1}^{d} (x_{1i} - x_{2i})^2\right)^{1/2}$$

↳ **Euclidean Distance.** $\|x_1 - x_2\|_2$

↳ L2 norm of a vector

Manhatan Distance

$$\sum_{i=1}^{d} |x_{i2} - x_{2i}| \quad \text{i.e. } |(x_{21} - x_{11}) + (x_{22} - x_{12})|$$



$d = d_1 + d_2$

$$\|x_1 - x_2\|_1 = \sum_{i=1}^{d} |x_{i2} - x_{2i}|$$

↳ L1 norm of vector $(x_1 - x_2)$

Lp norms → **Minkowski distance**

↳ p could be any norm

$$\|x_1 - x_2\|_p = \left(\sum_{i=1}^{d} |x_{1i} - x_{2i}|^p\right)^{1/p}$$

$p = 1$ → manhatan

$p = 2$ → Ecclidian distance.

Lp norm of a vector

$$\|x_1\|_p = \left(\sum_{i=1}^{d} |x_{i1}|^p \right)^{1/p} \qquad p \neq 0, \; p > 0$$

So, distance one between 2 points, however the Norms ~~one~~ is for a vector.

- So, we have learnt that Euclidean distance b/w 2 points: = L2 norms of $(x_1 - x_2)$

$$= \|x_2 - x_2\|_2$$

- Similary manhatan distance b/w 2 points is L1 norm $\|(x_1 - x_2)\|_1 = \sum_{i=1}^{d} |x_{1i} - x_{2i}|$

- Minkowski distance b/w two points is the P-norm $\|(x_1 - x_2)\|_p$

$$\|x_1 - x_2\|_p = \left[\sum_{i=1}^{d} (x_{1i} - x_{2i})^p \right]^{1/p}$$

→ There are many types of distances available, here we discuss those we are going to use a lot. when to use what we will see later.

Hamming Distance → Mainly going to use in text processing (boolean vector)

x_1, x_2 → boolean vector

Hamming-dist(x_1, x_2) = # locations/dimensions where the binary vectors differs.

Haming diptance also used when x_1 and x_2 are strings = # locations/features/dimensions they are different.

Cosine Distance and cosine Similarity

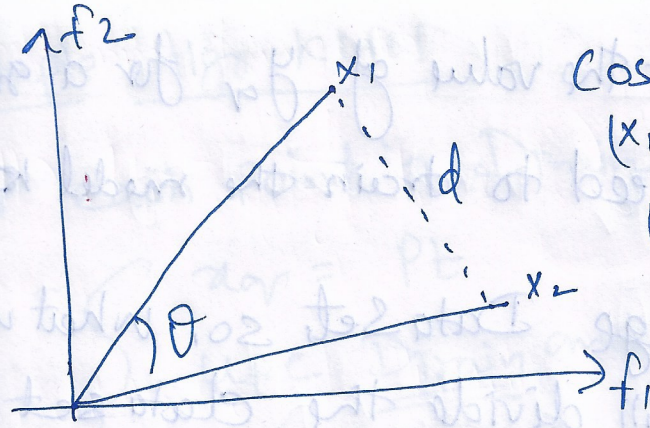
Similarity ↓/↑ distance ↑/↓

As distances ↑ similarity ↓.

$$1\text{-Cos-Sin}(x_1, x_2) = \text{Cos-dis}(x_1, x_2)$$

→ If the 2 points are very similar IS $\cos\theta = 1$,

If the 2 points are very dissimilar $\cos\theta = 0$



Cos-Sin = Cosθ

(x_1, x_2)

θ → angle b/w x_1 and x_2

→ How to measure distance in KNN

If we talk about the Amazon Dataset

$$D = \begin{bmatrix} x^T \end{bmatrix} \begin{bmatrix} y_i \end{bmatrix} \qquad y_i \in \{0, 1\}$$

0 → -ve, 1 → +ve

↳ Text BoW / tf-IDF / W2V

This is the dataset we are having

Problem

what is the problem ? Given a new text review, what is its polarity i.e. +ve/-ve.

$x_q \rightarrow (y_q) \rightarrow$ +ve / -ve

In order to get the value of y_q for a given new x_q, we need to train the model KNN

As we have huge Data Set, so, what we will do, we will divide the data set into 2 parts D_n → D_{Train}, D_{Test}

$$D_{Train} + D_{Test} = D_n$$

$$D_{Train} \cap D_{Test} = \phi$$

Any given data point can either belong to D_{Train} or D_{Test} but not both.

How to split D_n to D_{Train} and D_{Test}

One simple answer is randomly. It could be but not always be the best case.

We will see more methods in coming section

Simple Algo KNN

for each point in Dtrain

loop {
- $x_{qr} = Pt$
- use Dtrain and KNN determine y_q
- $y_{qr} == y_{Pt}$
- count += 1

Count = # Pt for which Dtrain + KNN gave a correct class label

$$\text{Accuracy} = \frac{\text{Count}}{n2}$$

Count = # Pts which Dtrain + KNN given a correct class label

$n2$ ↓ # points in Dtest

$$0 \leq \text{Accuracy} \leq 1$$

Say Acc = 0.91 ⇒ 91% of times we predict y_{qr} for a given x_{qr}.

Conclude K-NN on Amazon food review using Dtrain gives me the accuracy of 91%.