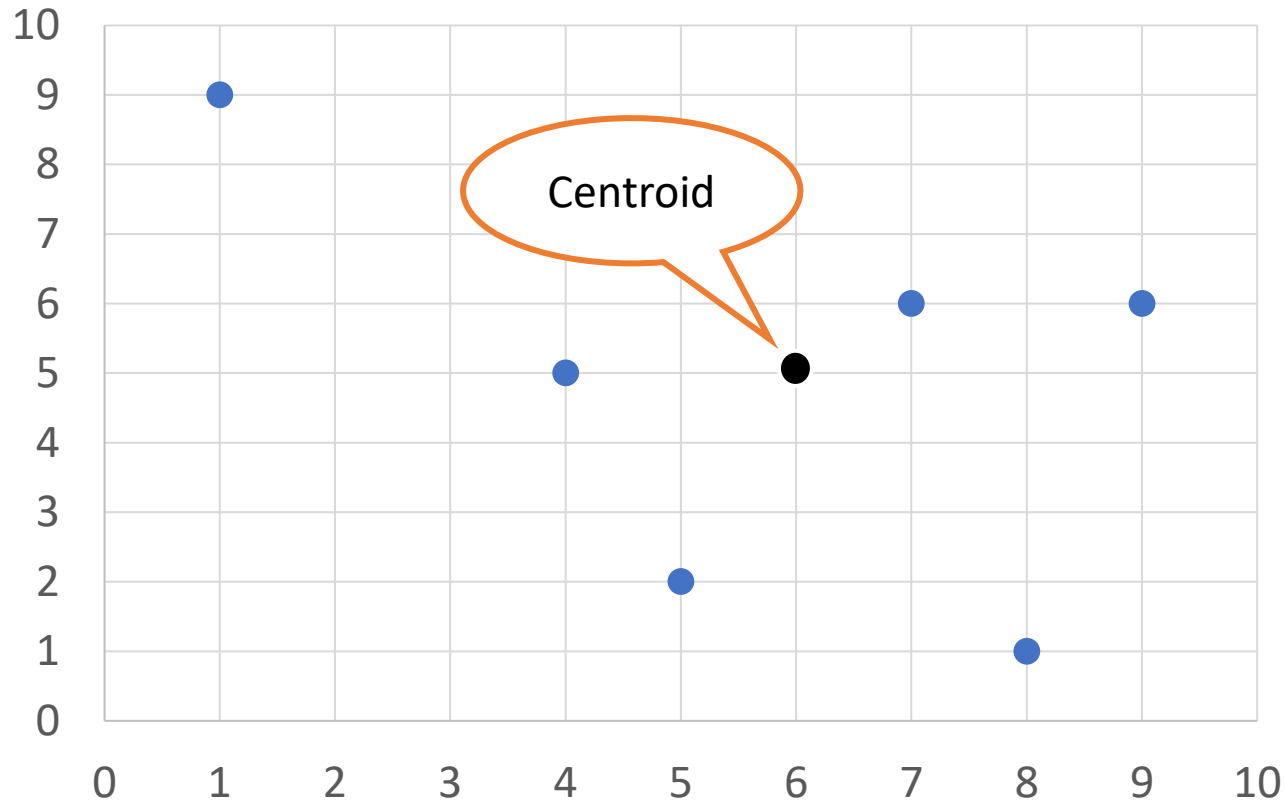
K Dimension Tree

K Dimension Tree

Sl No	X	Y	Target
1	9	6	0
2	7	6	1
3	8	1	0
4	1	9	1
5	4	5	0
6	5	2	1
Centroid	6	5	

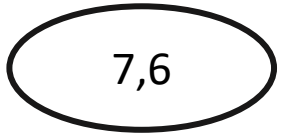


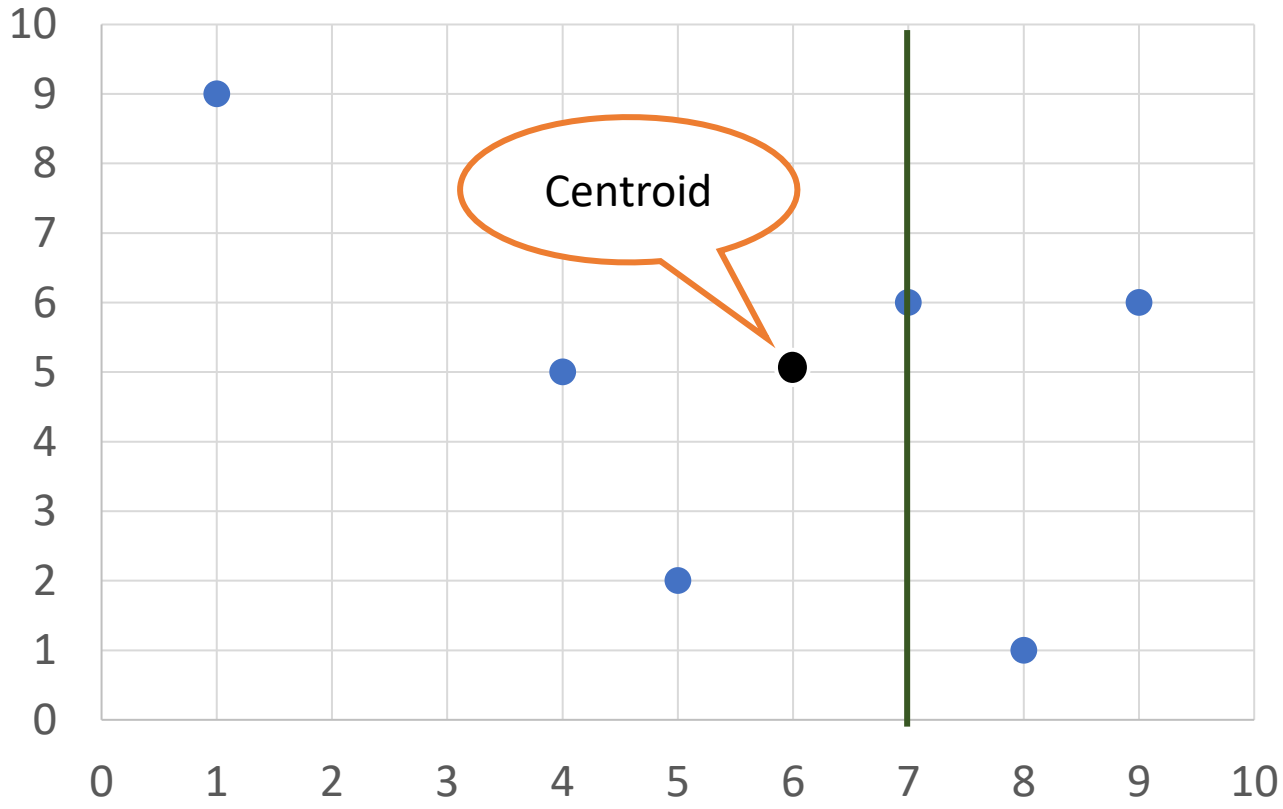
- We have a data set of X,Y, plotted the same on the graph
- K D Tree follows the centroid method, so we calculated the centroid and plotted on the graph
- **Centroids are rounded to single digit for better understanding**

K Dimension Tree

- Select the nearest point to the centroid to split the data and to construct a tree
- Choosing the point 7,6, then draw a line parallel to Y axis on X

Construction of K D Tree



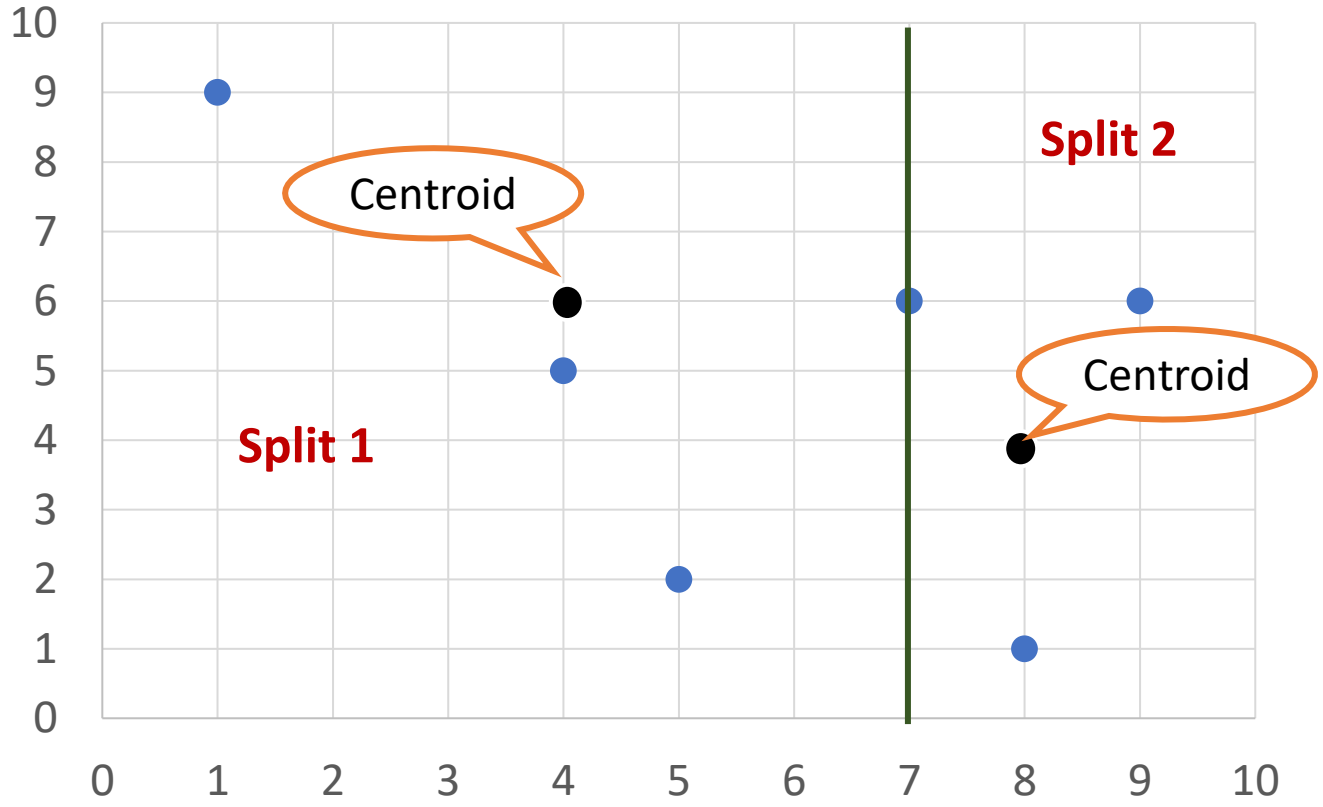


K Dimension Tree

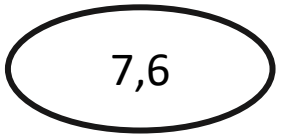
- Data split/ partitioned into two parts split 1 and split 2
- Calculate and find the centroids of the two splits.

Split 1		
Sl No	X	Y
2	7	6
4	1	9
5	4	5
6	5	2
Centroid	**4**	**6**

Split 2		
Sl No	X	Y
1	9	6
2	7	6
3	8	1
Centroid	**8**	**4**

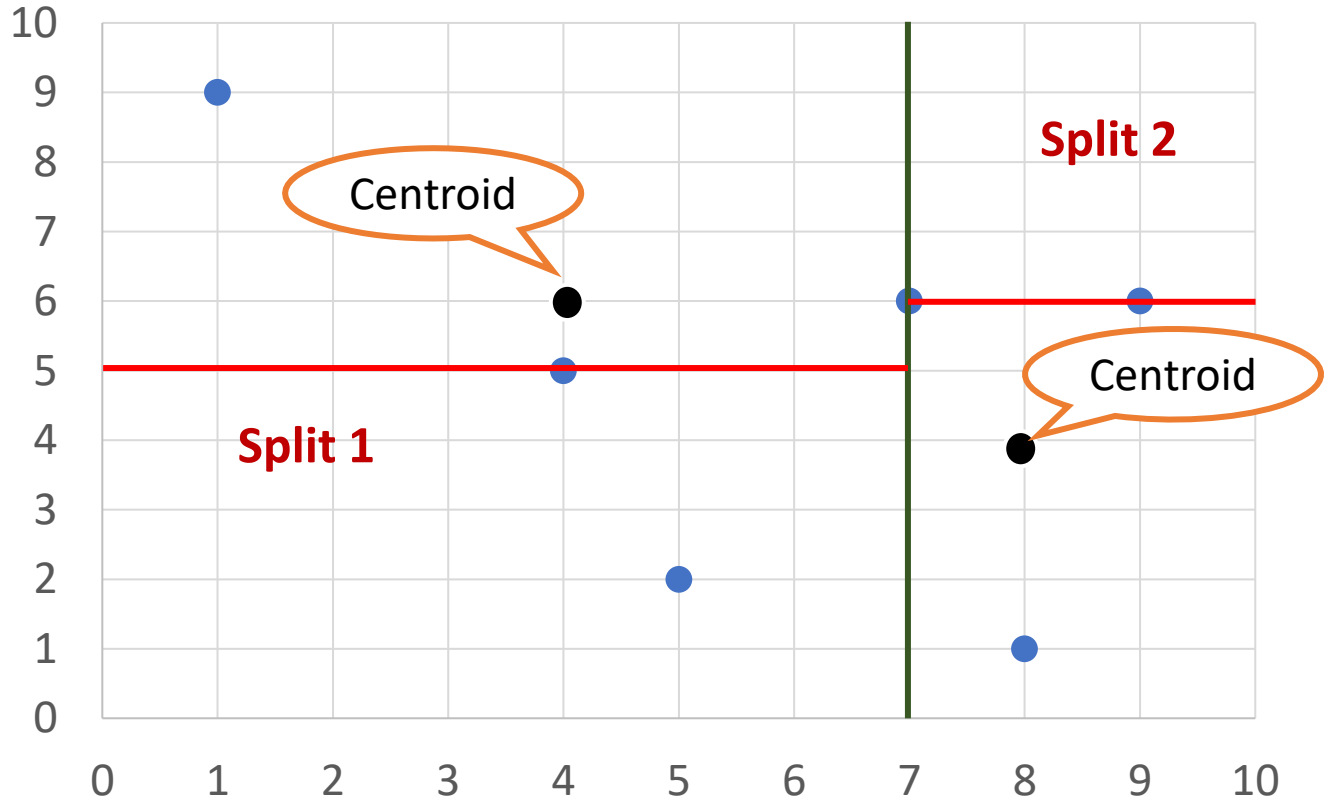


Construction of K D Tree

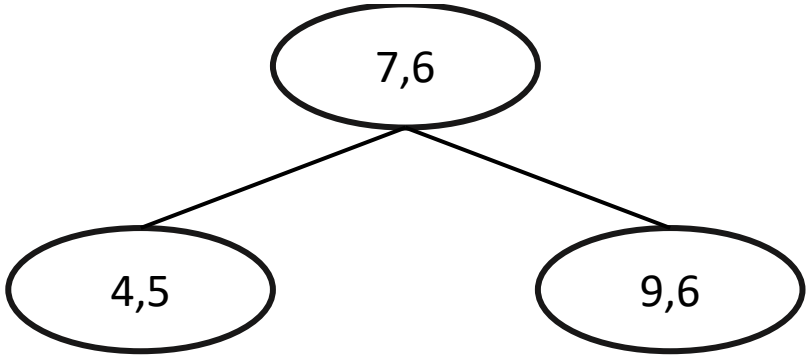


K Dimension Tree

- Find the nearest observation to the centroid and split the data plotting a line parallel to x axis on y axis
- K D Tree splits data by drawing parallel lines to axis alternatively ie first parallel to y axis, next parallel to x axis, again parallel to y axis and so on..
- Selected 4,5 from split 1
- In split 2 distance b/w centroid to 8,1 is 3.3 & to 9,6 is 1.9 using Euclidean. Therefore, selected 9,6



Construction of K D Tree

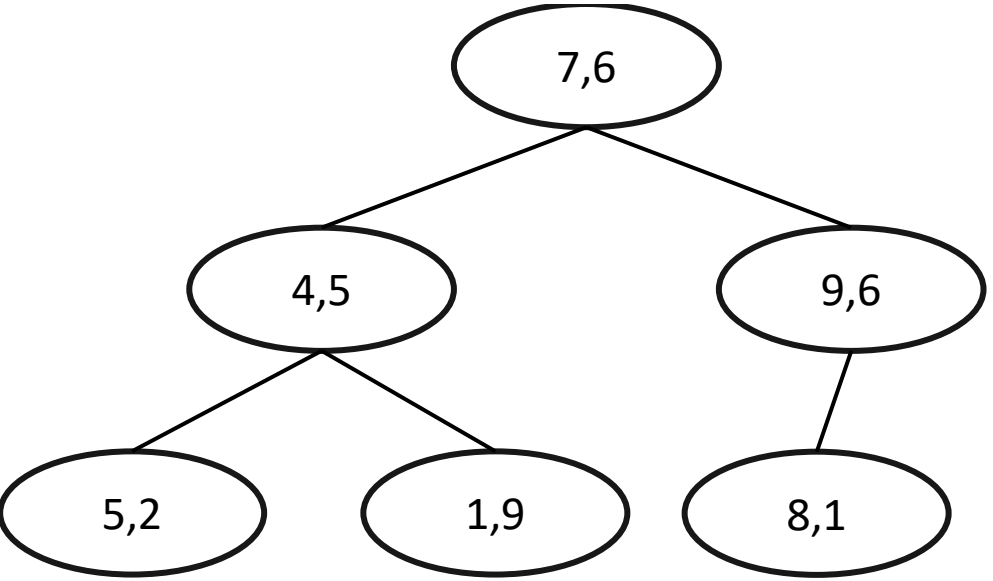


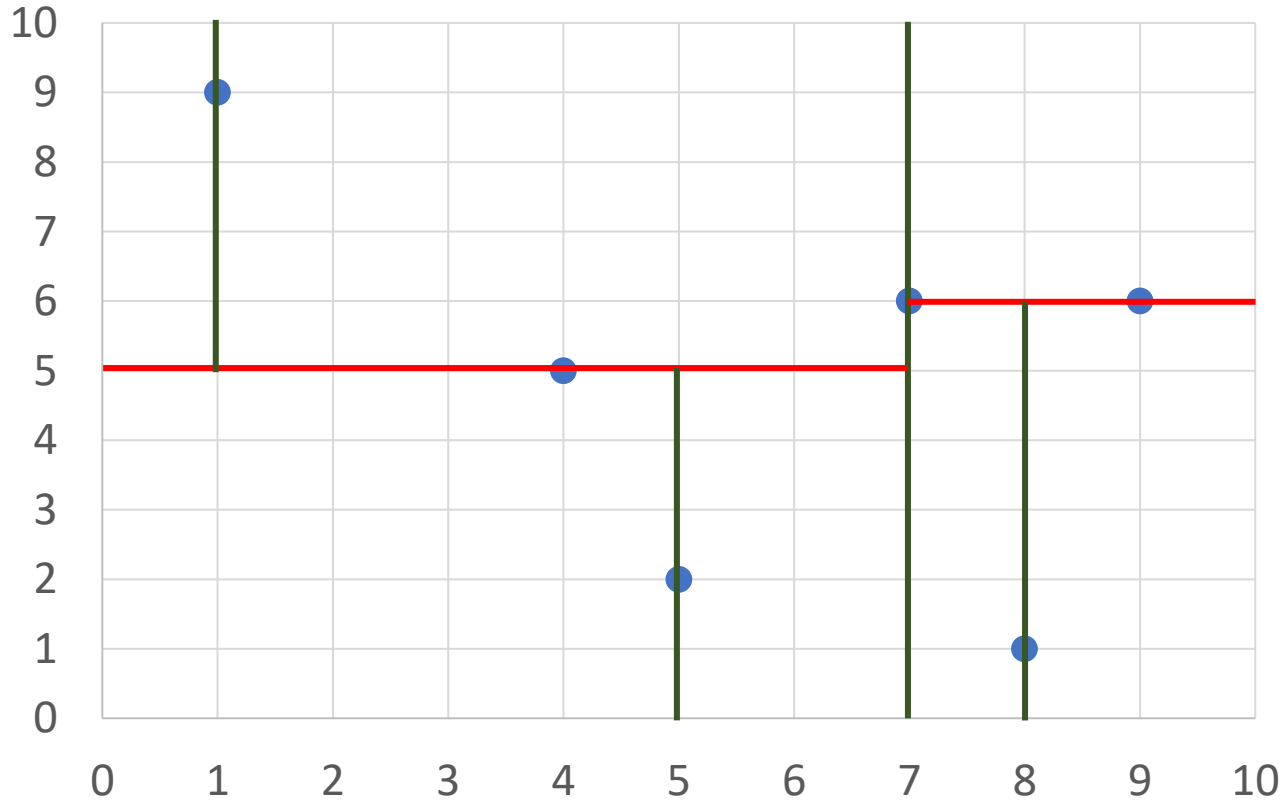
Left side of tree is lower values and right side is higher values

K Dimension Tree

- Split 1 & split 2 further partitioned.
- Splitting of the data continues till each observation centroid is itself.
- Further splitting data by plotting line parallel to y- axis

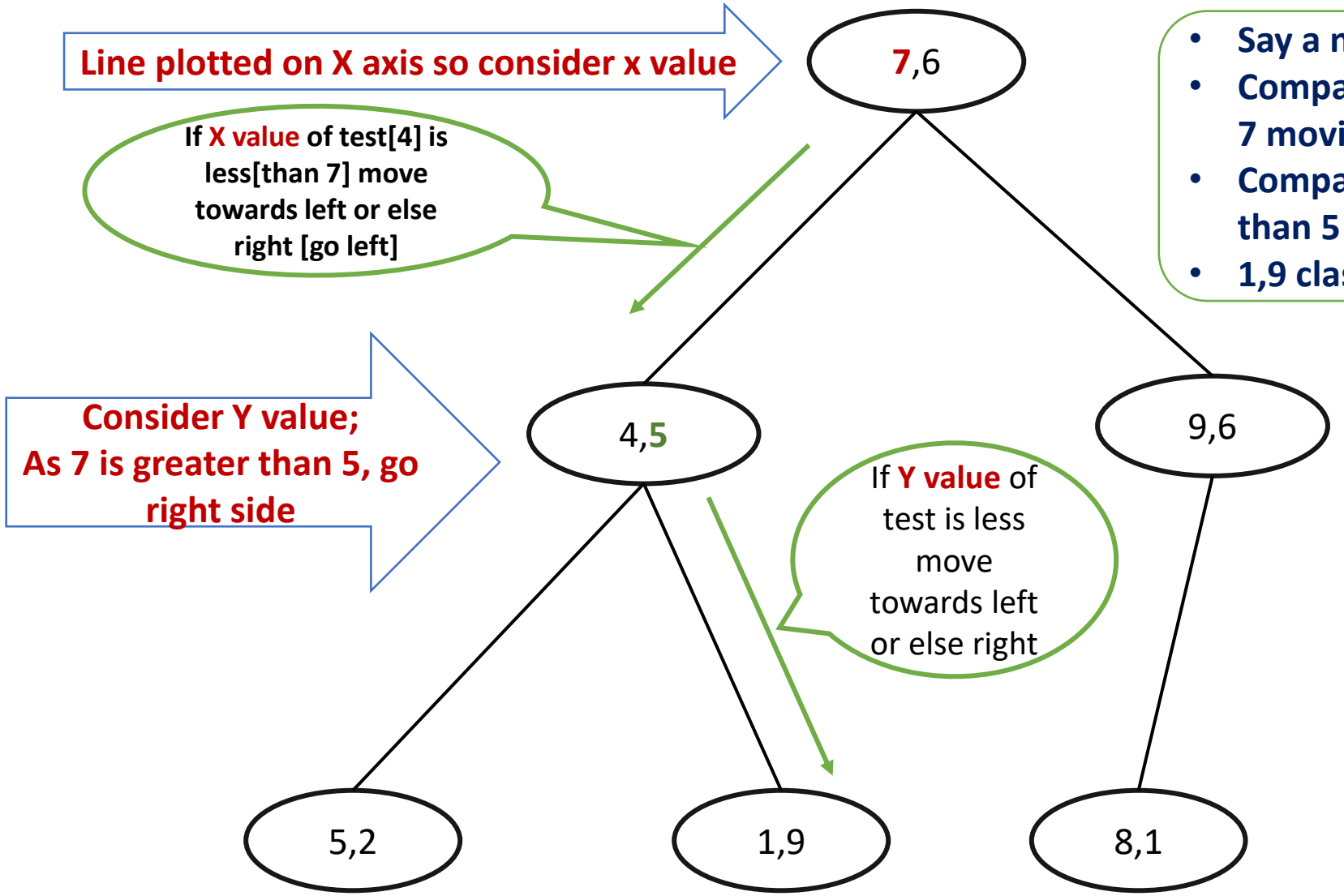
Construction of K D Tree





Left side of tree is lower values and right side is higher values

K Dimension Tree - Prediction



- Say a new value (4,7)
- Comparing X values ie 4 is less than 7 moving towards 4,5
- Comparing Y values ie 7 greater than 5 moving towards right ie 1,9.
- 1,9 classified as 1 therefore 4,7 is 1

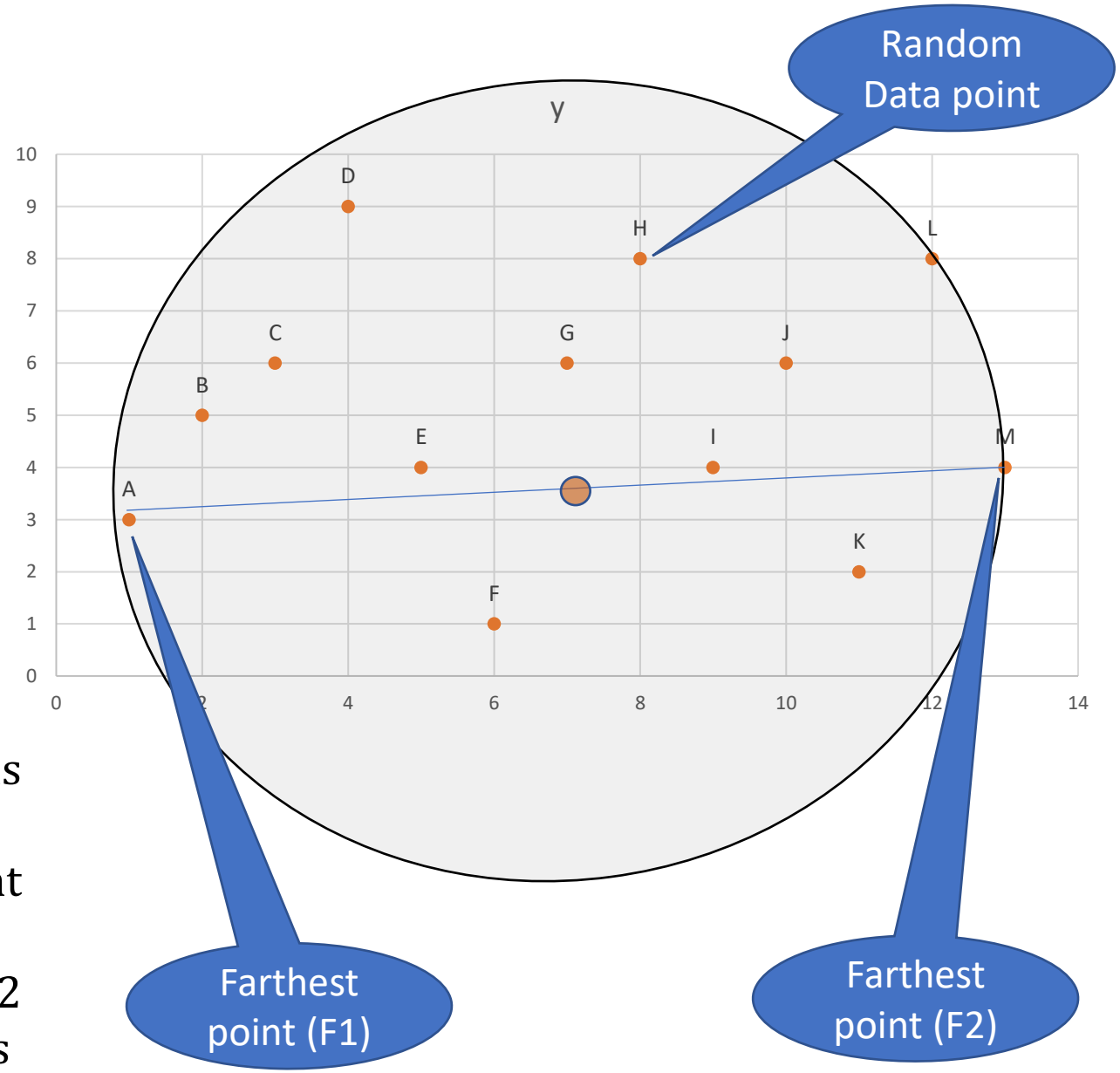
X	Y	Target
9	6	0
7	6	1
8	1	0
1	9	1
4	5	0
5	2	1

Ball Tree

Ball Tree

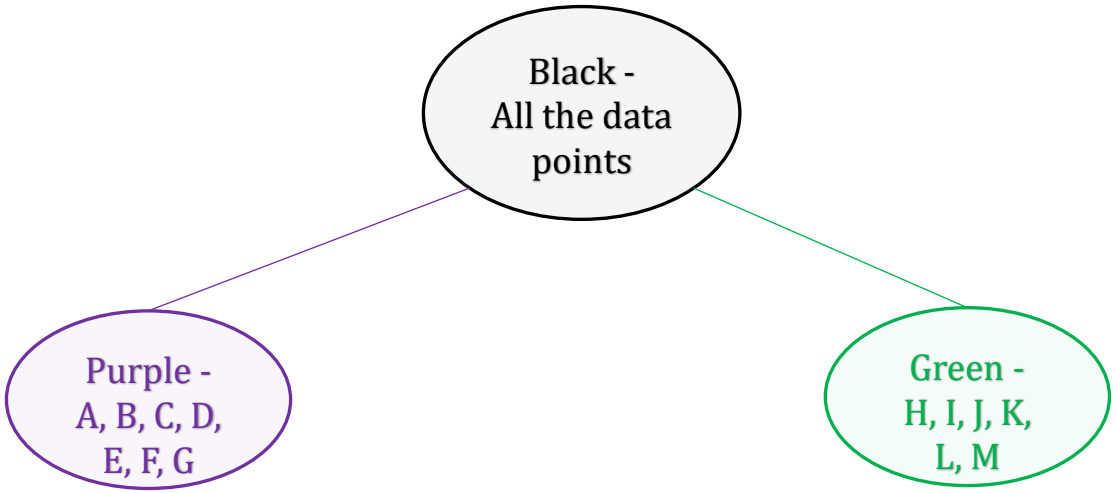
Labels	x	y	z
A	1	3	0
B	2	5	0
C	3	6	0
D	4	9	1
E	5	4	1
F	6	1	1
G	7	6	0
H	8	8	1
I	9	4	1
J	10	6	0
K	11	2	0
L	12	8	1
M	13	4	0

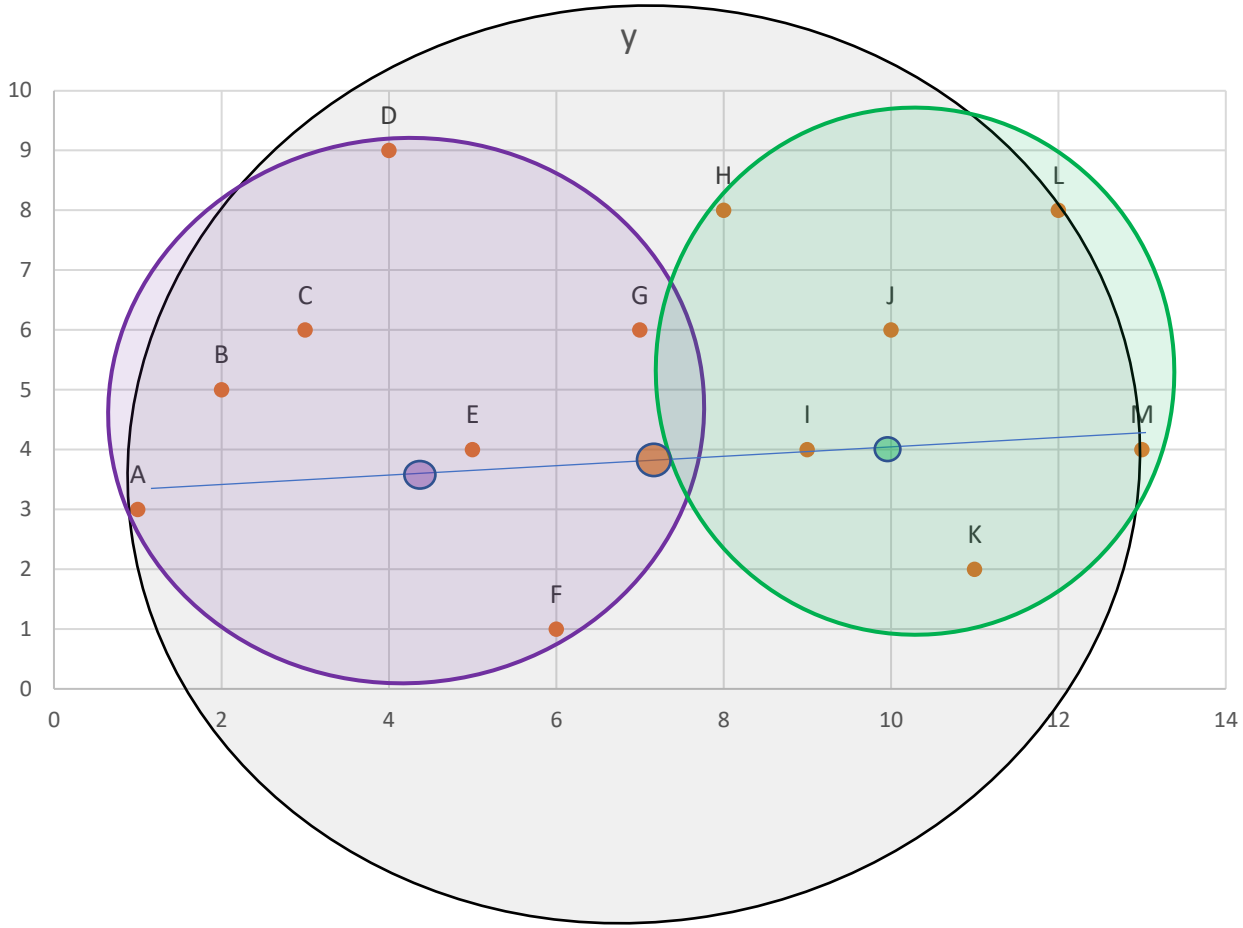
1. Randomly select a data point, Say H
2. Find the farthest data (F1) point from this randomly selected data point (H).
3. From farthest data F1 find the farther data point F2
4. Find the center of these two data points F1 & F2 and draw a circle which covers all the data points



Ball Tree

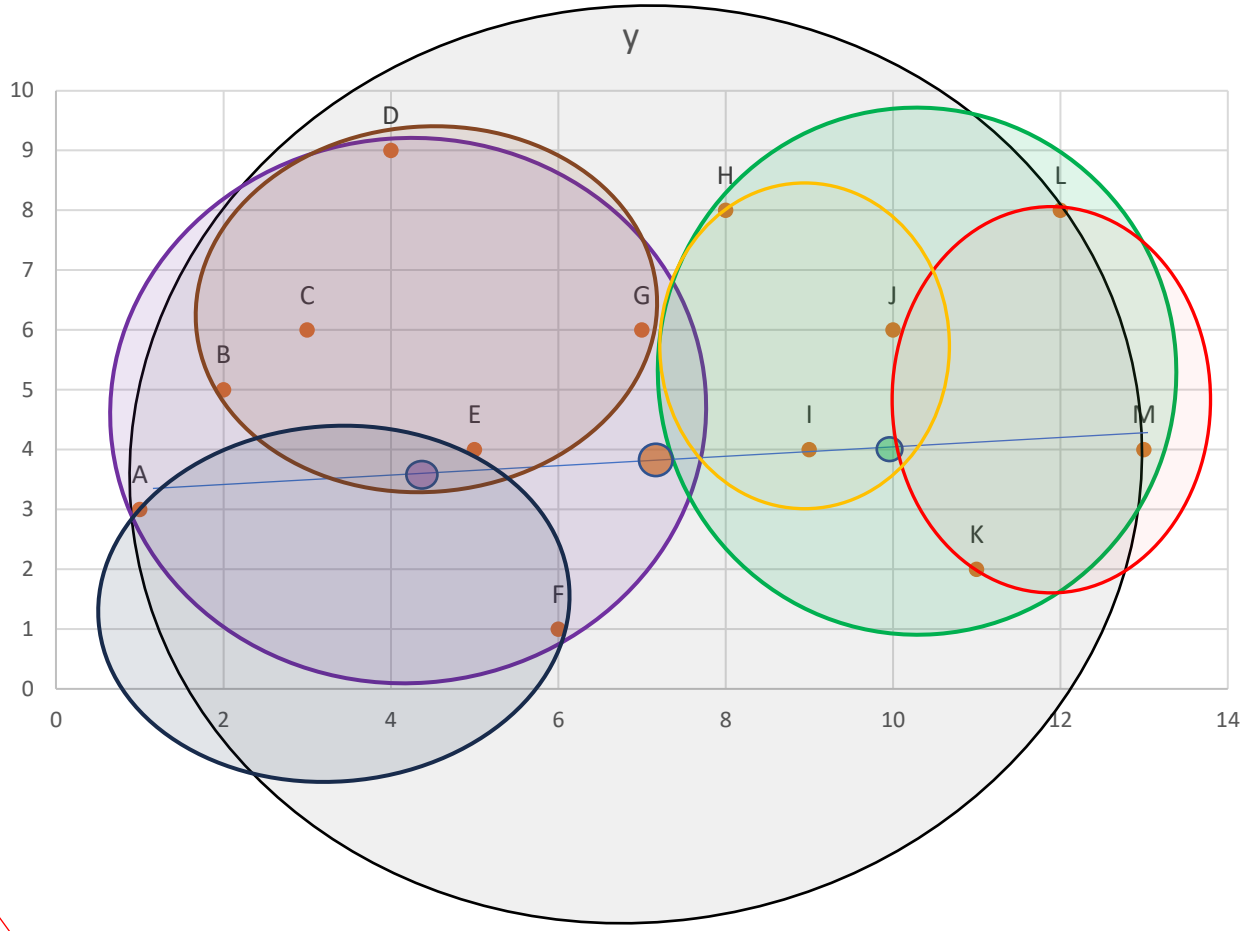
5. Find the centers between center to the farthest point. We get 2 centers
6. Draw 2 circles keeping them as the center of the circles. All the data points should be part of any one of these circles.

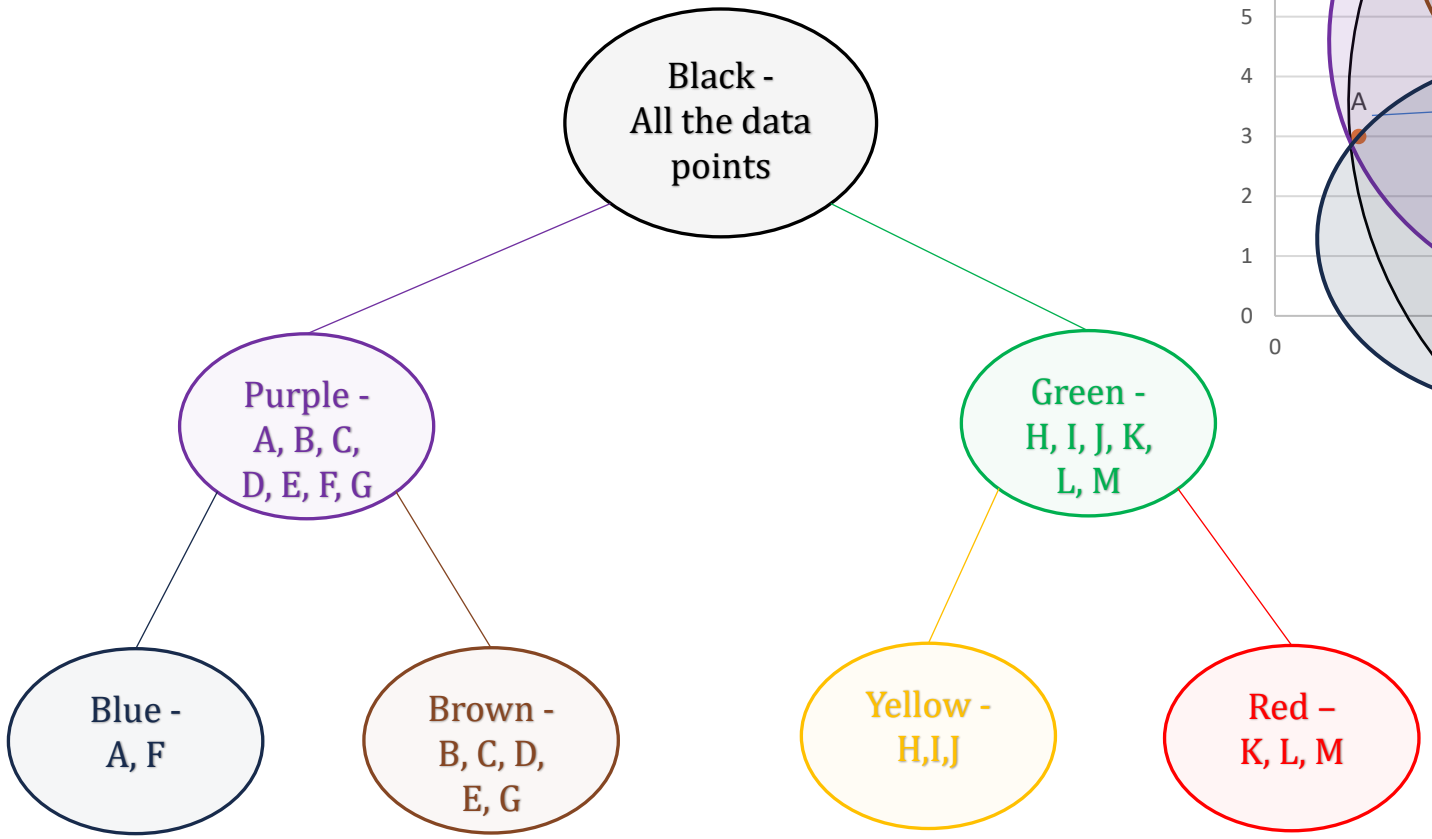




Ball Tree

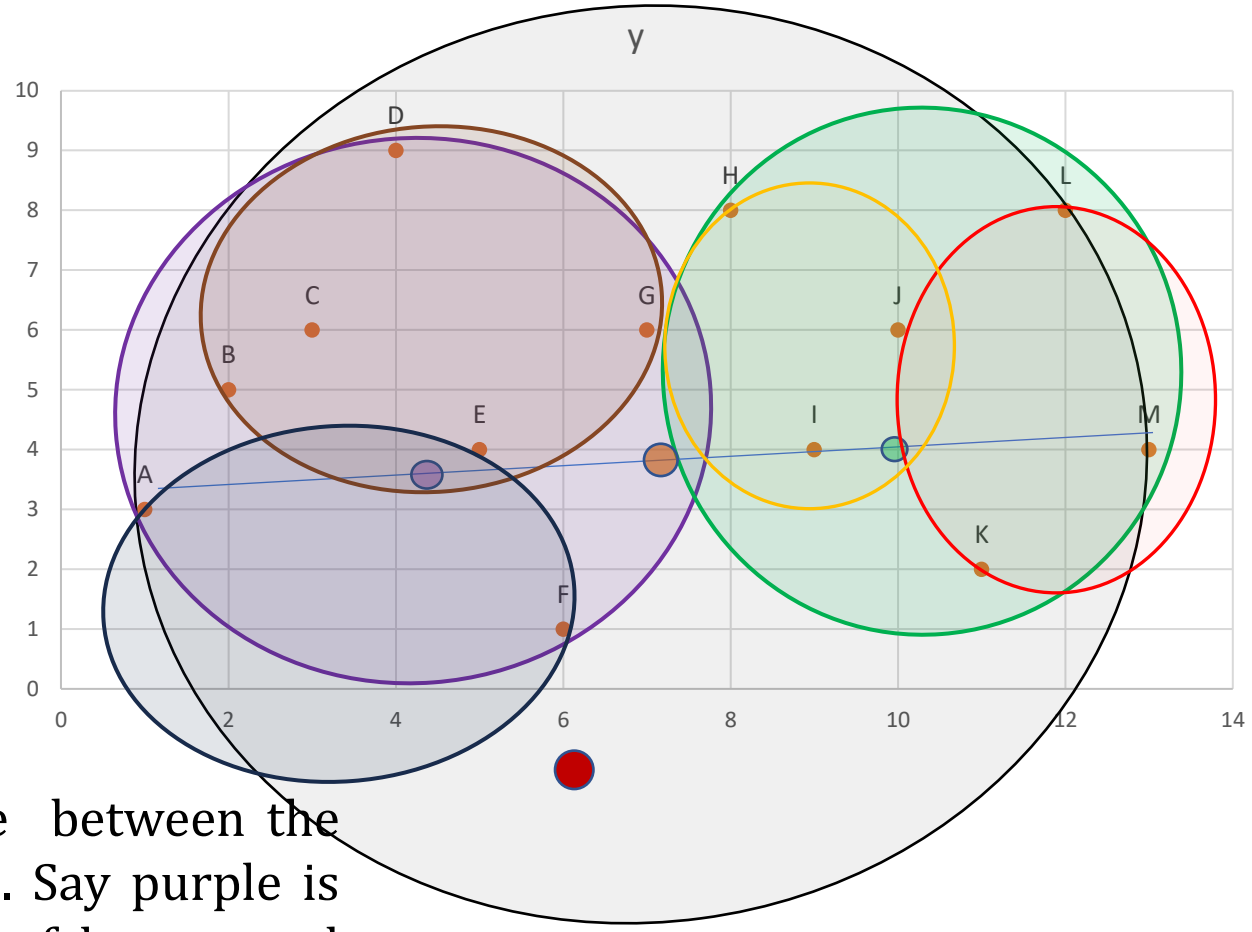
7. From the center of the purple circle find the farther point, say D. Find the center between these 2 points and draw a circle.
8. Find the farthest point from the remaining data points and find its center and draw a circle.
9. Similarly in green circle





Ball Tree – Prediction

Labels	x	y	z
A	1	3	0
B	2	5	0
C	3	6	0
D	4	9	1
E	5	4	1
F	6	1	1
G	7	6	0
H	8	8	1
I	9	4	1
J	10	6	0
K	11	2	0
L	12	8	1
M	13	4	0



A new data point enters, it calculates the distance between the point and centers of purple and green color circle. Say purple is near, then it calculates distance between centers of brown and blue. Say blue is near, in that circle, whatever the maximum category that will be allocated to this new data point. If k is specified it pics the k number of nearest data point and whatever category is max that will be allocated to this new data point

Brute Force

Brute Force

1. Brute force method — calculate distance from new point to every point in training data matrix X, sort distances and take k nearest, then do a majority vote. There is no need for separate training, so we only consider prediction complexity.