Say you have a large amount of data

You want to explore it and see if any patterns emerge

One way to do this is using a technique called

Clustering

Clustering

Anything

and I mean Anything in the world

can be described using
a set of numbers

Clustering

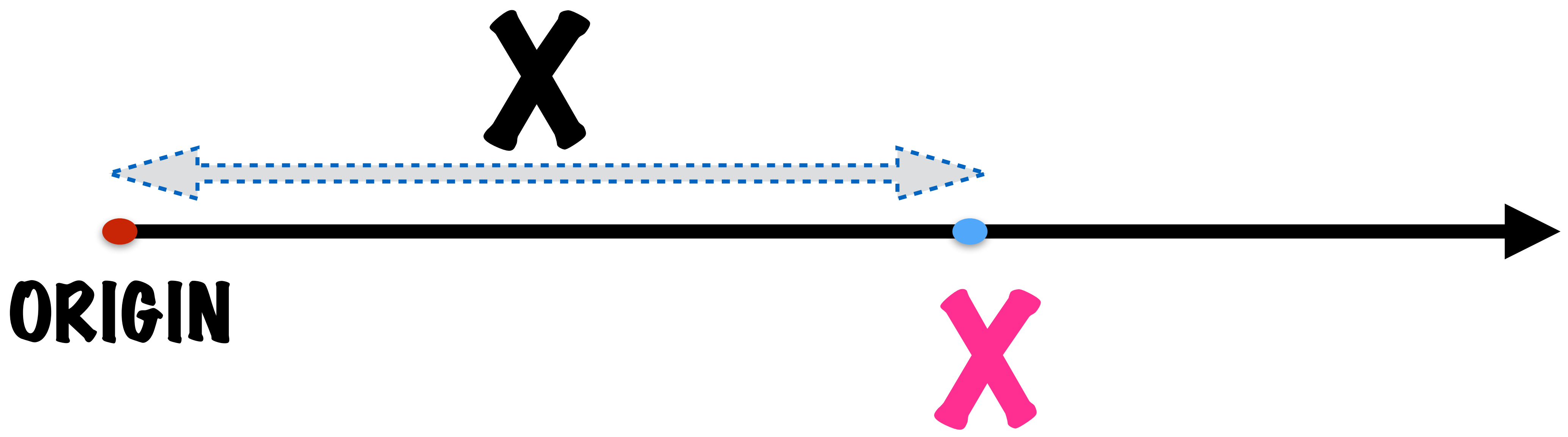
A person	(Age, Height, Weight)
A sale	(Time, Amount, Product Id)
A webpage	(Length , Frequencies of words)

Clustering

A set of N numbers represents a point in an N-Dimensional Hypercube

N-Dimensional Hypercube

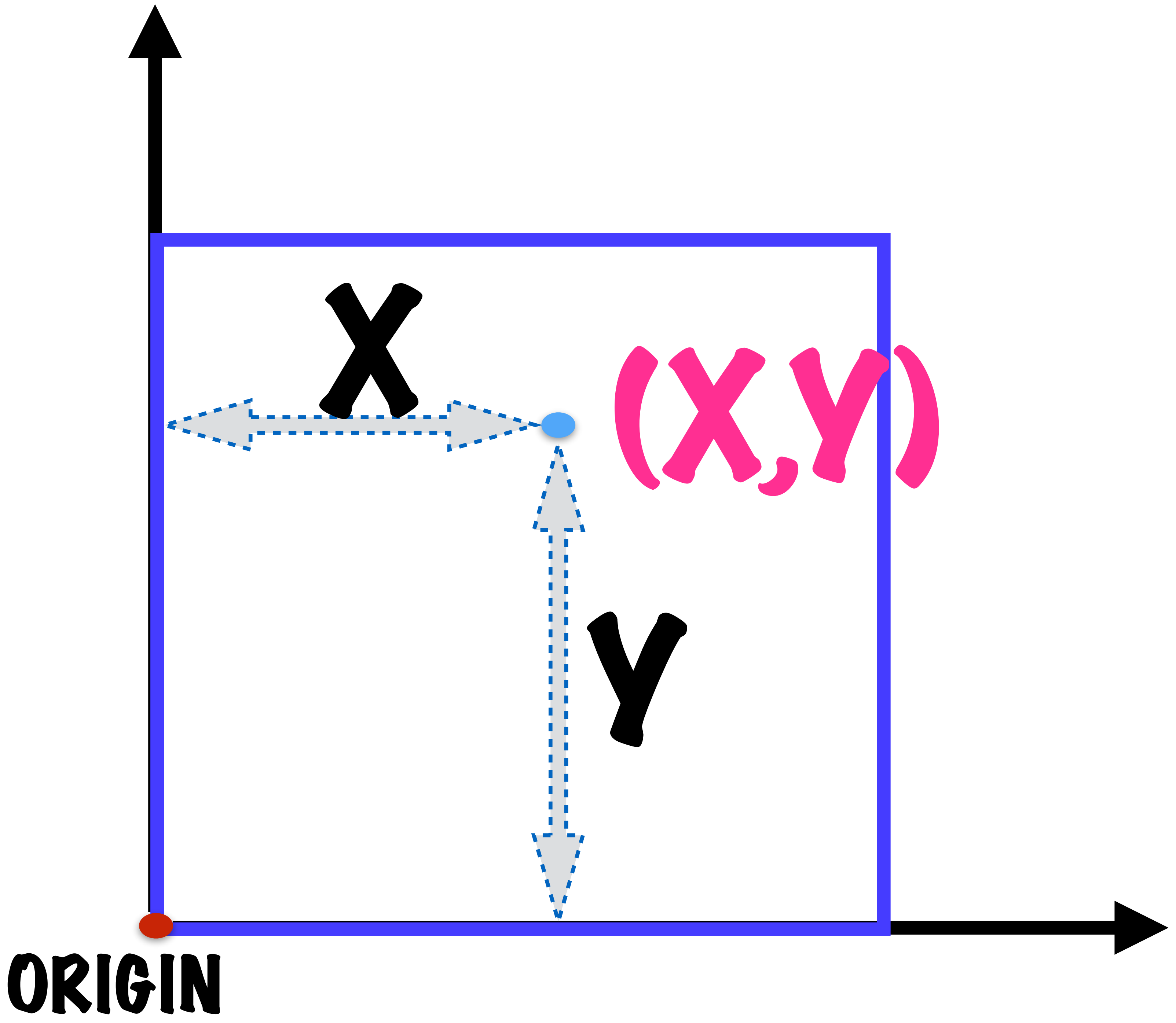
A LINE IS A 1 DIMENSIONAL SHAPE



Any point on a line can be represented using 1 number

N-Dimensional Hypercube

A SQUARE IS A 2 DIMENSIONAL SHAPE

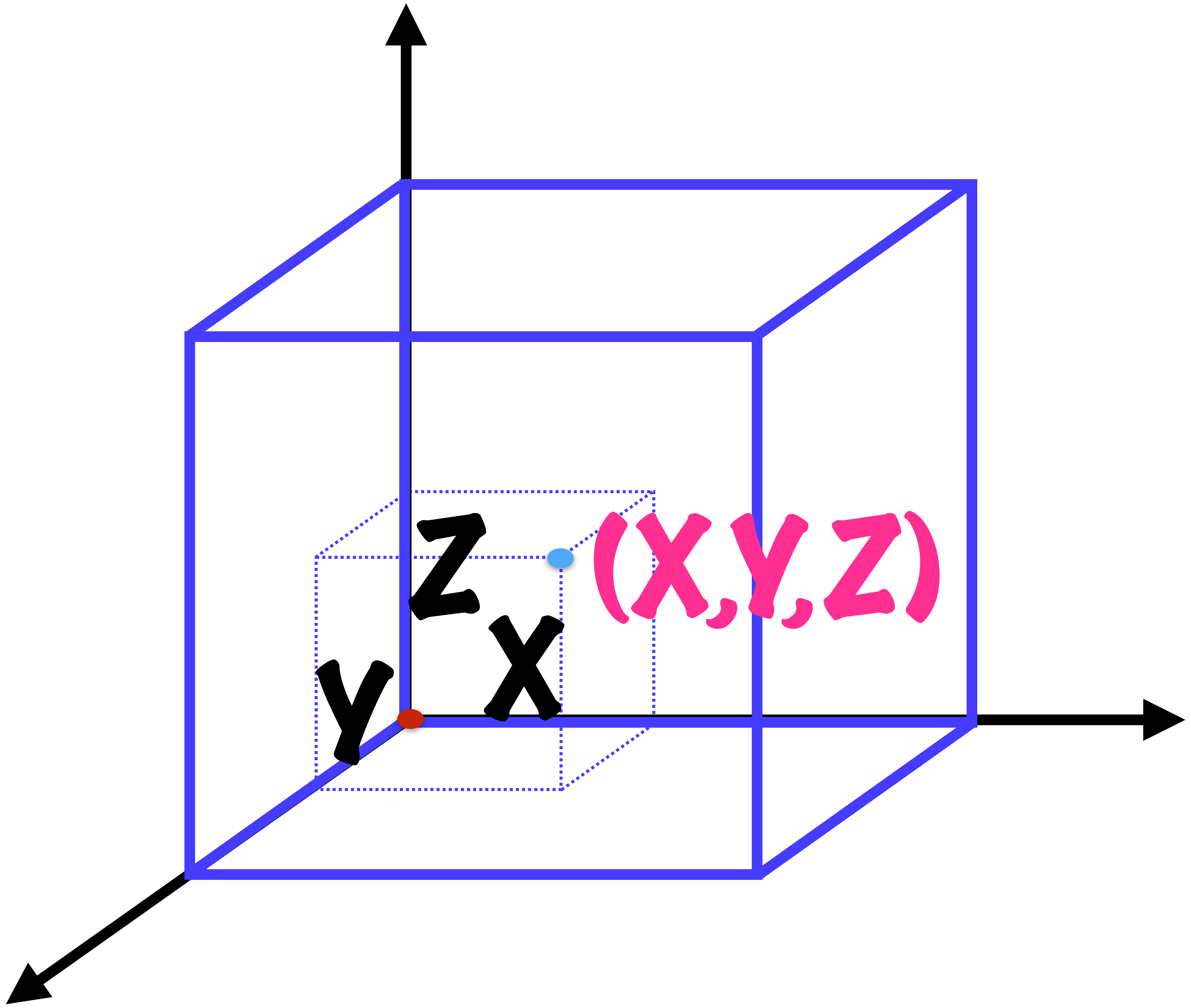


Any point in a square can be represented using 2 numbers

N-Dimensional Hypercube

A CUBE IS A 3 DIMENSIONAL SHAPE

Any point in a cube can be represented with 3 numbers



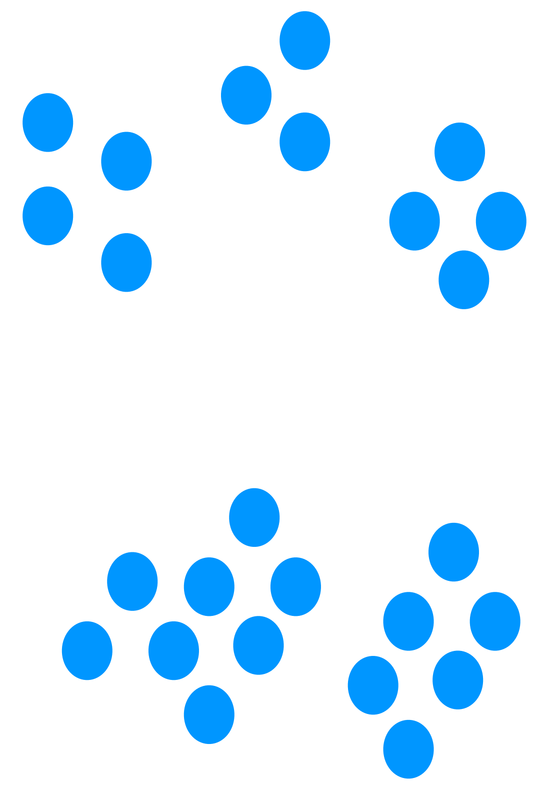
Clustering

A set of N numbers represents a point in an N-Dimensional Hypercube

This is just to say that any data in the world can be represented as points in an N-Dimensional space

Clustering

THE OBJECTIVE OF CLUSTERING IS TO DIVIDE UP THESE USERS INTO GROUPS



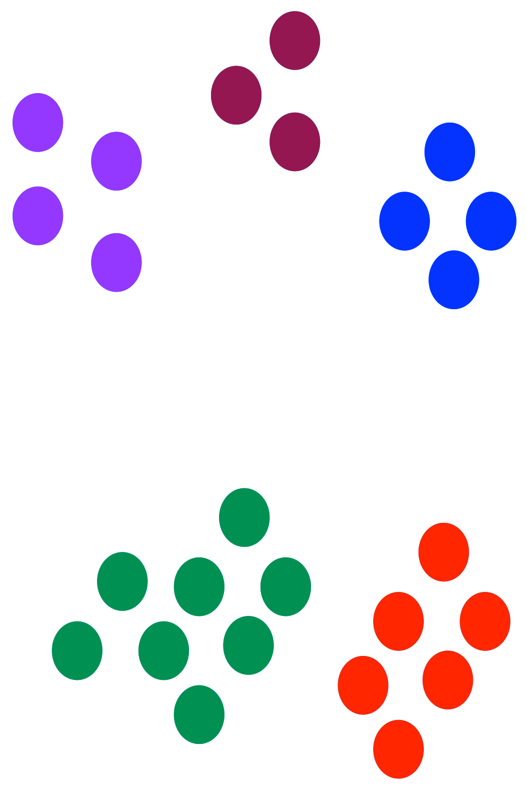
Here is a set of points - each representing a Facebook User

USERS IN A GROUP ARE SIMILAR TO ONE ANOTHER

USERS FROM DIFFERENT GROUPS ARE VERY DIFFERENT FROM EACH OTHER

Clustering

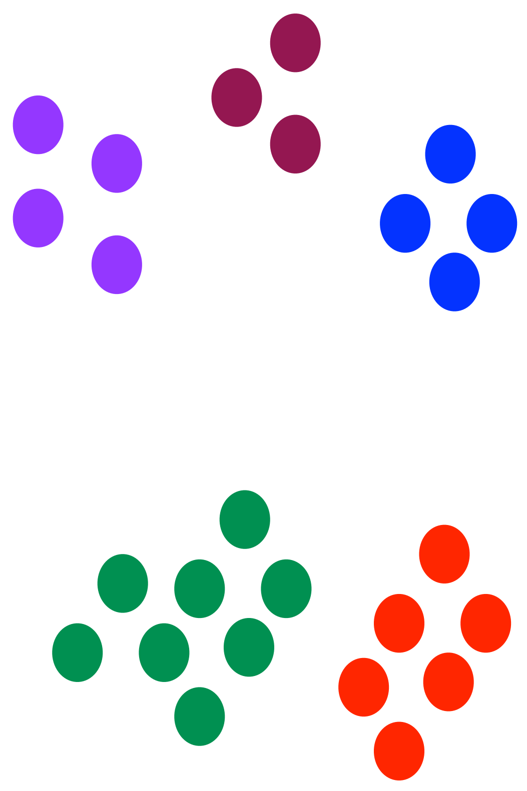
HERE'S ONE WAY OF DOING THIS..

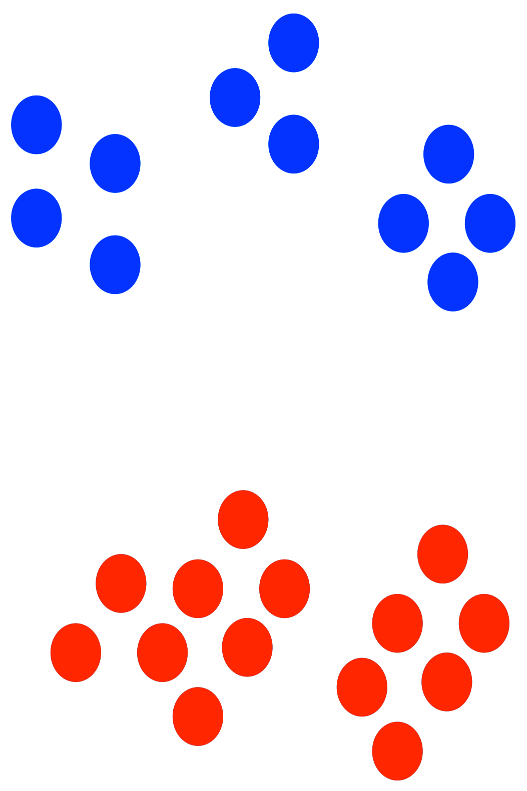


USERS IN A GROUP ARE SIMILAR TO ONE ANOTHER

USERS FROM DIFFERENT GROUPS ARE VERY DIFFERENT FROM EACH OTHER

Clustering



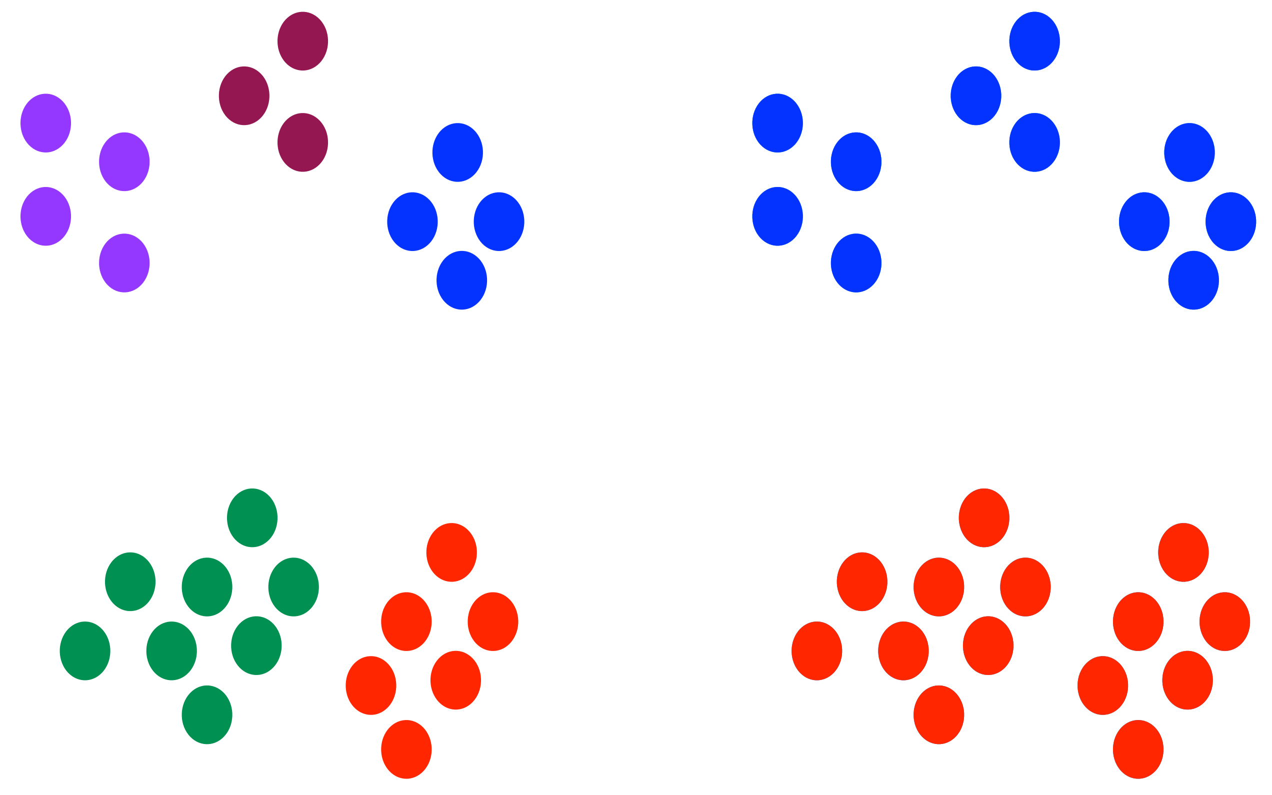


HERE'S ANOTHER WAY ..

USERS IN A GROUP
ARE SIMILAR
TO ONE ANOTHER

USERS FROM DIFFERENT
GROUPS ARE
VERY DIFFERENT FROM
EACH OTHER

Clustering



IN REAL LIFE THIS DISTANCE MIGHT TRANSLATE TO SOMETHING MEANINGFUL

1. MAYBE THEY LIKE/FOLLOW THE SAME THINGS
2. MAYBE THEY ARE FROM THE SAME STATE
3. OR BOTH OF THE ABOVE

BASICALLY, THE DISTANCE BETWEEN THE POINTS IS USED TO INDICATE THE SIMILARITY BETWEEN USERS

Clustering

THE CLUSTERS SHOULD BE SUCH THAT WE

USERS IN A GROUP ARE SIMILAR TO ONE ANOTHER

USERS FROM DIFFERENT GROUPS ARE VERY DIFFERENT FROM EACH OTHER

MAXIMIZE INTRACLUSTER SIMILARITY

MINIMIZE INTERCLUSTER SIMILARITY

Clustering

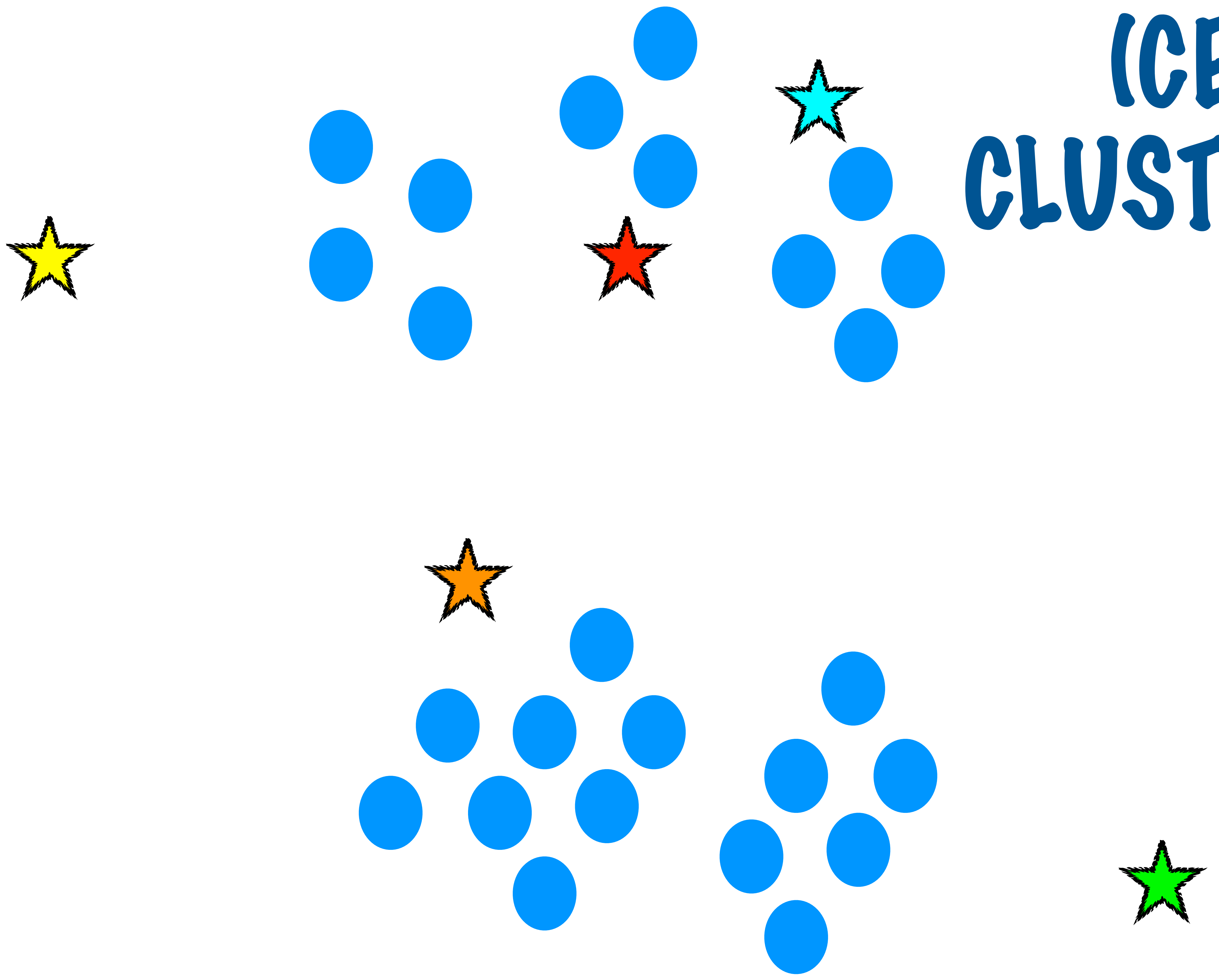
MAXIMIZE INTRACLUSTER SIMILARITY

MINIMIZE INTERCLUSTER SIMILARITY

K-MEANS CLUSTERING IS A VERY FAMOUS ALGORITHM TO ACHIEVE EXACTLY THIS

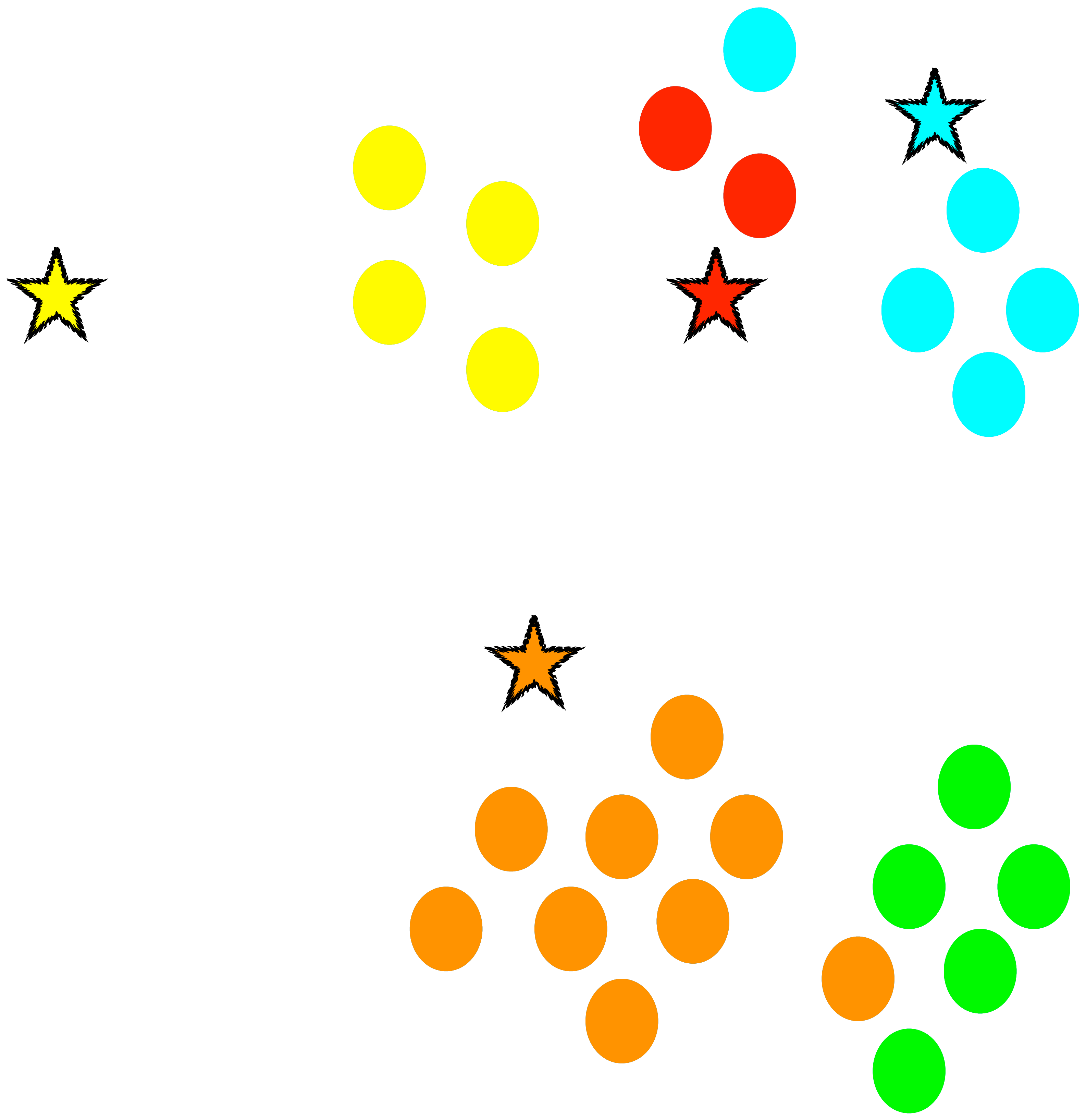
K-MEANS CLUSTERING

1 . INITIALIZE A SET OF POINTS AS THE "K" MEANS (CENTROIDS OF THE CLUSTERS YOU WANT TO FIND)



YOU WANT TO DIVIDE THIS DATA INTO K CLUSTERS

K-MEANS CLUSTERING



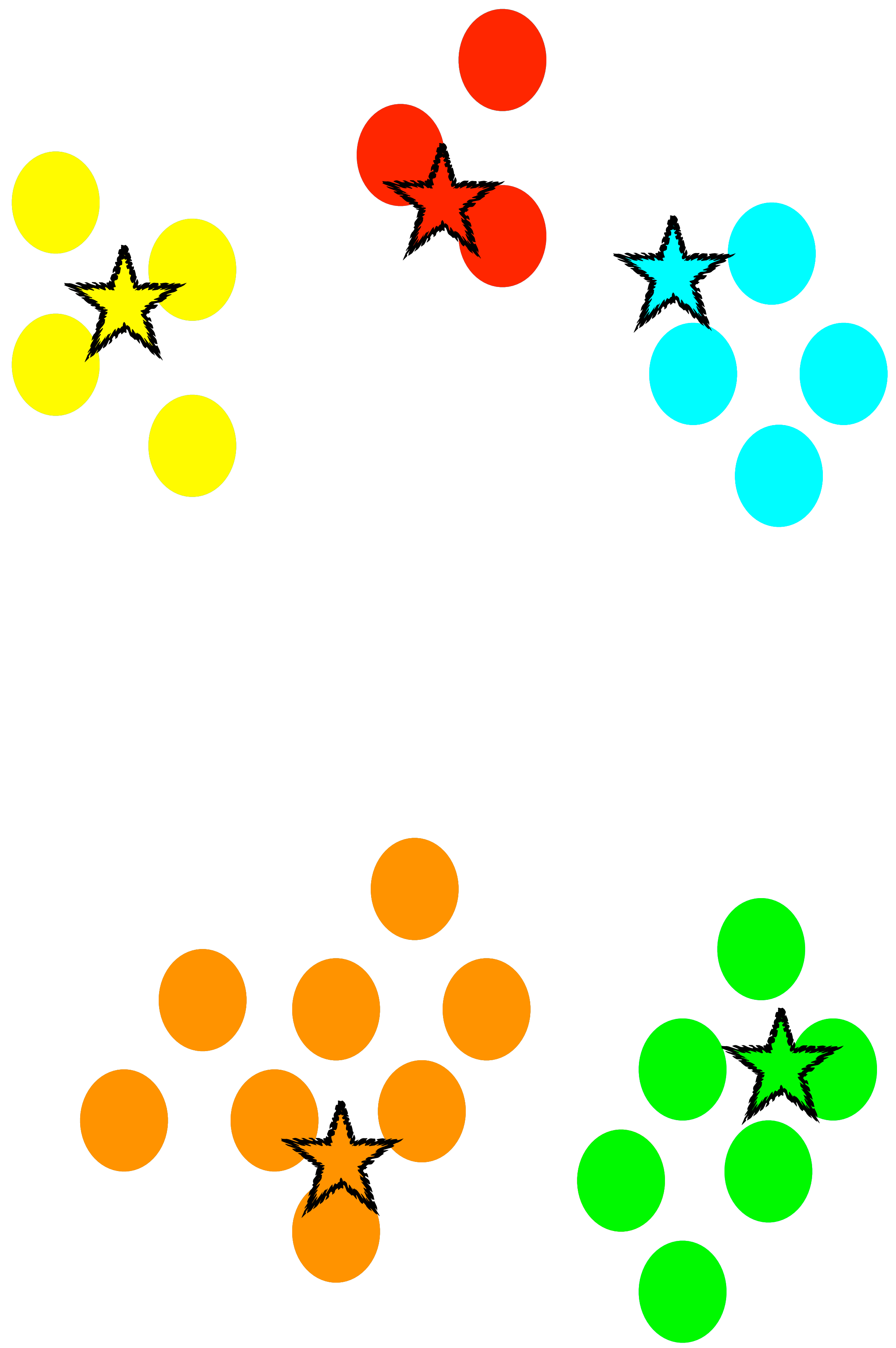
1 . INITIALIZE A SET OF POINTS AS THE "K" MEANS (CENTROIDS OF THE CLUSTERS YOU WANT TO FIND)

2 . ASSIGN EACH POINT TO THE CLUSTER BELONGING TO THE NEAREST MEAN

3 . FIND THE NEW MEANS /CENTROIDS OF THE CLUSTERS

K-MEANS CLUSTERING

1 . INITIALIZE A SET OF POINTS AS THE "K" MEANS (CENTROIDS OF THE CLUSTERS YOU WANT TO FIND)



2 . ASSIGN EACH POINT TO THE CLUSTER BELONGING TO THE NEAREST MEAN

3 . FIND THE NEW MEANS / CENTROIDS OF THE CLUSTERS

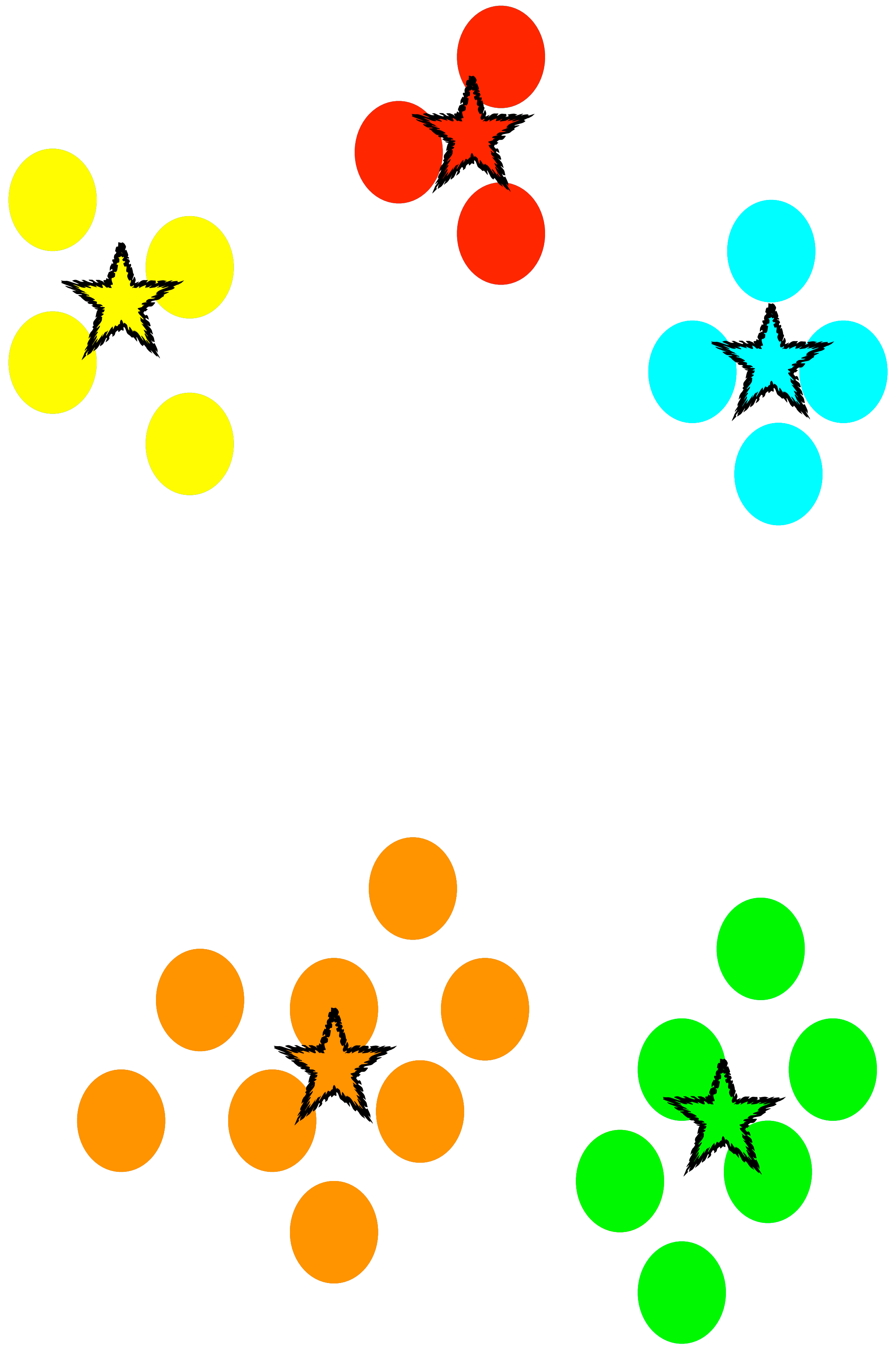
RINSE AND REPEAT STEPS 2, 3 UNTIL THE MEANS DON'T CHANGE ANY MORE

K-MEANS CLUSTERING

1 . INITIALIZE A SET OF POINTS AS THE "K" MEANS (CENTROIDS OF THE CLUSTERS YOU WANT TO FIND)

CONVERGENCE

RINSE AND REPEAT
STEPS 2, 3 UNTIL THE MEANS DON'T
CHANGE ANY MORE



2 . ASSIGN EACH POINT TO THE CLUSTER BELONGING TO THE NEAREST MEAN

3 . FIND THE NEW MEANS / CENTROIDS OF THE CLUSTERS

K-MEANS CLUSTERING

Let's see how to achieve this using MapReduce

K-MEANS CLUSTERING

One MapReduce job will run for each iteration that we just described

K-MEANS CLUSTERING

There are a few questions we need to answer

1) What is the input and output of each iteration?

2) What are the Map and Reduce transformations in an iteration?

3) How do we specify the stopping condition?

K-MEANS CLUSTERING

1) What is the input and output of each iteration?

In each iteration 1 MapReduce job will run

Input

1) The current set of cluster centers

2) All the data points

Output

1) A new set of cluster centers

2) Mapping of data point to Cluster center

K-MEANS CLUSTERING

There are a few questions we need to answer

1) What is the input and output of each iteration?

2) What are the Map and Reduce transformations in an iteration?

3) How do we specify the stopping condition?

K-MEANS CLUSTERING

There are a few questions we need to answer

1) What is the input and output of each iteration?

2) What are the Map and Reduce transformations in an iteration?

3) How do we specify the stopping condition?

K-MEANS CLUSTERING

Map and Reduce

Input

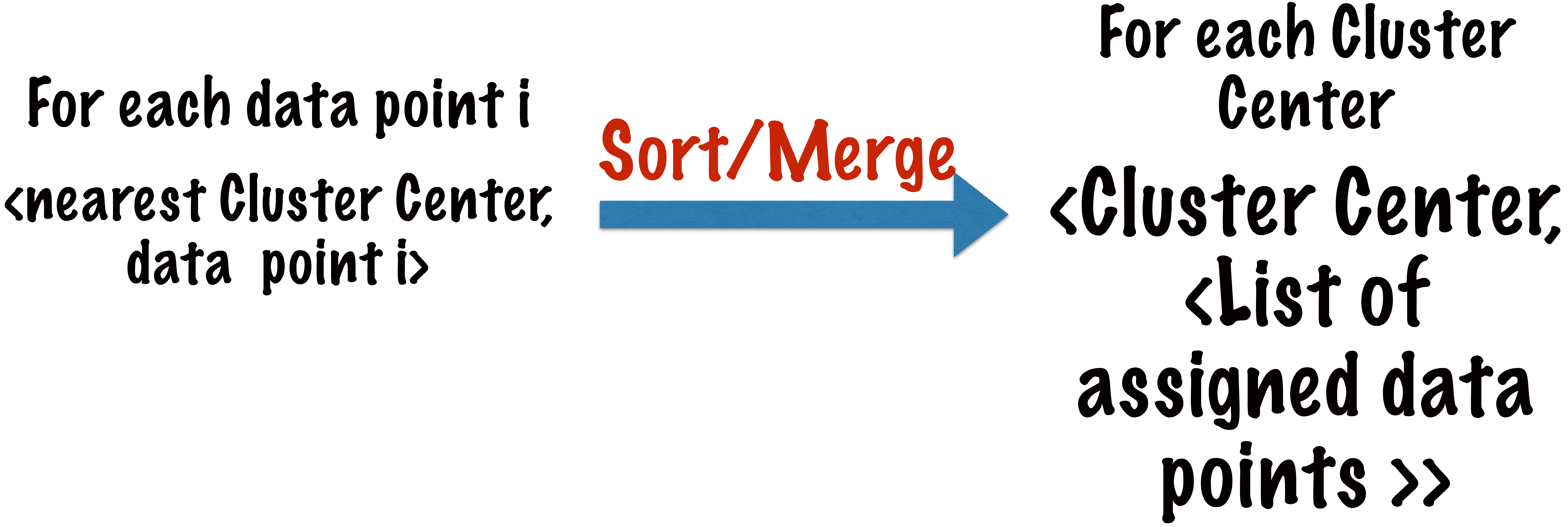
1) The current set of cluster centers

2) All the data points

For each data point i

<nearest Cluster Center, data point i>

Map and Reduce



For each data point i

<nearest Cluster Center, data point i>

Sort/Merge

For each Cluster Center

<Cluster Center, <List of assigned data points >>

K-MEANS CLUSTERING

Map and Reduce

For each Cluster Center

<Cluster Center, <List of assigned data points >>

Reduce

For each data point

<New Cluster Center, <Data Point>>

K-MEANS CLUSTERING

There are a few questions we need to answer

1) What is the input and output of each iteration?

2) What are the Map and Reduce transformations in an iteration?

3) How do we specify the stopping condition?

K-MEANS CLUSTERING

There are a few questions we need to answer

1) What is the input and output of each iteration?

2) What are the Map and Reduce transformations in an iteration?

3) How do we specify the stopping condition?

MapReduce Counters

MapReduce allows users to define Counters that are set by the Mapper / Reducer

MapReduce Counters

The value of this counter at any time is accessible to the Job

MapReduce Counters

In each iteration, if the

new Cluster centers <> old Cluster centers

The Reducer can

increment a counter

MapReduce Counters

The Job will keep checking this to say if it wants to move on to another iteration or not