CPSC 340: Machine Learning and Data Mining

Density-Based Clustering

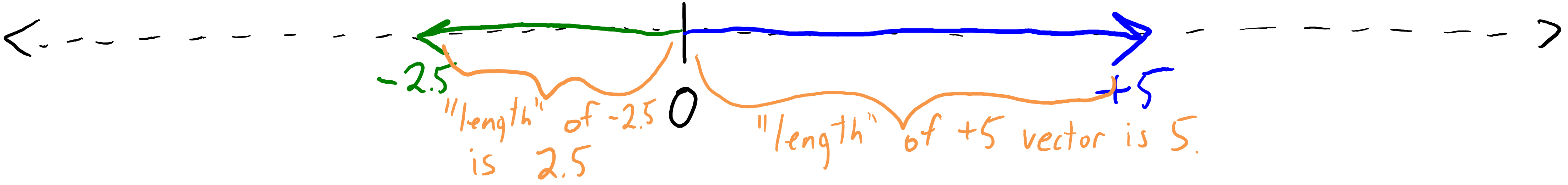
Fall 2016

Admin

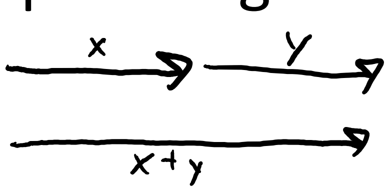
- Assignment 1 :
  - 2 late days to hand it in before Wednesday’s class.
  - 3 late days to hand it in before Friday’s class.
  - 0 after that.
- Assignment 2 coming tonight.

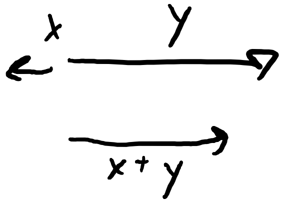
Norms in 1-Dimension

- 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 |α|:
     - “Absolute homogeneity”: |αx| = |α||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| <= |x| + |y|.
     - Think of “how far you travel”.



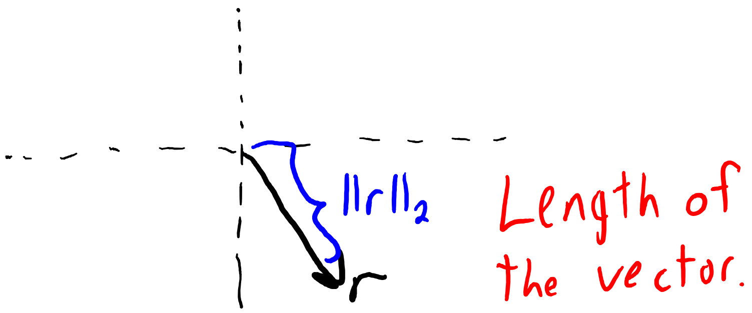


Norms in 2-Dimensions

- 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:

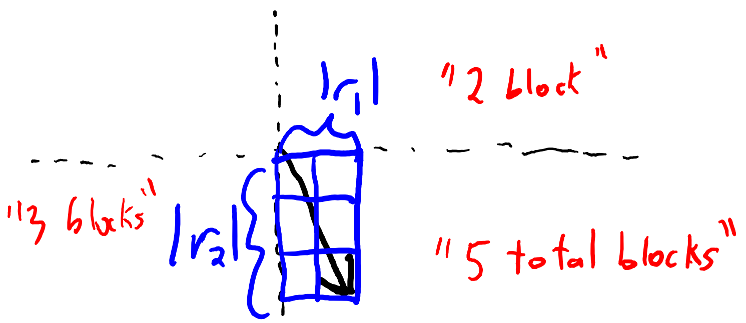
L_2 or "Euclidean" norm.

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



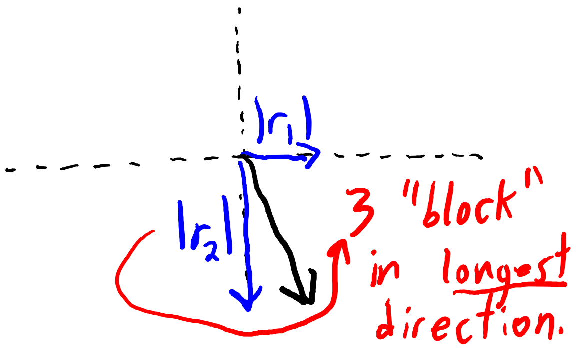
L_1 or "Manhattan" norm:

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



L_∞ or "max" norm:

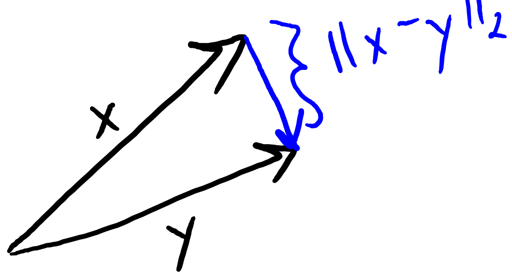
$$\|r\|_\infty = \max\{|r_1|, |r_2|\}$$



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_∞: $\max_j \{|r_j|\}$

E.g., in 3-dimensions:

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

in 4-dimensions:

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

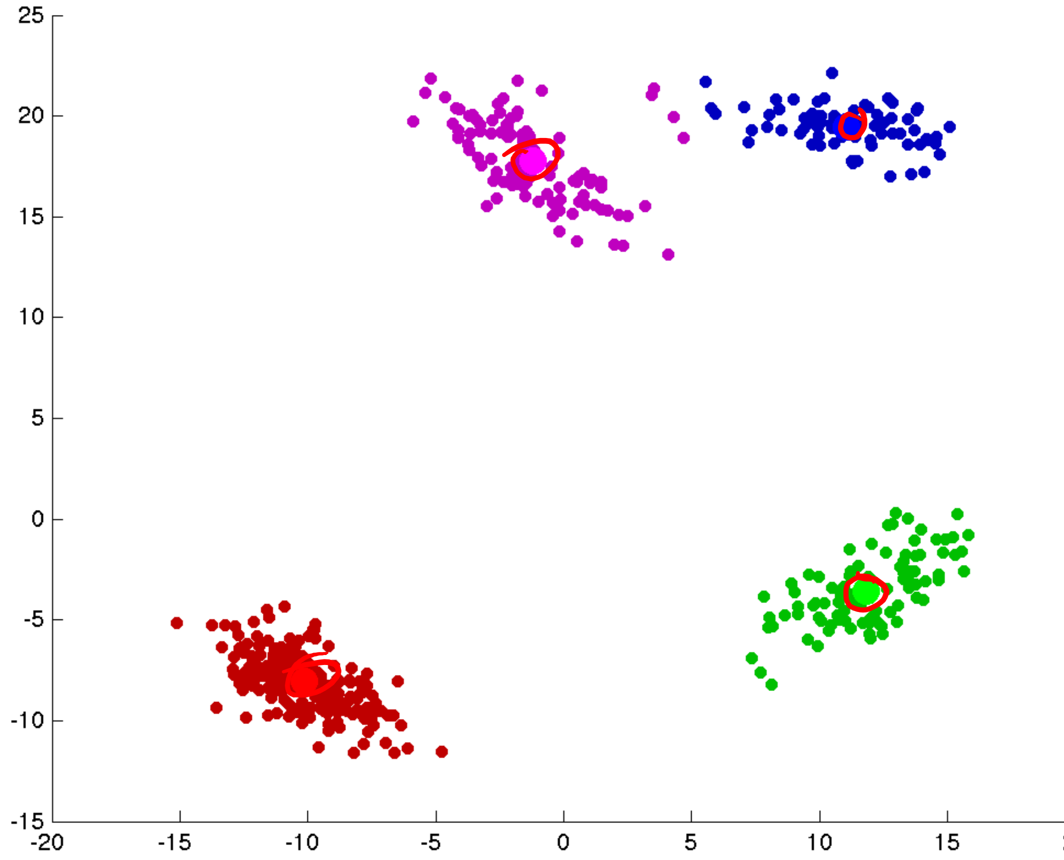
Notation: $\|r\|_2^2 = (\|r\|_2)^2 = \left(\sqrt{\sum_{j=1}^{d} r_j^2}\right)^2 = \sum_{j=1}^{d} r_j^2 = r^T r$

Different ways to write the same thing

- 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.

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 $(d_i)^2$.

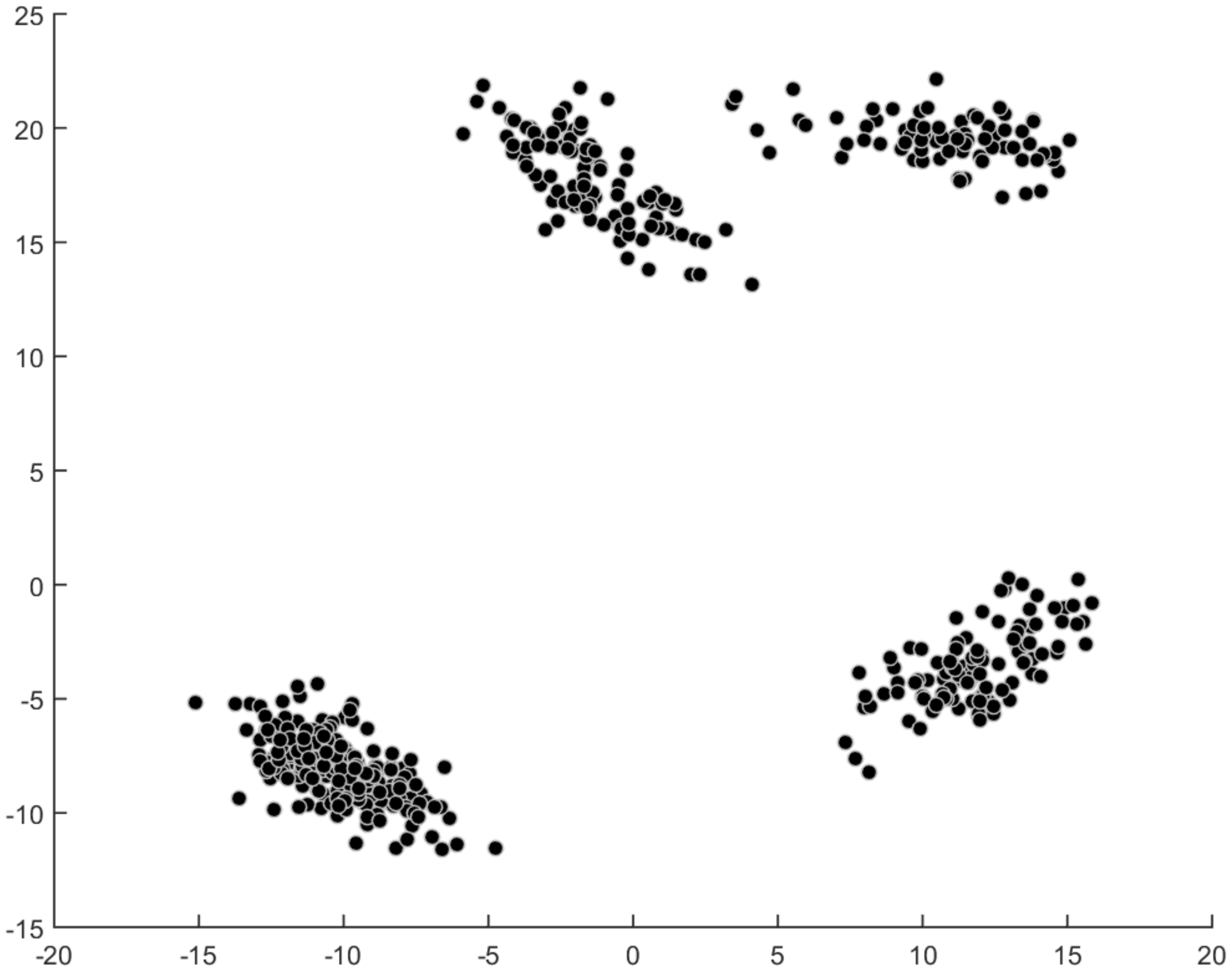
$$p_i \propto d_i^2 \Rightarrow p_i = \frac{d_i^2}{\sum_{j=1}^{n} d_j^2}$$

"probability that we choose x_i as next mean"

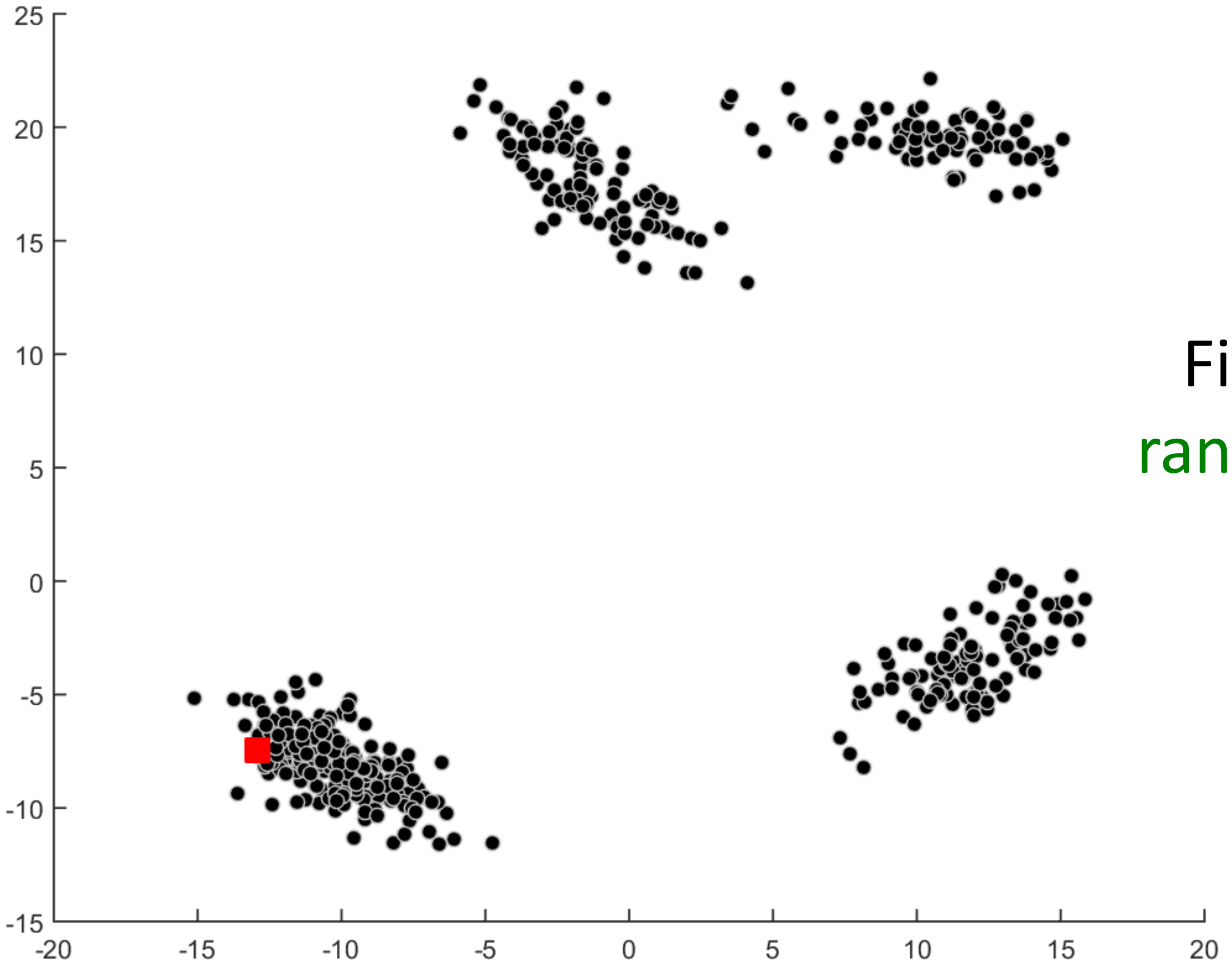
Can be done in $O(n)$.

  5. Keep returning to step 2 until we have k-means.
- Expected approximation ratio is O(log(k)).

K-Means++

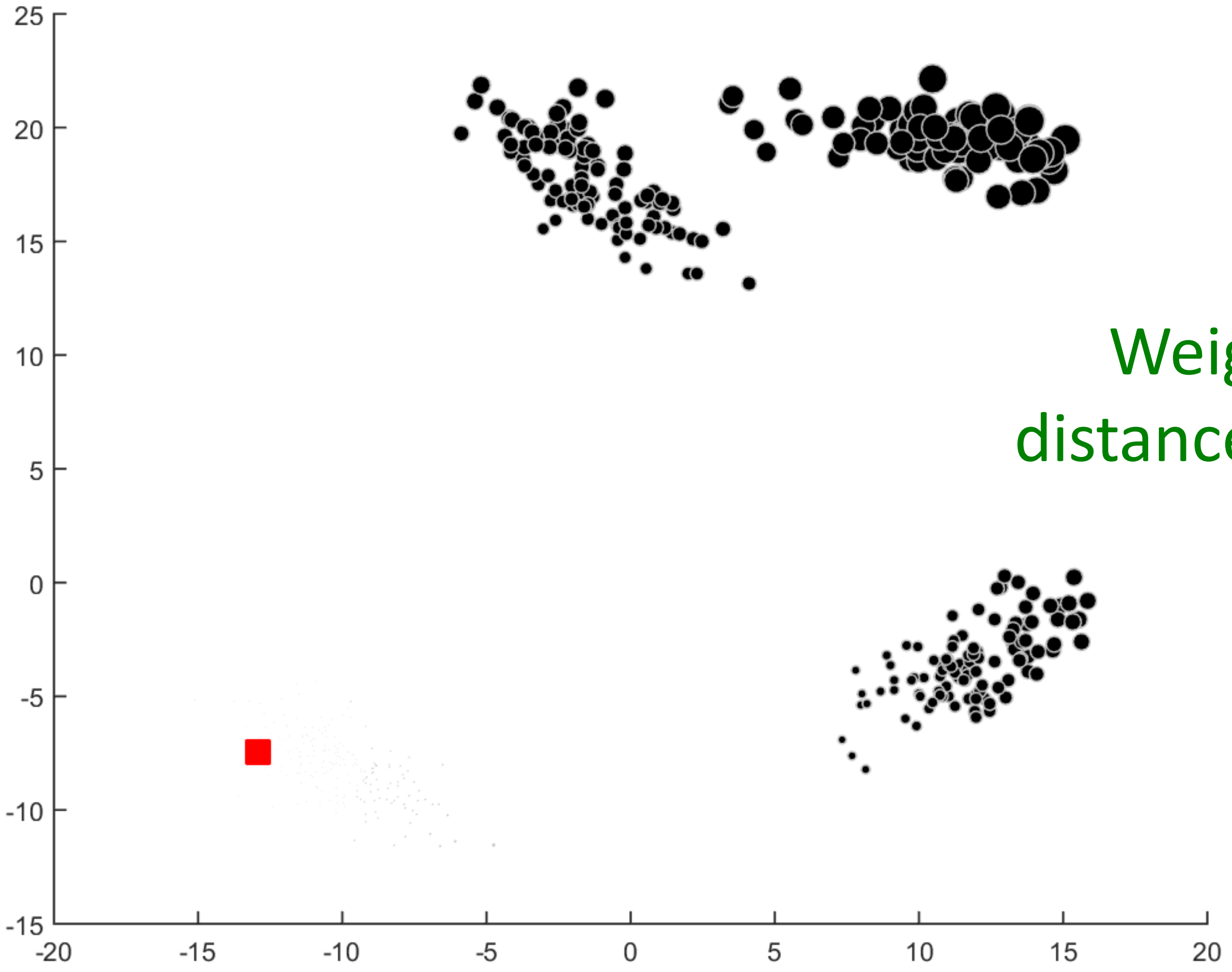


K-Means++



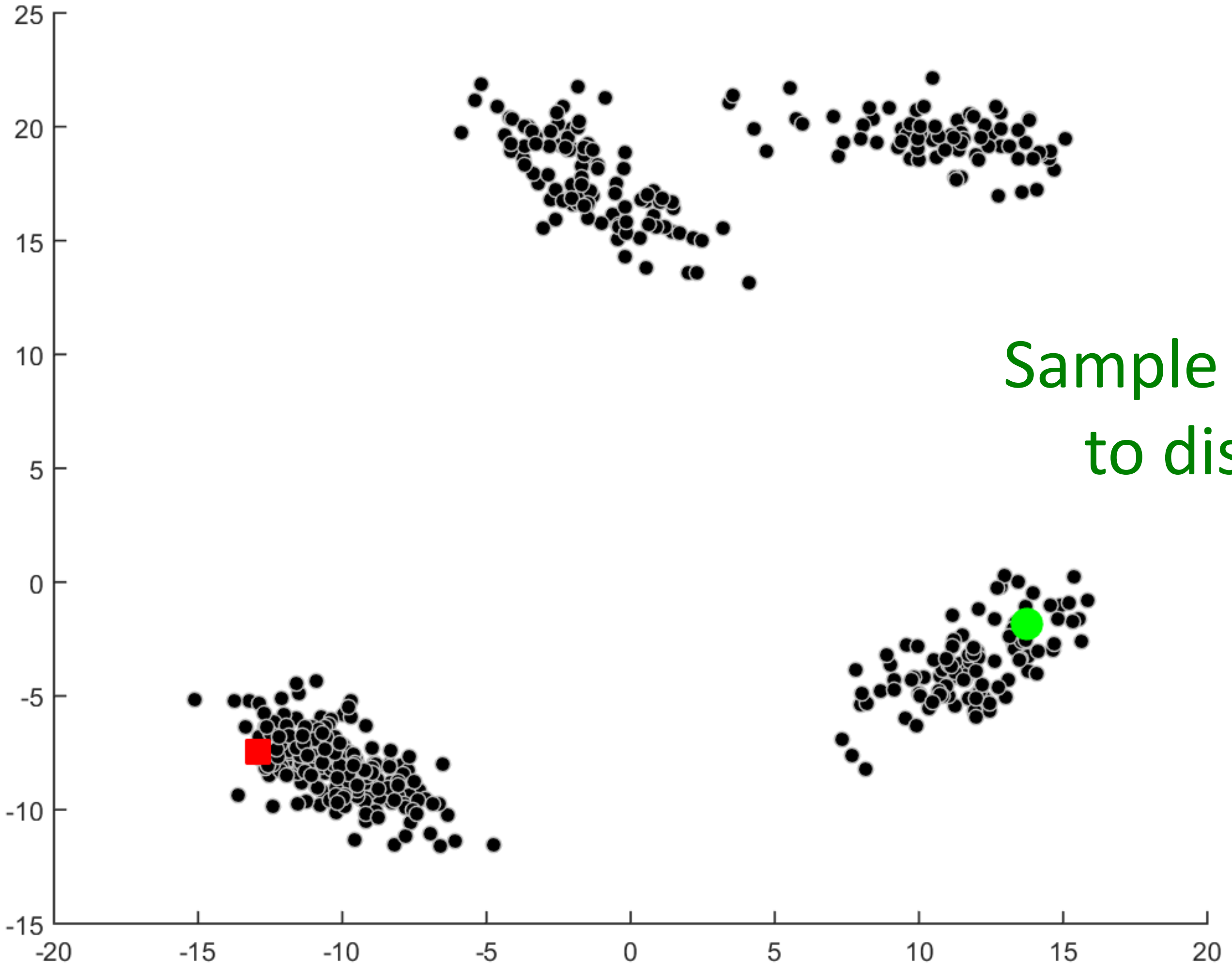
First mean is a random example.

K-Means++



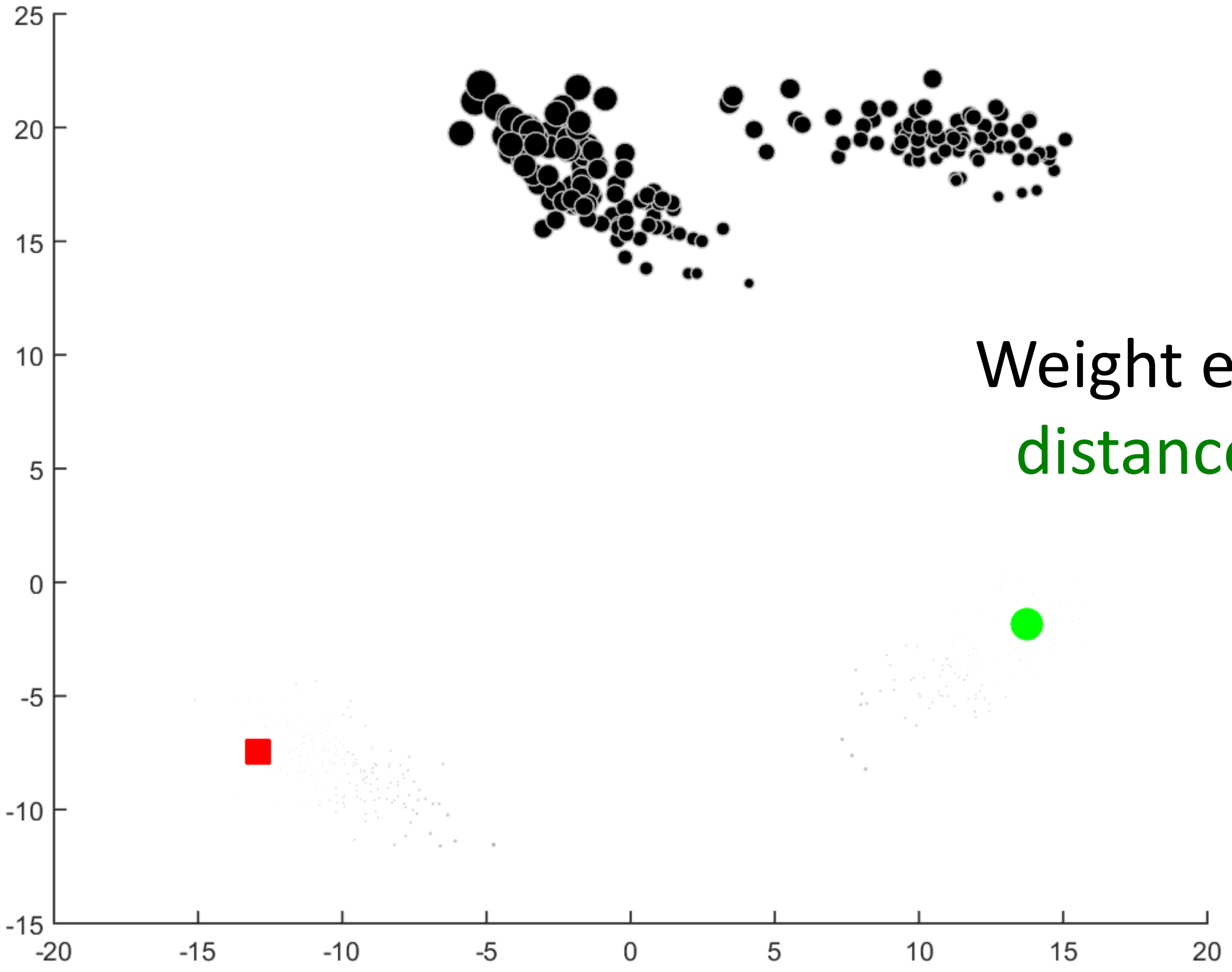
Weight examples by distance to mean squared.

K-Means++



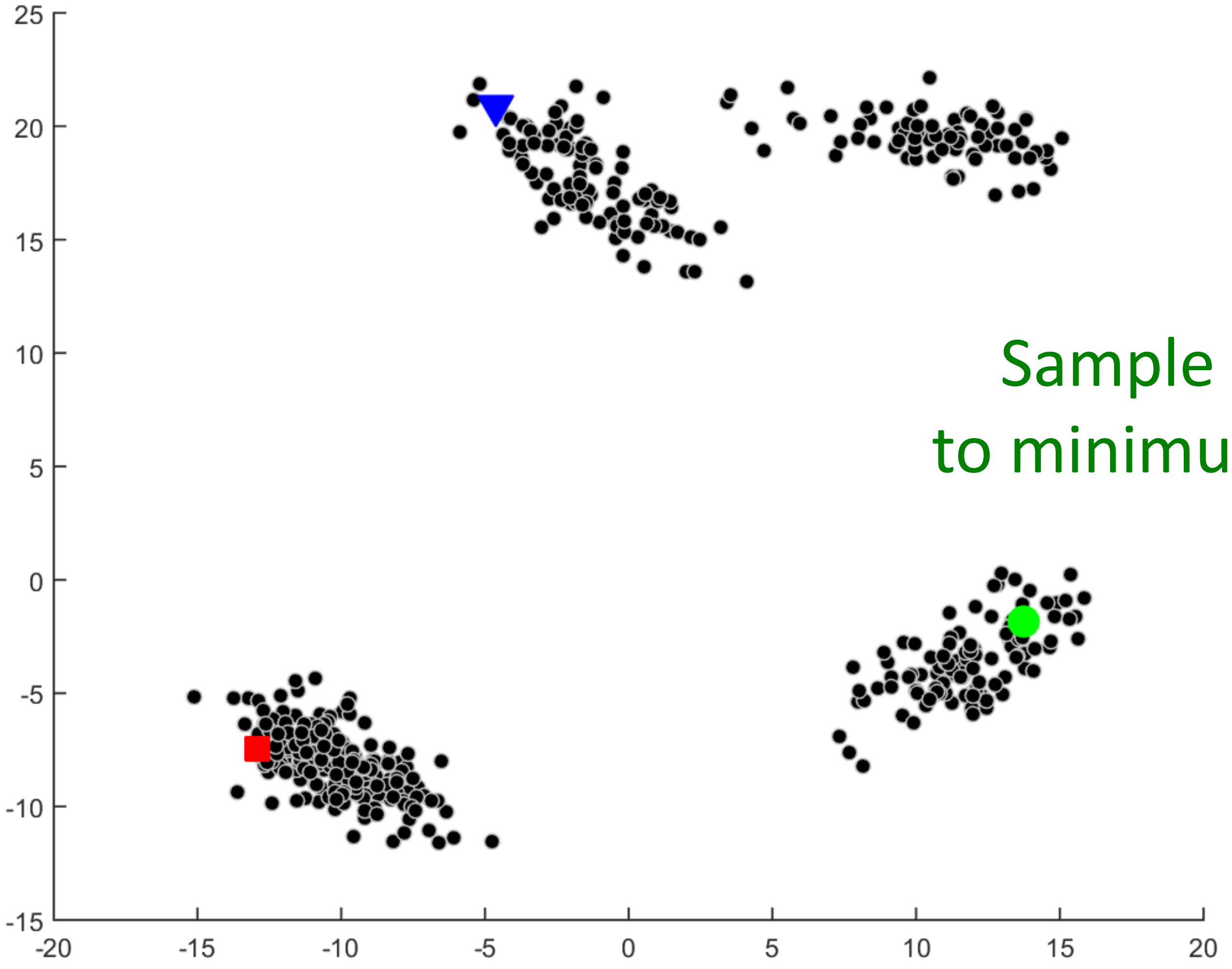
Sample mean proportional to distances squared.

K-Means++



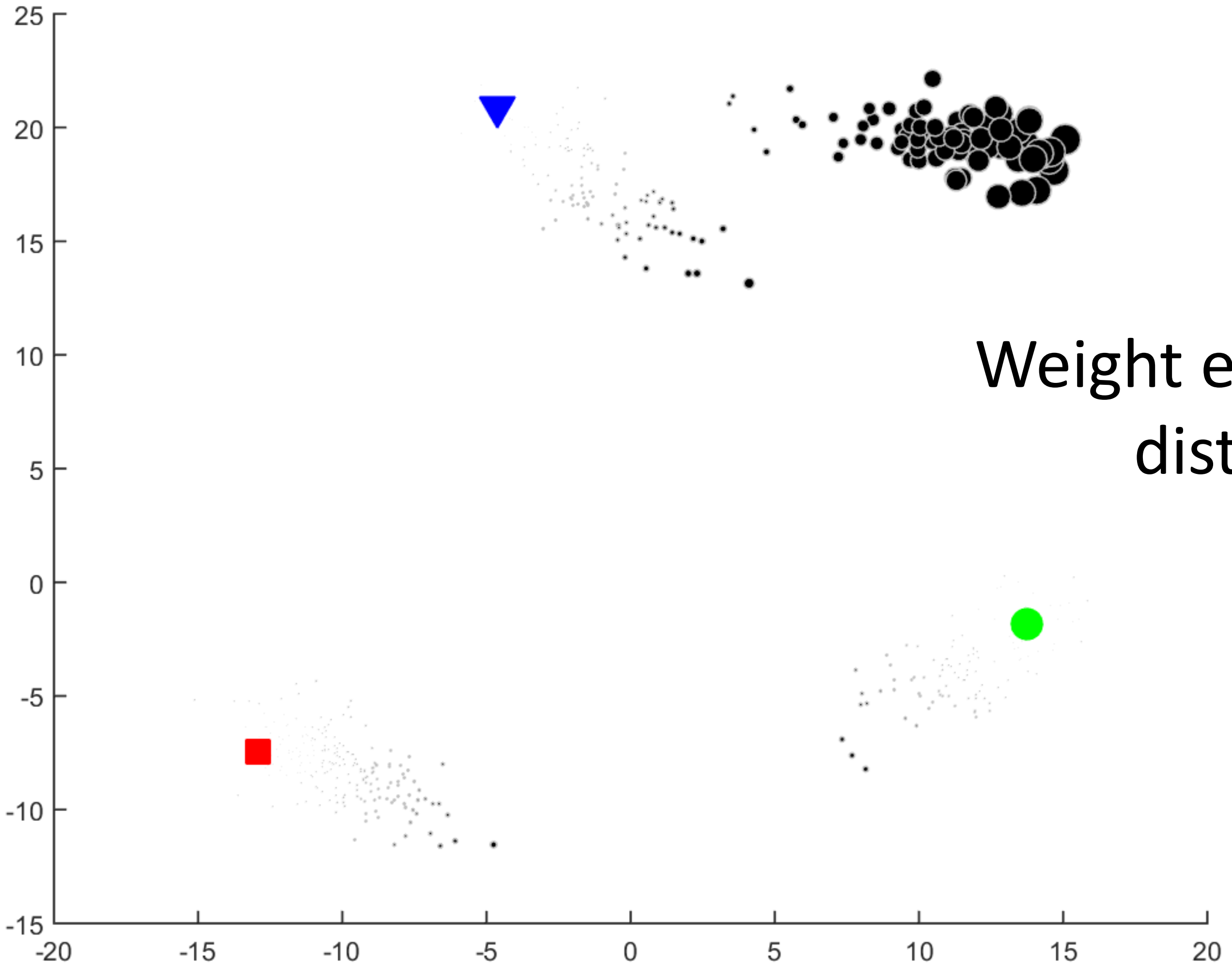
Weight examples by squared distance to nearest mean.

K-Means++



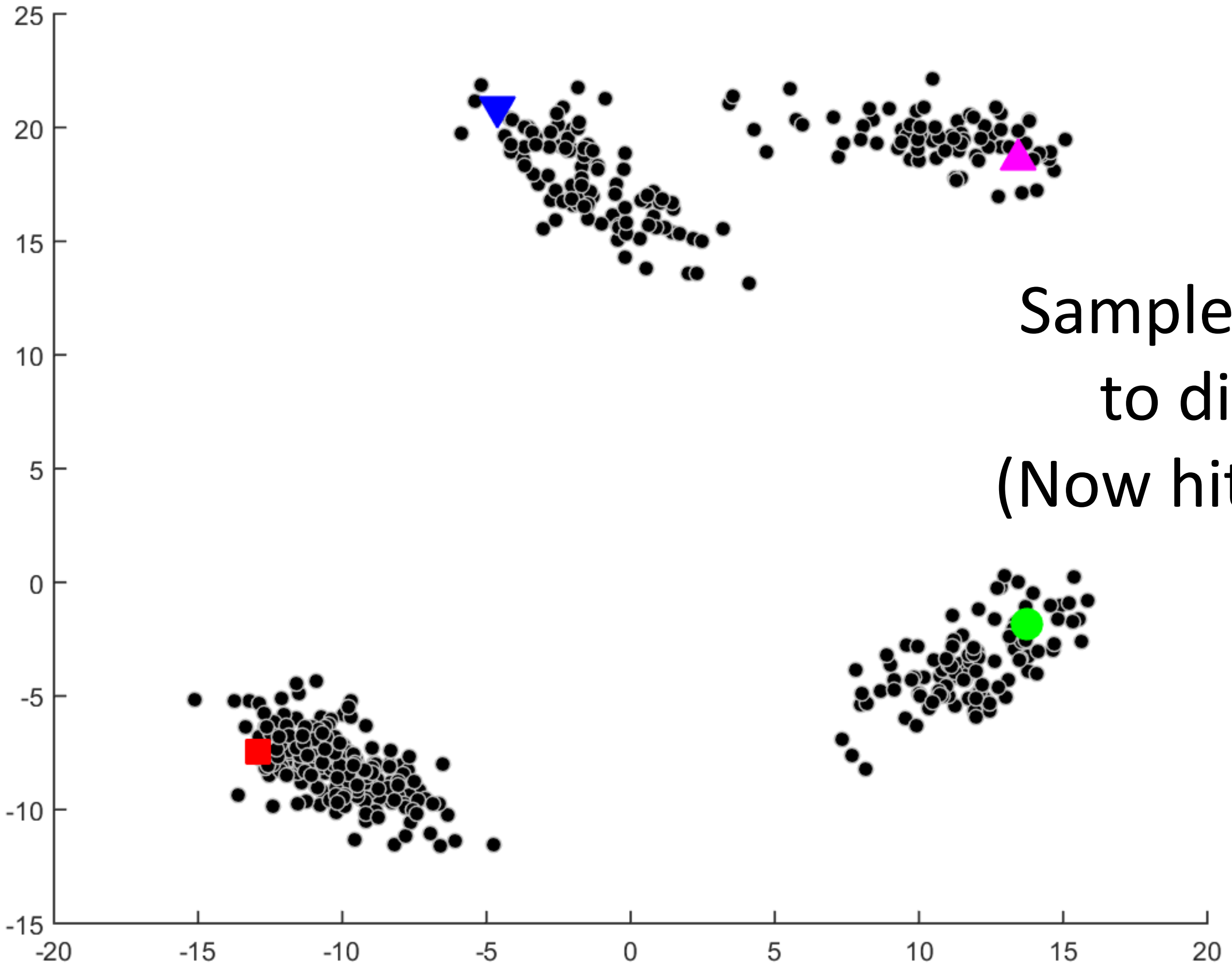
Sample mean proportional
to minimum distances squared.

K-Means++



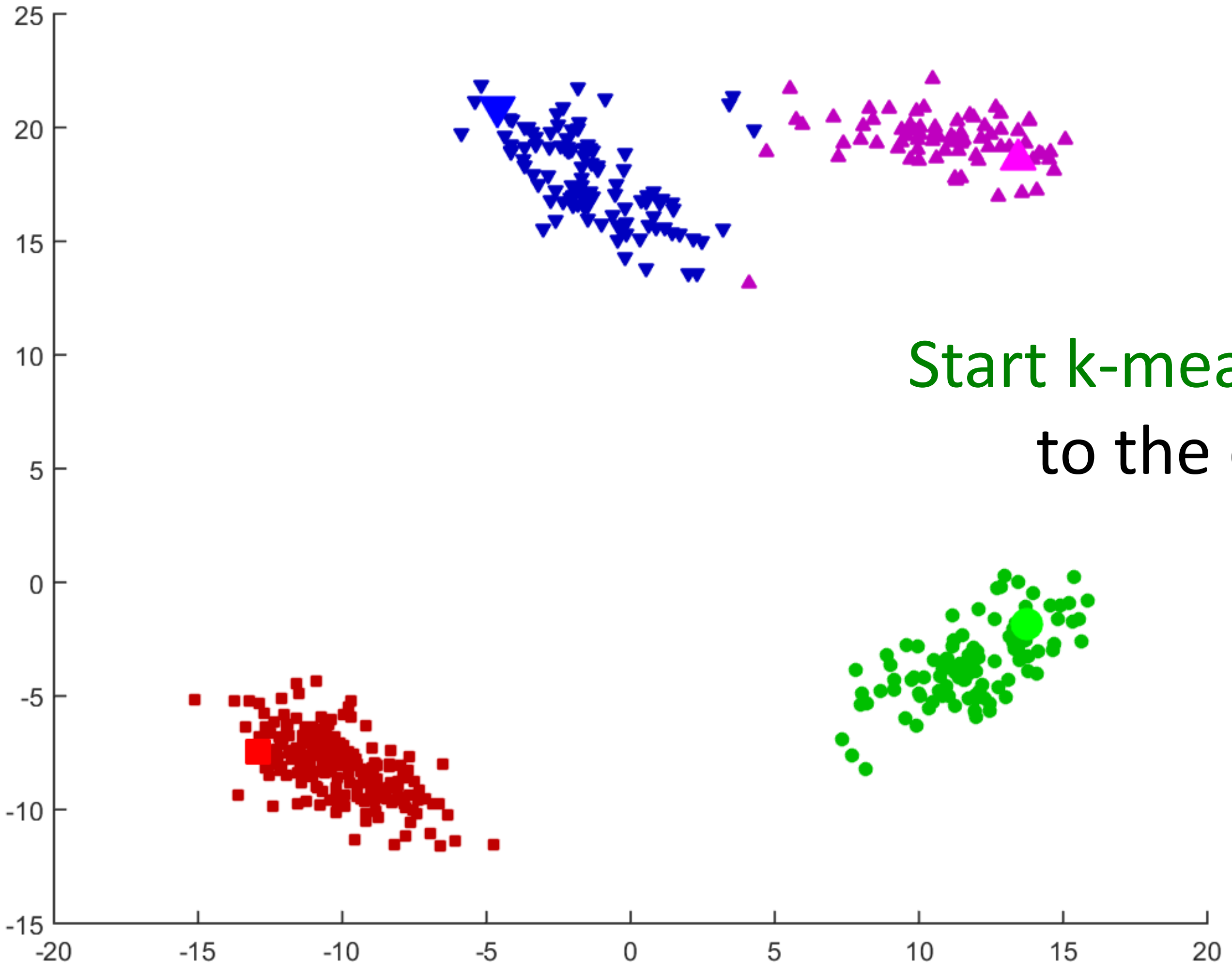
Weight examples by squared distance to mean.

K-Means++



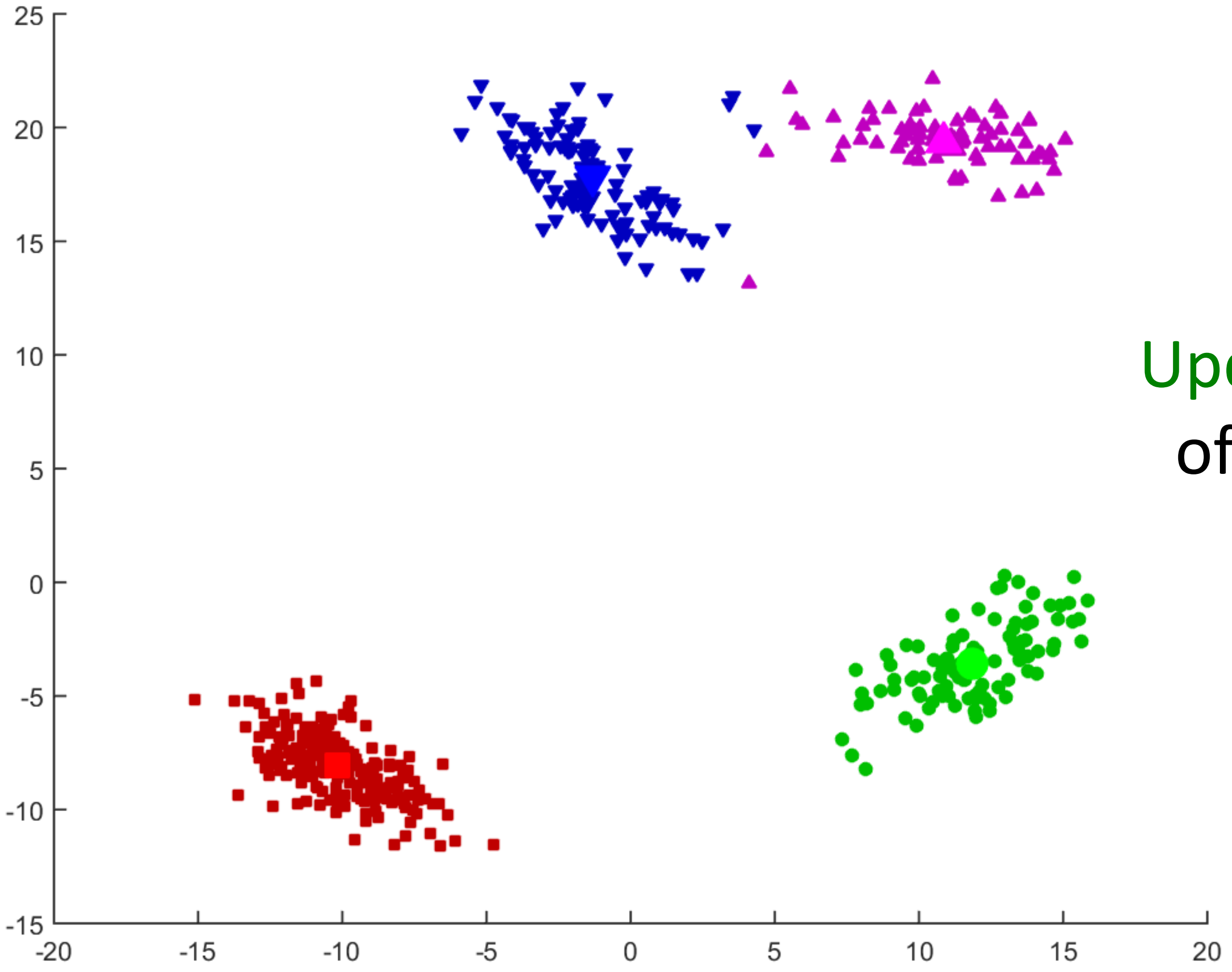
Sample mean proportional to distances squared.
(Now hit chosen target k=4.)

K-Means++



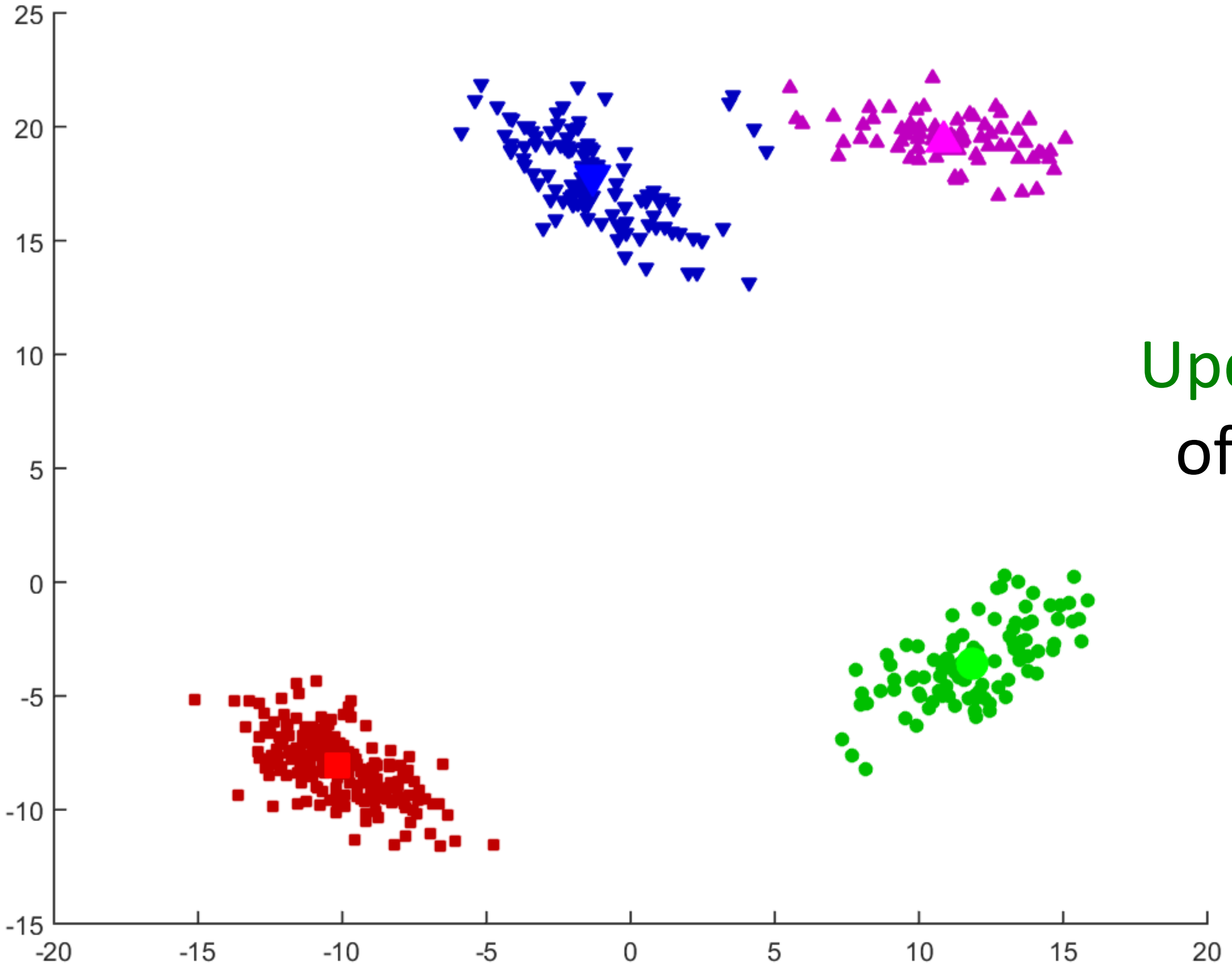
Start k-means: assign objects to the closest mean.

K-Means++



Update the mean of each cluster.

K-Means++



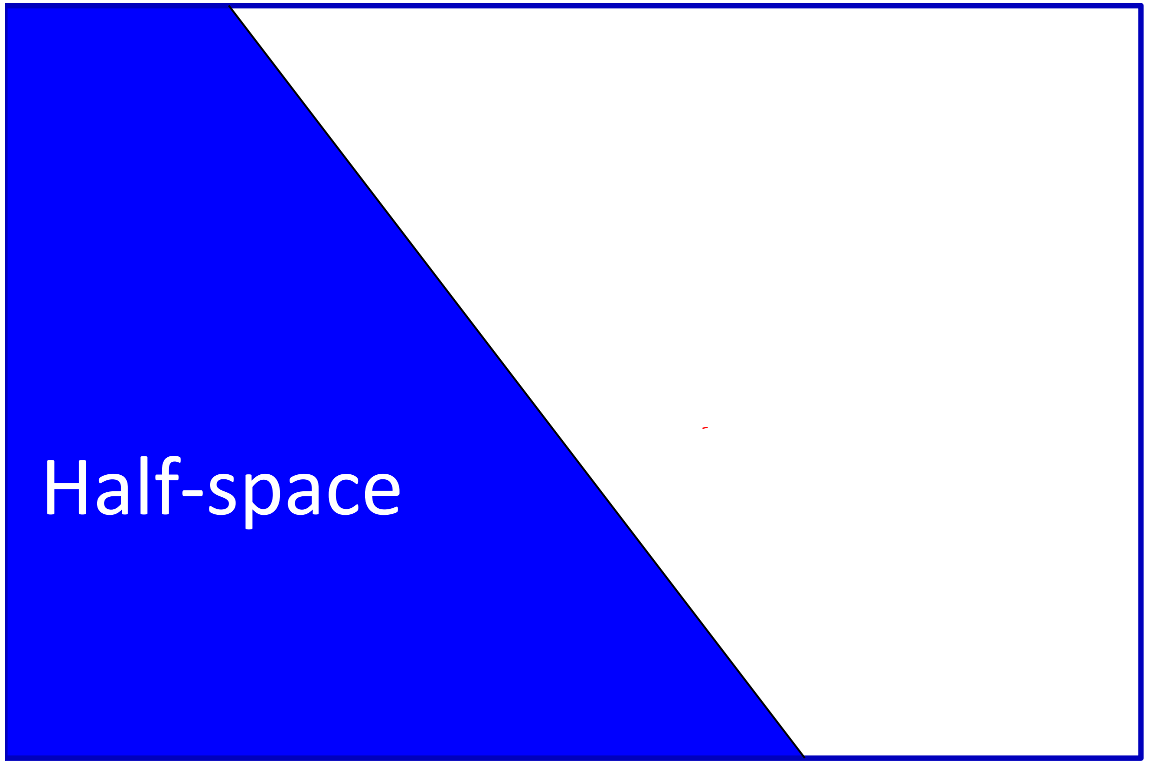
Update the mean
of each cluster.

In this case: just 2 iterations!

Shape of K-Means Clusters

- K-means clusters are formed by the intersection of half-spaces.

Half-space is a set of points satifying a linear inequality 'x' like $\sum_{j=1}^{d} a_j x_j \leq b$

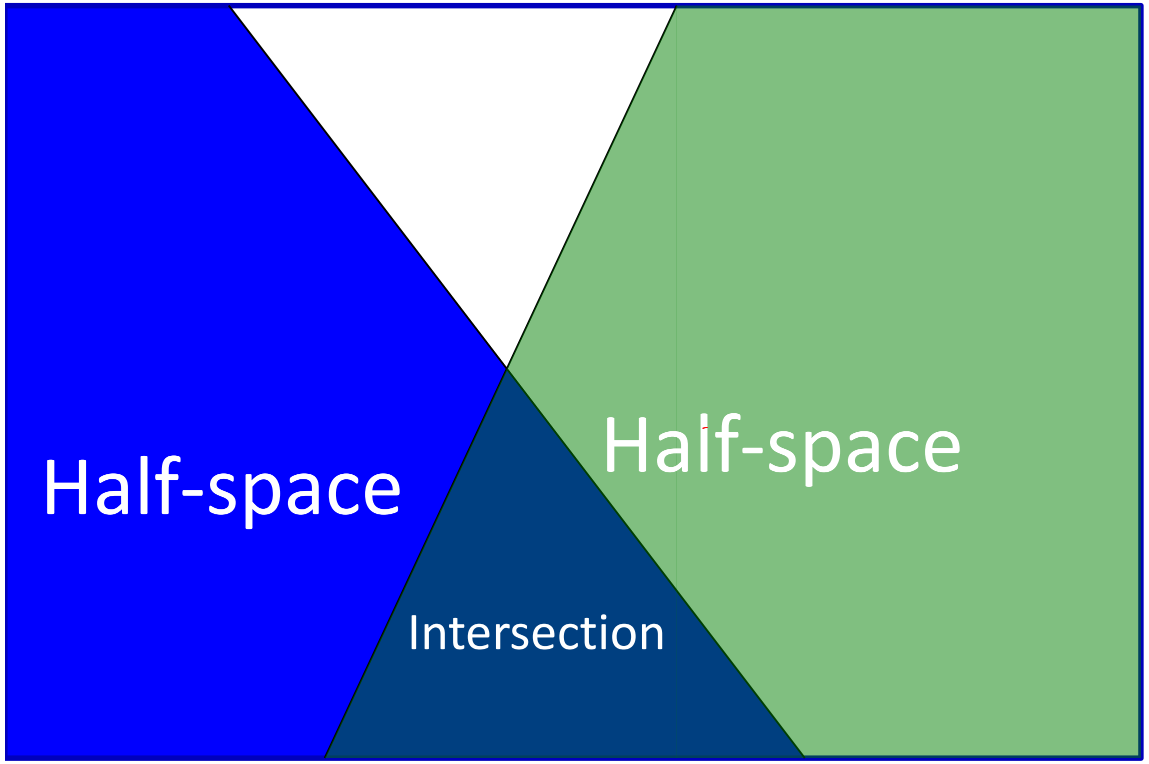


Shape of K-Means Clusters

- K-means clusters are formed by the intersection of half-spaces.

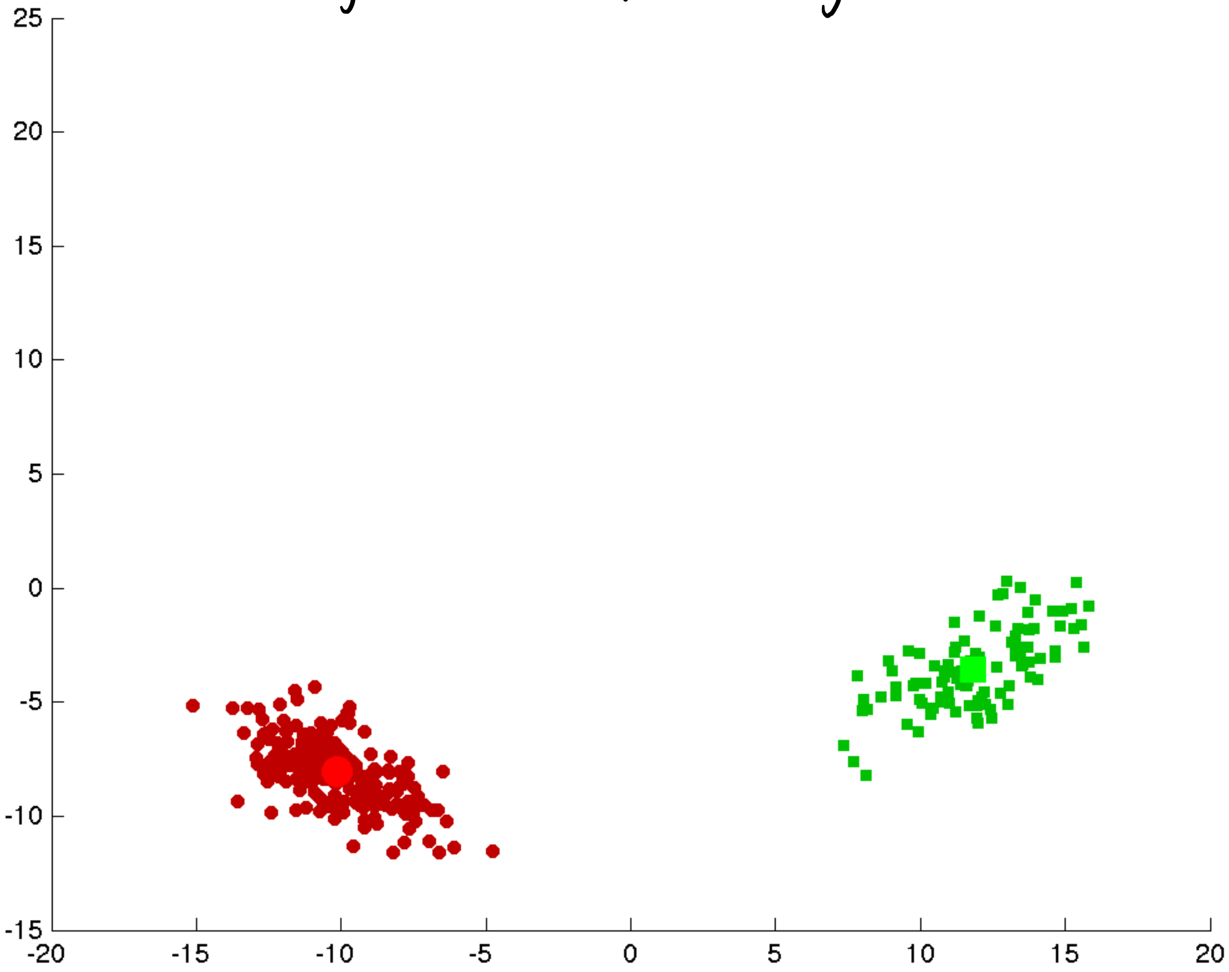
Half-space is a set of points satifying a linear inequality, like $\sum_{j=1}^{d} a_j x_j \leq b$

$a^T x$



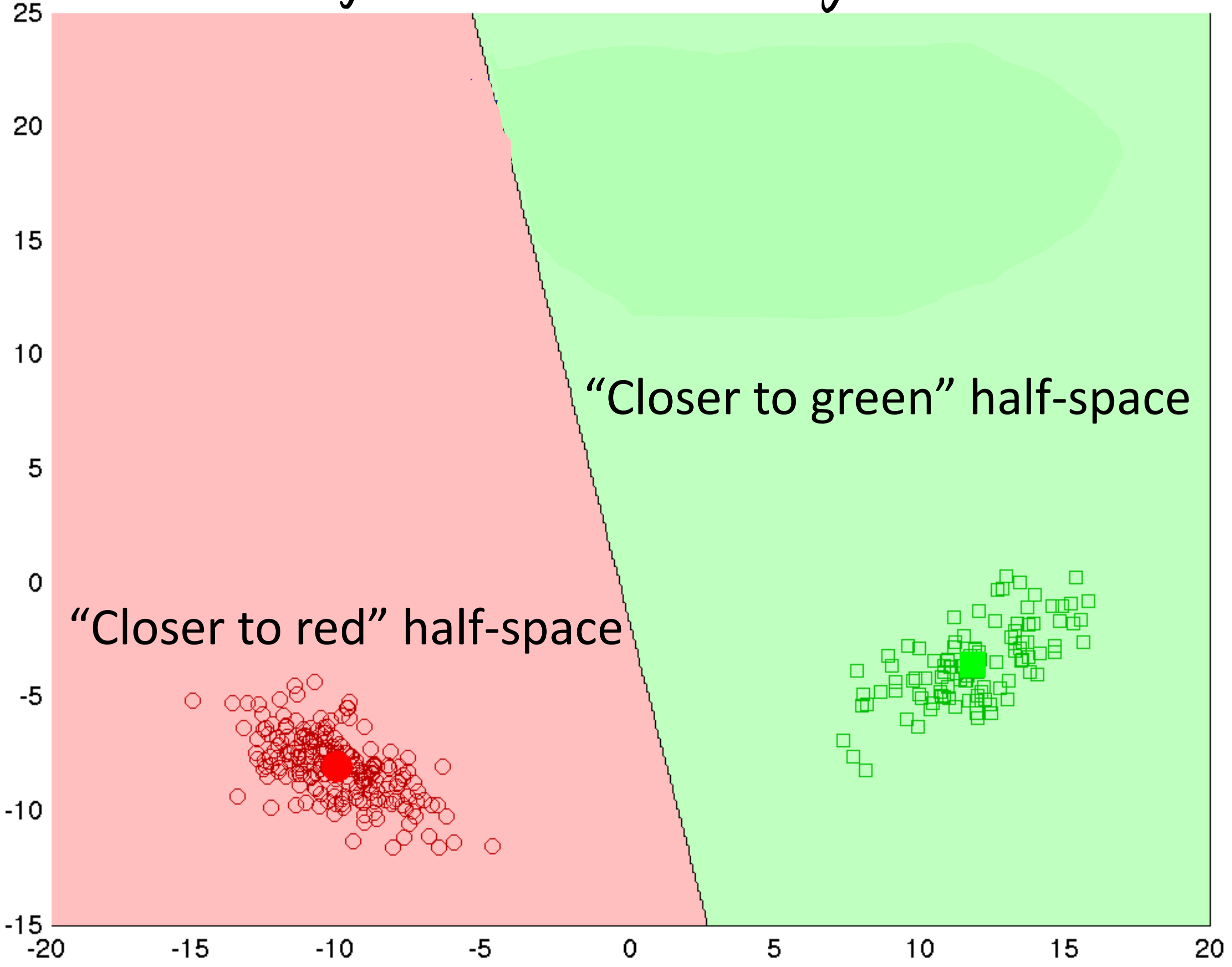
Shape of K-Means Clusters

Which regions are put in green cluster?



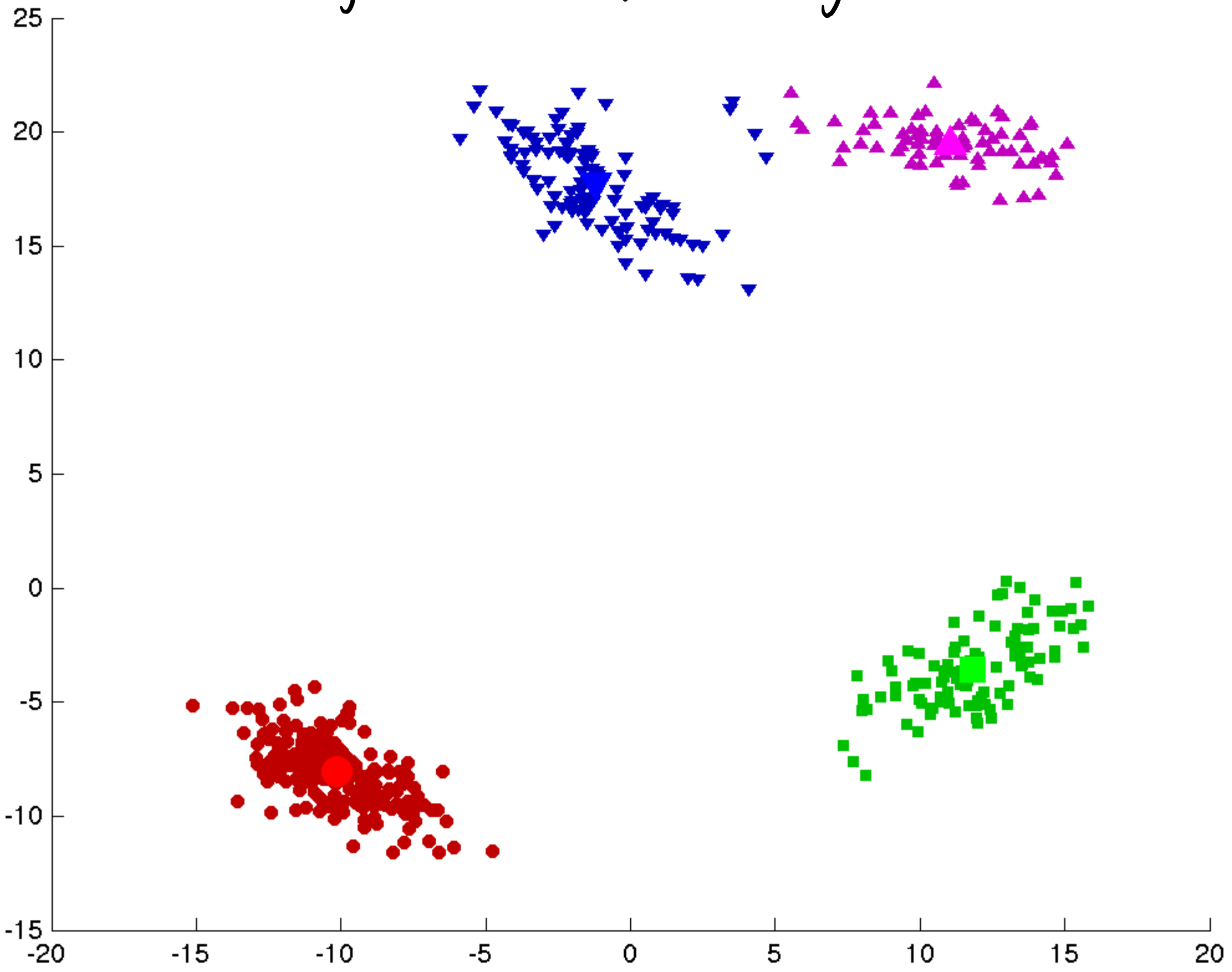
Shape of K-Means Clusters

Which regions are put in green cluster?



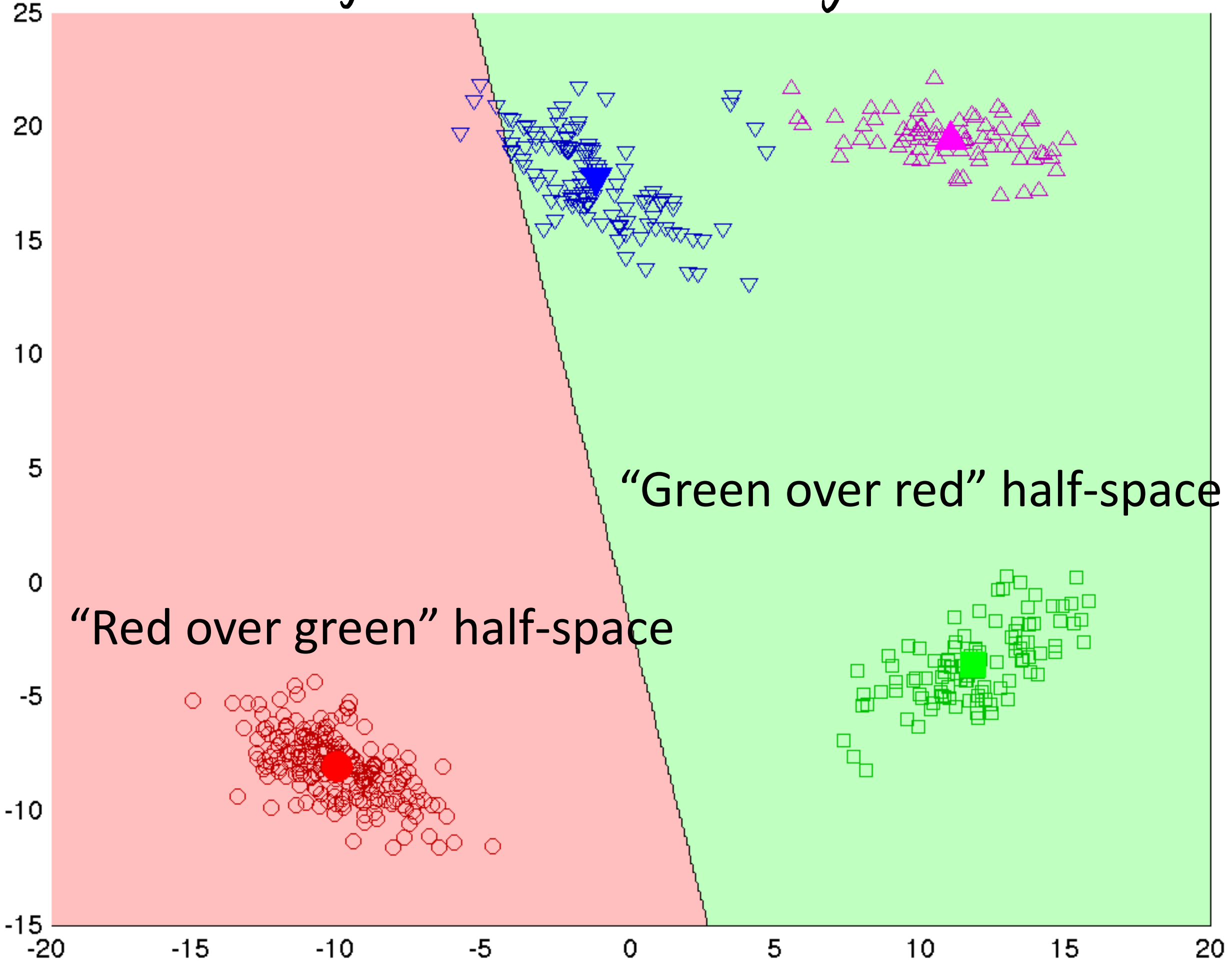
Shape of K-Means Clusters

Which regions are put in green cluster?



Shape of K-Means Clusters

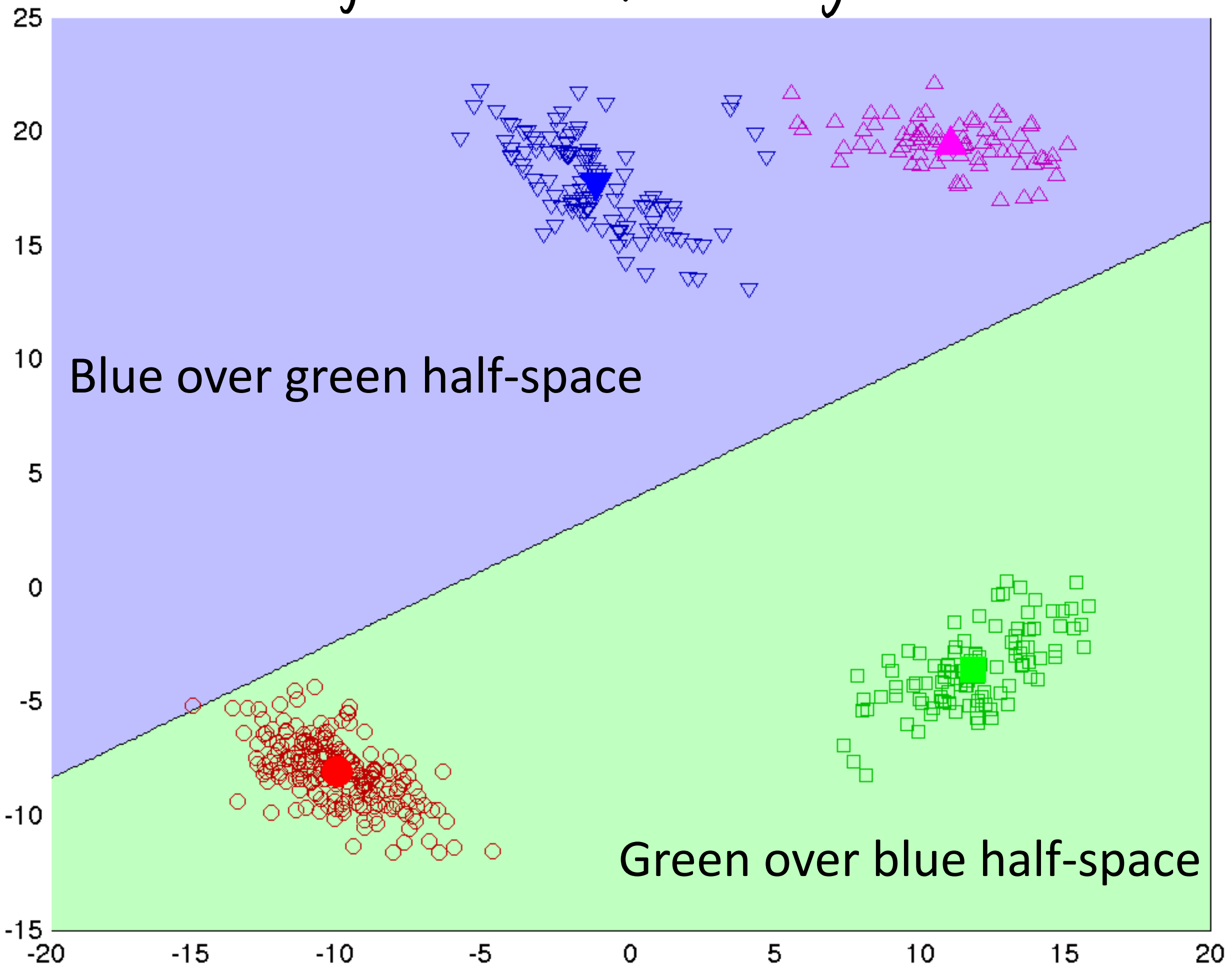
Which regions are put in green cluster?



Red vs. green decision stays the same with more clusters.

Shape of K-Means Clusters

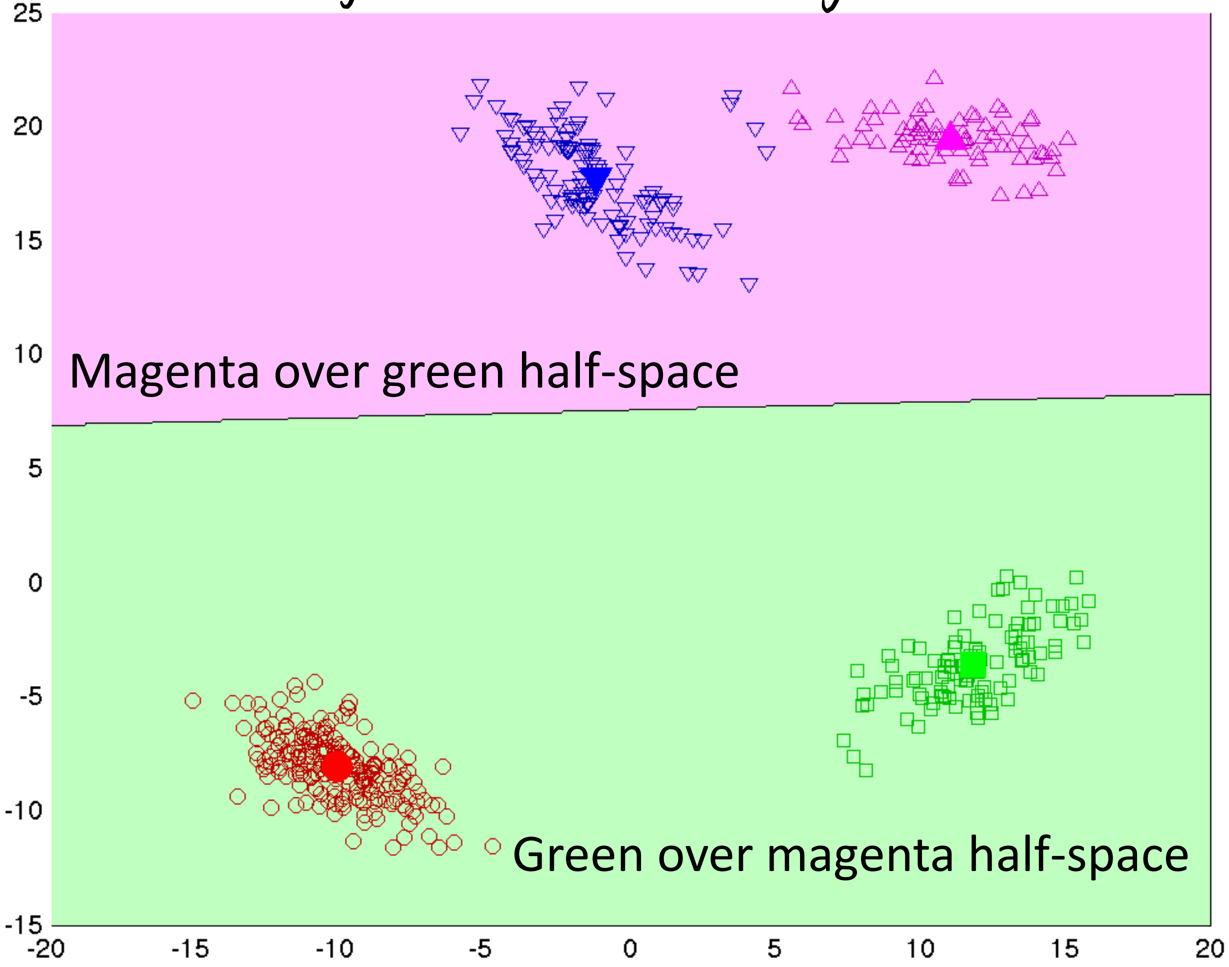
Which regions are put in green cluster?



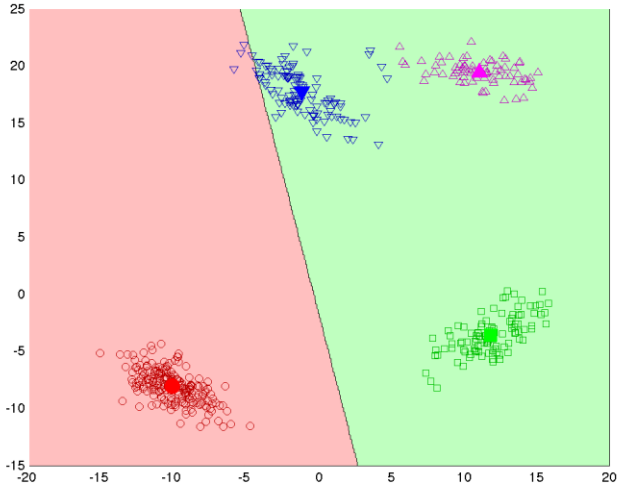
Blue vs. green decision defines different half-spaces.

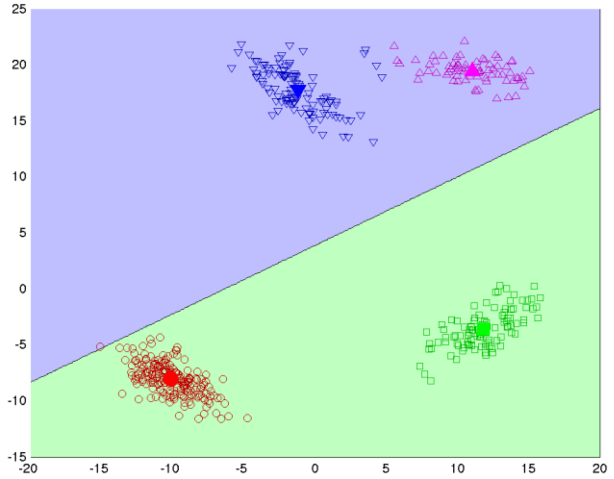
Shape of K-Means Clusters

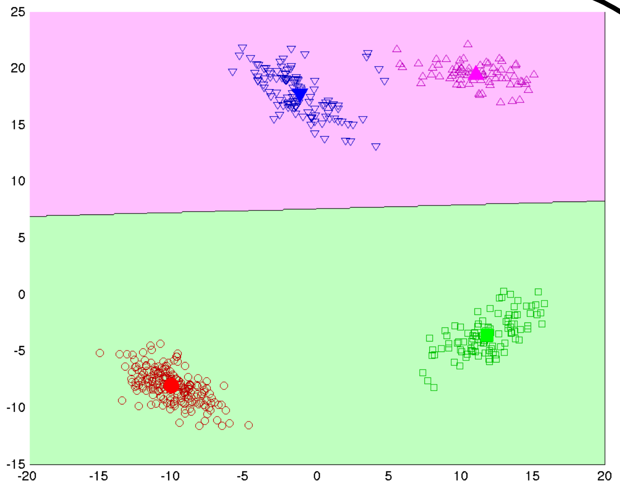
Which regions are put in green cluster?



Shape of K-Means Clusters

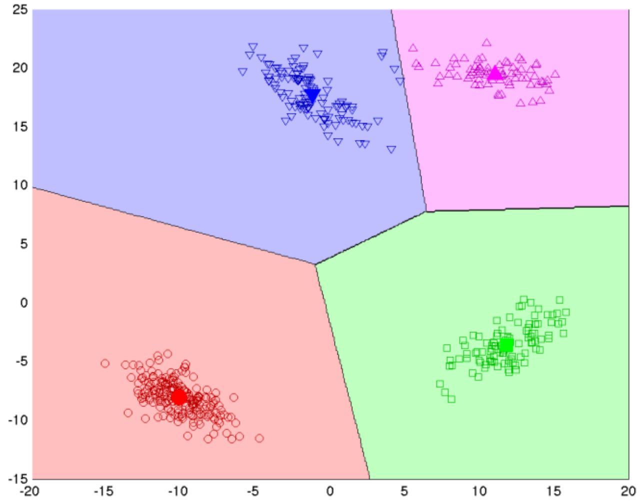






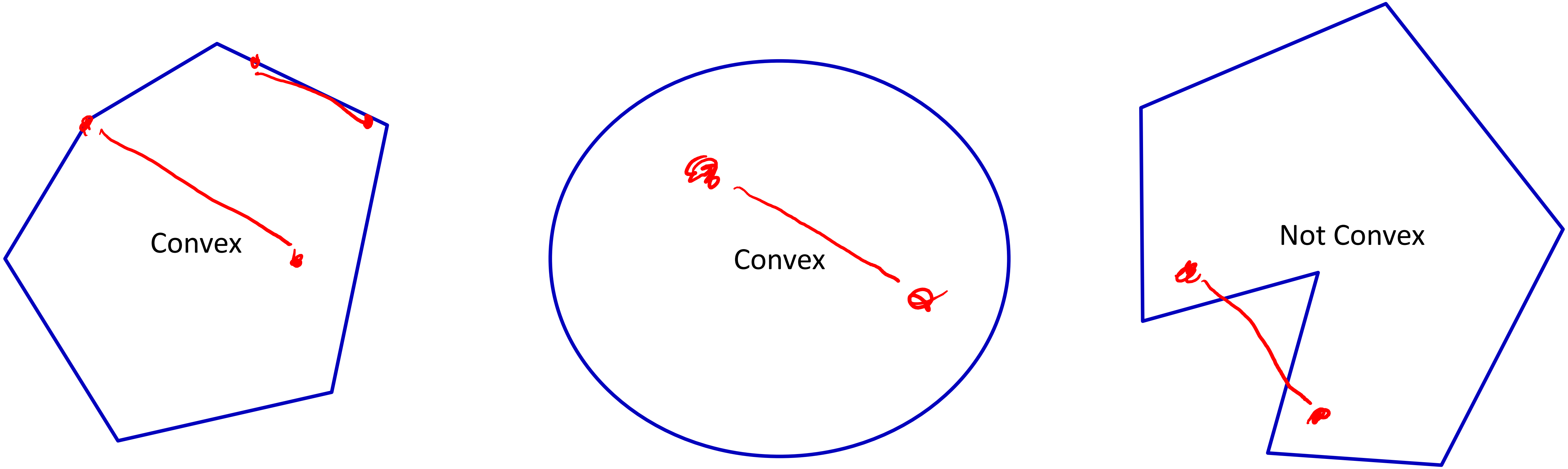
Green "cluster" is the intersection of these three half-spaces.

Here is what the four clusters look like:



Shape of K-Means Clusters

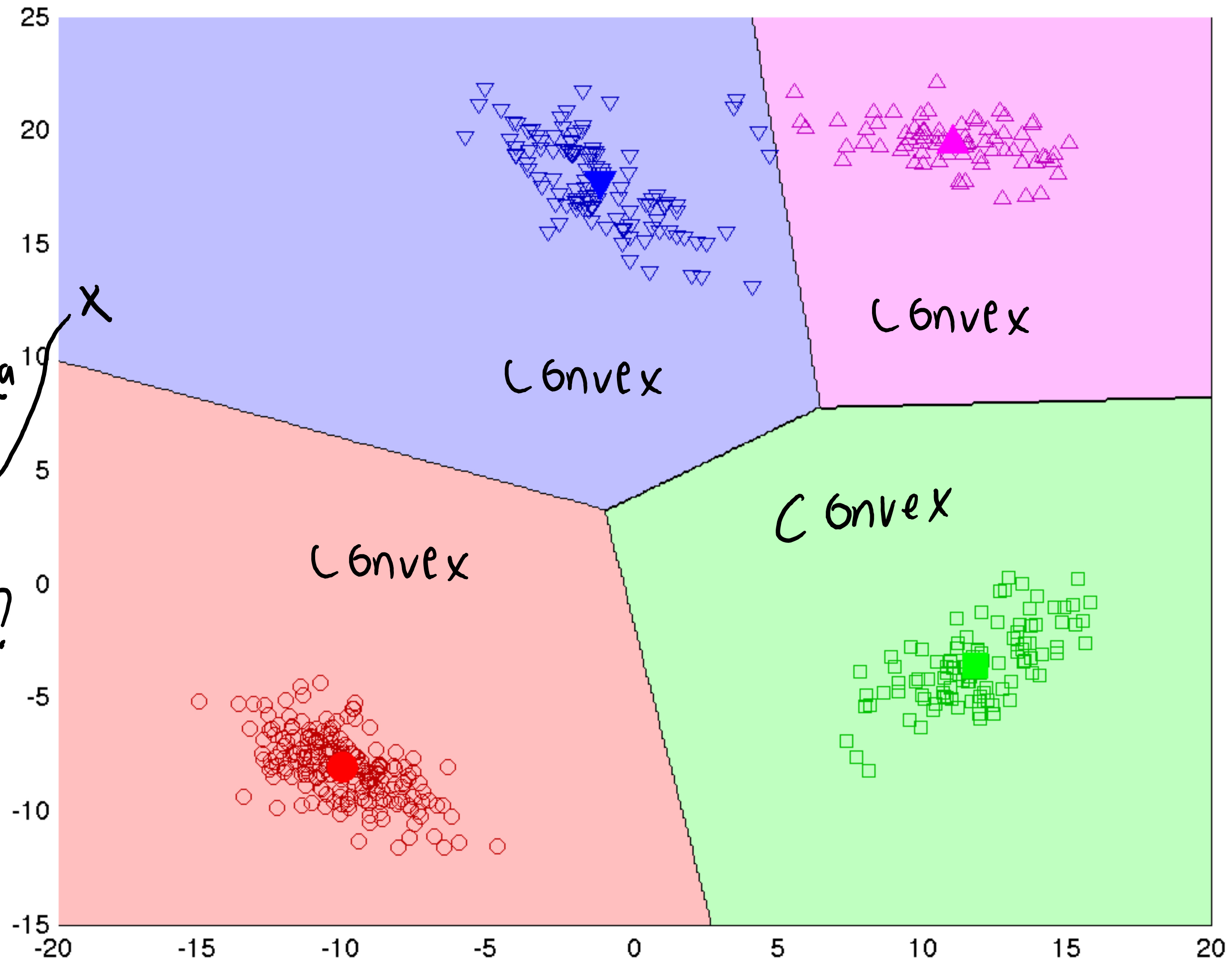
- Intersection of half-spaces form a convex set:
  - Line between any two points in the set stays in the set.



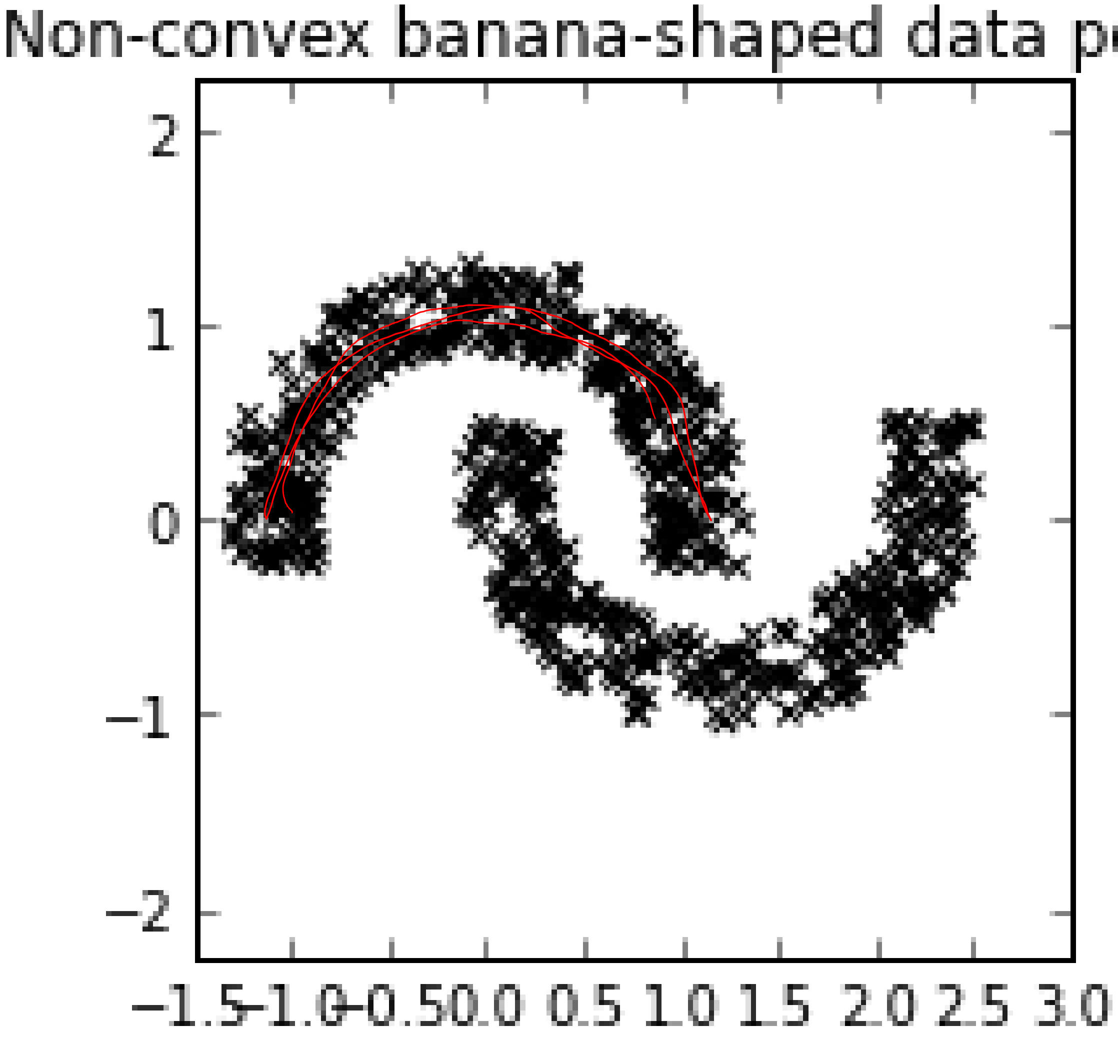
Shape of K-Means Clusters

New issues:

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

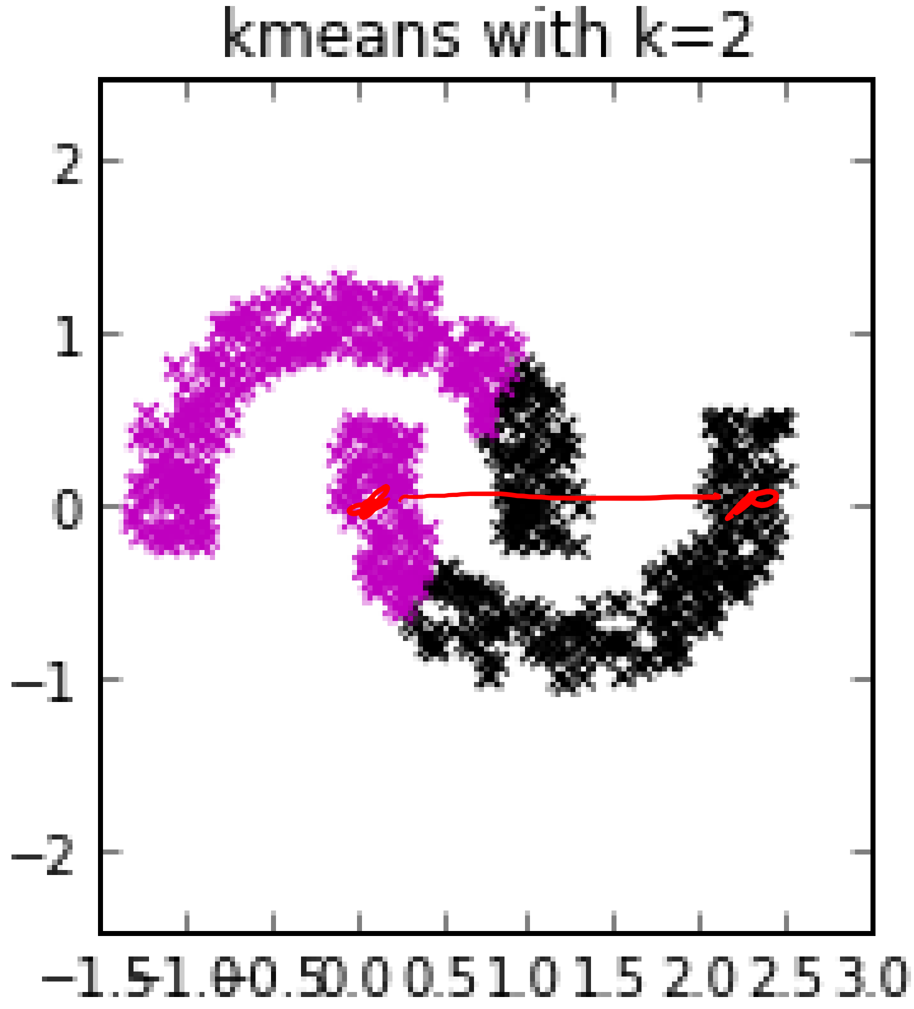


K-Means with Non-Convex Clusters



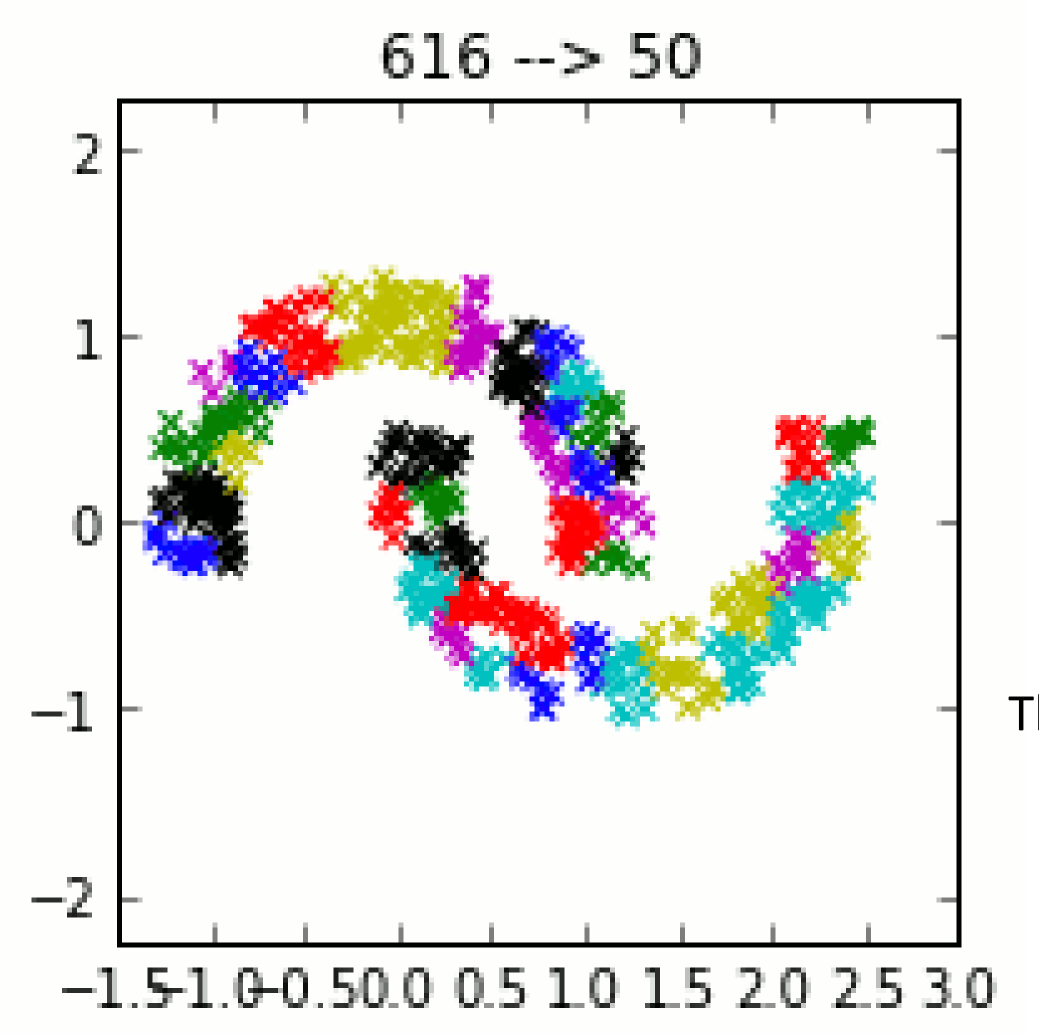
https://corelifesciences.com/human-long-non-coding-rna-expression-microarray-service.html

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)

https://corelifesciences.com/human-long-non-coding-rna-expression-microarray-service.html

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.

http://www.defenders.org/elephant/basic-facts

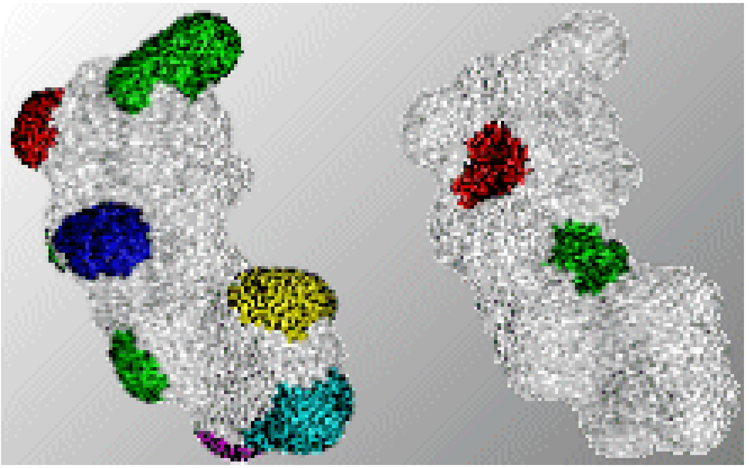
Motivation for Density-Based Clustering

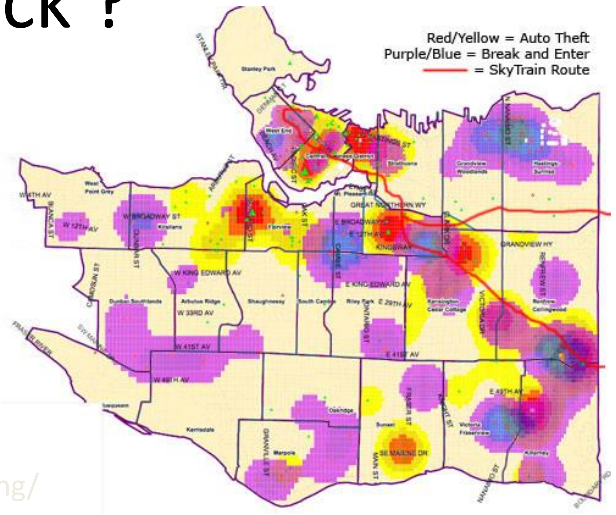
- Density-based clustering:
  - Clusters are defined by all the objects in “dense” regions.
  - Objects in non-dense regions don’t get clustered.
- It’s a non-parametric clustering method:
  - Clusters can become more complicated the more data we have.
  - No fixed number of clusters ‘k’.

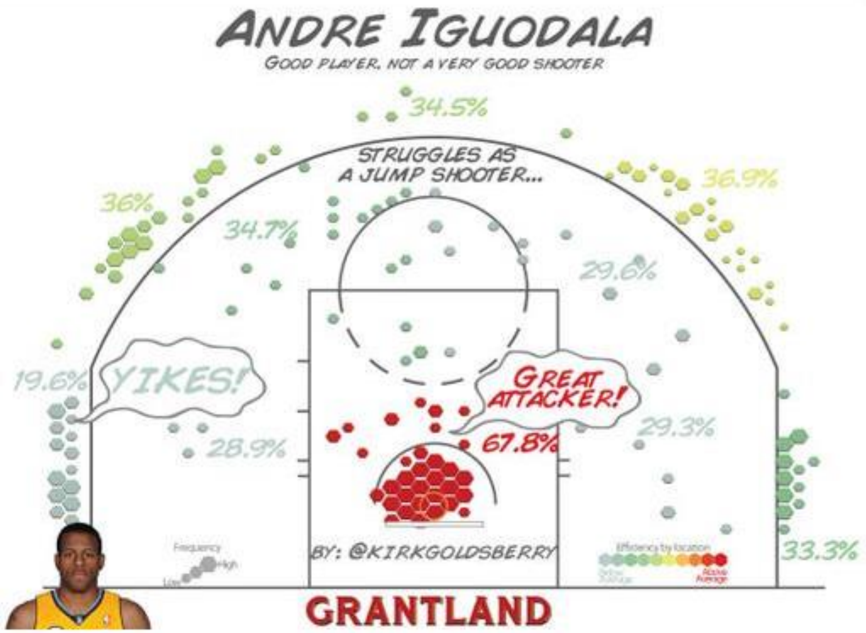
http://www.defenders.org/elephant/basic-facts

Other Potential Applications

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



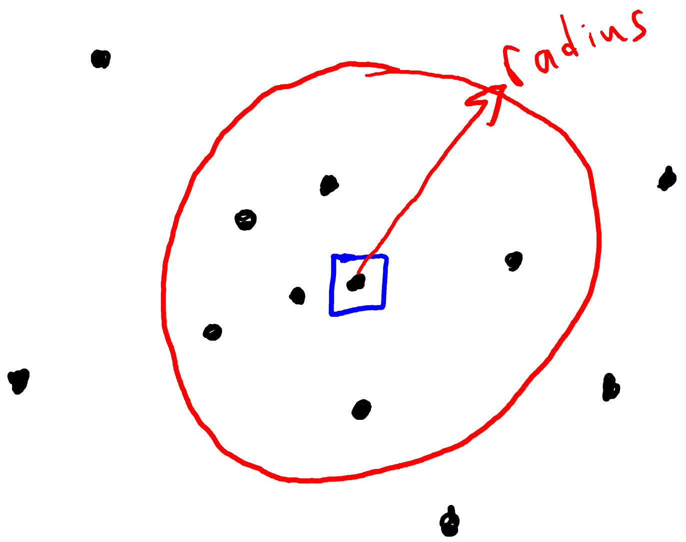




https://en.wikipedia.org/wiki/Cluster_analysis
https://www.flickr.com/photos/dbarefoot/420194128/
http://letsgowarriors.com/replacing-jarrett-jack/2013/10/04/
http://www.dbs.informatik.uni-muenchen.de/Forschung/KDD/Clustering/

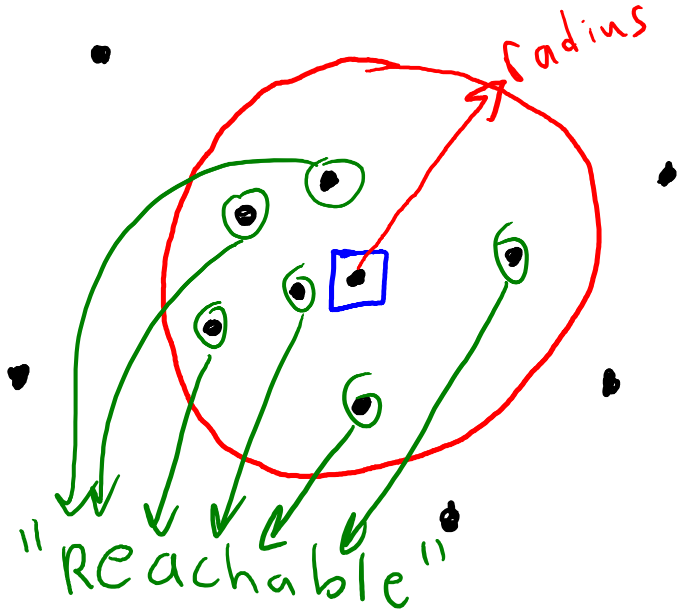
Density-Based Clustering

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



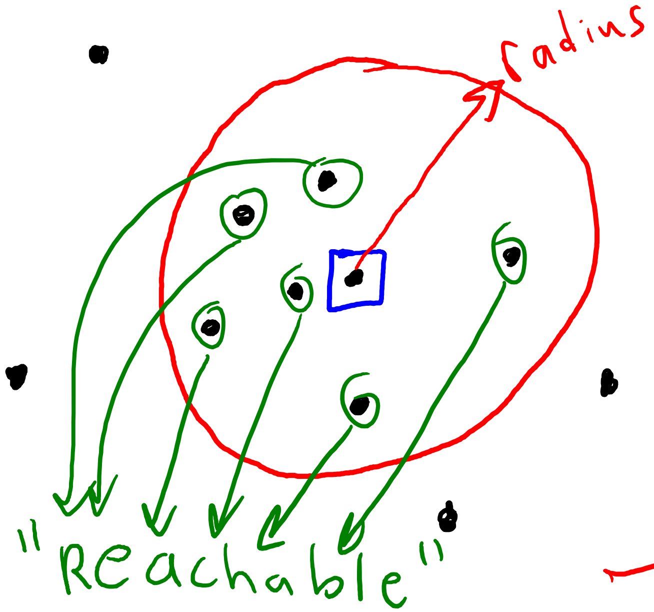
Density-Based Clustering

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



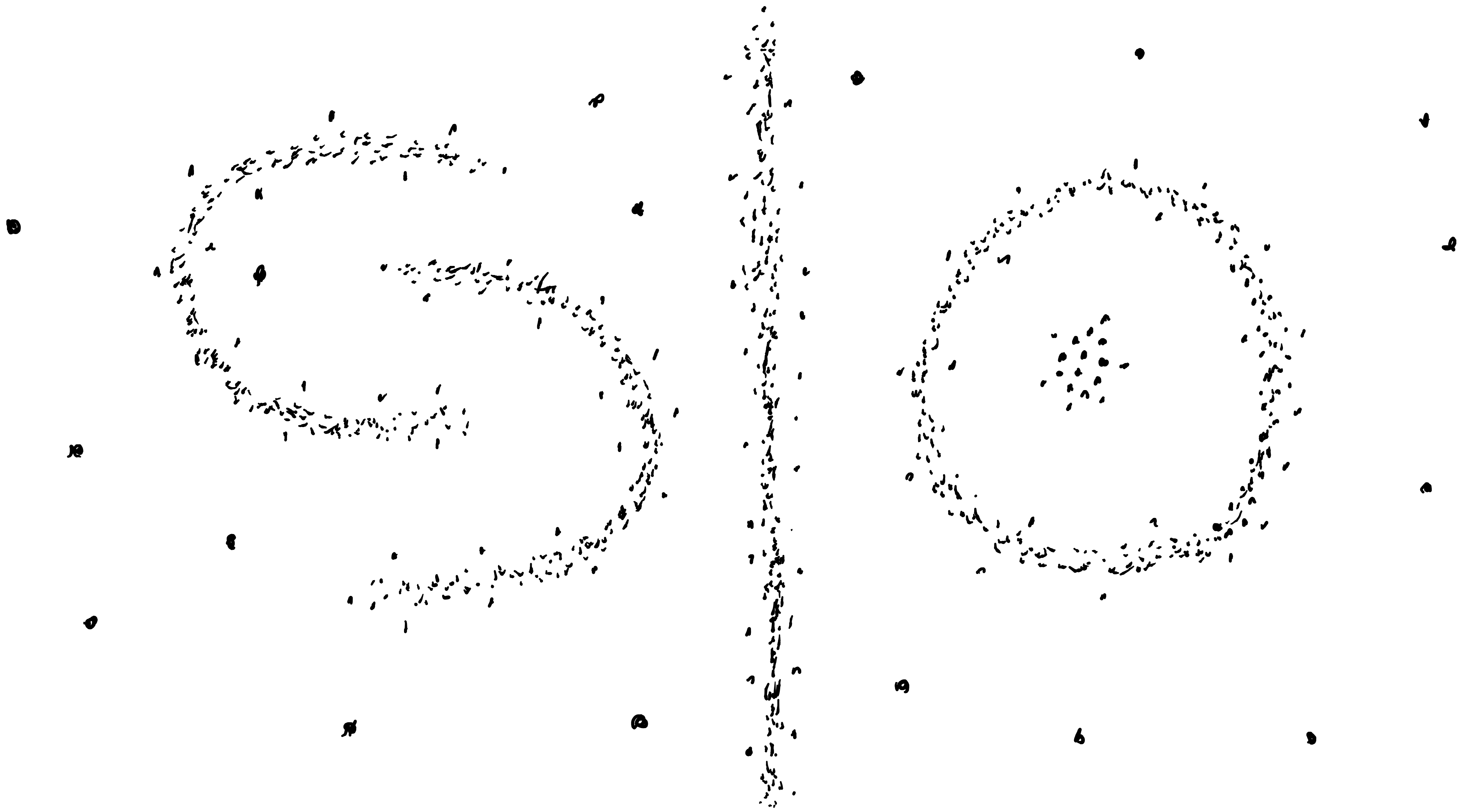
Density-Based Clustering

- 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.

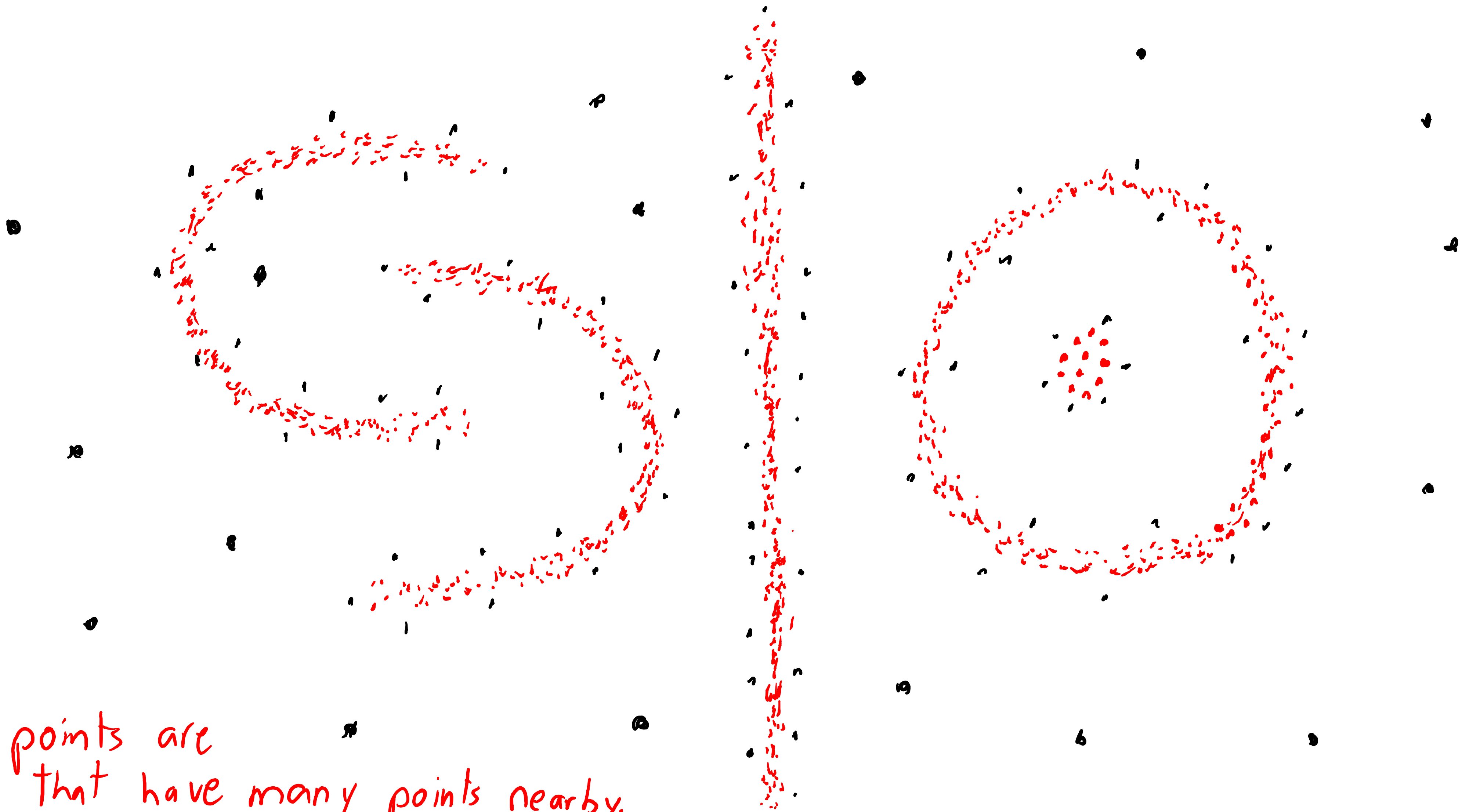


E.g., if minPoints = 3 then this is a "core" point since 6 points are "reachable"

Density-Based Clustering

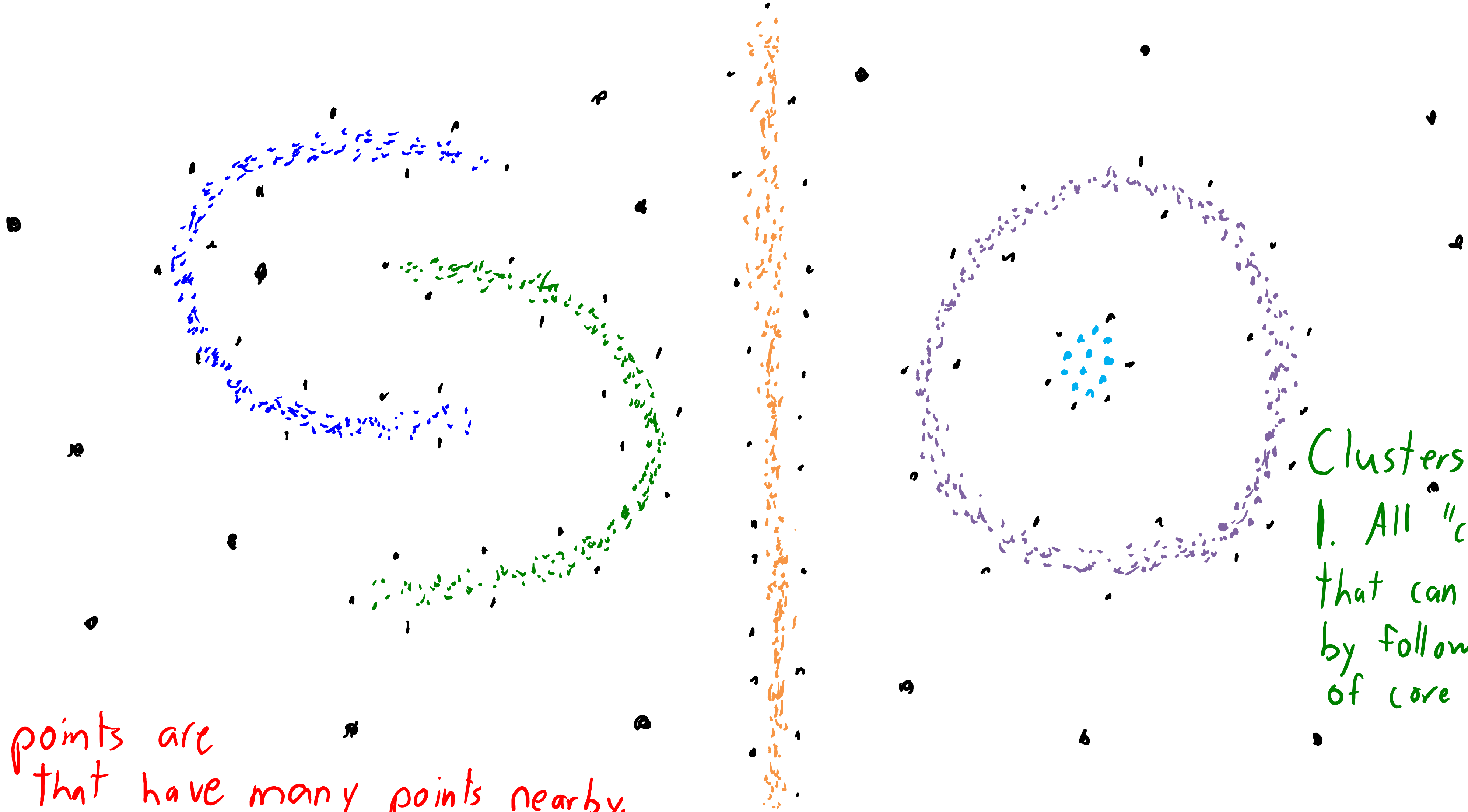


Density-Based Clustering



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

Density-Based Clustering

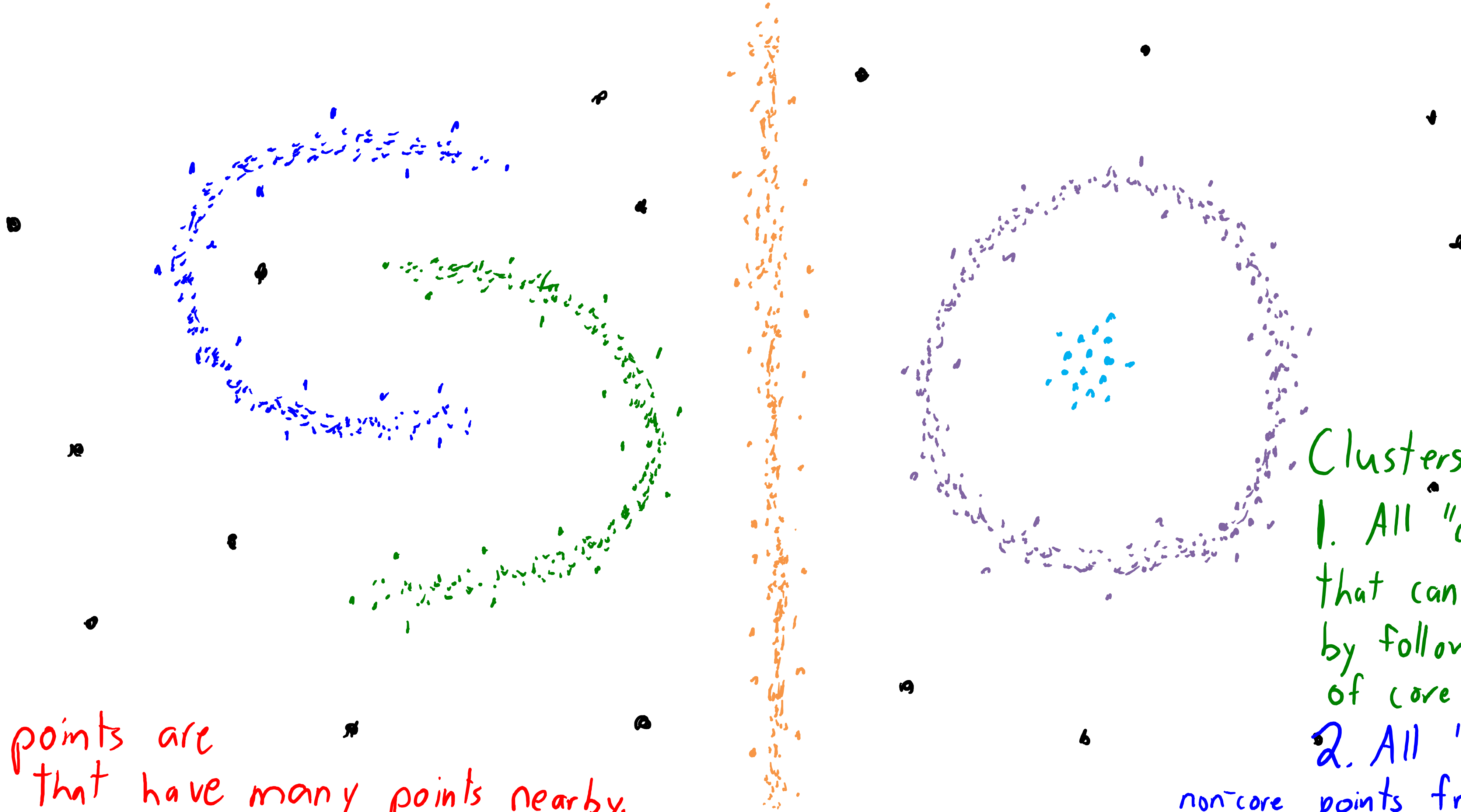


Clusters contain:

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

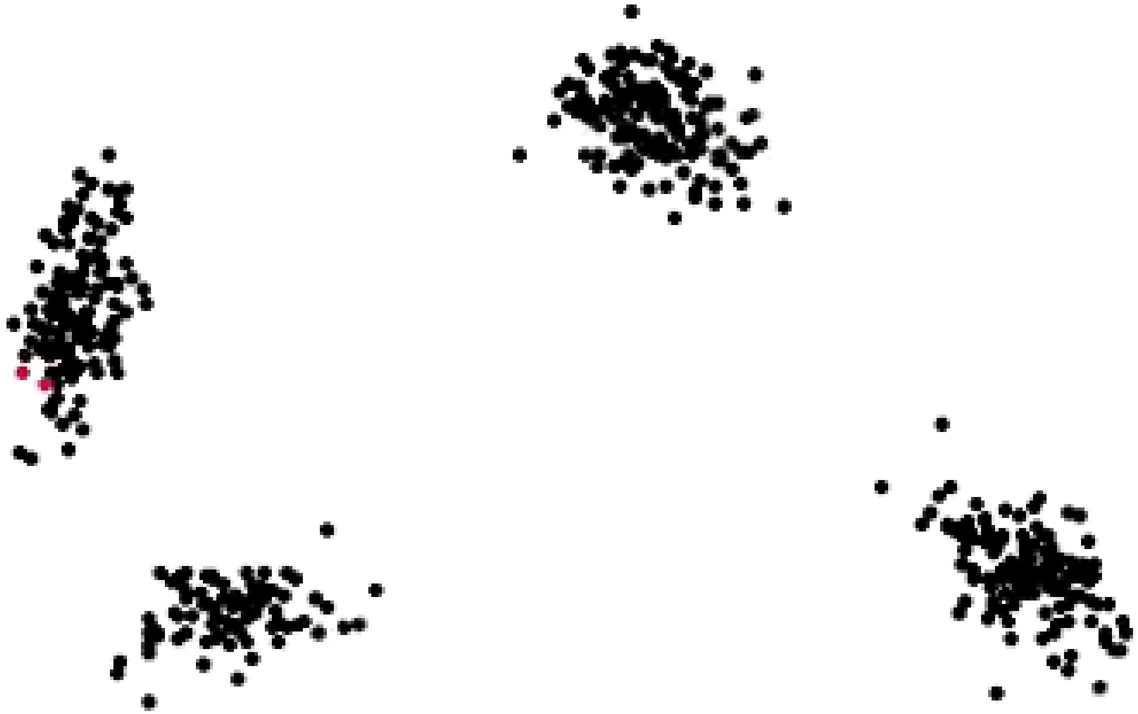


"Core" points are points that have many points nearby.

Clusters contain:

1. All "core" points that can be reached by following a <u>sequence</u> of core points.
2. All "reachable" non-core points from these "core" points ("boundary" points)

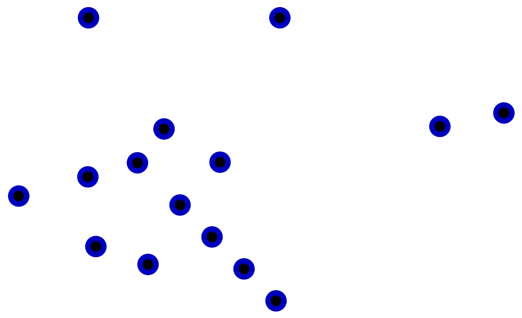
Density-Based Clustering in Action



Interactive demo:
https://www.naftaliharris.com/blog/visualizing-dbscan-clustering

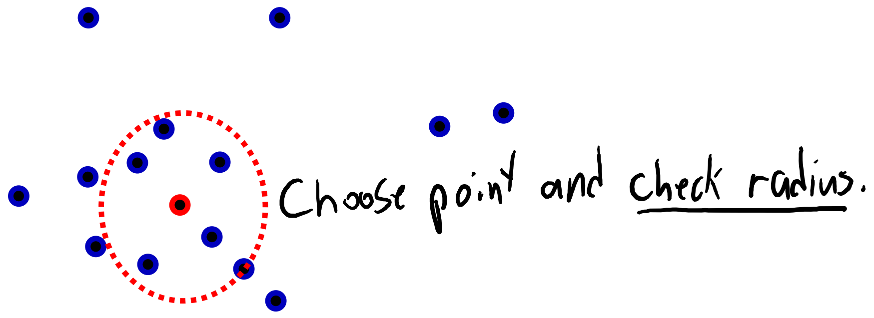
Density-Based Clustering

- 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.



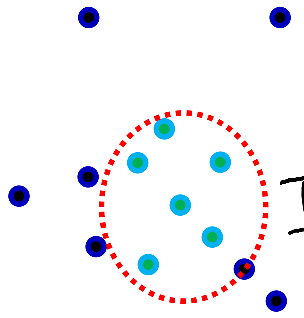
Density-Based Clustering

- 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

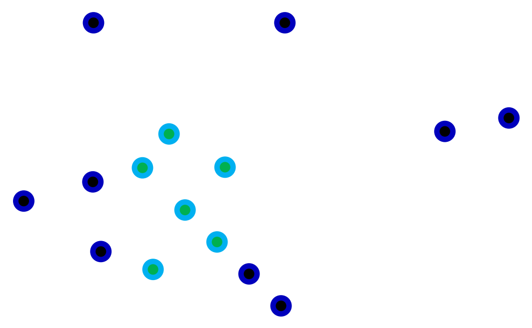
- Each "core" point defines a cluster:
  - Consisting of "core" point and all its "reachable" points.
- Merge clusters if "core" points are "reachable" from each other.



It's a "core" point, so make a new cluster with points within radius.

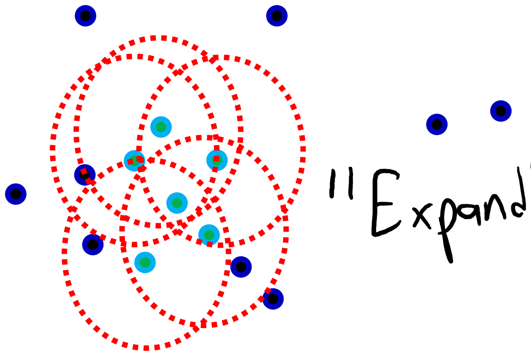
Density-Based Clustering

- 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

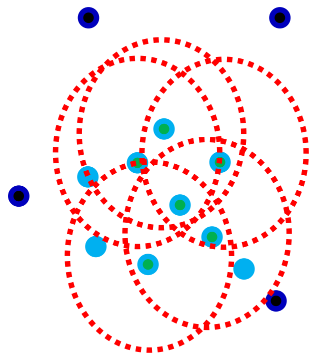
- Each "core" point defines a cluster:
  - Consisting of "core" point and all its "reachable" points.
- Merge clusters if "core" points are "reachable" from each other.



"Expand" the new cluster points that are "core" points.

Density-Based Clustering

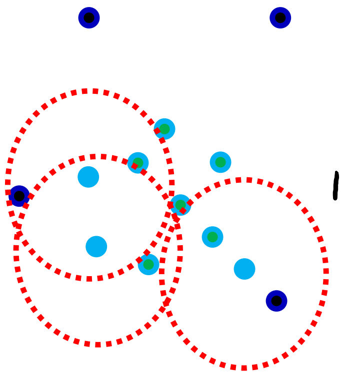
- Each "core" point defines a cluster:
  - Consisting of "core" point and all its "reachable" points.
- Merge clusters if "core" points are "reachable" from each other.



Add "reachable" points to cluster.

Density-Based Clustering

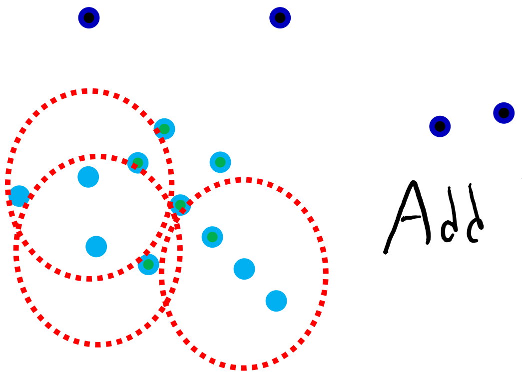
- Each "core" point defines a cluster:
  - Consisting of "core" point and all its "reachable" points.
- Merge clusters if "core" points are "reachable" from each other.



"Expand" the new cluster points that are "core" points.

Density-Based Clustering

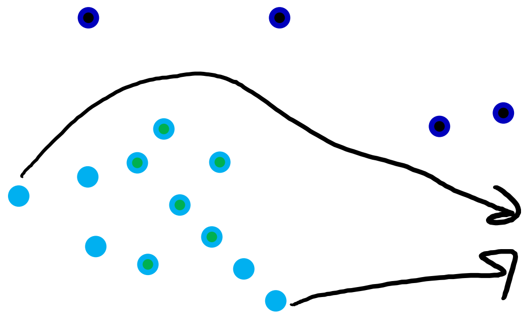
- Each "core" point defines a cluster:
  - Consisting of "core" point and all its "reachable" points.
- Merge clusters if "core" points are "reachable" from each other.



Add "reachable" points to cluster.

Density-Based Clustering

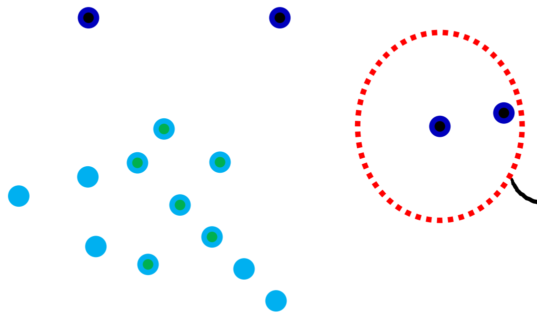
- Each "core" point defines a cluster:
  - Consisting of "core" point and all its "reachable" points.
- Merge clusters if "core" points are "reachable" from each other.



"Reachable" but not "core" points, so do not expand them.

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.



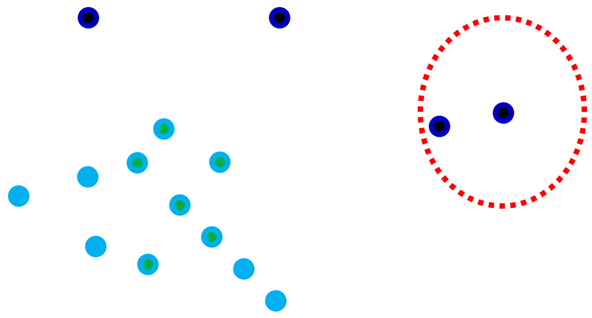
We're done "expanding", try a new point.

Not enough points in radius => not a "core" point.

Don't assign cluster

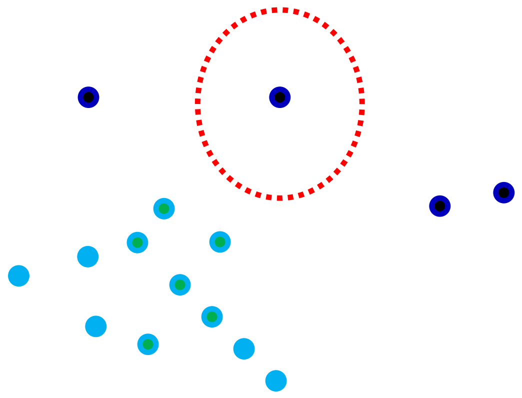
Density-Based Clustering

- 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.



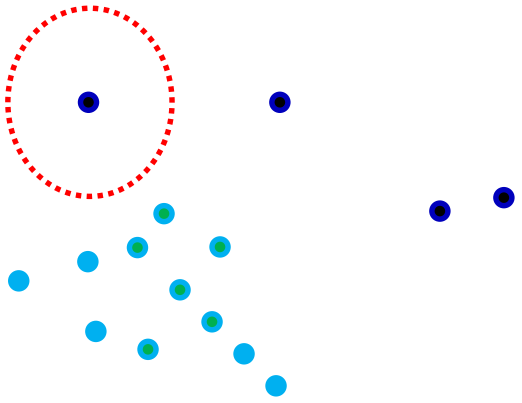
Density-Based Clustering

- 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

- Pseudocode for DBSCAN:

  - For each example x_i:
    - If x_i is already assigned to a cluster, do nothing.
    - Test whether xi is a 'core' point (less than minPoints neighbours with distances ≤ '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:

Points at "boundary" between clusters.

- 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.

Bonus Slide: Lp-norms

- The L_1-, L_2-, and L_∞-norms are special cases of Lp-norms:

$$||x||_p = \left(\sum_{j=1}^{d} x_j^p \right)^{1/p}$$

- This gives a norm for any (real-valued) $p \geq 1$.
  - The L_∞-norm is limit as 'p' goes to ∞.

- For $p < 1$, not a norm because triangle inequality not satisfied.

Bonus Slide: Squared/Euclidean-Norm Notation

We're using the following conventions:

The subscript after the norm is used to denote the p-norm, as in these examples:

$$\|x\|_2 = \sqrt{\sum_{j=1}^{d} w_j^2}.$$
$$\|x\|_1 = \sum_{j=1}^{d} |w_j|.$$

If the subscript is omitted, we mean the 2-norm:

$$\|x\| = \|x\|_2.$$

If we want to talk about the *squared* value of the norm we use a superscript of "2":

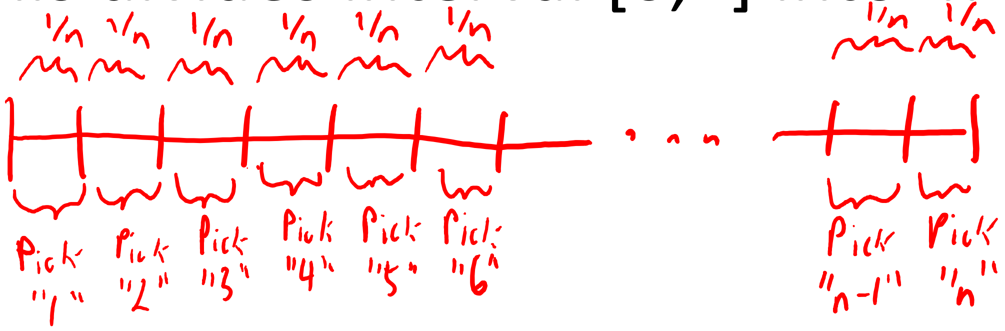
$$\|x\|_2^2 = \sum_{j=1}^{d} w_j^2.$$
$$\|x\|_1^2 = \left(\sum_{j=1}^{d} |w_j|\right)^2.$$

If we omit the subscript and have a superscript of "2", we're taking about the squared L2-norm:

$$\|x\|^2 = \sum_{j=1}^{d} w_j^2.$$

Bonus Slide: Uniform Sampling

- Standard approach to generating a random number from {1,2,…,n}:
  1. Generate a uniform random number ‘u’ in the interval [0,1].
  2. Return the largest index ‘i’ such that $u \leq i/n$.
- Conceptually, this divides interval [0,1] into ‘n’ equal-size pieces:

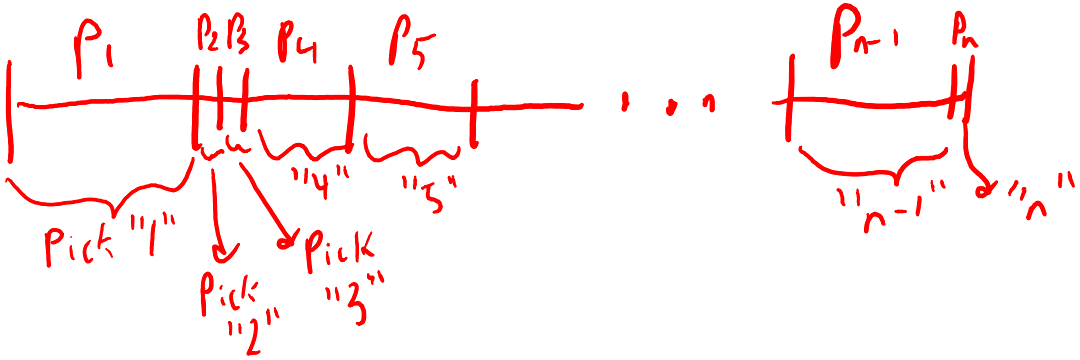


- This assumes $p_i = 1/n$ for all ‘i’.

probability of picking number 'i'.

Bonus Slide: Non-Uniform Sampling

- Standard approach to generating a random number for general p_i.
  1. Generate a uniform random number 'u' in the interval [0,1].
  2. Return the largest index 'i' such that $u \leq \sum_{j=1}^{i} p_i$
- Conceptually, this divides interval [0,1] into non-equal-size pieces:



- Can sample from a generic discrete probability distribution in O(n).
- If you need to generate 'm' samples:
  – Cost is O(n + m log (n)) with binary search and storing cumulative sums.

Bonus Slide: Discussion of K-Means++

- Recall the objective function k-means tries to minimize:

$$\underbrace{f(W, c)}_{} = \sum_{i=1}^{n} \|x_i - w_{c(i)}\|_2^2$$

all means   all assigments

- The initialization of 'W' and 'c' given by k-means++ satisfies:

$$\frac{E[f(W,c)]}{f(W^*,c^*)} = O(\log(k))$$

expectation over random samples

"Best" mean and clustering according to objective.

- Get good clustering with high probability by re-running.
- However, there is no guarantee that c^* is a good clustering.