

CPSC 340: Machine Learning and Data Mining

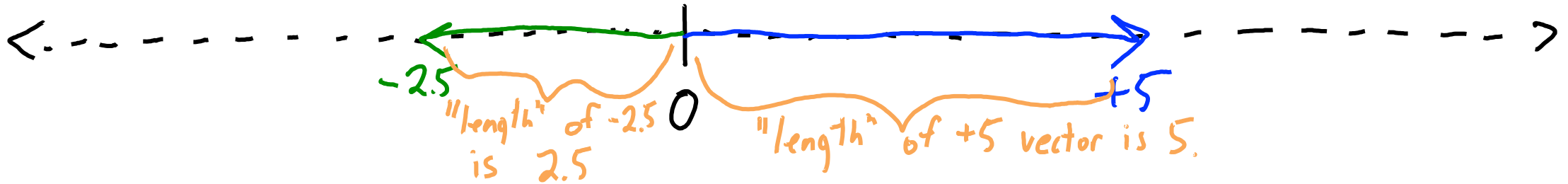
Density-Based Clustering

Admin

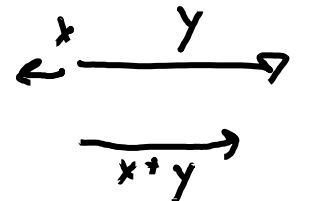
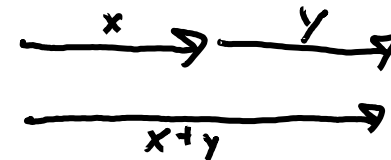
- **Assignment 1**
 - Due last night
 - Solutions coming Thursday
- Assignment 2: coming soon.
 - [If you want to work with a partner](#), open the issue ASAP (ideally today!)
- Private posts
 - Disabled to avoid repeated questions
 - But you need a way of asking about code
 - For hw2 onwards, open an issue in your hw repo
 - Make sure to tag @cpsc340/staff
 - We can't guarantee we'll get to all of them, depending on volume

Norms in 1-Dimension

- We can view absolute value, $|x|$, as 'size' or 'length' of a number:



- It satisfies three intuitive properties of 'length':
 1. Only '0' has a 'length' of zero.
 2. Multiplying 'x' by constant ' α ' multiplies length by $|\alpha|$:
 - "Absolute homogeneity": $|\alpha x| = |\alpha| |x|$.
 - "If will twice as long if you multiply by 2".
 3. Length of ' $x+y$ ' is not more than length of ' x ' plus length of ' y ':
 - "Triangle" inequality: $|x + y| \leq |x| + |y|$.
 - Think of "how far you travel".

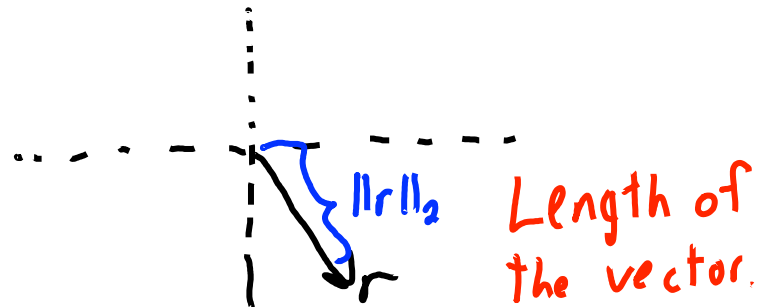


Norms in 2-Dimensions

- In 1-dimension, **only scaled absolute values** satisfy the 3 properties.
- In 2-dimensions, there is **no unique function** satisfying them.
- We call any function satisfying them a **norm**:
 - Measures of “size” or “length” in 2-dimensions.
- Three most common examples:

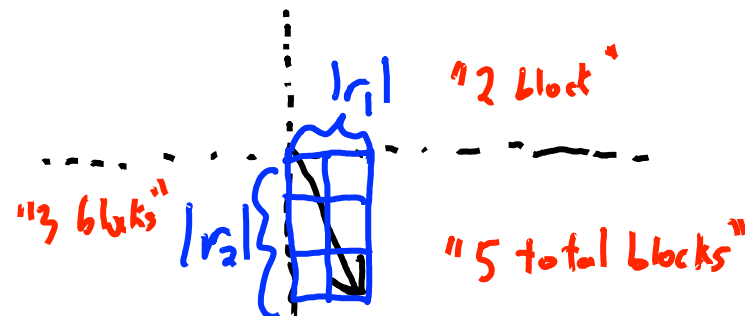
L_2 or “Euclidean” norm.

$$\|r\|_2 = \sqrt{r_1^2 + r_2^2}$$



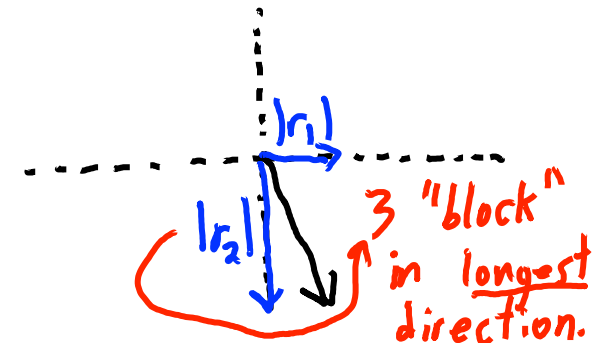
L_1 or “Manhattan” norm:

$$\|r\|_1 = |r_1| + |r_2|$$



L_∞ or “max” norm:

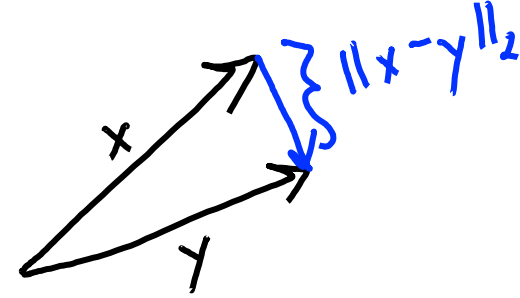
$$\|r\|_\infty = \max\{|r_1|, |r_2|\}$$



Norms as Measures of Distance

- By taking norm of difference, we get a “distance” between vectors:

$$\|x - y\|_2 = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}$$



$$\|x - y\|_1 = |x_1 - y_1| + |x_2 - y_2|$$

"Number of blocks you need to walk to get from x to y."

$$\|x - y\|_\infty = \max \{ |x_1 - y_1|, |x_2 - y_2| \}$$

"Most number of blocks in any direction you would have to walk."

Norms in d-Dimensions

- We can generalize these common norms to d-dimensional vectors:

$$L_2: \|r\|_2 = \sqrt{\sum_{j=1}^d r_j^2}$$

$$L_1: \|r\|_1 = \sum_{j=1}^d |r_j|$$

$$L_\infty: \max_j \{ |r_j| \}$$

E.g., in 3-dimensions:

$$\|r\|_2 = \sqrt{r_1^2 + r_2^2 + r_3^2}$$

in 4-dimensions:

$$\|r\|_2 = \sqrt{r_1^2 + r_2^2 + r_3^2 + r_4^2}$$

- These norms place different “weights” on large values:
 - L_1 : all values are equal.
 - L_2 : bigger values are more important (because of squaring).
 - L_∞ : only biggest value is important.

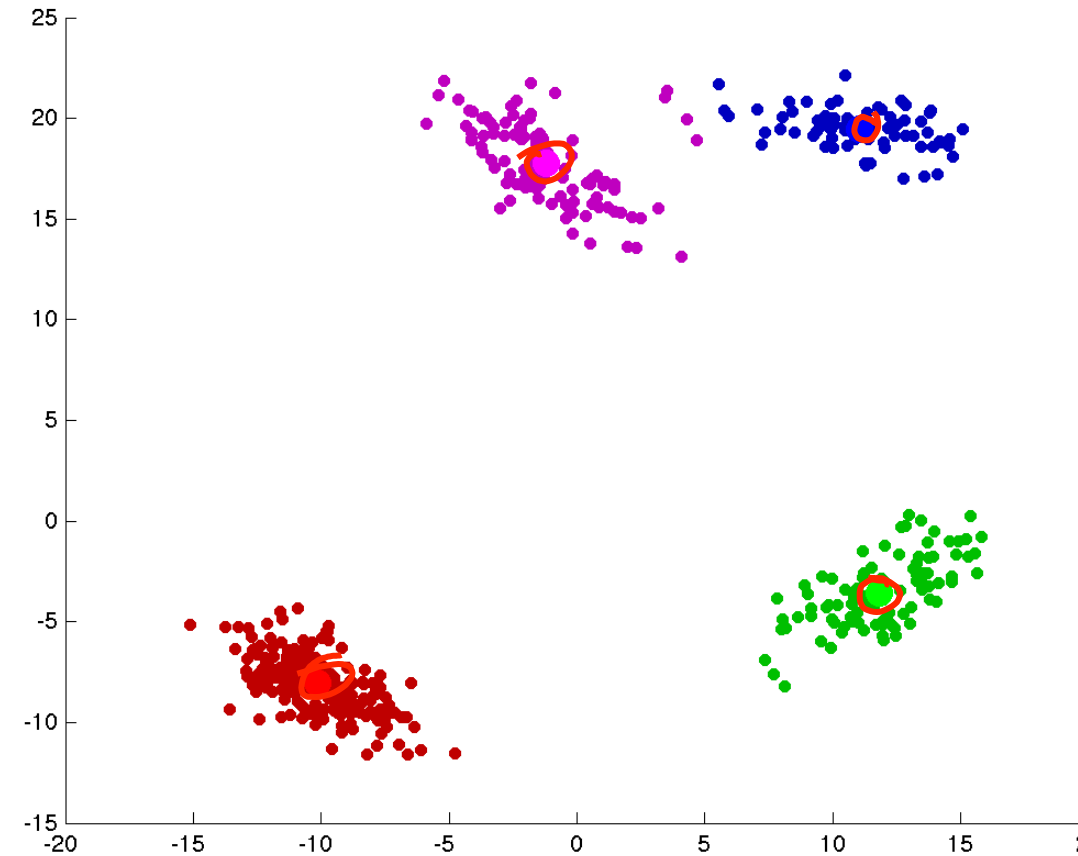
Notation: $\|r\|_2^2 = (\|r\|_2)^2$

$$= \left(\sqrt{\sum_{j=1}^d r_j^2} \right)^2$$
$$= \sum_{j=1}^d r_j^2$$
$$= r^T r$$

Different ways
to write the
same thing

Last Time: K-Means Clustering

- We want to **cluster** data:
 - Assign objects to groups.
- **K-means clustering**:
 - Define groups by “means”
 - Assign objects to nearest mean.
(Then update means during training.)
- Also used for **vector quantization**:
 - Use **means as prototypes** of groups.



K-Means Initialization

- K-means is fast but **sensitive to initialization**.
- Classic approach to initialization: **random restarts**.
 - Run to convergence using different random initializations.
 - Choose the one that minimizes average squared distance of data to means.
- Newer approach: **k-means++**
 - Random initialization that **prefers means that are far apart**.
 - Yields **provable bounds** on expected approximation ratio.

K-Means++

- Steps of **k-means++**:

1. Select initial mean w_1 as a **random** x_i .

2. **Compute distance** d_{ic} of each object x_i to each mean w_c .

$$d_{ic} = \sqrt{\sum_{j=1}^d (x_{ij} - w_{cj})^2} = \|x_i - w_c\|_2$$

3. For each object 'i' **set d_i to the distance to the closest mean.**

$$d_i = \min_c \{d_{ic}\}$$

4. Choose next mean by **sampling an example 'i' proportional to $(d_i)^2$.**

$$p_i \propto d_i^2 \Rightarrow p_i = \frac{d_i^2}{\sum_{j=1}^n d_j^2}$$

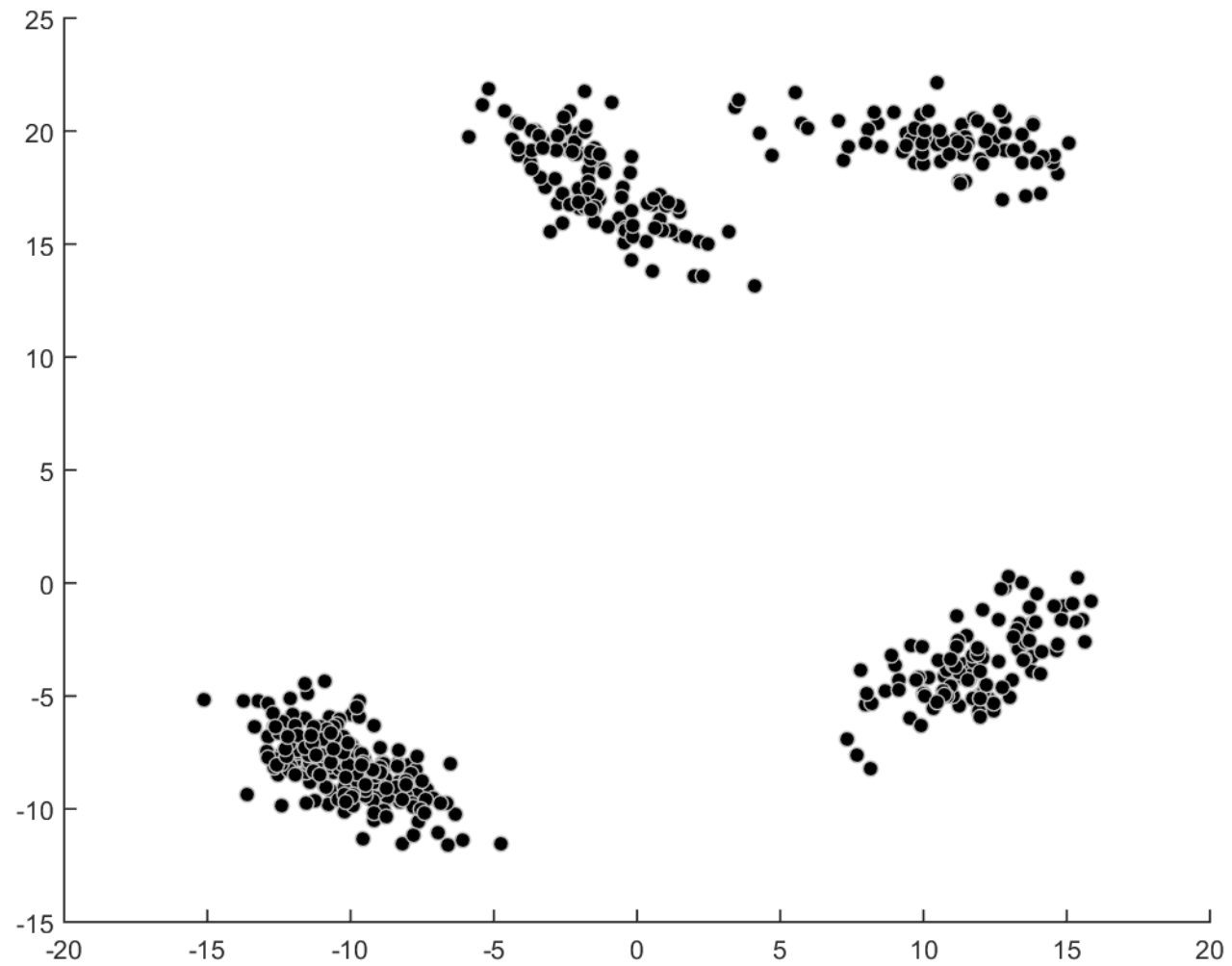
5. Keep returning to step 2 until we have k-means.

- Expected approximation ratio is $O(\log(k))$.

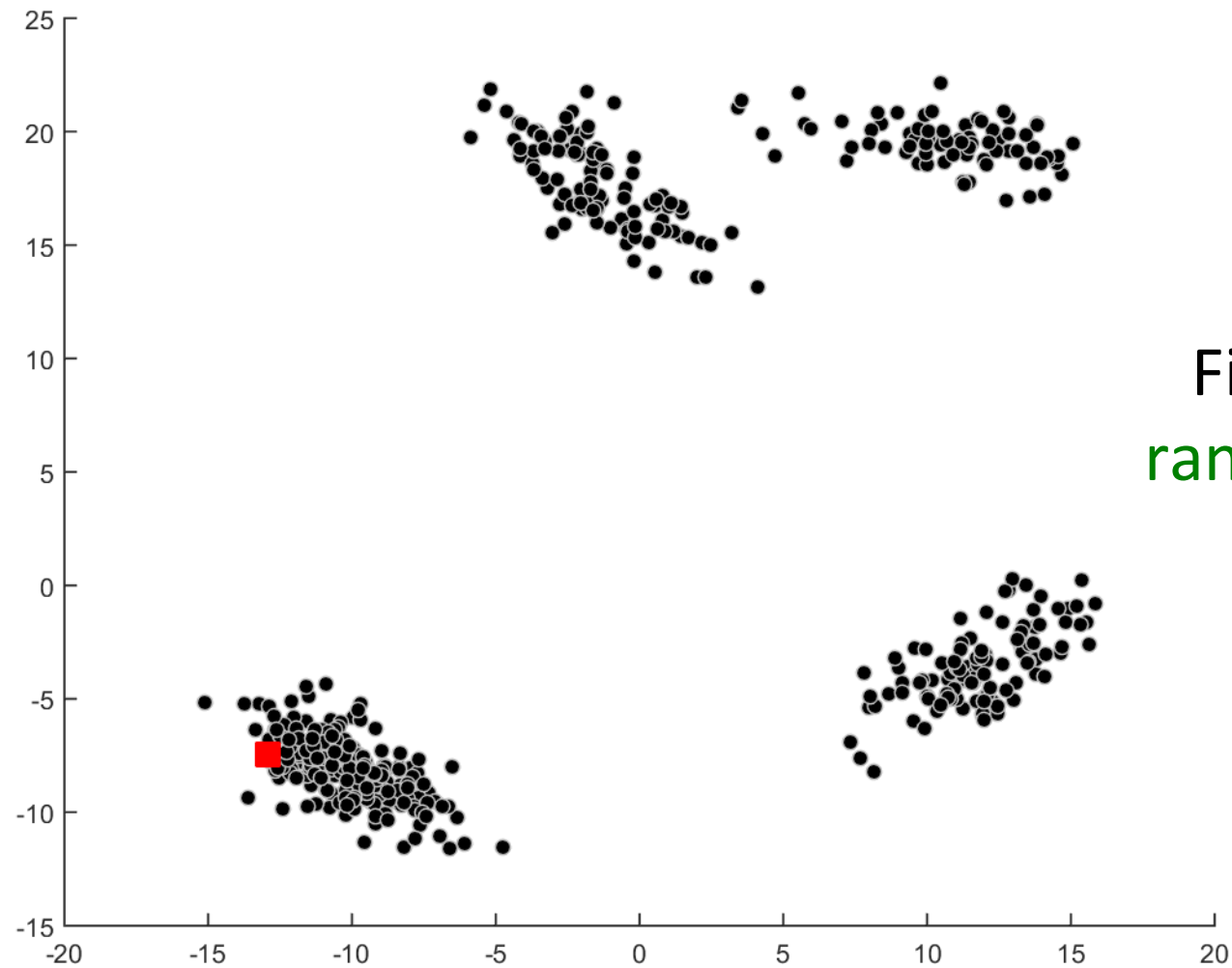
Can be
done in
 $O(n)$.

"probability that we
choose x_i as next mean"

K-Means++

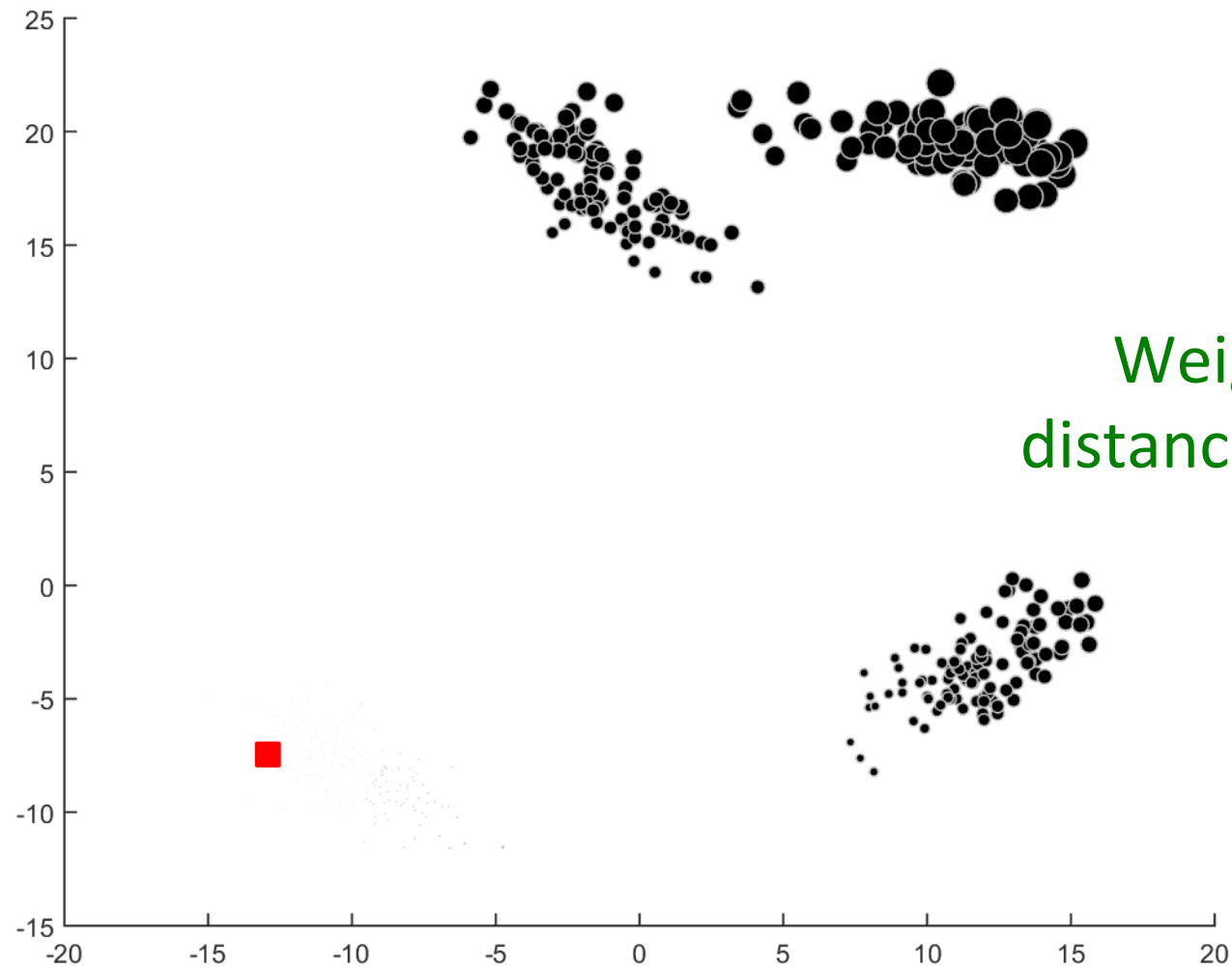


K-Means++

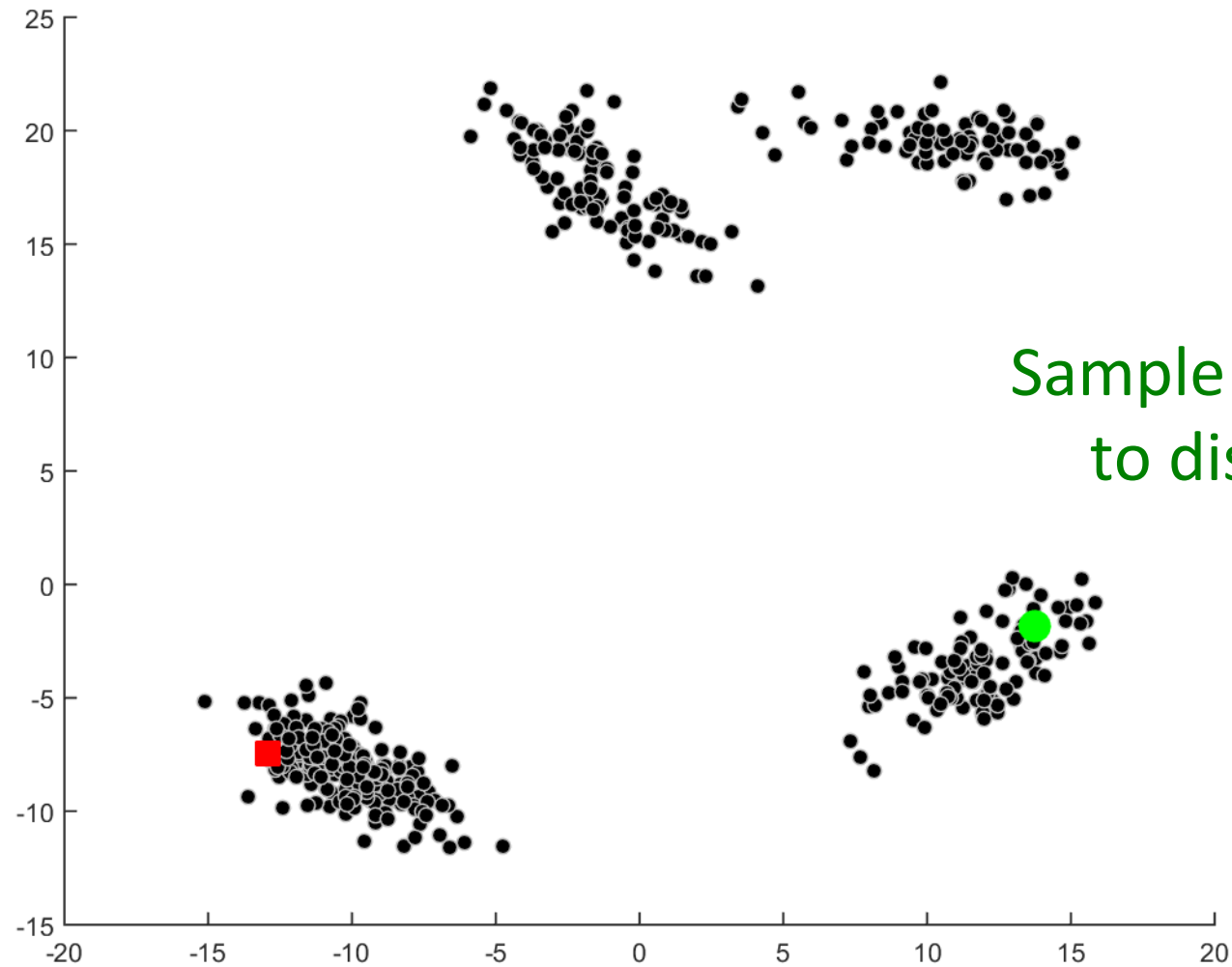


First mean is a
random example.

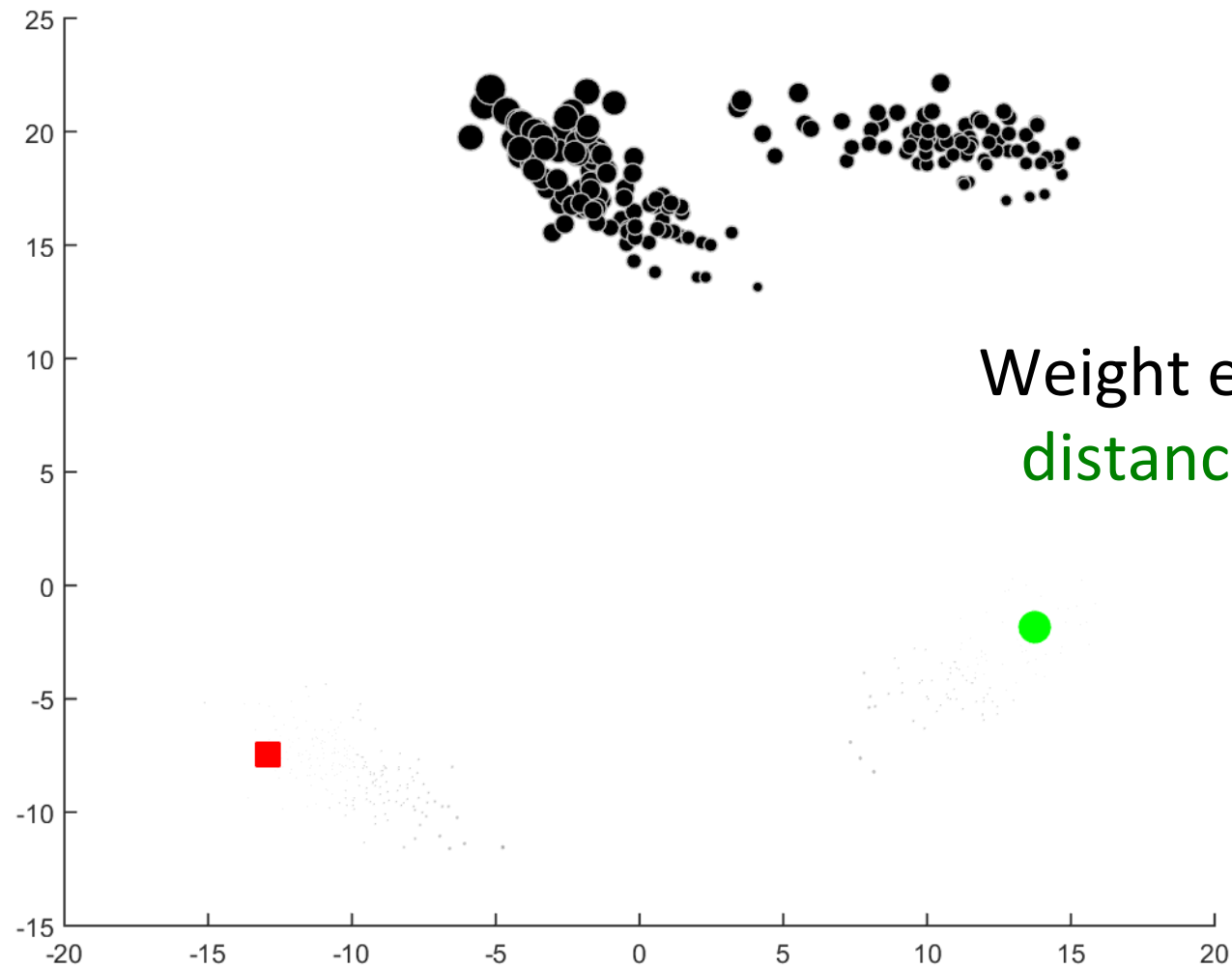
K-Means++



K-Means++

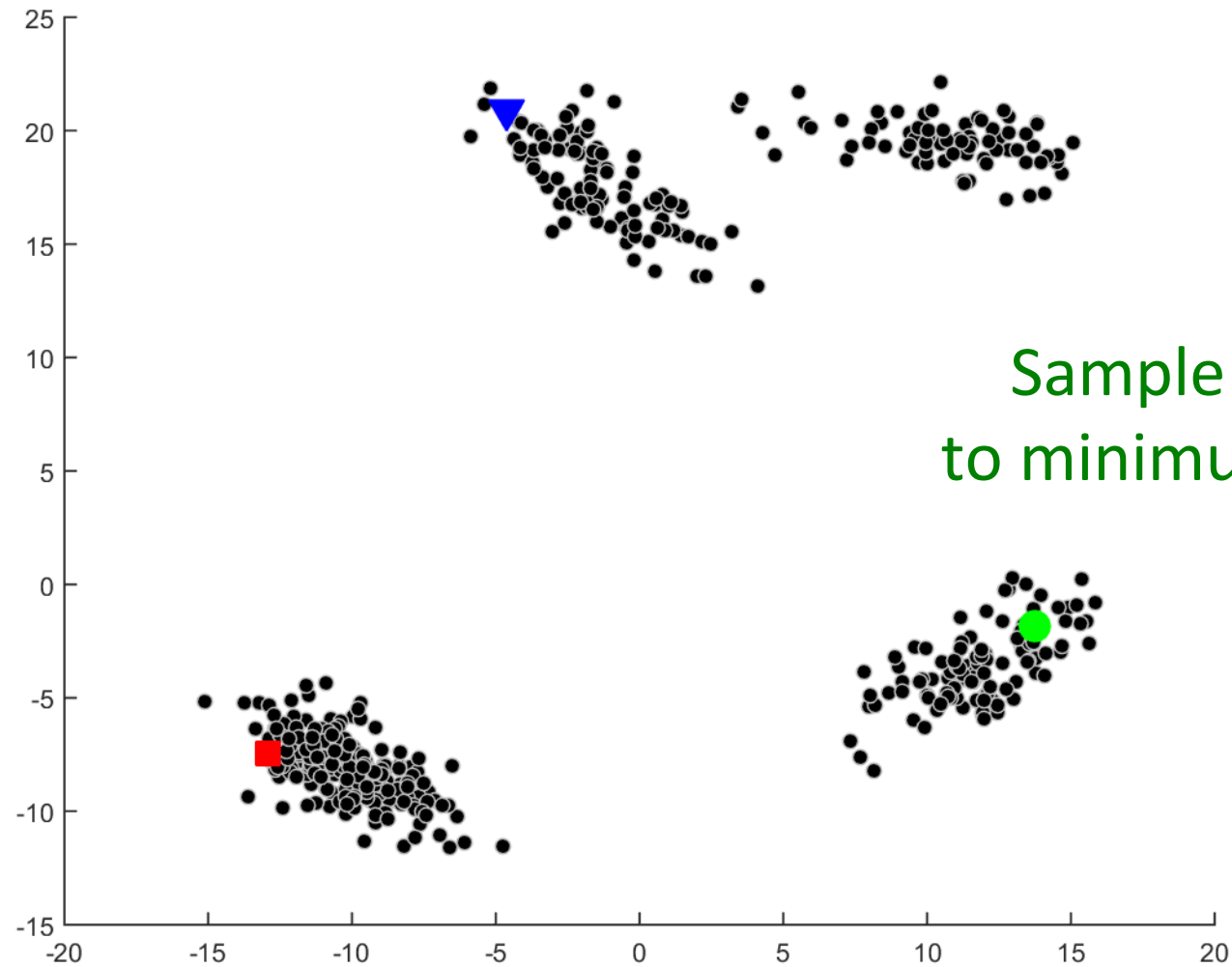


K-Means++

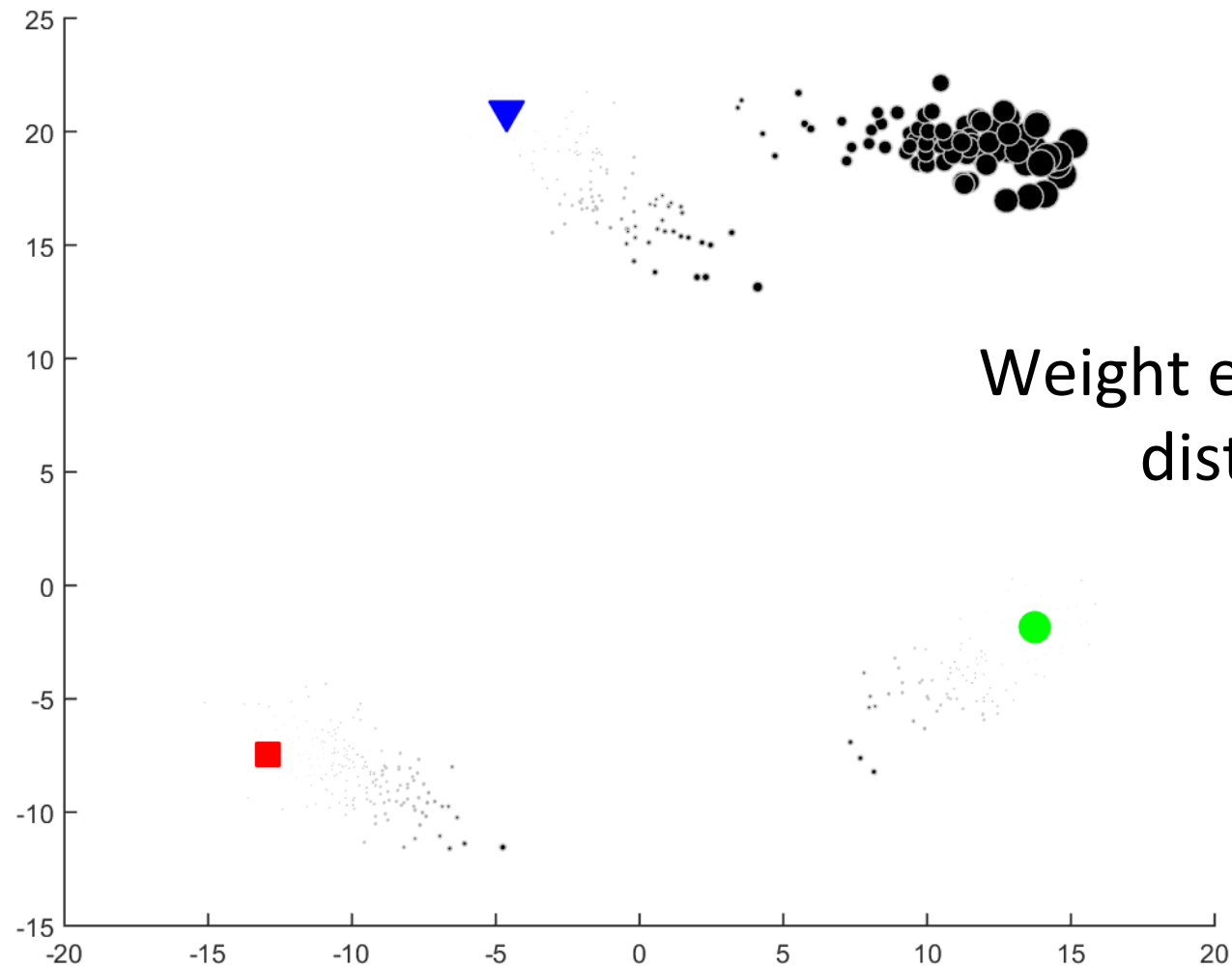


Weight examples by squared
distance to nearest mean.

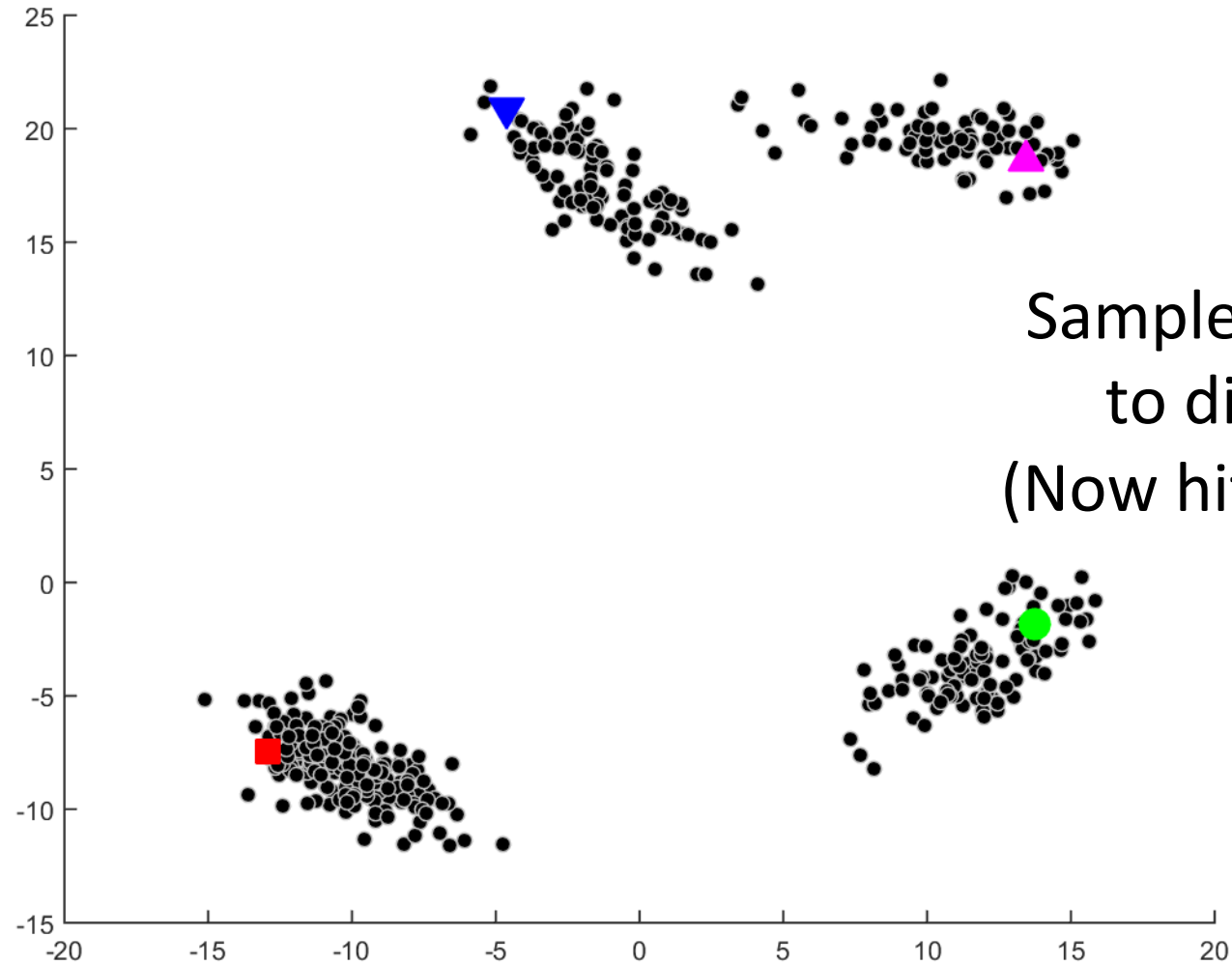
K-Means++



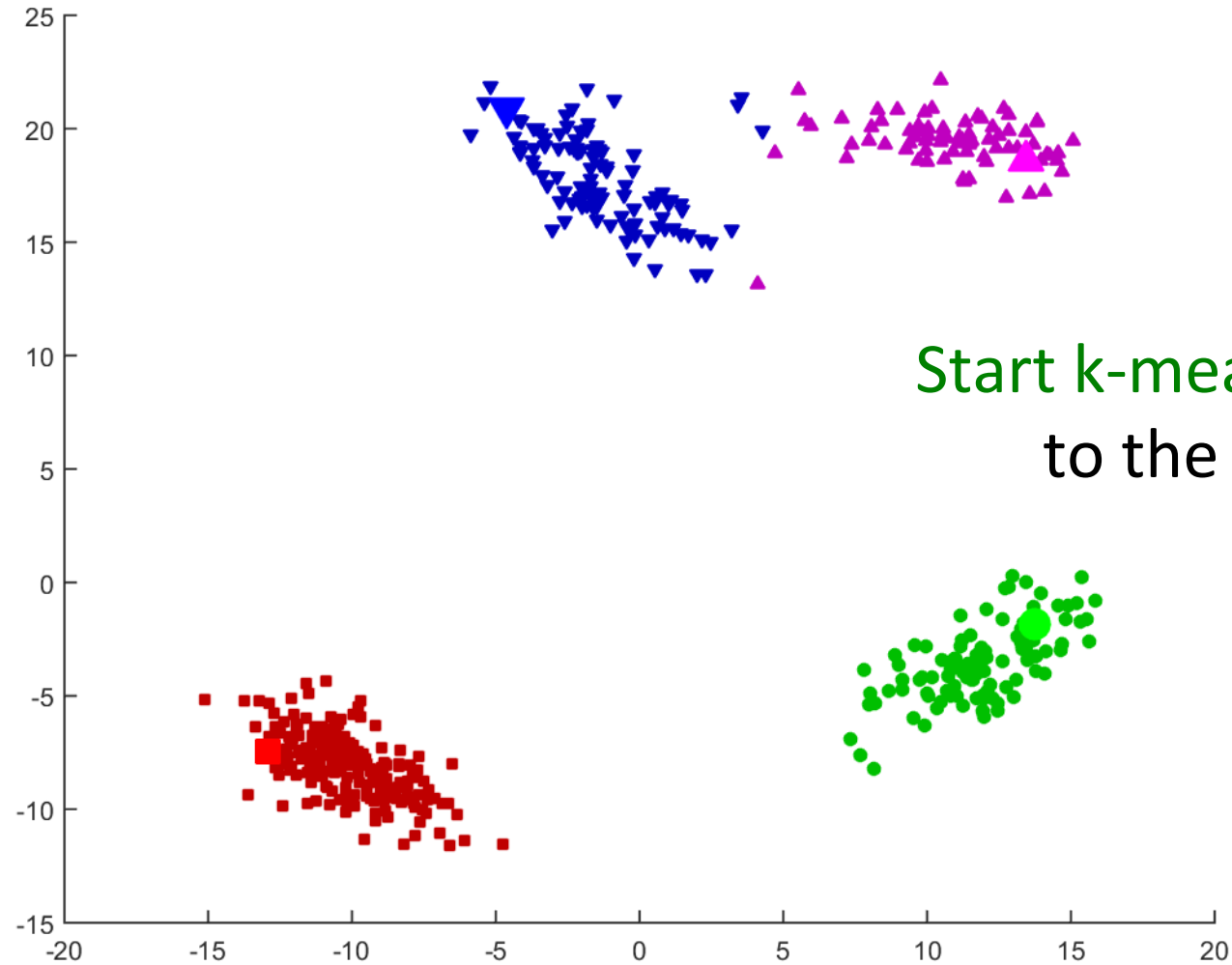
K-Means++



K-Means++

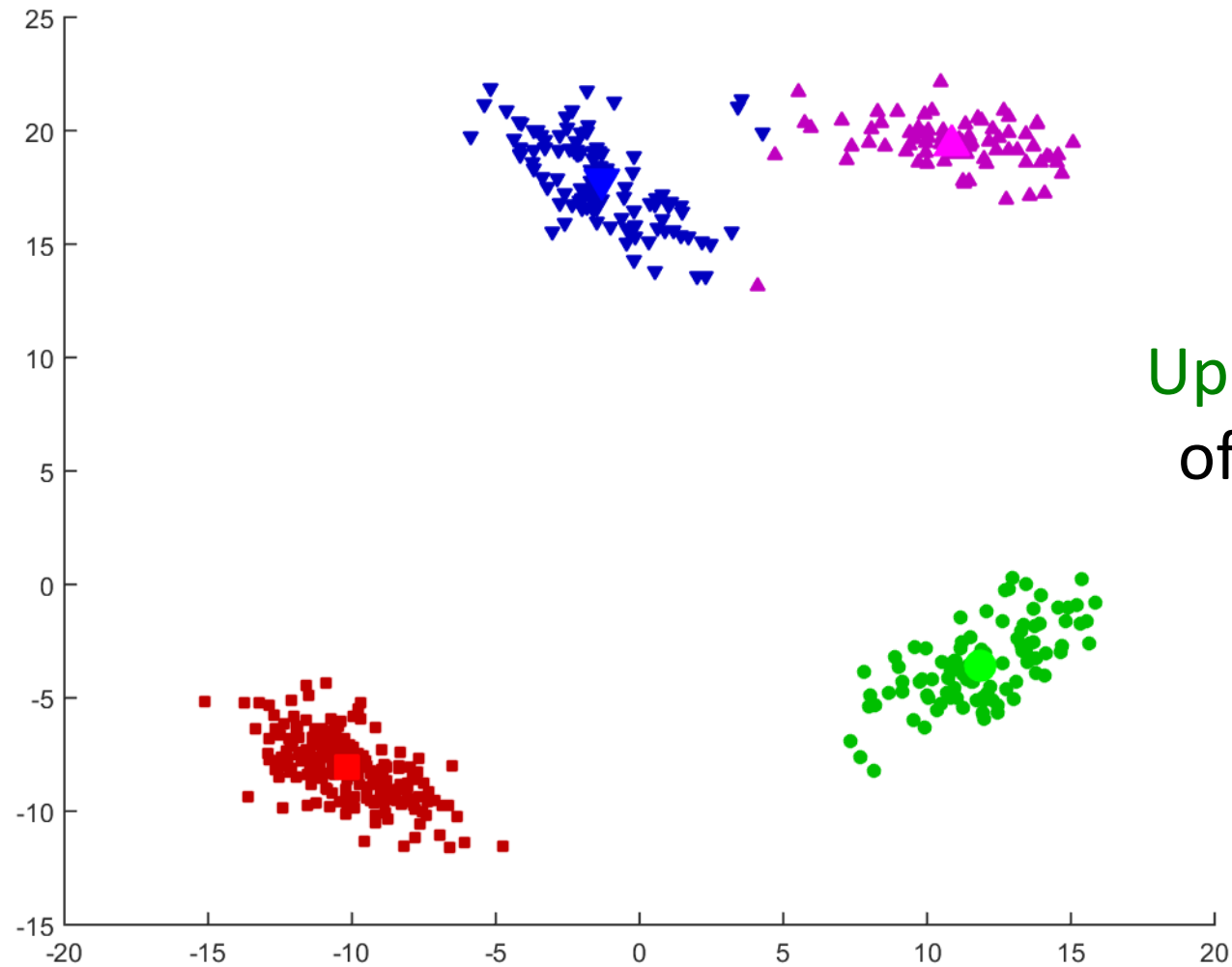


K-Means++



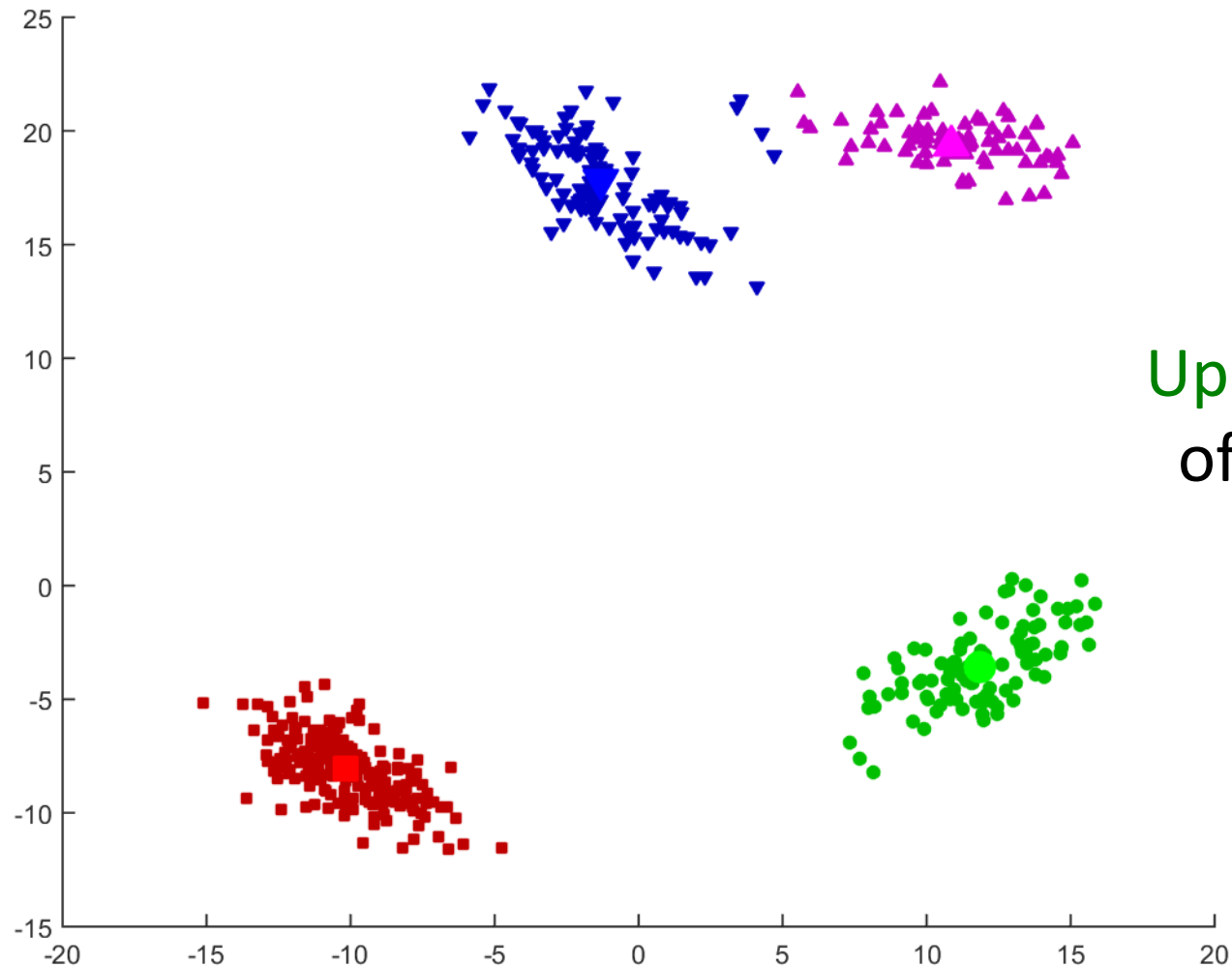
Start k-means: assign objects to the closest mean.

K-Means++



Update the mean
of each cluster.

K-Means++



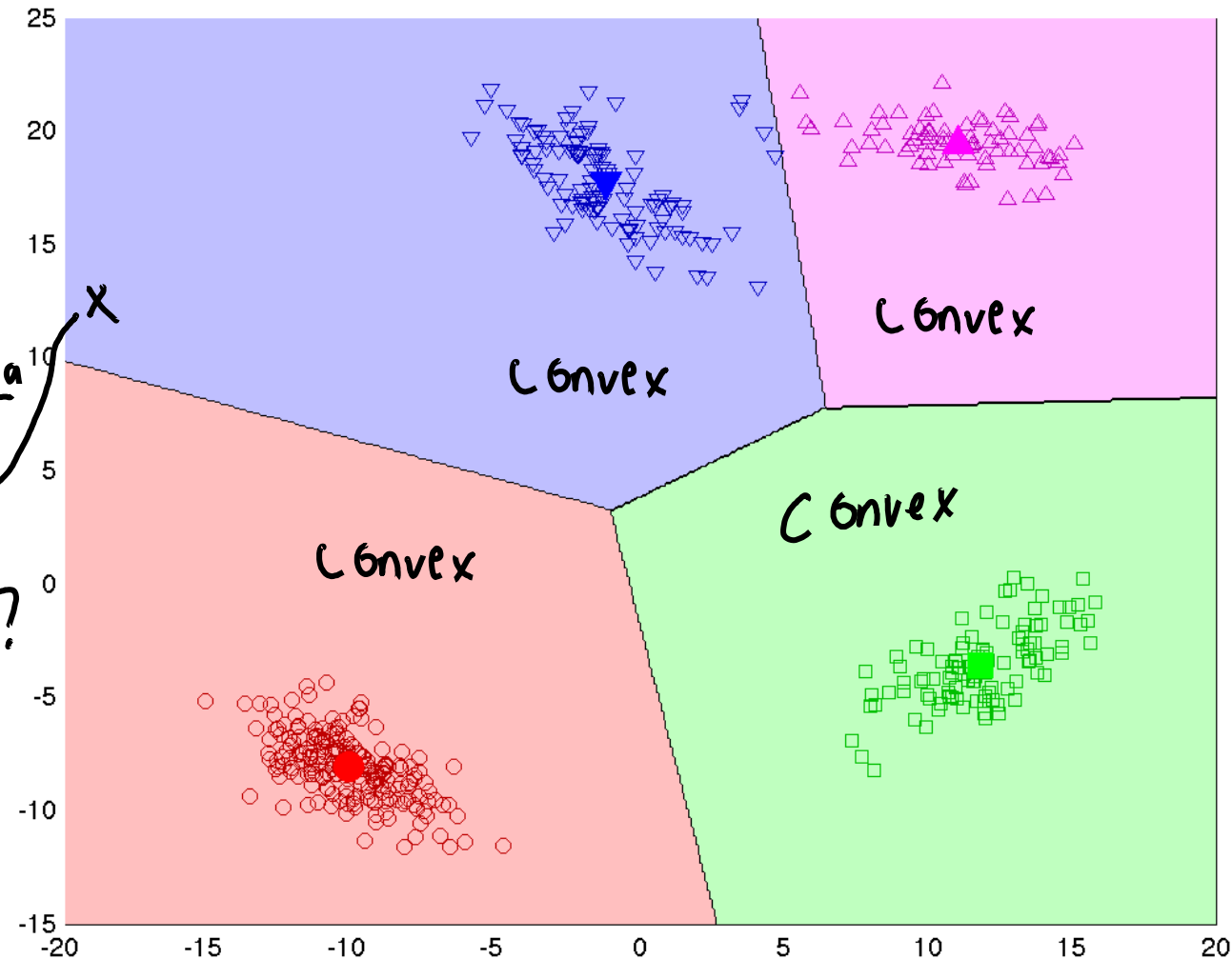
Update the mean
of each cluster.

In this case: just 2 iterations!

Shape of K-Means Clusters

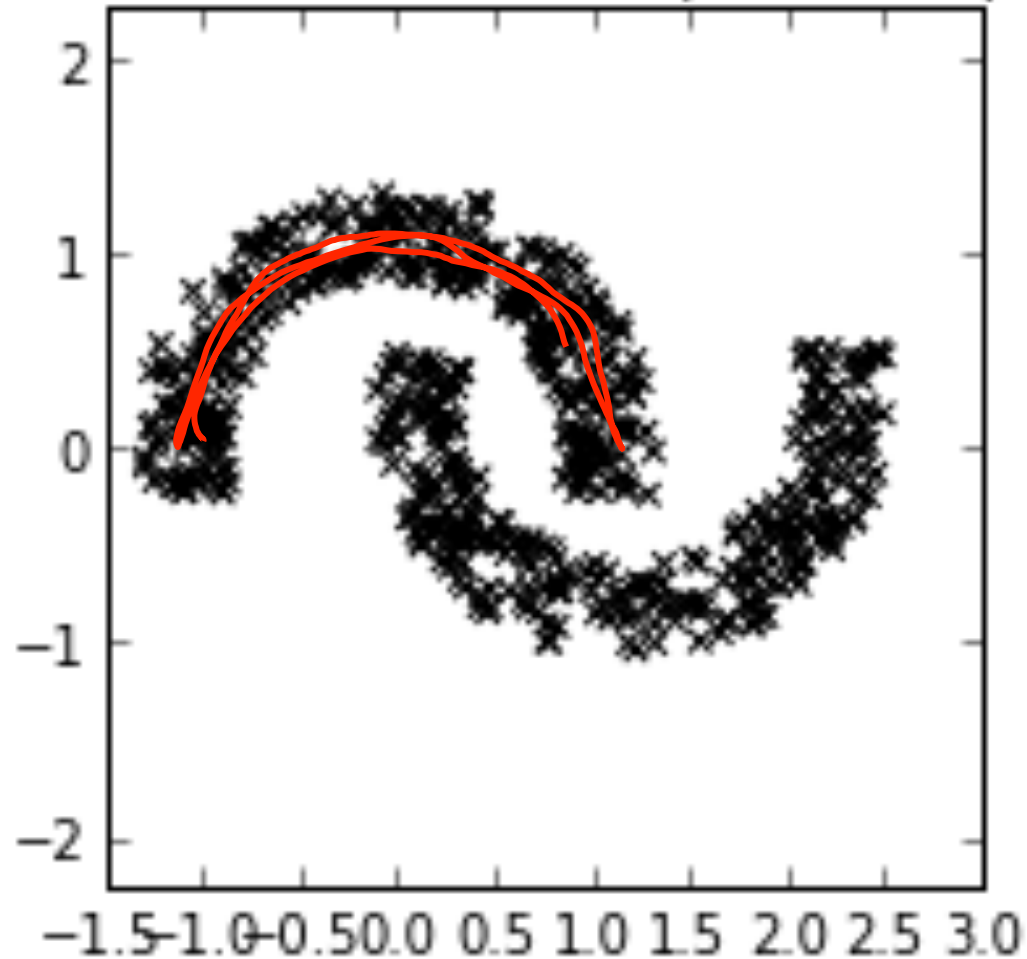
New issues:

1. Are clusters in data always convex?
2. Is this really part of the blue cluster?

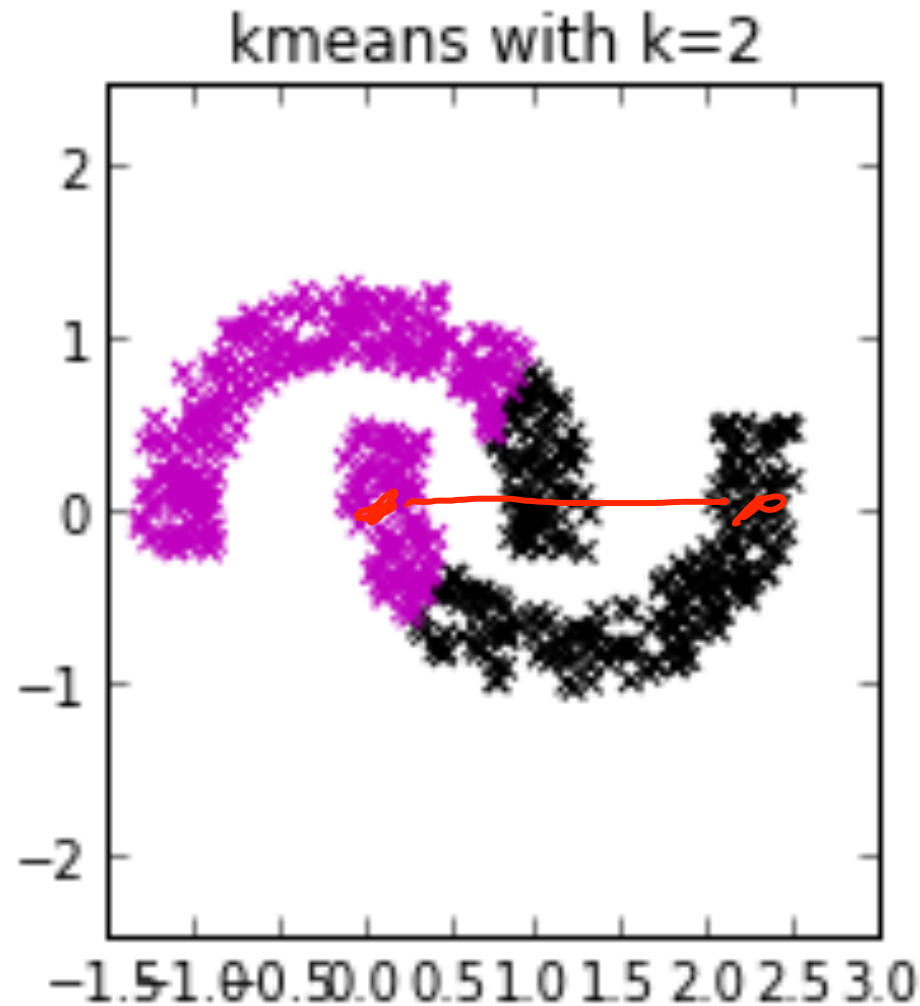


K-Means with Non-Convex Clusters

Non-convex banana-shaped data points

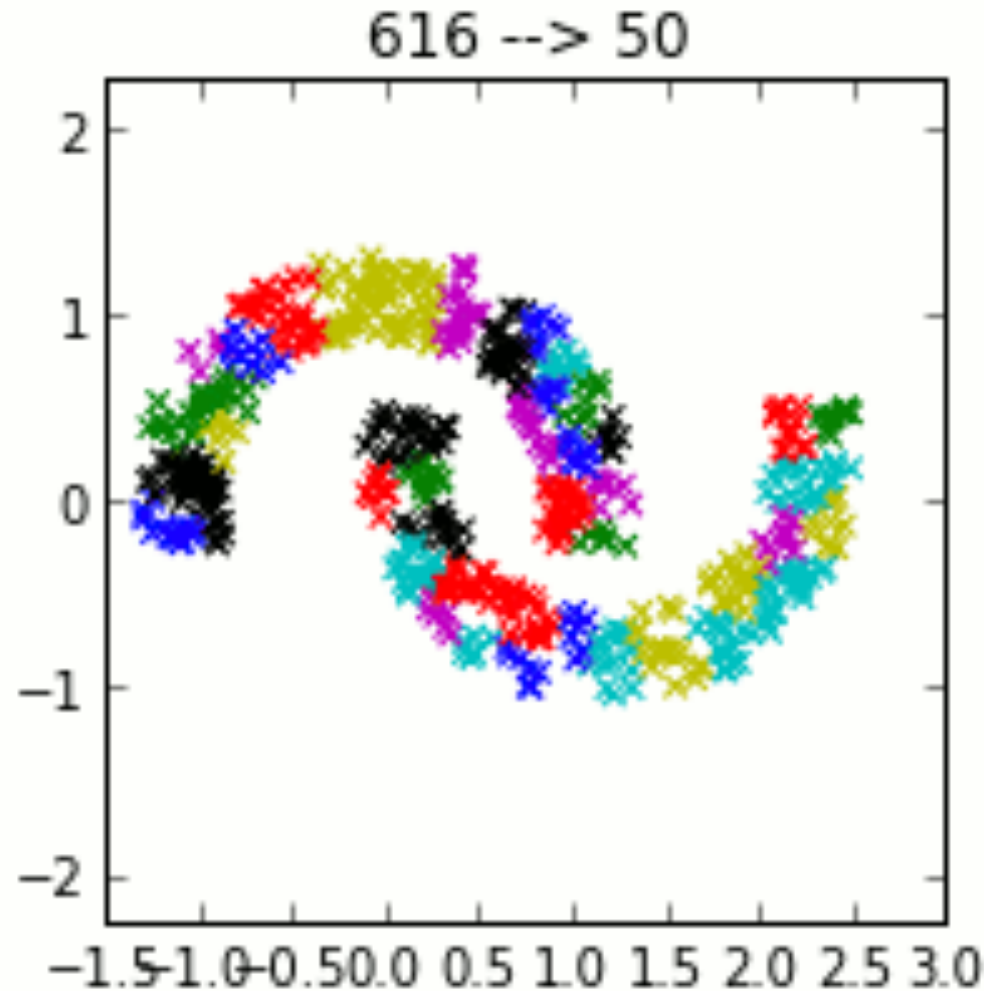


K-Means with Non-Convex Clusters



K-means **cannot separate**
non-convex clusters

K-Means with Non-Convex Clusters



K-means **cannot separate**
non-convex clusters

Though over-clustering can help
(next class)

Application: Elephant Range Map

- Find habitat area of African elephants.
 - Useful for assessing/protecting population.
- Build clusters from observations of locations.
- Clusters are **non-convex**:
 - affected by vegetation, mountains, rivers, water access, etc.
- We don't want to “partition” data:
 - Some **points have no cluster**.



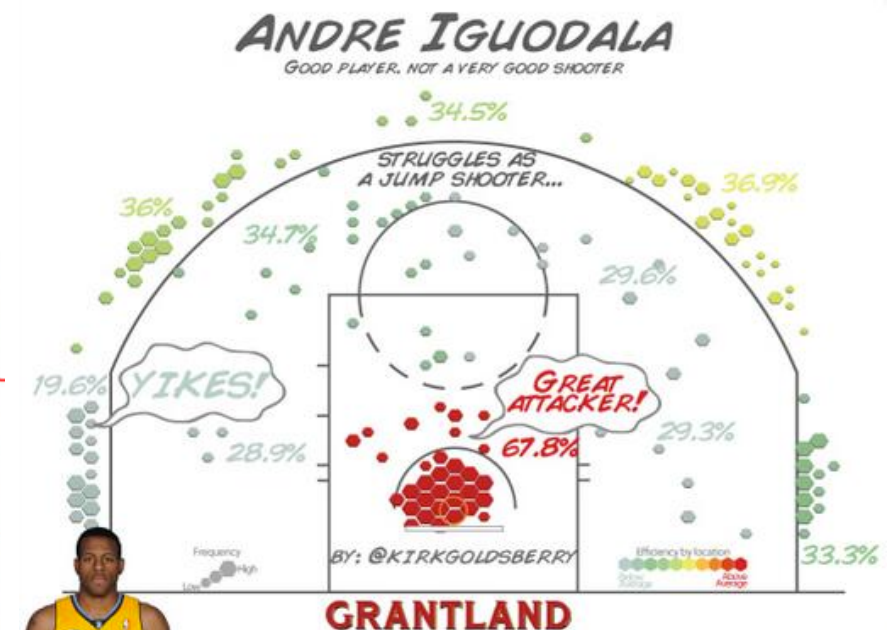
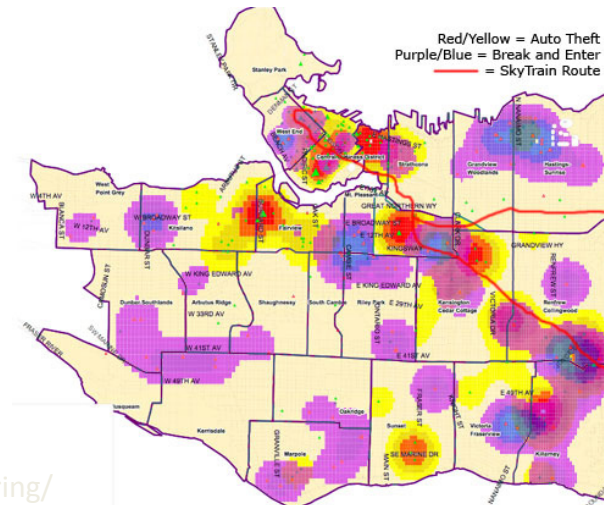
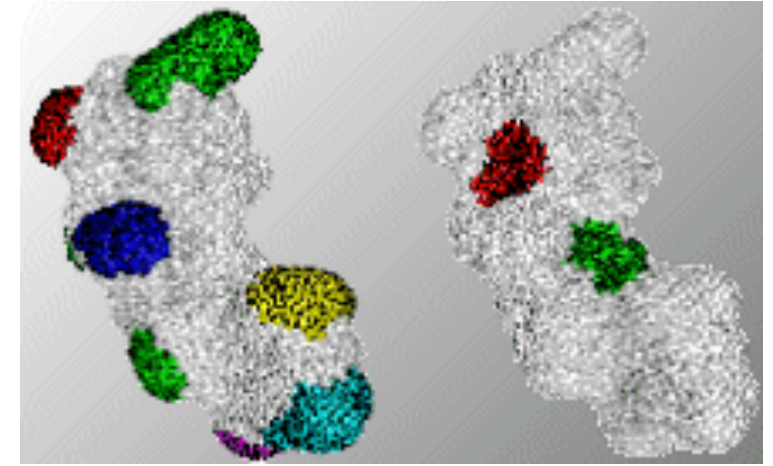
Motivation for Density-Based Clustering

- Density-based clustering:
 - Clusters are defined by all the objects in “dense” regions.
 - Objects in non-dense regions don’t get clustered.
- It’s a non-parametric clustering method:
 - Clusters can become more complicated the more data we have.
 - No fixed number of clusters ‘k’.



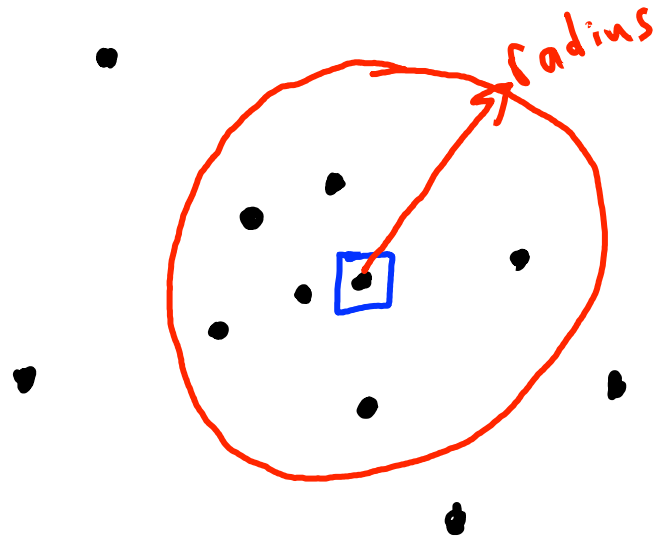
Other Potential Applications

- Where are high crime regions of a city?
- Where should taxis patrol?
- Which products are similar to this one?
- Which pictures are in the same place?
- Where can protein 'dock'?



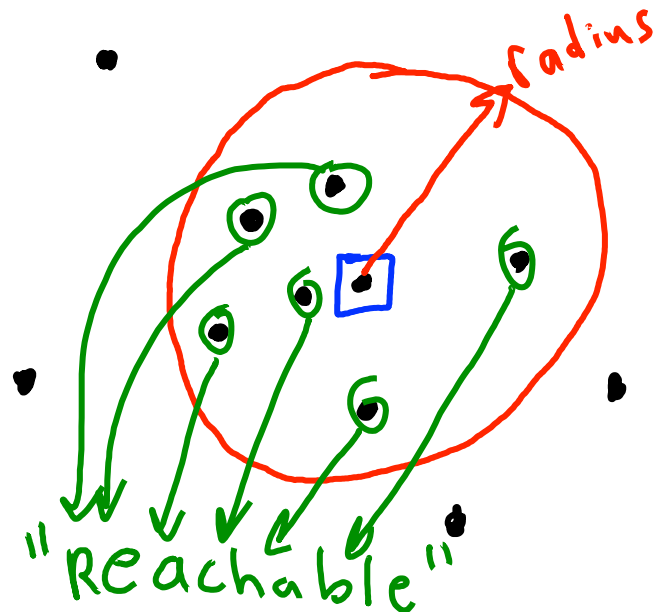
Density-Based Clustering

- Density-based clustering algorithm (DBSCAN) has two parameters:
 - Radius: minimum distance between points to be considered 'close'.
 - Objects within this radius are called "reachable".



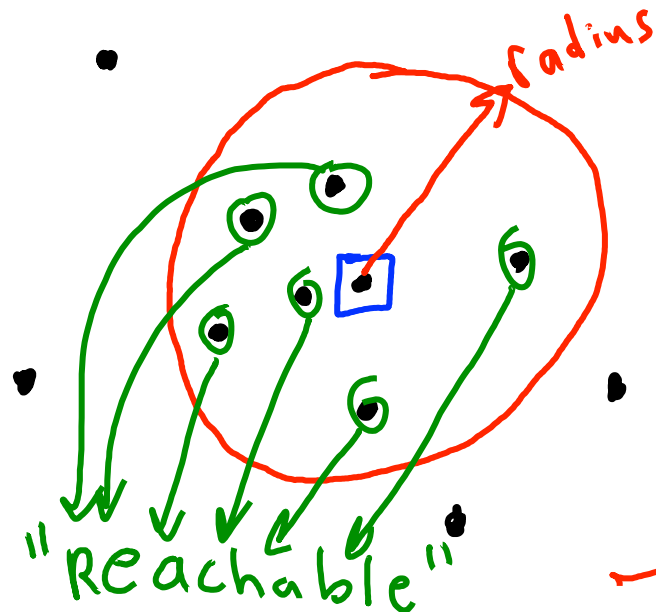
Density-Based Clustering

- Density-based clustering algorithm (DBSCAN) has two parameters:
 - Radius: minimum distance between points to be considered 'close'.
 - Objects within this radius are called "reachable".



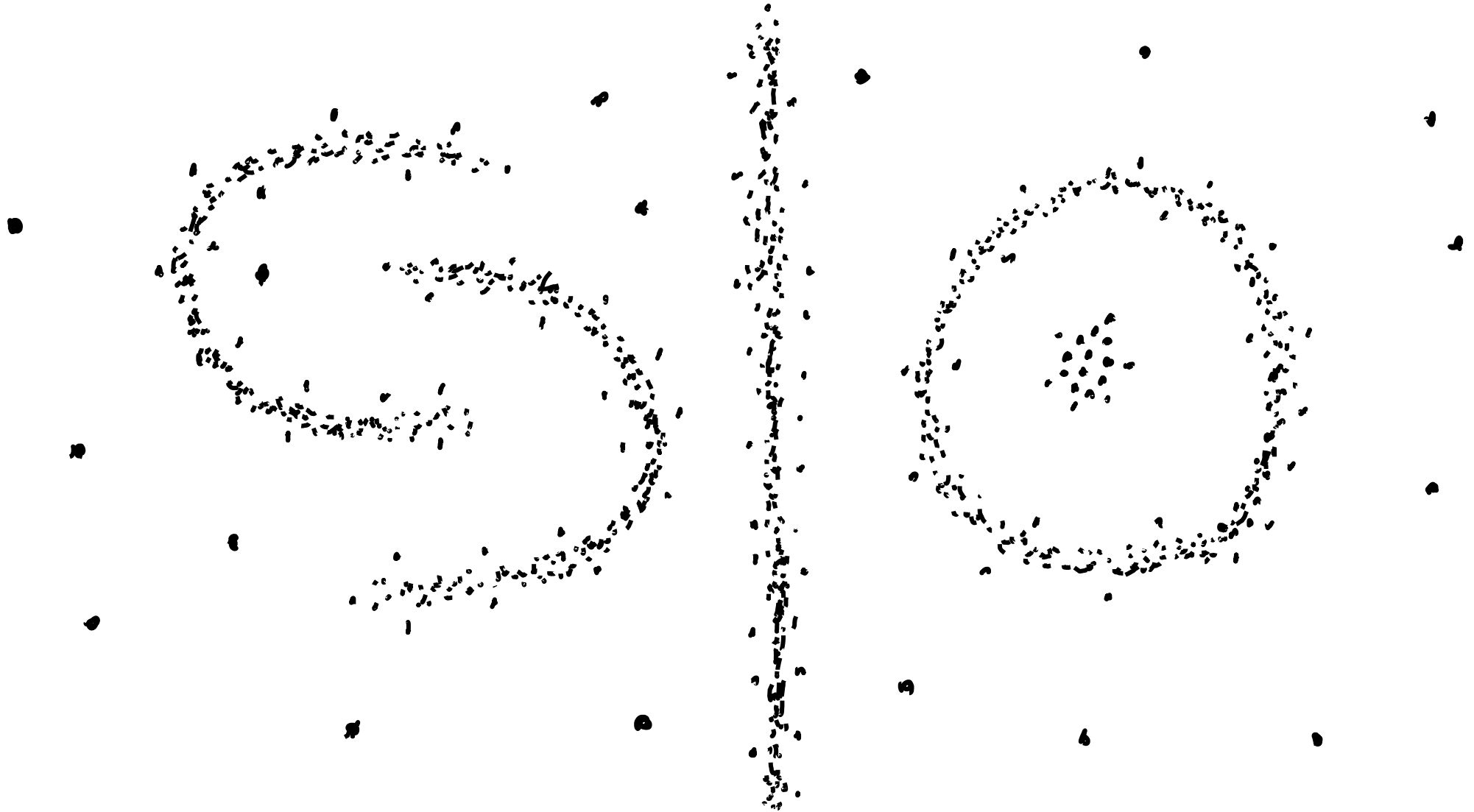
Density-Based Clustering

- Density-based clustering algorithm (DBSCAN) has two parameters:
 - Radius: minimum distance between points to be considered 'close'.
 - Objects within this radius are called "reachable".
 - MinPoints: number of reachable points needed to define a cluster.
 - If you have minPoints "reachable points", you are called a "core" point.

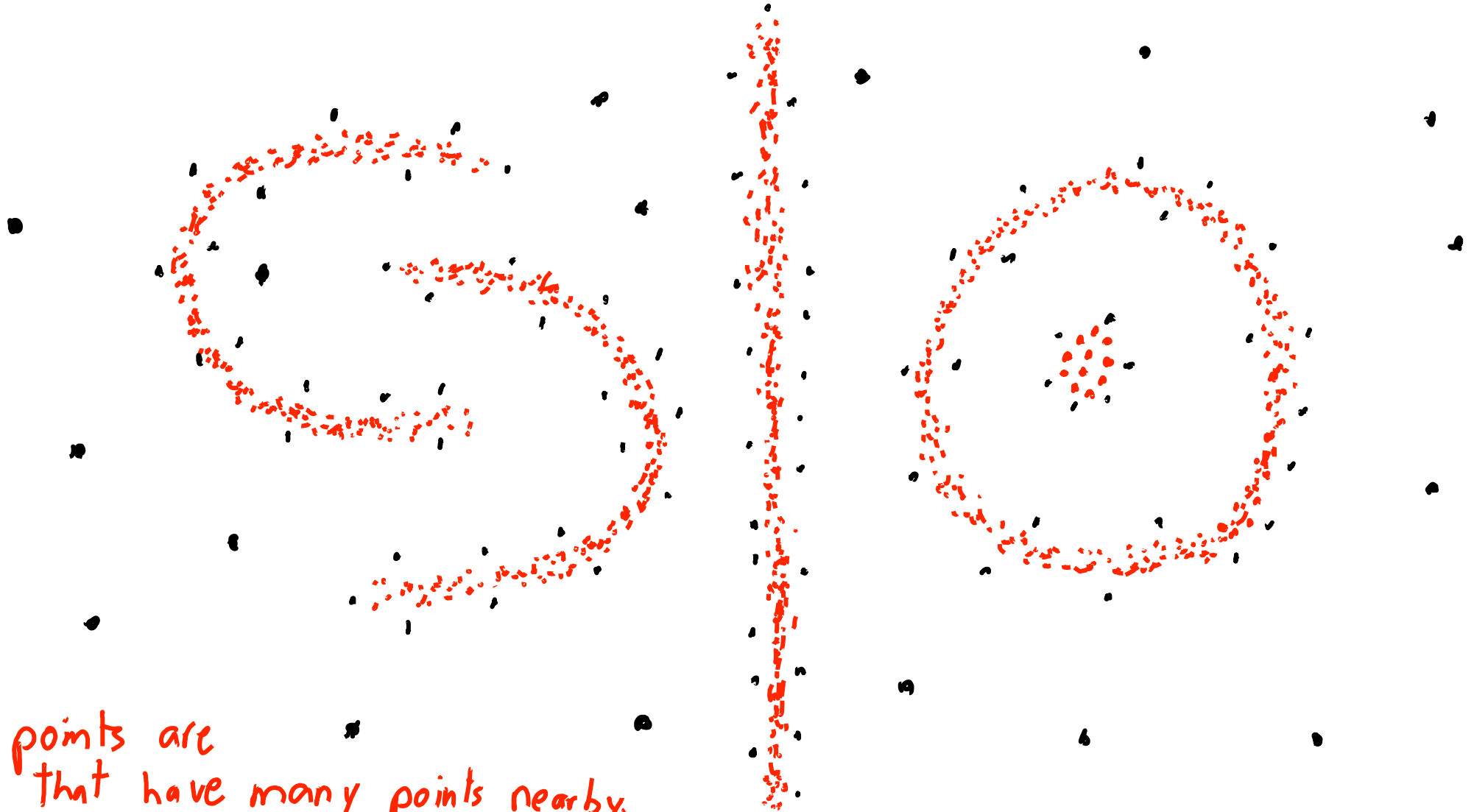


→ E.g., if minPoints = 3
then this is a "core"
point since 6 points are
"reachable"

Density-Based Clustering

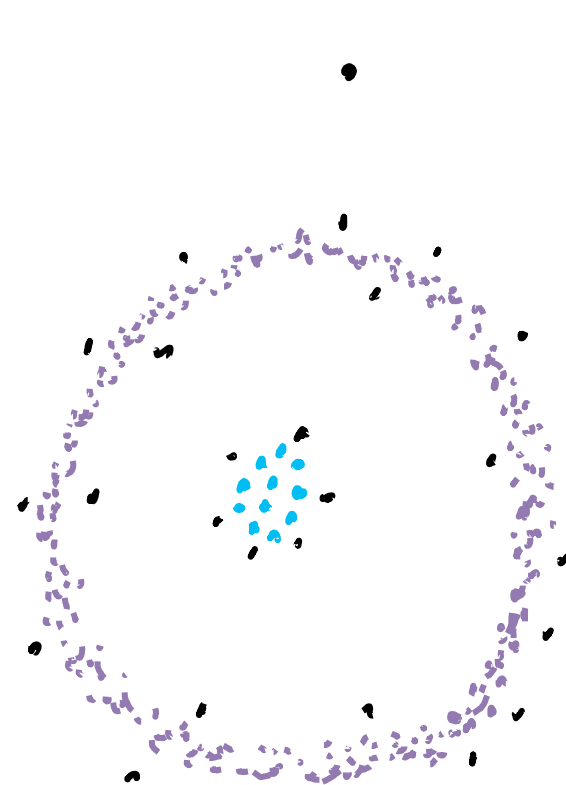
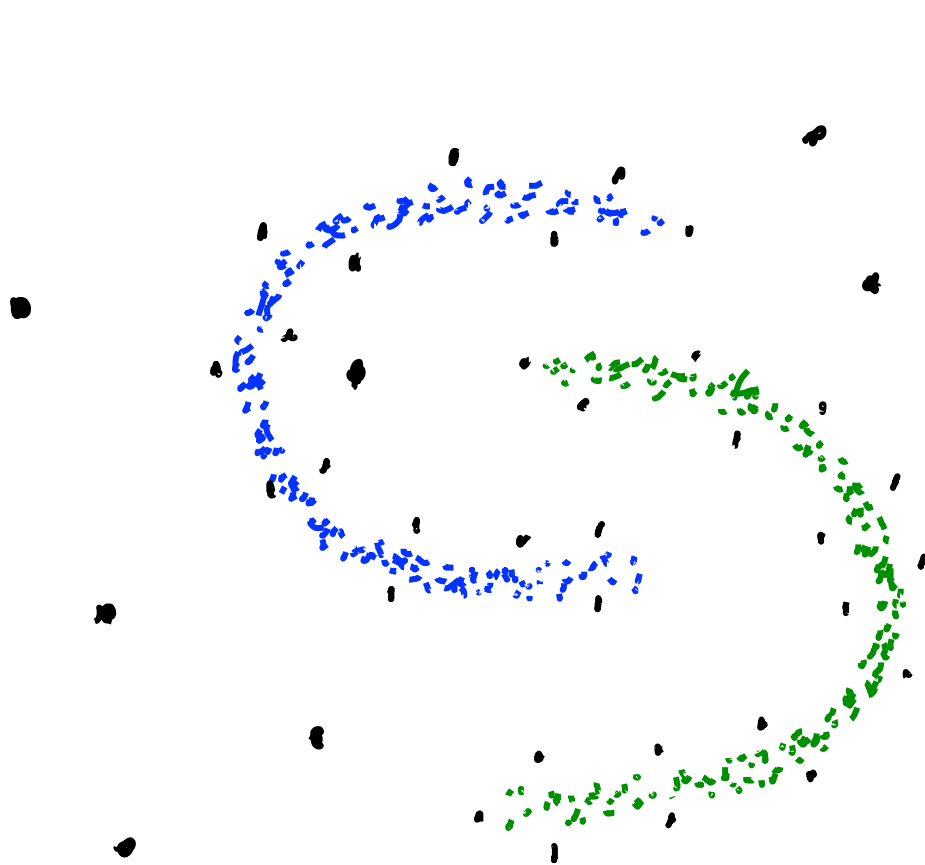


Density-Based Clustering



"Core" points are points that have many points nearby.

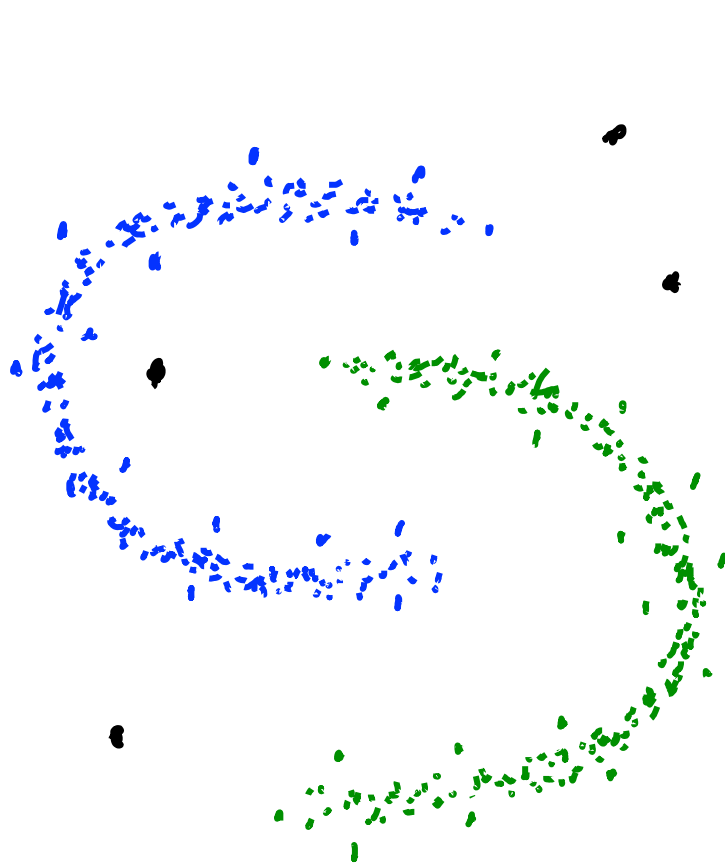
Density-Based Clustering



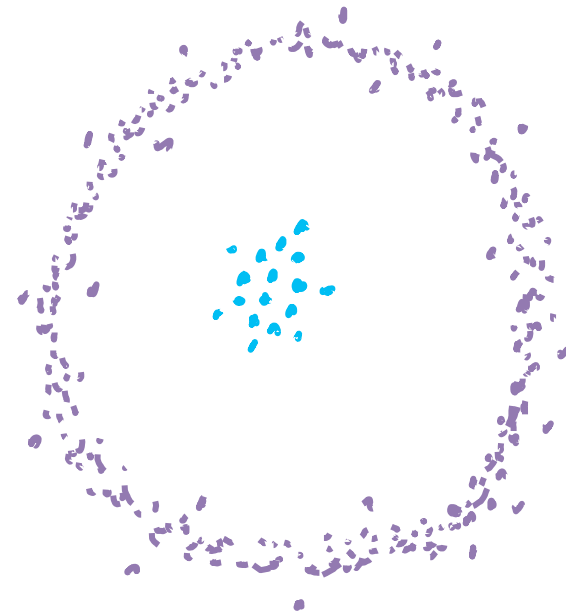
Clusters contain:
1. All "core" points
that can be reached
by following a sequence
of core points.

"Core" points are
points that have many points nearby.

Density-Based Clustering



"Core" points are points that have many points nearby.

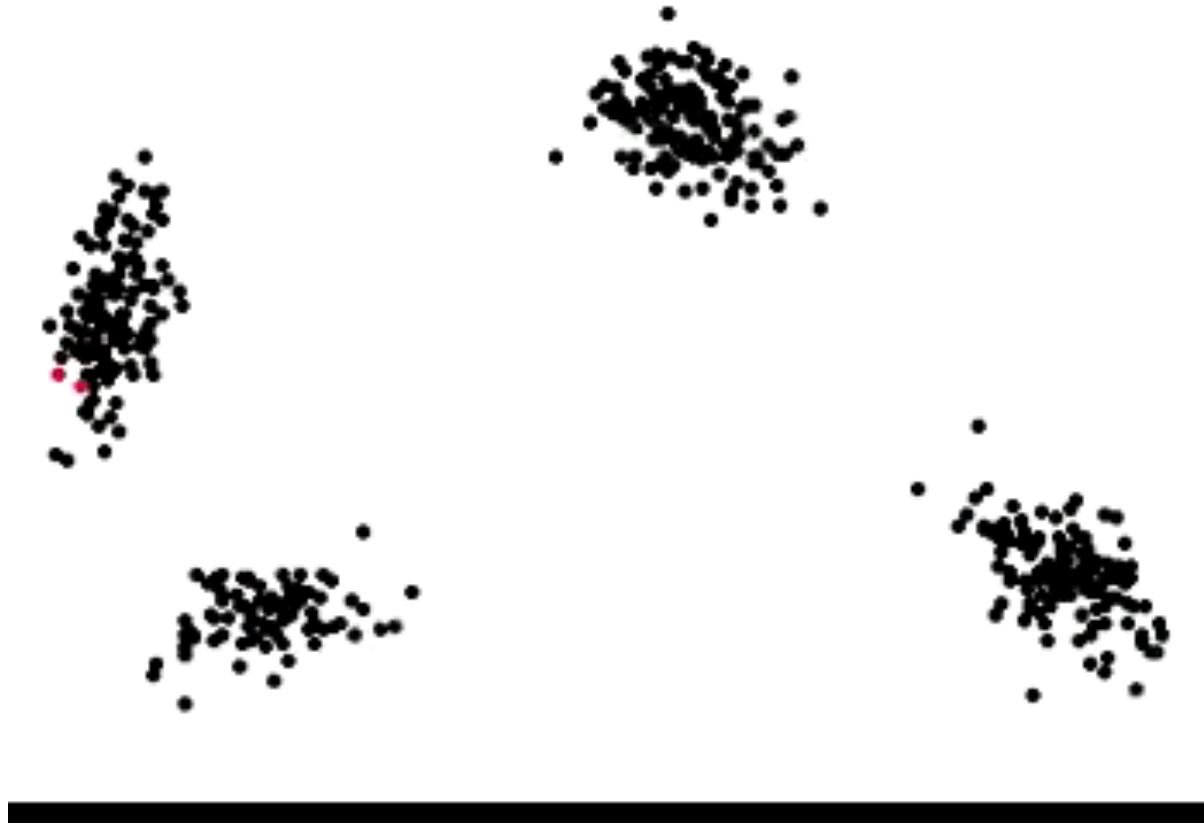


Clusters contain:

1. All "core" points that can be reached by following a sequence of core points.

2. All "reachable" non-core points from these "core" points ("boundary" points)

Density-Based Clustering in Action

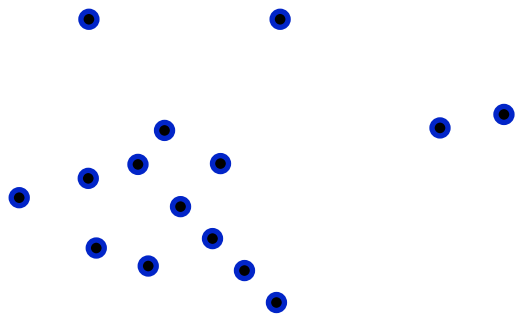


Interactive demo:

<https://www.naftaliharris.com/blog/visualizing-dbscan-clustering>

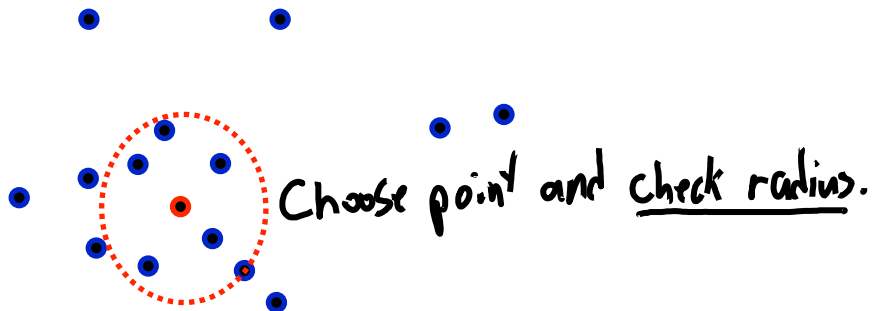
Density-Based Clustering

- Each “core” point defines a cluster:
 - Consisting of “core” point and all its “reachable” points.
- Merge clusters if “core” points are “reachable” from each other.



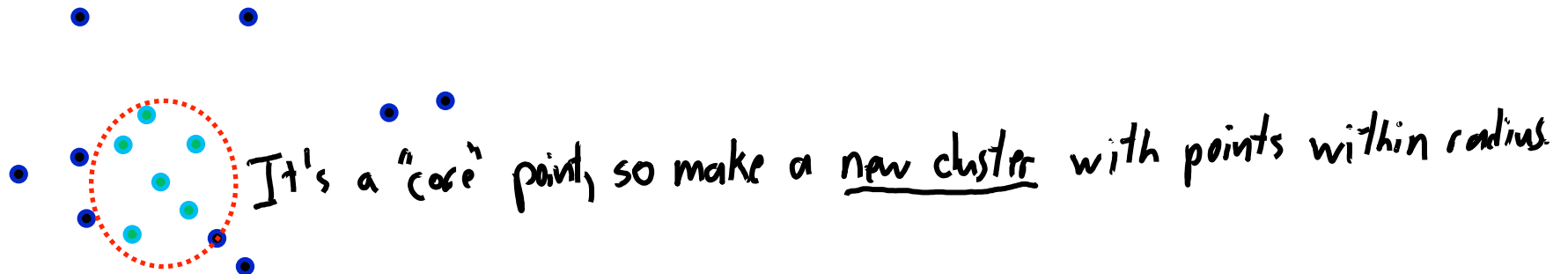
Density-Based Clustering

- Each “core” point defines a cluster:
 - Consisting of “core” point and all its “reachable” points.
- Merge clusters if “core” points are “reachable” from each other.



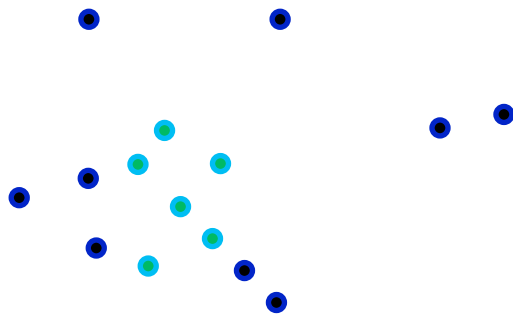
Density-Based Clustering

- Each “core” point defines a cluster:
 - Consisting of “core” point and all its “reachable” points.
- Merge clusters if “core” points are “reachable” from each other.



Density-Based Clustering

- Each “core” point defines a cluster:
 - Consisting of “core” point and all its “reachable” points.
- Merge clusters if “core” points are “reachable” from each other.



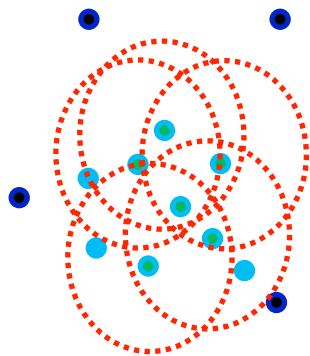
Density-Based Clustering

- Each “core” point defines a cluster:
 - Consisting of “core” point and all its “reachable” points.
- Merge clusters if “core” points are “reachable” from each other.



Density-Based Clustering

- Each “core” point defines a cluster:
 - Consisting of “core” point and all its “reachable” points.
- Merge clusters if “core” points are “reachable” from each other.



Add "reachable" points to cluster.

Density-Based Clustering

- Each “core” point defines a cluster:
 - Consisting of “core” point and all its “reachable” points.
- Merge clusters if “core” points are “reachable” from each other.



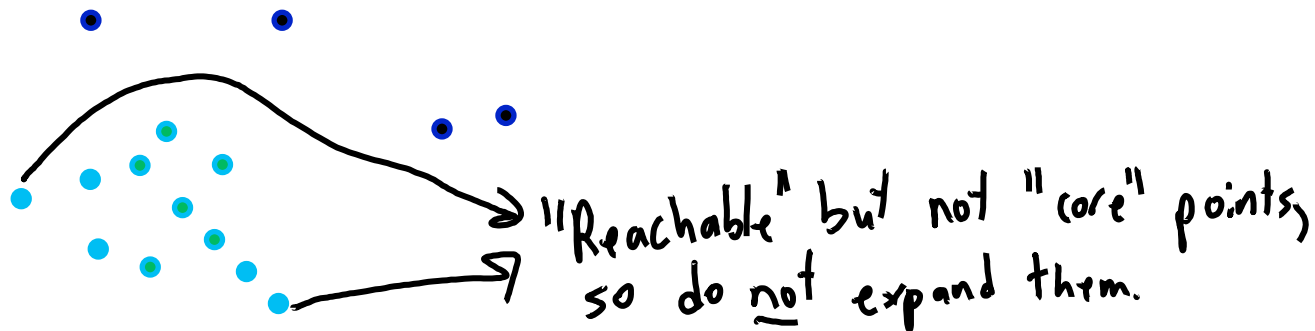
Density-Based Clustering

- Each “core” point defines a cluster:
 - Consisting of “core” point and all its “reachable” points.
- Merge clusters if “core” points are “reachable” from each other.



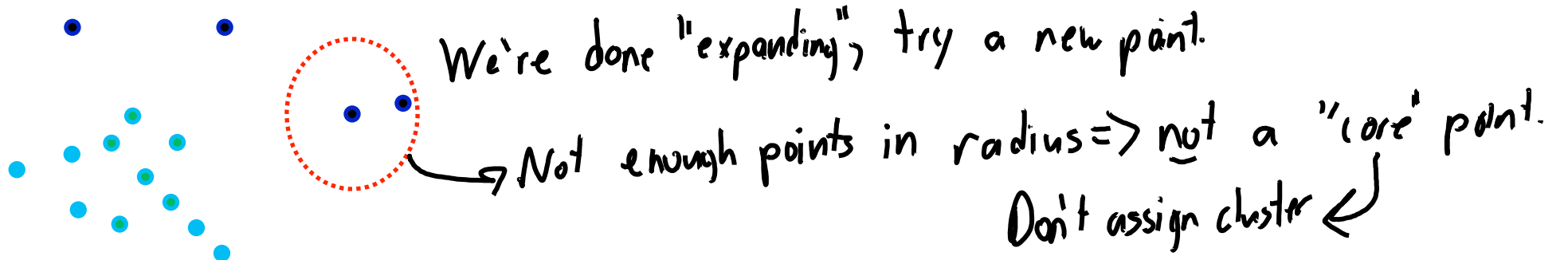
Density-Based Clustering

- Each “core” point defines a cluster:
 - Consisting of “core” point and all its “reachable” points.
- Merge clusters if “core” points are “reachable” from each other.



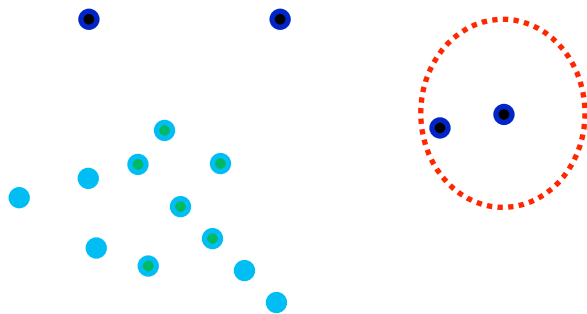
Density-Based Clustering

- Each “core” point defines a cluster:
 - Consisting of “core” point and all its “reachable” points.
- Merge clusters if “core” points are “reachable” from each other.



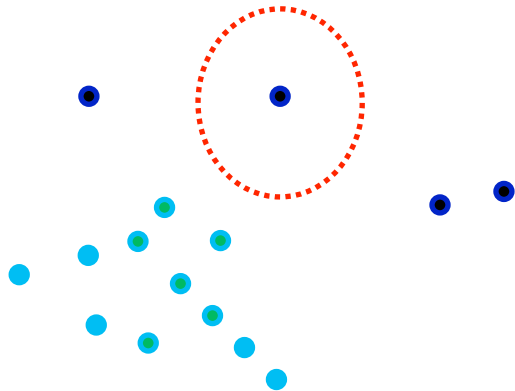
Density-Based Clustering

- Each “core” point defines a cluster:
 - Consisting of “core” point and all its “reachable” points.
- Merge clusters if “core” points are “reachable” from each other.



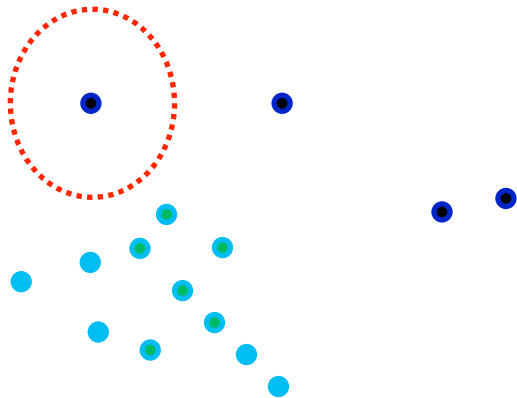
Density-Based Clustering

- Each “core” point defines a cluster:
 - Consisting of “core” point and all its “reachable” points.
- Merge clusters if “core” points are “reachable” from each other.



Density-Based Clustering

- Each “core” point defines a cluster:
 - Consisting of “core” point and all its “reachable” points.
- Merge clusters if “core” points are “reachable” from each other.

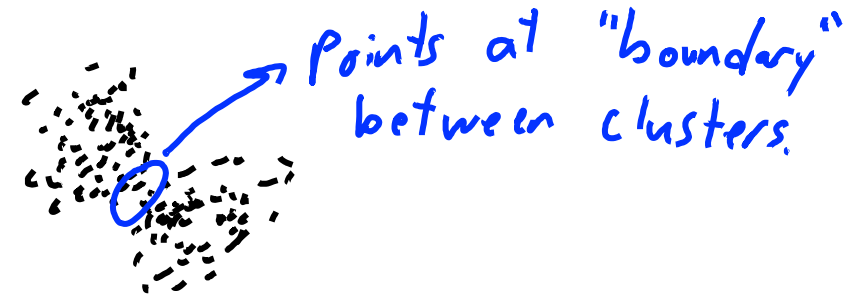


Density-Based Clustering

- Pseudocode for DBSCAN:
 - For each example x_i :
 - If x_i is already assigned to a cluster, do nothing.
 - Test whether x_i is a 'core' point (less than minPoints neighbours with distances $\leq 'r'$).
 - If x_i is not core point, do nothing.
 - If x_i is a core point, “expand” cluster.
 - “Expand” cluster function:
 - Assign all x_j within distance ' r ' of core point x_i to cluster.
 - For each newly-assigned neighbour x_j that is a core point, “expand” cluster.

Density-Based Clustering Issues

- Some points are not assigned to a cluster.
 - Good or bad, depending on the application.
- Ambiguity of “non-core” (boundary) points:
- Sensitive to the choice of radius and minPoints.
 - Otherwise, not sensitive to initialization (except for boundaries).
- If you get a new example, finding cluster is expensive.
 - Need to compute distances to training points.
- In high-dimensions, need a lot of points to ‘fill’ the space.



Summary

- Norms:
 - Ways to measure “size” in higher dimensions.
- K-means++:
 - Randomized initialization of k-means with good expected performance.
- Shape of K-means clusters:
 - Intersection of half-spaces, which forms convex sets.
- Density-based clustering:
 - “Expand” and “merge” dense regions of points to find clusters.
 - Useful for finding non-convex connected clusters.
- Next time:
 - Discovering the tree of life.