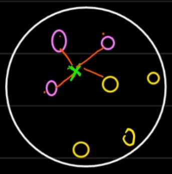
K-Nearest Neighbour

① KNN Classifier
② KNN Regressor

* KNN Classifier

f1	f2	y
—	—	0
—	—	1
—	—	0
		0
		1
		1

Sce-1



Scen-1

x (Assign a class)

* 4 nearest point

⇓

Pink   yellow

3   ↓ 1

⇓

majority ⇒ Pink

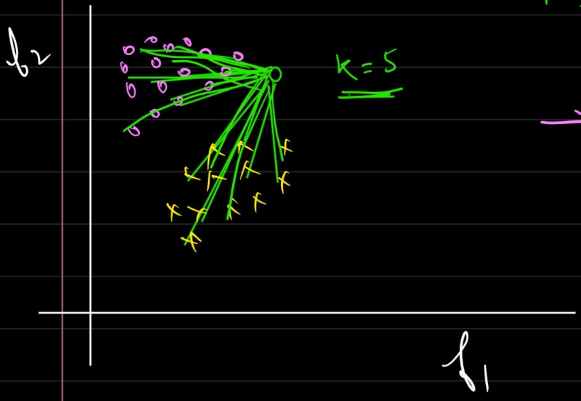
*⇒ Pink class

K → neares dp.

Scen-2

instead of
4 nearest dp,
take 7 nearest
dp.

* → yellow.



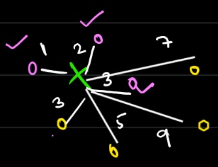
k = 5

→ As k changes,
the class of new dp
might also change.

⇓

K is a hyperparameter.

4 nearest dp



o — 1
o — 2
□ — 3
o — 3
o — 5
o — 7
o — 9

→ Prediction will be
o

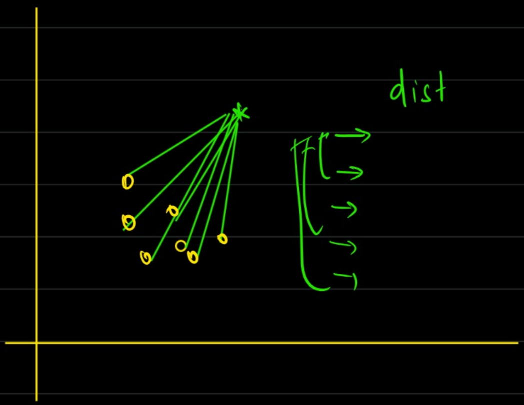
binary class. → K as odd value.

1, 5, 7, 9, 11, 13

2 class:

How K?? ⇒ K is hyperparameter.

Algorithm



① Plot the dp in n-d space.

② Initialise the K-value. (No of neighbours you want to consider)

$K \in 1 \rightarrow \infty$ (Generally $k > 3$)

→ Calculate the distance of new dp wrt to all dp's.

→ Sort the distance

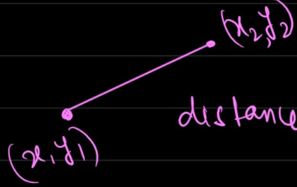
→ based on K find class of that K nearest dp's

③ Find the mode of the class

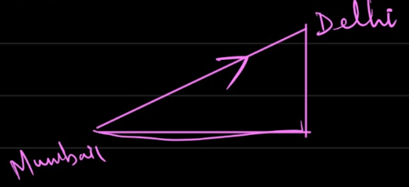
④ Assign the class.

distance

① Euclidean distance.

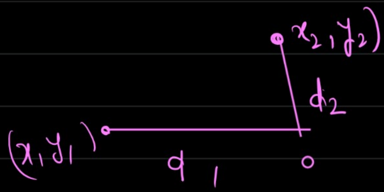


$$\text{distance} = \sqrt{(x_2-x_1)^2 + (y_2-y_1)^2}$$

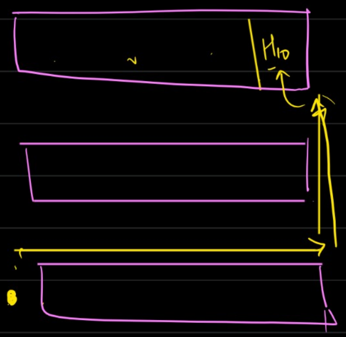


$$d(x,y) = \sqrt{(x_1-y_1)^2 + (x_2-y_2)^2 + \cdots (x_n-y_n)^2}$$

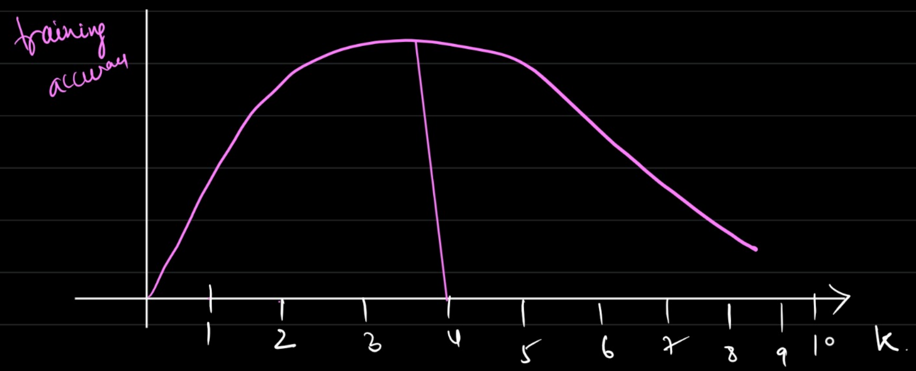
② Manhattan distance.



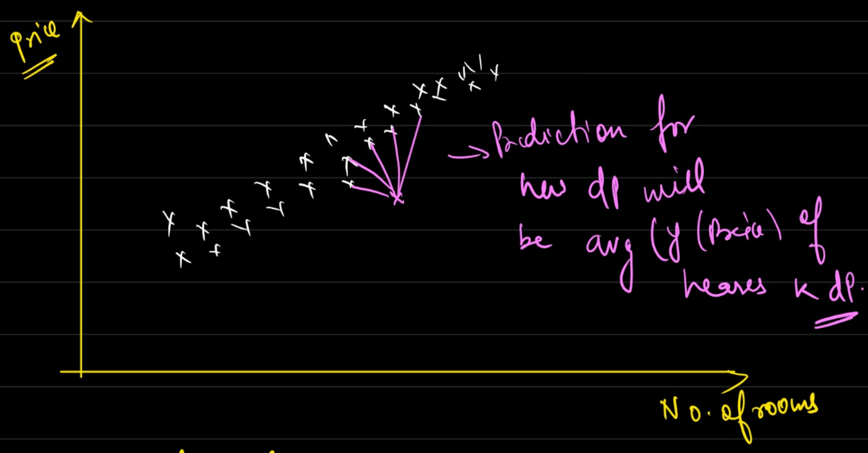
$$\text{distance} = d_1 + d_2$$



How to decide K.



* KNN Regressor



→ Prediction for new dp will be avg (y (price) of nearest k dp.

* Advantag of KNN

→ Easy to Understand | very intuitive.

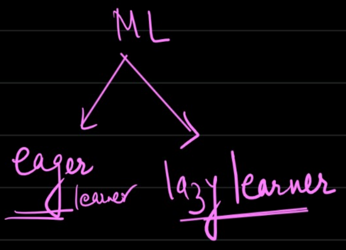
→ Performance of model in terms evaluatic metric is good

* disadvantage

→ Lazy learner.

⇓

Computation time is more.



- fit → Coeff's
- predict.

$\beta_0 + \beta_1 x_1 + \beta_2 x_2$

* All the model parameter are calculatef while training & used for Prediction.

→ you are Calculatig Parameter of model in real time.

↓

KNN.

Some more lazy learners

1. Locally weighted learning (LWL)
2. Case-Based Reasoning
3. Lazy Bayesian rules.✓

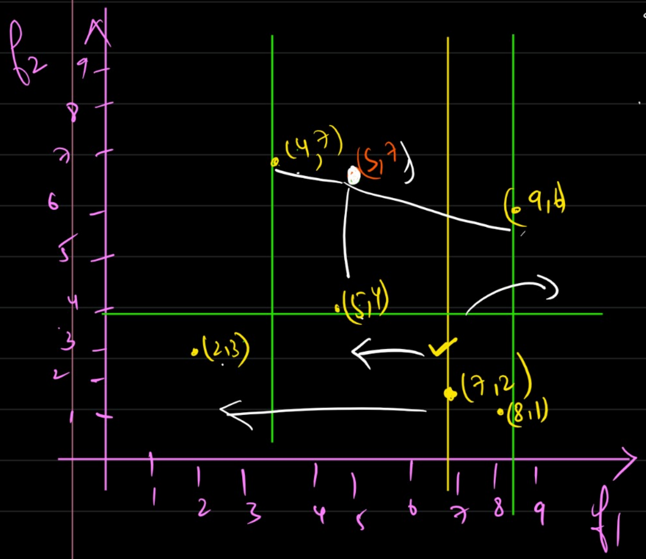
* Variants of KNN.

1. K-D tree
2. Ball tree.

why KNN is lazy learner ?

→ In real time the distance of test data from each of dp is Calculated ⇒ Brute Force

* Variants of KNN



f1	f2
7	2
5	4
9	6
2	3
4	7
8	1

* K-D-tree.

↳ Partition the data in a binary tree.

Step-1 sort the feature f1 & f2

Step-2 Calculate the median of f1 and f2.

Step-3 Partition the data based on median

Step-4 Partition all the data recursively in each of Subpartition.

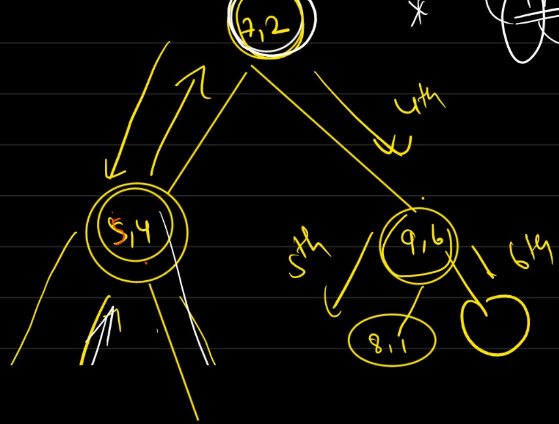
↓

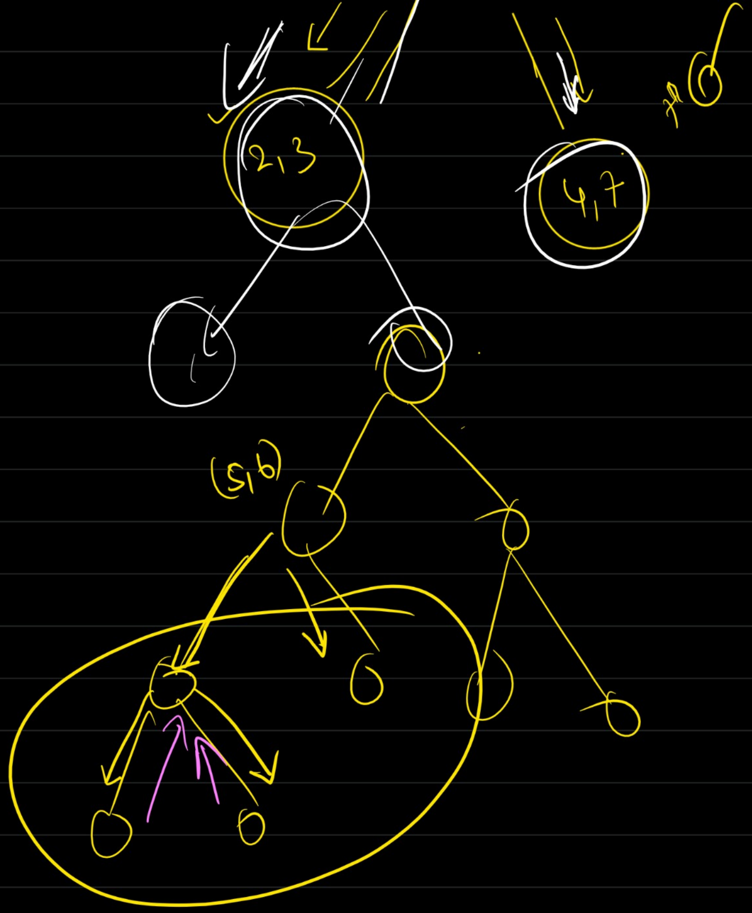
Untill in one partition only one dp is there.

→ f1 - 2, 4, 5, 7, 8, 9.

→ median → 7 as median

* (9, 6) (8, 1) → 9, 8





BST

Binary ⇒ O(logn)

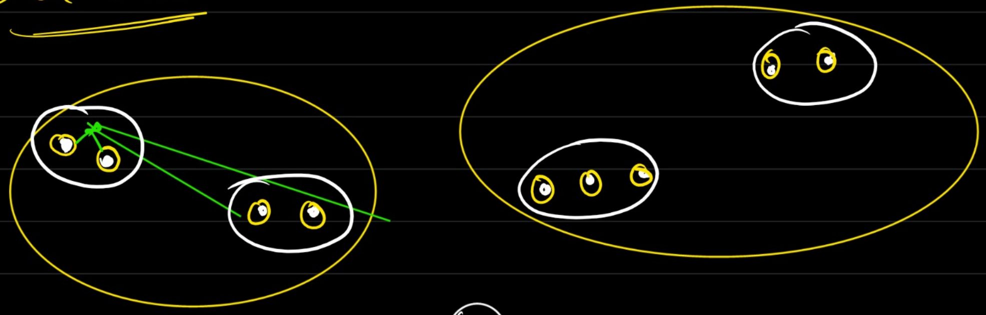
worst = O(n) → Brute force

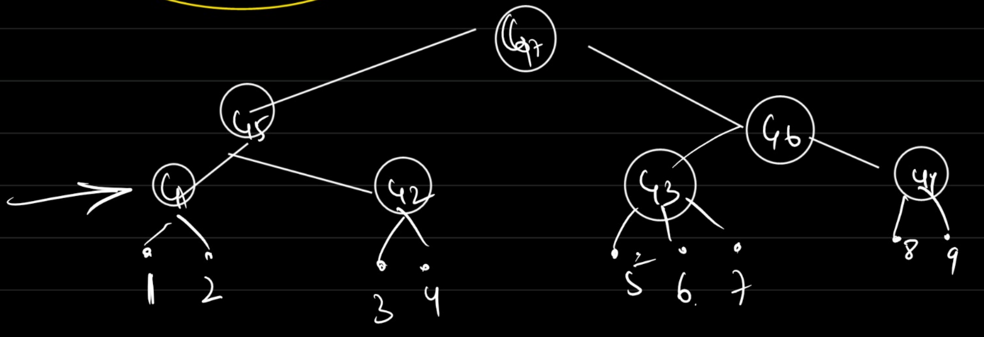
* Conclusion → reduced the search space.

② Ball tree

Advantage

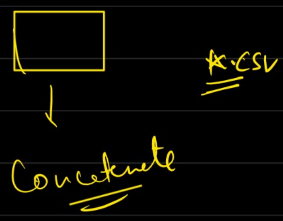
→ Only need to calculate distance of the nearest group element.





if small dataset → Use KNN

→ read all CSV in one go


*.csv
Concatenate