# Contents

# TASK 4 : CLUSTERING ALGORTIHMS AND K NEAREST NEIGHBOURS

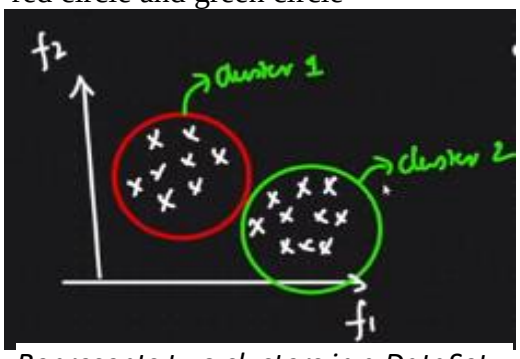
**Author: Rida Aimen Mirza**

# CLUSTERING ALGORITHMS

1. K Means Clustering
2. Hierarchical Clustering
3. Silhouette Score
4. DBSCAN Clustering

## K Means Clustering

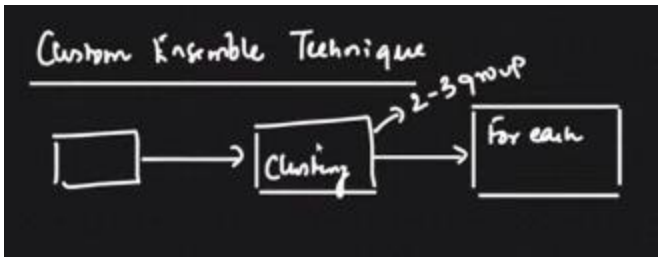
This is a kind of unsupervised learning which means that you don't have any specific outputs. We basically try to create different clusters based on the data. These data show which data are similar kind of data.
To understand how K Means clustering takes place. Suppose you have two features f1 and f2 and you plot them on a 2-D graph. So, our main purpose is to cluster datapoints into groups. Let's consider we have groups marked by red circle and green circle




*Represents two clusters in a DataSet*

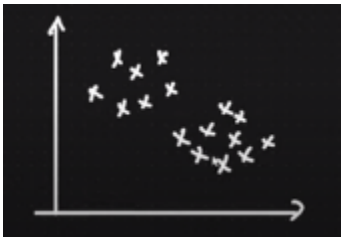
## Where does clustering gets used?

In Custom Ensemble Techniques, whenever we are probably creating a model. we create clusters in our data set.
During our model creating, the first algo that we apply will probably be Clustering algo and after that its better to apply regression or classification. Suppose we get 2-3 clusters from clustering algo . now we can apply a different algo to each group if we know the specific output that we really want.



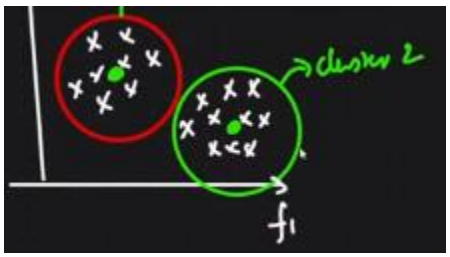

*Working of Custom Ensemble Techniques*

In KMeans clustering algo .
K → centroids
Suppose you have a data set looks like this :



*Example Data set*

By looking at this you can say K=2 . Which means that you will be having two groups and each will b having a centroid




*Two Centroids*

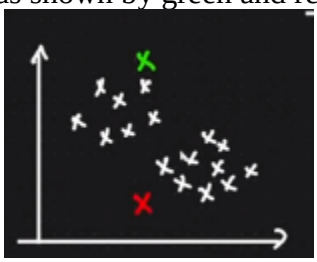
This centroid determines that you have two separate groups. Here and here.
Now you can just look and say that yes here are just 2 groups but how do you actually come to a conclusion that YES, THERE ARE ONLY __ number of groups? We can't just look and say because you will be having a high dimension data.
Here the diagram is just 2-Dimensional data..
SO for this we perform some steps :
**1.We try with different K values that means that we try with different centroids.**
Lets say that this is out plotted data :

And we start with

K=2
How to come up with a perfect K value ? We will talk about this later . This is base on a concept called Within cluster sum of square

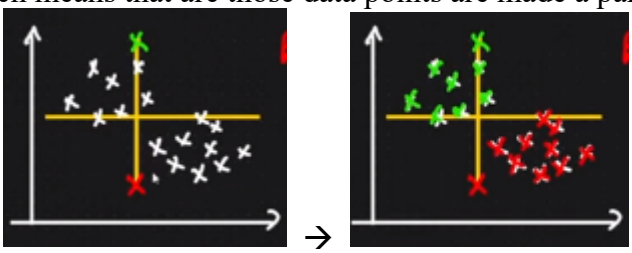
**2.We initialize K number of centroids**

So in in particular case we initialize 2 centroids randomly.
For example we initialize centroids as shown by green and red cross as follows:



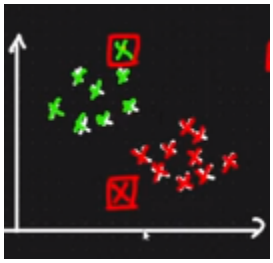
**3.Now you need to find the points that are close to each centroid . This is done by using Euclidian Distance**

In easier words . if I join the two centroids that I have placed randomly and then put a line perpendicular to it that is intersecting it . All the points closer to the particular centroid are given that centroid's color which means that are those data points are made a part of one cluster / group



*Perpendicular lines Separating two clusters*

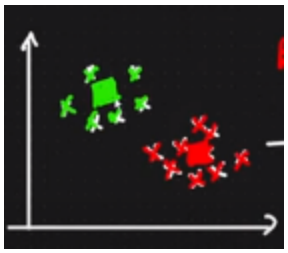
So now we can see that this is based on the shortest distance
So now for the next step let me remove these lines



*Placing Two Centroids at random positions*

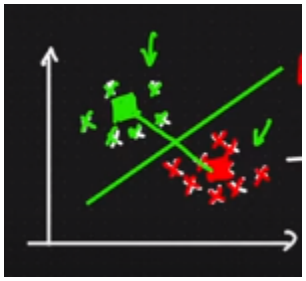
**Compute average of the points in each cluster to update centroids**

So now , when you will compute the average the centroid will before more close to the center
As a result :



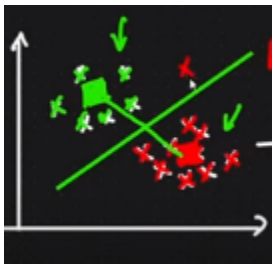
*Updated Centroids*

**THEN AGAIN THE STEP NO 3**
Like this :



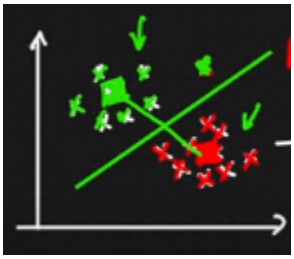
*Repetition Of Step 3*

Now if the Centroids are already in correct place . No Update will be made .
Yes if suppose if we had a red  point here :



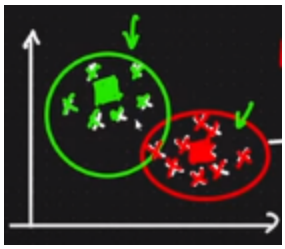
*Test Point*

Then this would have been converted to Green



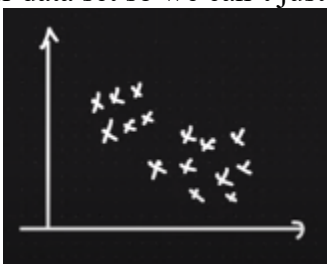
*Test Point Classification*

But its not the case here .
So we finally have two groups :

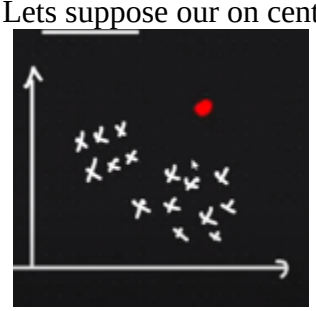


## How do we decide this K value ?

We use a concept called **“Elbow method”**
This will help us find
**the optimized K value** (whether k value should 1,2,3)
Suppose, this is the datapoints of our data set so we can’t just choose a value

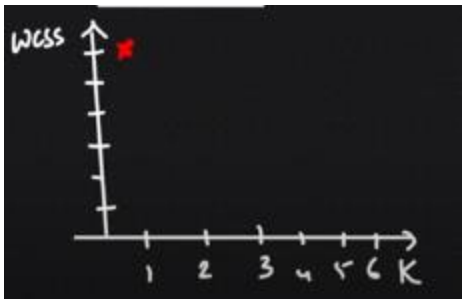


So we will have have to go with iteration
for i=1, i=10
So for every K-Value we construct a graph with respect to K and WCSS
WCSS -> Within cluster Sum of square
Initially we start with one centroid . Lets suppose our on centroid is place here :



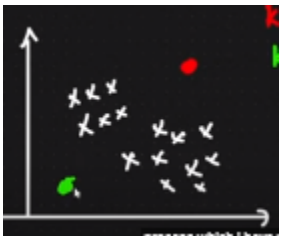
*Showing first supposed centroid*

If we go and compute the distance of each and every point from the centroid and if we try to find out the distance . Will it be greater or smaller ?
IT WILL ALWAYS BE VERY GREATER
Lets suppose For K=1
WCSS comes out to be this :

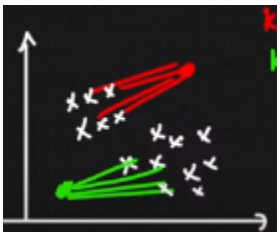



*Error Graph*

Now when we go with next value K =2 . I have 2 initialized  points and then we will repeat the entire set of steps mentioned above

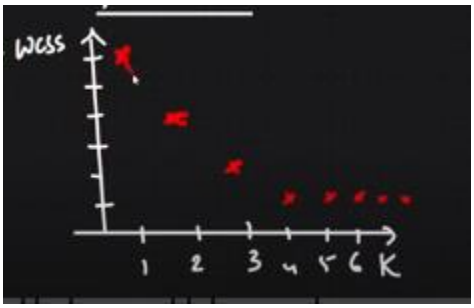


*Two centroids*



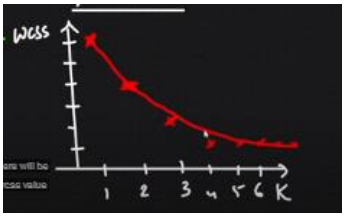
*Showing distance of two centroids from Different datapoints*

Now obviously the distances will bi lesser than the previous ones. we will get a point on graph like this:

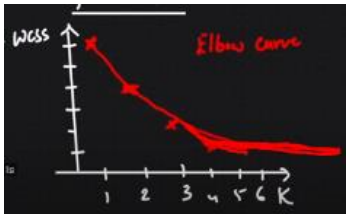



*Graph representing decrease in WCSS*

So, you can see that the WCSS value will decrease abruptly with increase in K value




after this the curve becomes straight →




*ELBOW CURVE*

No this is called an elbow curve
FOR FINDING K VALUE. we use → Elbow method
FOR VALIDATING PURPOSE, we use → Silhouette Score
The suitable K value is the one where the final abrupt change in curve occurs and it becomes a straight line
Obviously complexity will be high

## Problem with K-Means

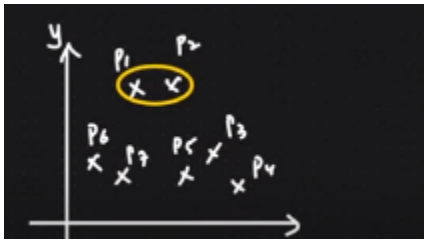
## 1.One conditioning for proper grouping is to place the centroids very far away

. only then they would update successively and come to proper correct positions. Otherwise you would get incorrect output.
To fix this "K-Means ++" is used

## Hierarchal clustering

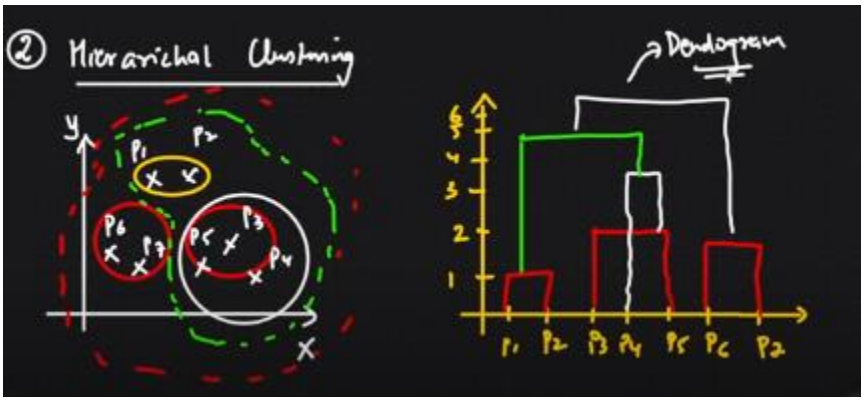
This algo goes step by step

Suppose we have these points



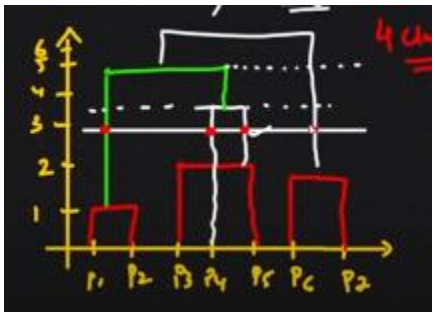

*Datapoint of a dataset*

*So it groups the closest ones togethers one by one . and it moves forward from smallest distance to largest distance*




*Hierarchical Clustering Graph and its Dendrogram*

This graph is called dendrogram.
Now, how do you find . How many groups should be there in the graph?
You need to find the longest vertical line that has no horizontal line passing through it.
In this case that vertical line is the white one




*DENDOGRAMS*

And then wee put a horizontal line total number of vertical lines that this horizontal line passes through. Is 4 . clusters are 3

## BY WHICH ALGO IS MAXIMUM TIME TAKEN ? K Means or Hierarical Clustering ?

**ANSWER : Hierarchical**
Because lets say , u have toooooo many data points then it will continue making many dendrograms . so it will take too much time.
So if your data set is small go ahead with hierarchical. if datagram large, go ahead with Kmeans.

## VALIDATE CLUSTERING MODELS

In regression problems we do validation using performance matrix line Confusion matrix , accuracy , recall etc
But for clustering models for validation we use

## Silhouette Score

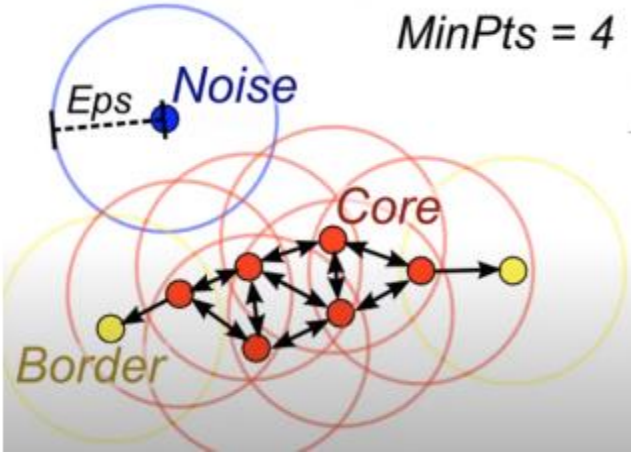
## DBSCAN → Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

This is one of the better algos .

## 4 important components of DBSCAN

- Epsilon
- Min Points
- Core Points
- Border Points
- Noise point

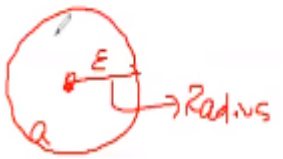
This and unsupervised learning technique as we already know. So, we know that such an algo will form clusters of closely present points.
So what happens is that . We have initially some points



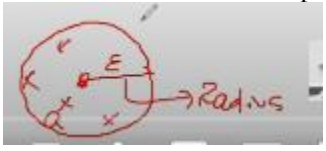

*Representation of Epsilon , min point, core point,border point,noise point*

So what we do is that , we initially consider some epsilon value and some Min Points value
Suppose we consider Minpnts value = 4 , epsilon value = 3
Epsilon value indicates something very important .


Suppose this is my value a .
Then epsilon is the radius of the circle




Now after creating particular radius . The concept of min point comes .
As I have initialized min pnts= 4 . let they be arranged like



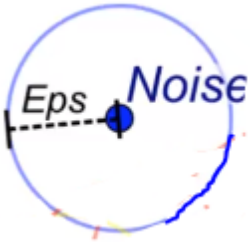
And a will be considered as core point

## TWO CONDITIONS TO BE THE CORE POINT

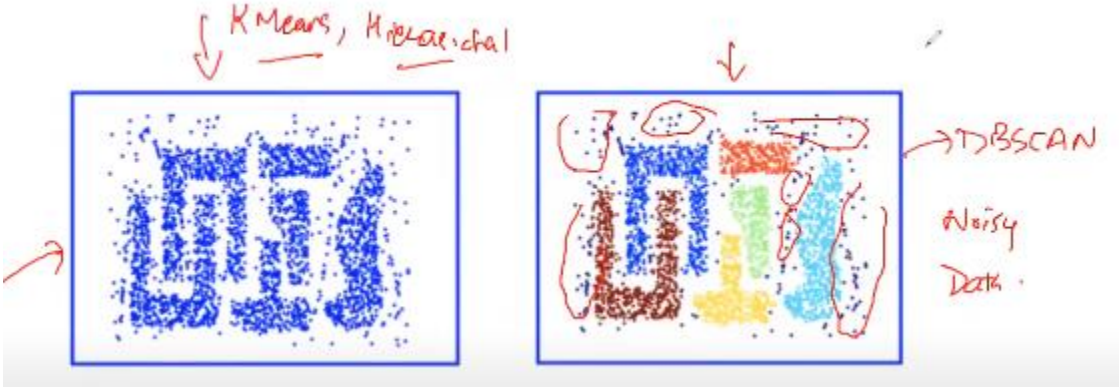
- I have to consider a boundry with epsilon value radius
- Min number of points that fall within this boundry should be atleast equal to the initialized min points value

Suppose we have a point C that does not satisfy the conditions to be a core point but it has atleast 1 core point inside its boundry . its called a border point


Suppose I have a point E and this is neither  Border point nor a core point
The its called a noise point




Noise point is basically like an outlier . so DBSCAN  skips the noise point and never considers it as a separate cluster




## Advantages :

- Great at separating clusters of high density  versus clusters of low density .
- Great with handling outliers (noisy data points)

## Disadvantages:

- Doesnot work well when dealing with clusters of varying densities .
- Struggles with clusters of similar densityy
- Struggles with high dimentionality data

# K- Nearest Neighbors

KNN can be used in two types of problem statements :

- Classification
- Regression

## FOR CLASSIFICATION PROBLEM

Suppose I have a problem statement where I have two features and 2 fixed out puts

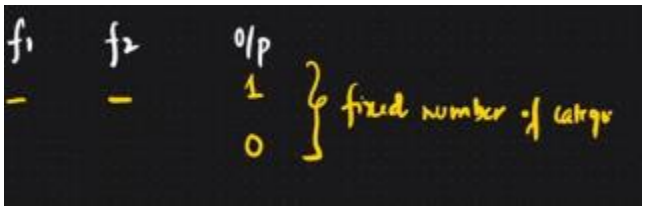
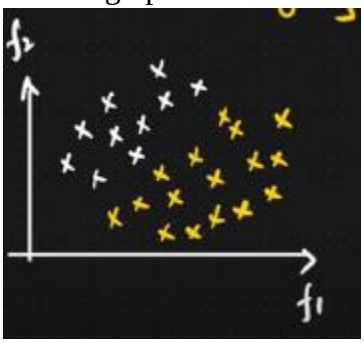


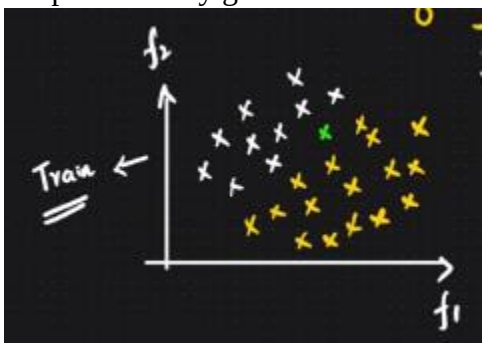

*Figure 1Classification problem example for KNN*

Lets say we have plotted the data in a 2-D graph as below :




Suppose this is our training dataset .

If we take a new test data point represented by green




*Test point in KNN example*

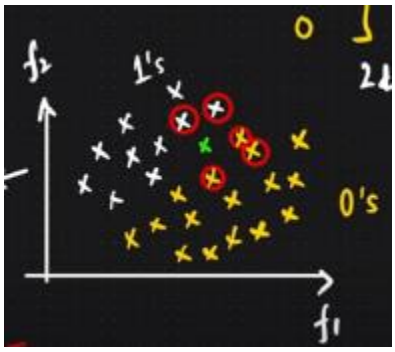
Now we have to decided to which category does this belong to ?

How doest KNN figure this out ?

KNN finds out a K value

Let K =5

So this test data point see 5 nearest data points to it




*Nearest K points found*

Therefore , to recap . The steps are :
Find value of K
Find K nearest data point
Now find that amongst these 5 , How many data points belong to each category and the one with max number of data points
1's --> 2
0's –> 3

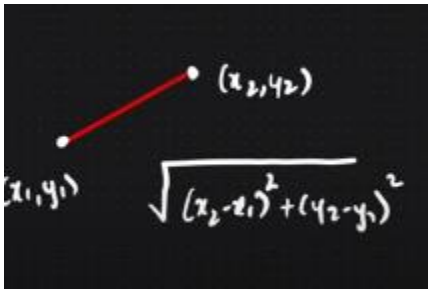
## So now , how to we actually find the distance ?

Two methods can be used :

- Euclidian Distance
- Manhattan Distance

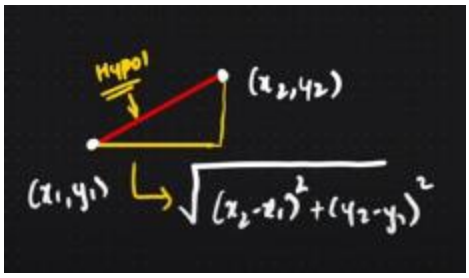
## Euclidian Distance

Suppose you have to find the distance between two points . as shown below . then the formula will be :




*Euclidian distance explanation(1)*

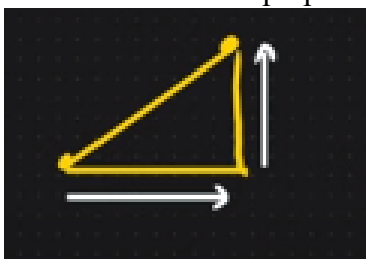
So , how did we get this ? Simply by using Pythagoras theorem




*Euclidian distance explanation(2)*

## Manhattan Distance

We are not into detail right now . Just know that we don't jump to finding the hypotenuse .
We calculate the base distance at shown and then then perpendicular's distance

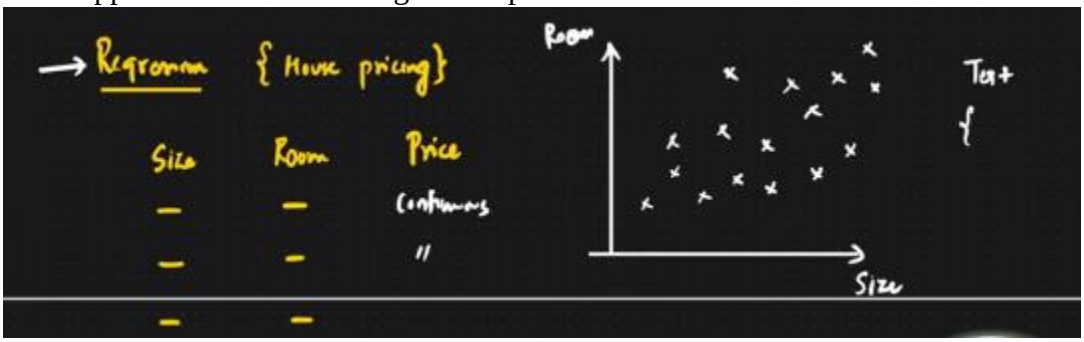


*Manhattan Distance*

Lets consider that we have taken the Euclidian distance in the example being mentioned and so it means that , based on that , the point nearest to me is the most maximum in zeros  so this new test point also belongs to zero category

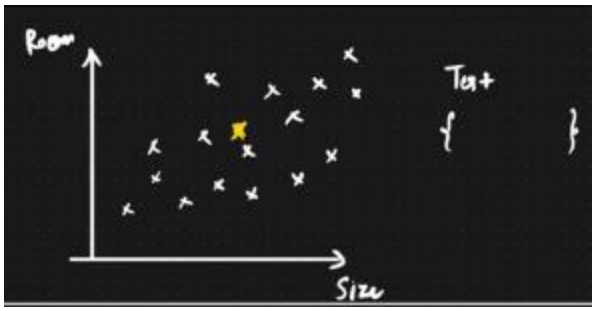
## FOR REGRESSION PROBLEM

Lets suppose that we have a regression problem that looks as follows and has following graph

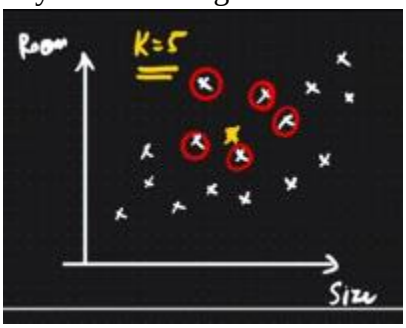



*Regression problem for KNN Example*

Now suppose I have a test data  and that new point is located some where as shown by yellow mark




So now how will we find the price of the house ?
Step 1 : Find a K value
Step 2 : Find K number of nearest data points
Steps 3: Take average of all those K values .
There fore the p=output will actually be the average of nearest data points



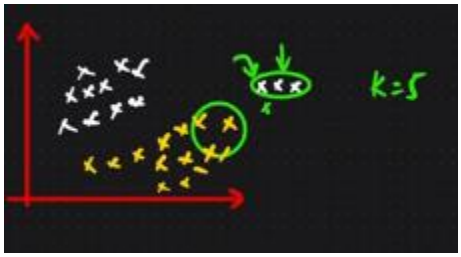

## LIMITATIONS OF KNN

### Wont be able to use KNN with very huge data sets.

Because itll create problems with large data set because if the value of K is large then youll have to calculate distance soo manyyy timeess
So itll create a problem

### Outliers will hugely impact this algorithm . it is very sensitive to outliers

Suppose you have a data set as shown below




*Representation of how Outliers effect KNN*

You can see that this way it will it say that the test data is from the white category. but that will be a wrong result. because the outer 3 white marks are actually outliers.

### Sensitive to missing values