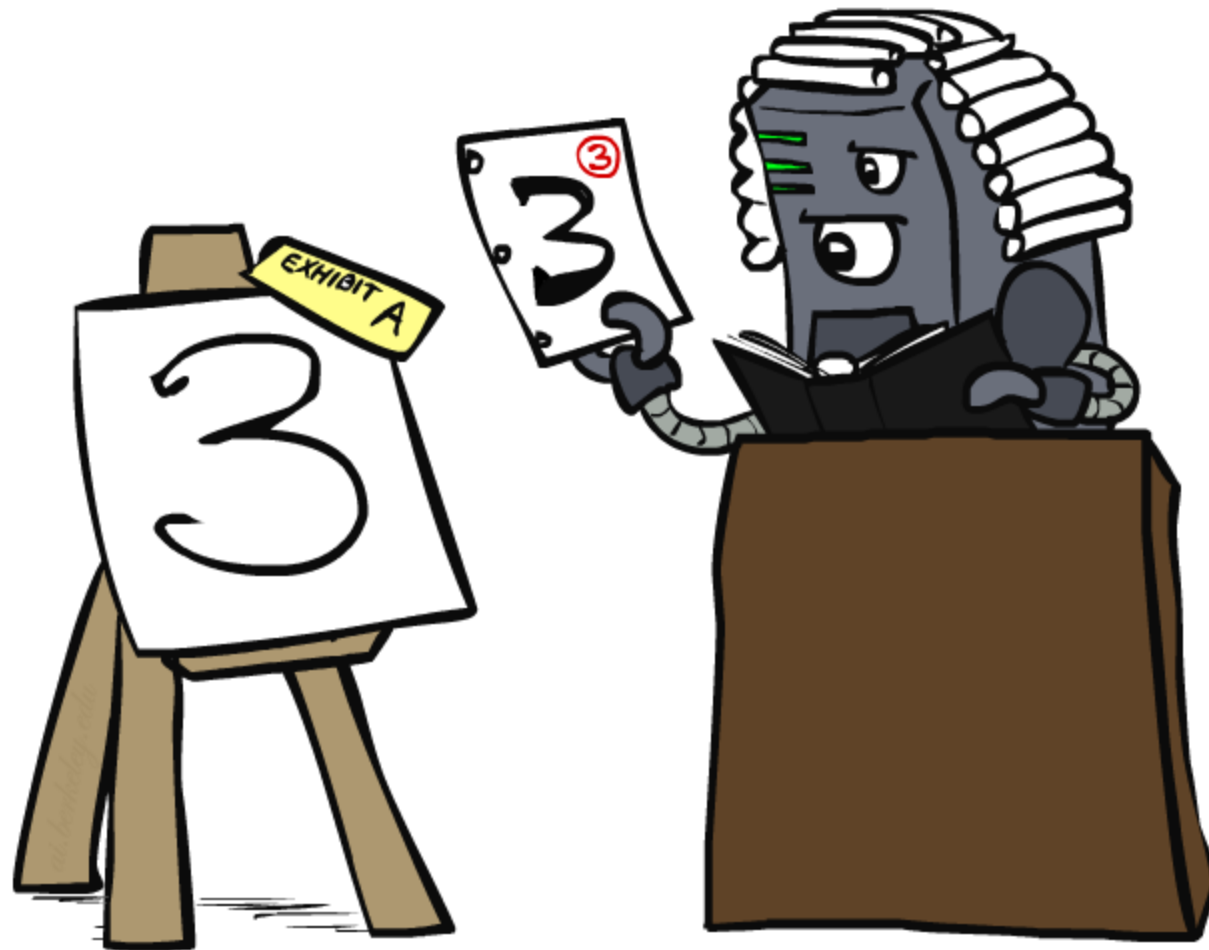


Kernels and Clustering

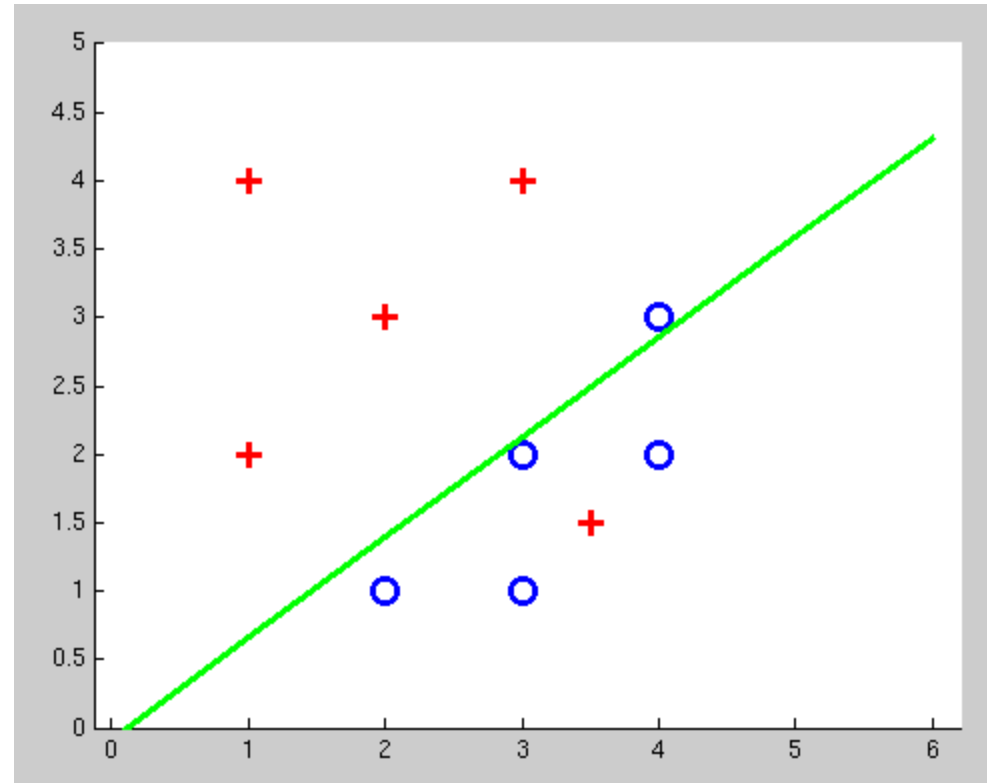
Robert Platt
Northeastern University

All slides in this file are adapted from CS188 UC
Berkeley

Case-Based Learning

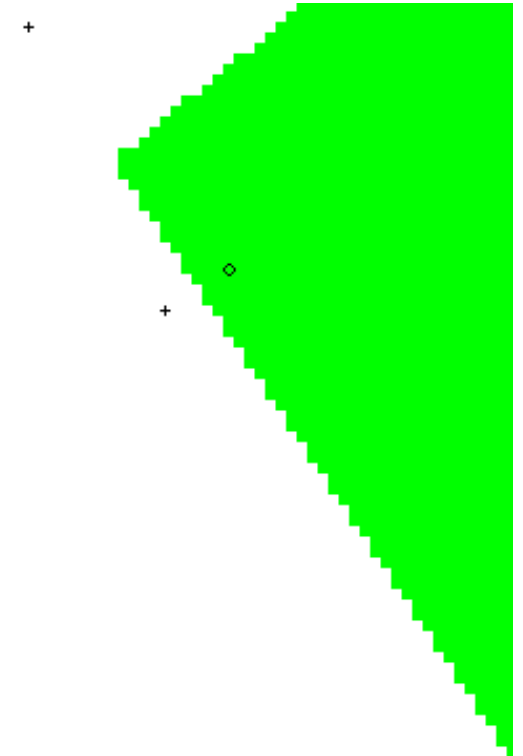


Non-Separable Data



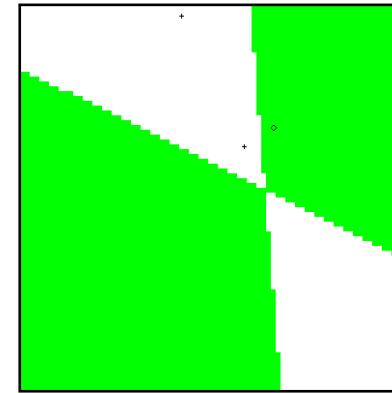
Case-Based Reasoning

- **Classification from similarity**
 - Case-based reasoning
 - Predict an instance's label using similar instances
- **Nearest-neighbor classification**
 - 1-NN: copy the label of the most similar data point
 - K-NN: vote the k nearest neighbors (need a weighting scheme)
 - Key issue: how to define similarity

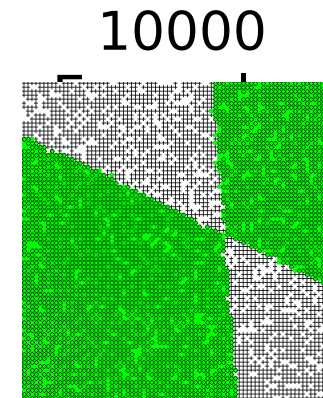
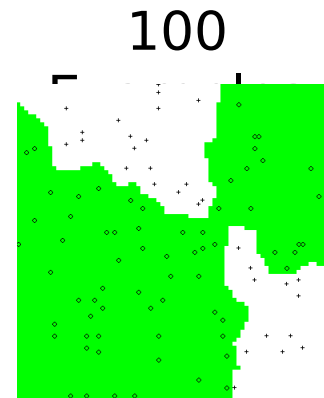
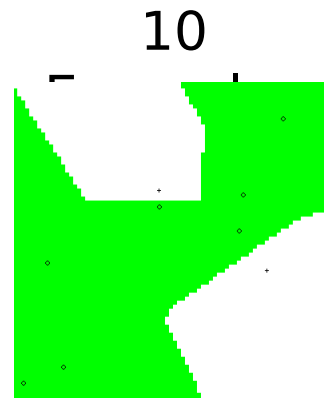
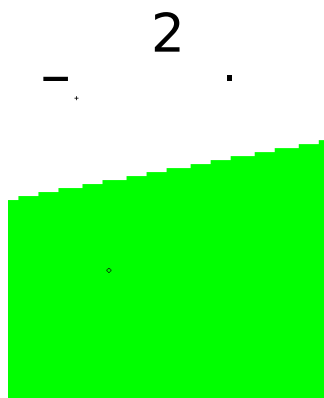


Parametric / Non-Parametric

- Parametric models:
 - Fixed set of parameters
 - More data means better settings
- Non-parametric models:
 - Complexity of the classifier increases with data
 - Better in the limit, often worse in the non-limit
- (K)NN is non-parametric



Truth



Nearest-Neighbor Classification

- Nearest neighbor for digits:
 - Take new image
 - Compare to all training images
 - Assign based on closest example

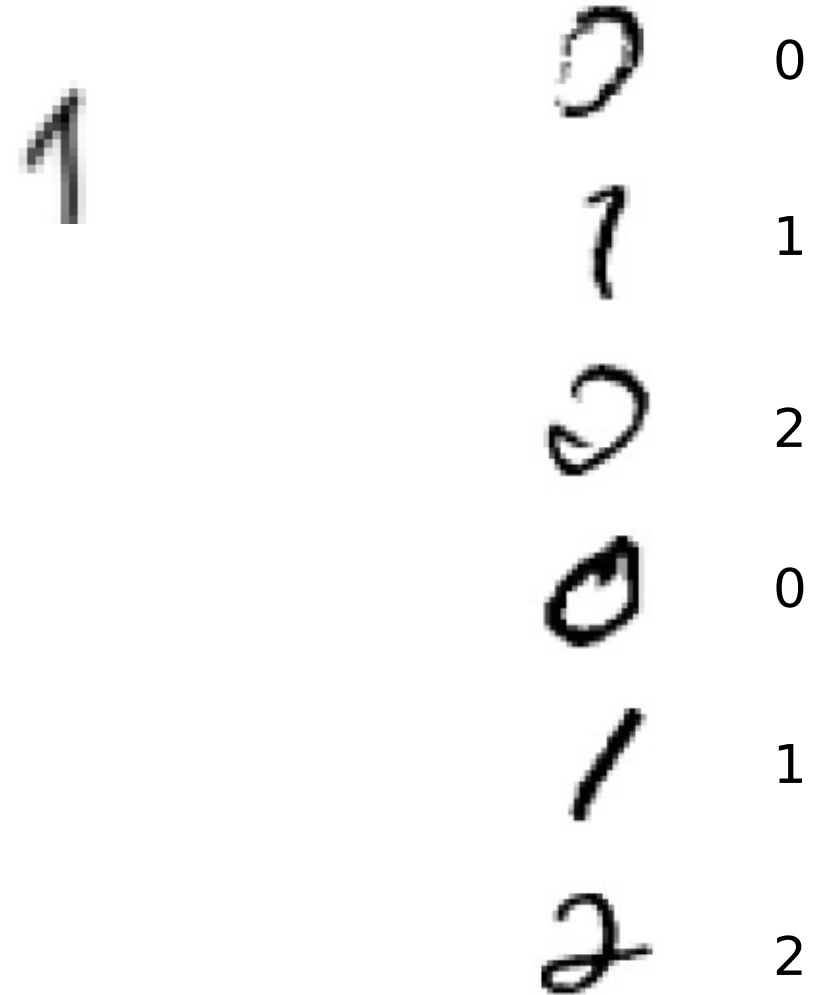
- Encoding: image is vector of intensities:

$$\mathbf{1} = \langle 0.0 \ 0.0 \ 0.3 \ 0.8 \ 0.7 \ 0.1 \ \dots \ 0.0 \rangle$$

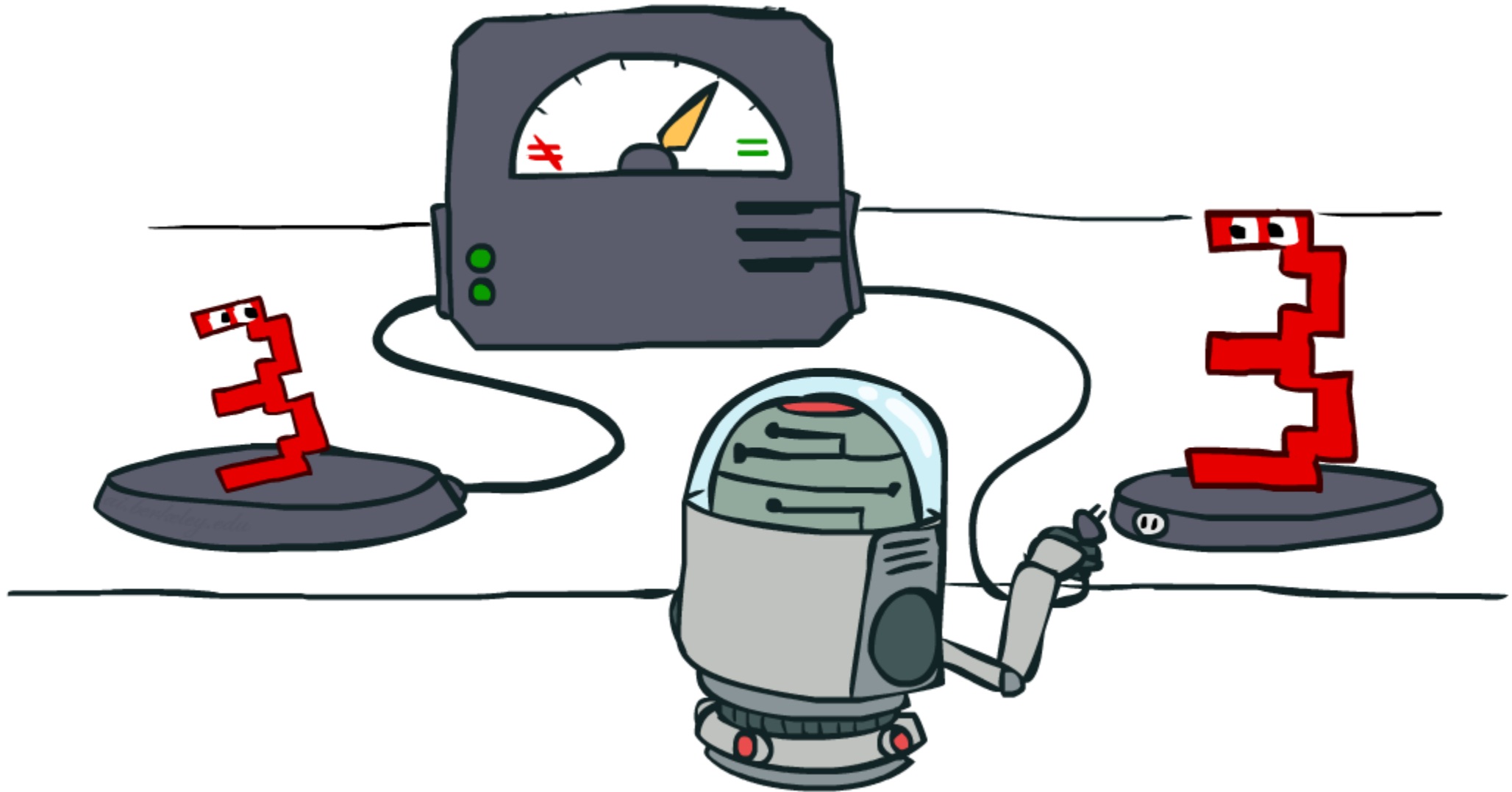
- What's the similarity function?
 - Dot product of two images vectors?

$$\text{sim}(x, x') = x \cdot x' = \sum_i x_i x'_i$$

- Usually normalize vectors so $\|x\| = 1$
- min = 0 (when?), max = 1 (when?)



Similarity Functions



Basic Similarity

- Many similarities based on **feature dot products**:

$$\text{sim}(x, x') = f(x) \cdot f(x') = \sum_i f_i(x) f_i(x')$$

- If features are just the pixels:

$$\text{sim}(x, x') = x \cdot x' = \sum_i x_i x'_i$$

- Note: not all similarities are of this form

Invariant Metrics

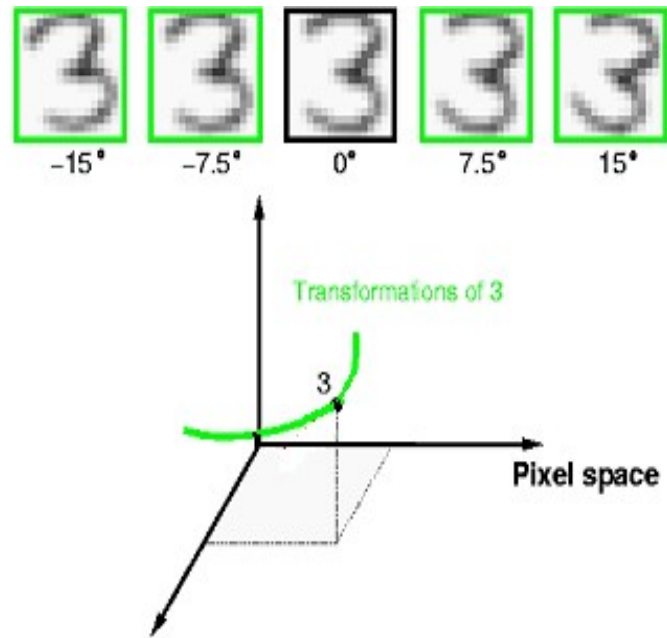
- Better similarity functions use knowledge about vision
- Example: invariant metrics:
 - Similarities are invariant under certain transformations
 - Rotation, scaling, translation, stroke-thickness...

▪ E.g:



- $16 \times 16 = 256$ pixels; a point in 256-dim space
 - These points have small similarity in \mathbb{R}^{256} (why?)
- How can we incorporate such invariances into our similarities?

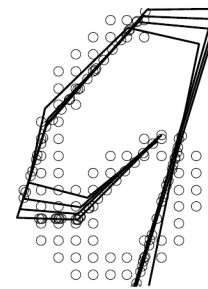
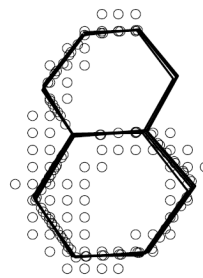
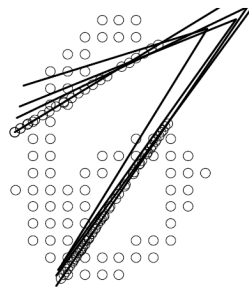
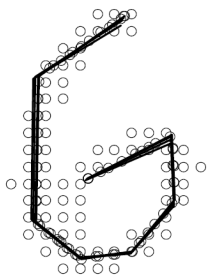
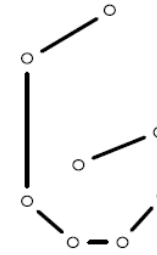
Rotation Invariant Metrics



- Each example is now a curve in \mathbb{R}^{256}
- Rotation invariant similarity:
$$s' = \max s(r(\text{3}), r(\text{3}))$$
- E.g. highest similarity between images' rotation lines

Template Deformation

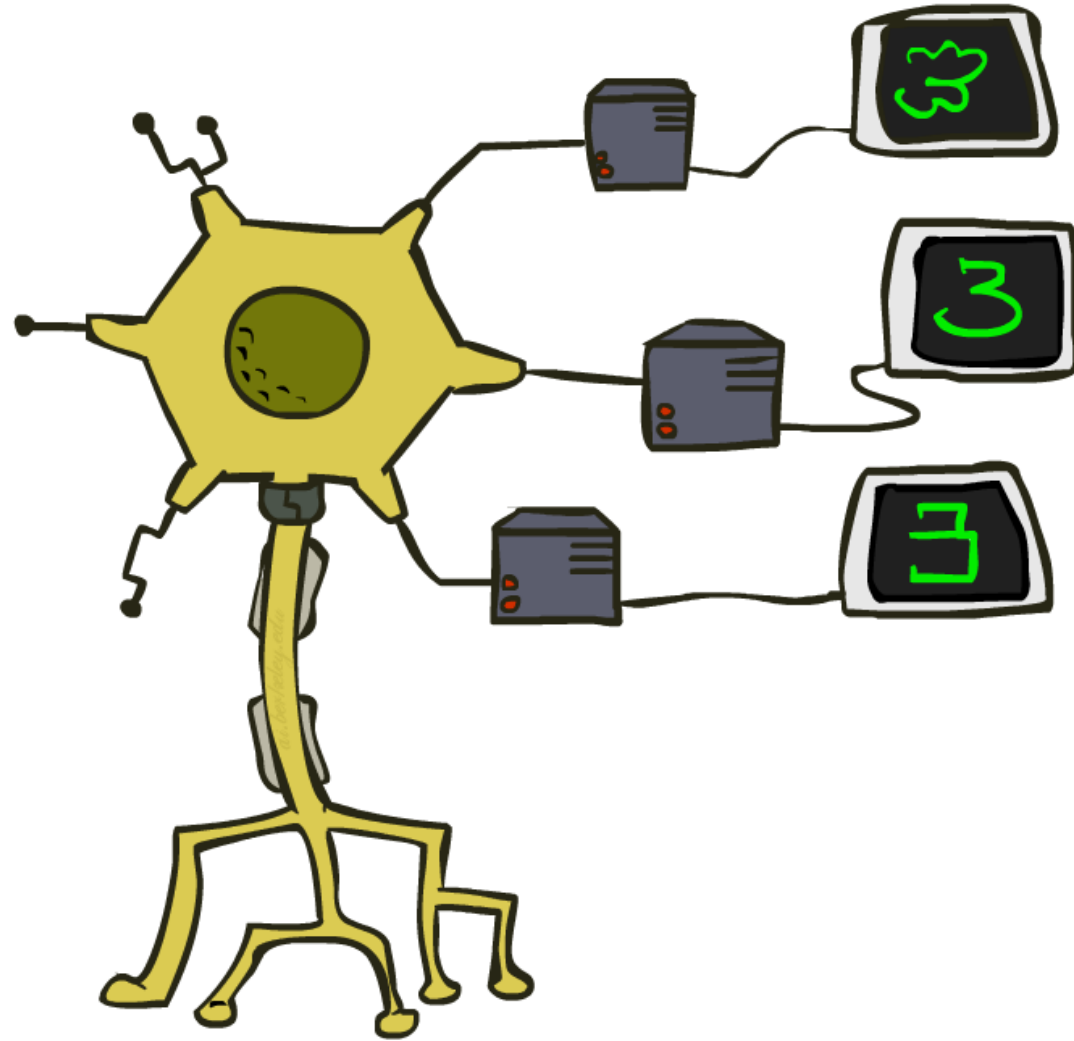
- Deformable templates:
 - An “ideal” version of each category
 - Best-fit to image using min variance
 - Cost for high distortion of template
 - Cost for image points being far from distorted template
- Used in many commercial digit recognizers



A Tale of Two Approaches...

- Nearest neighbor-like approaches
 - Can use fancy similarity functions
 - Don't actually get to do explicit learning
- Perceptron-like approaches
 - Explicit training to reduce empirical error
 - Can't use fancy similarity, only linear
 - Or can they? Let's find out!

Kernelization



Perceptron Weights

- What is the final value of a weight w_y of a perceptron?
 - Can it be any real vector?
 - No! It's built by adding up inputs.

$$w_y = 0 + f(x_1) - f(x_5) + \dots$$

$$w_y = \sum_i \alpha_{i,y} f(x_i)$$

- Can reconstruct weight vectors (the **primal representation**) from update counts (the **dual representation**) $\alpha_y = \langle \alpha_{1,y} \ \alpha_{2,y} \ \dots \ \alpha_{n,y} \rangle$

Dual Perceptron

- How to classify a new example x ?

$$\begin{aligned}\text{score}(y, x) &= w_y \cdot f(x) \\ &= \left(\sum_i \alpha_{i,y} f(x_i) \right) \cdot f(x) \\ &= \sum_i \alpha_{i,y} (f(x_i) \cdot f(x)) \\ &= \sum_i \alpha_{i,y} K(x_i, x)\end{aligned}$$

- If someone tells us the value of K for each pair of examples, never need to build the weight vectors (or the feature vectors)!

Dual Perceptron

- Start with zero counts (alpha)
- Pick up training instances one by one
- Try to classify x_n ,

$$y = \arg \max_y \sum_i \alpha_{i,y} K(x_i, x_n)$$

- If correct, no change!
- If wrong: lower count of wrong class (for this instance), raise count of right class (for this instance)

$$\alpha_{y,n} = \alpha_{y,n} - 1$$

$$w_y = w_y - f(x_n)$$

$$\alpha_{y^*,n} = \alpha_{y^*,n} + 1$$

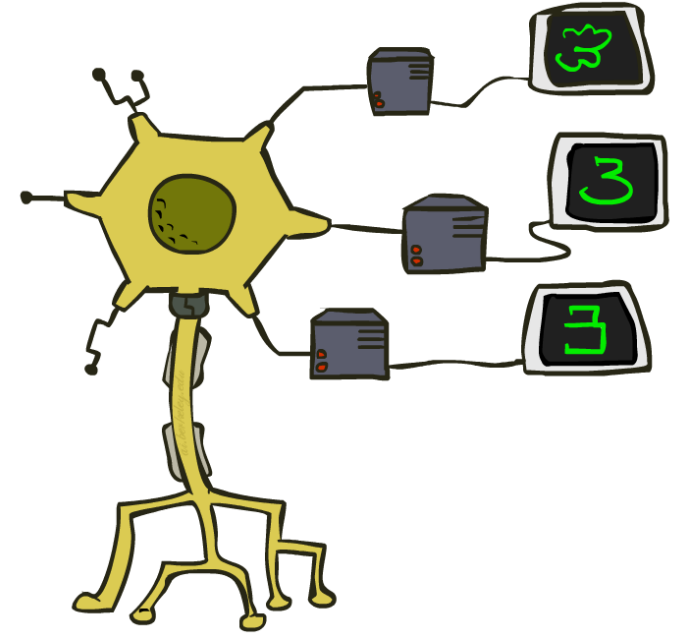
$$w_{y^*} = w_{y^*} + f(x_n)$$

Kernelized Perceptron

- If we had a black box (**kernel**) K that told us the dot product of two examples x and x' :
 - Could work entirely with the dual representation
 - No need to ever take dot products (“kernel trick”)

$$\text{score}(y, x) = w_y \cdot f(x)$$

$$= \sum_i \alpha_{i,y} K(x_i, x)$$



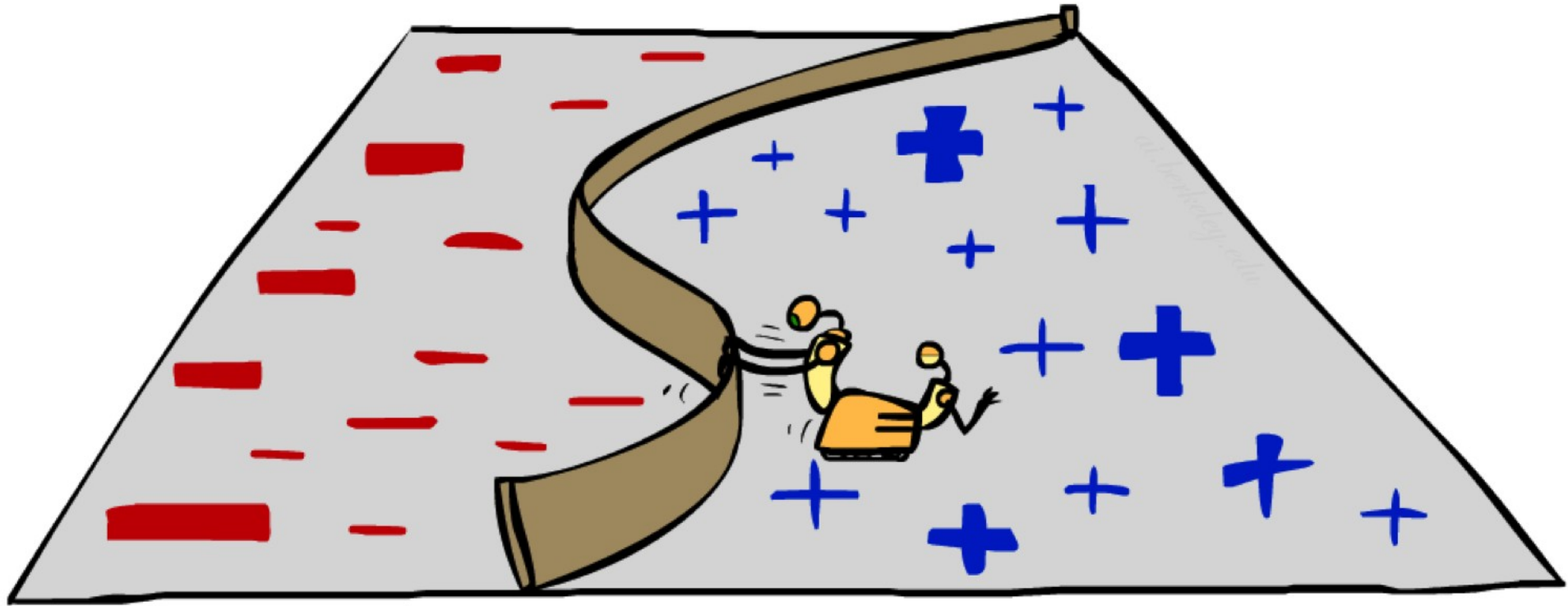
- Like nearest neighbor – work with black-box similarities
- Downside: slow if many examples get nonzero alpha

Kernels: Who Cares?

- So far: a very strange way of doing a very simple calculation
- “Kernel trick”: we can substitute any* similarity function in place of the dot product
- Lets us learn new kinds of hypotheses

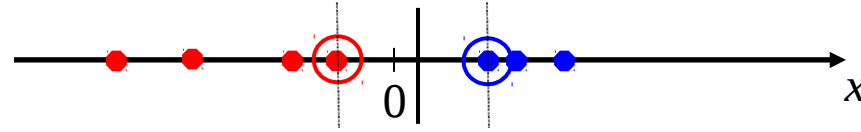
* Fine print: if your kernel doesn't satisfy certain technical requirements, lots of proofs break. E.g. convergence, mistake bounds. In practice, illegal kernels *sometimes* work (but not always).

Non-Linearity

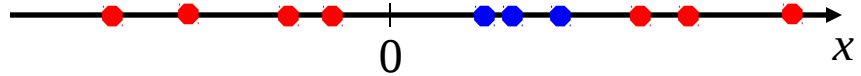


Non-Linear Separators

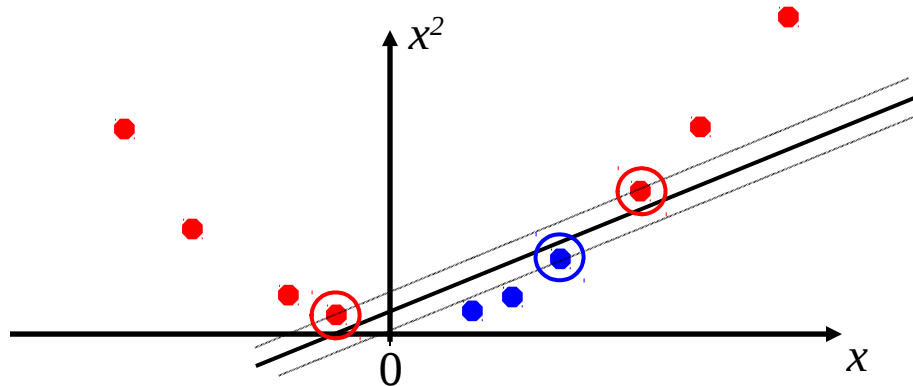
- Data that is linearly separable works out great for linear decision rules:



- But what are we going to do if the dataset is just too hard?

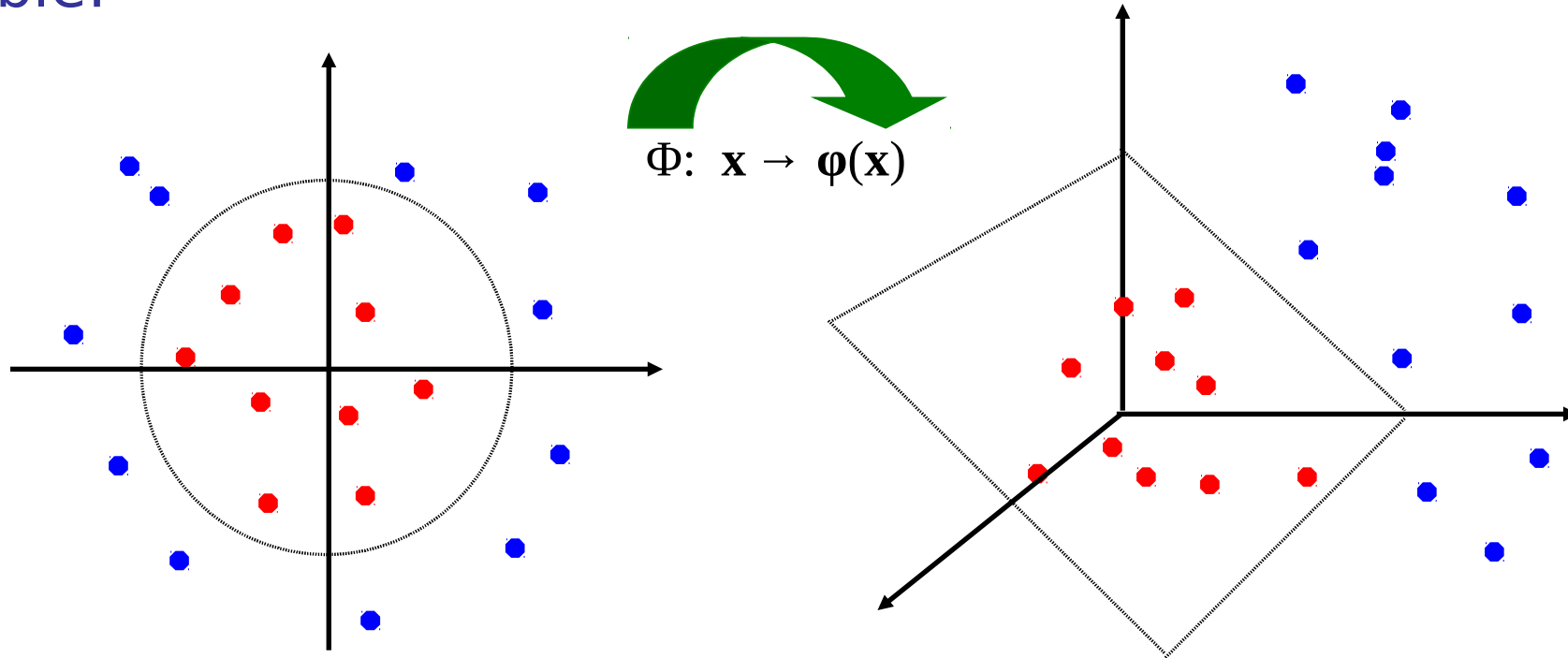


- How about... mapping data to a higher-dimensional space:



Non-Linear Separators

- General idea: the original feature space can always be mapped to some higher-dimensional feature space where the training set is separable:



Some Kernels

- Kernels **implicitly** map original vectors to higher dimensional spaces, take the dot product there, and hand the result back

- Linear kernel:
$$K(x, x') = x' \cdot x' = \sum_i x_i x'_i$$

- Quadratic kernel:
$$K(x, x') = (x \cdot x' + 1)^2$$
$$= \sum_{i,j} x_i x_j x'_i x'_j + 2 \sum_i x_i x'_i + 1$$

- RBF: infinite dimensional representation

$$K(x, x') = \exp(-\|x - x'\|^2)$$

- Discrete kernels: e.g. string kernels

Why Kernels?

- Can't you just add these features on your own (e.g. add all pairs of features instead of using the quadratic kernel)?
 - Yes, in principle, just compute them
 - No need to modify any algorithms
 - But, number of features can get large (or infinite)
 - Some kernels not as usefully thought of in their expanded representation, e.g. RBF kernels
- Kernels let us compute with these features implicitly
 - Example: implicit dot product in quadratic kernel takes much less space and time per dot product
 - Of course, there's the cost for using the pure dual algorithms: you need to compute the similarity to every training datum

Recap: Classification

- Classification systems:
 - Supervised learning
 - Make a prediction given evidence
 - We've seen several methods for this
 - Useful when you have labeled data

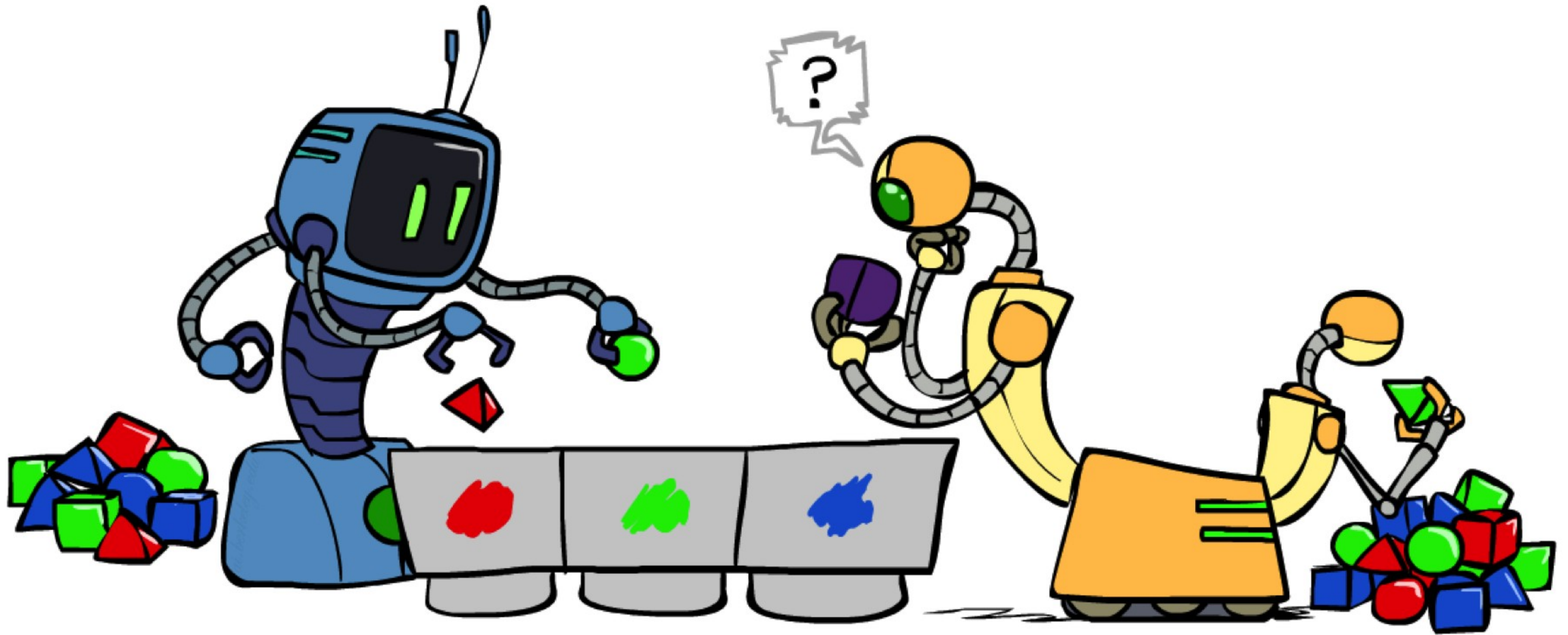


Clustering

- Clustering systems:
 - Unsupervised learning
 - Detect patterns in unlabeled data
 - E.g. group emails or search results
 - E.g. find categories of customers
 - E.g. detect anomalous program executions
 - Useful when don't know what you're looking for
 - Requires data, but no labels
 - Often get gibberish



Clustering



Clustering

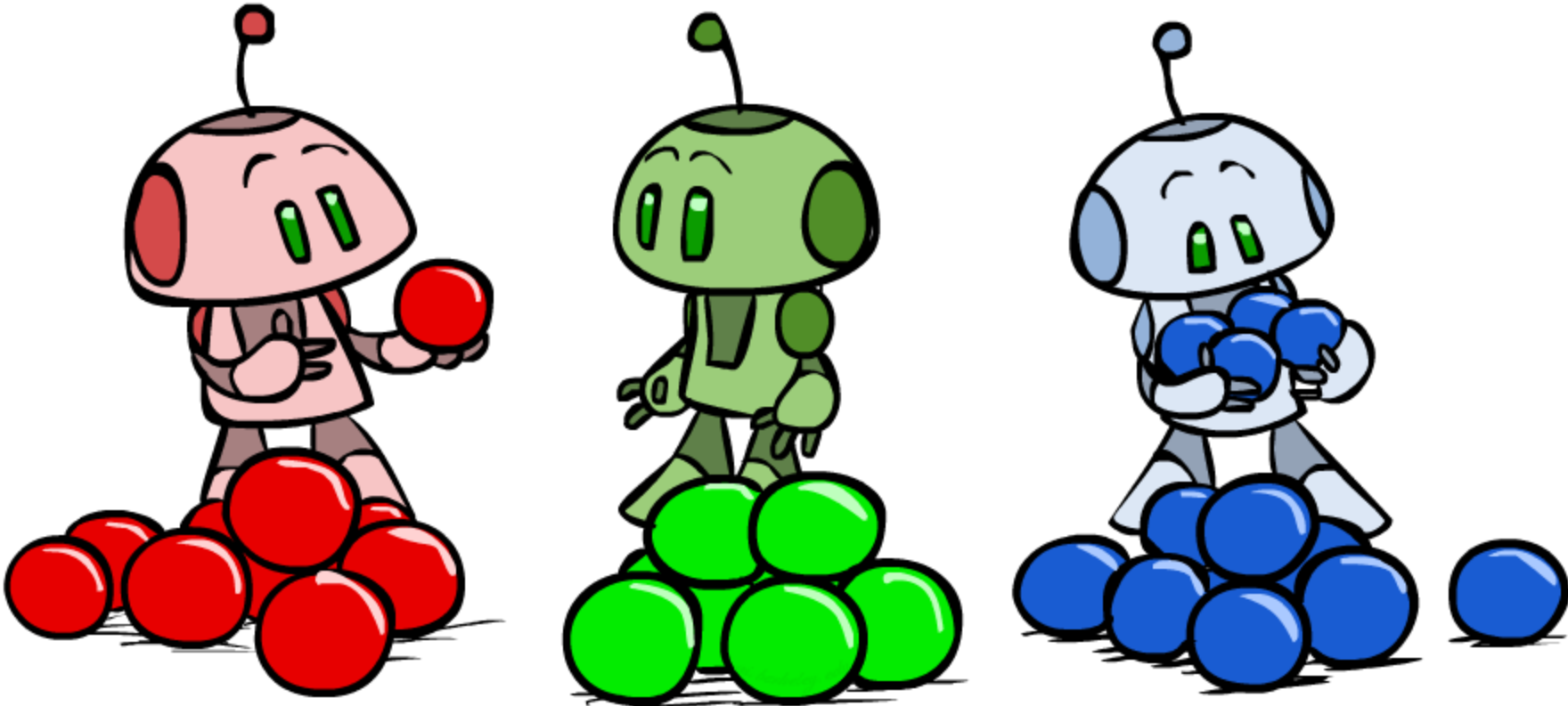
- Basic idea: group together similar instances
- Example: 2D point patterns



- What could “similar” mean?
 - One option: small (squared) Euclidean distance

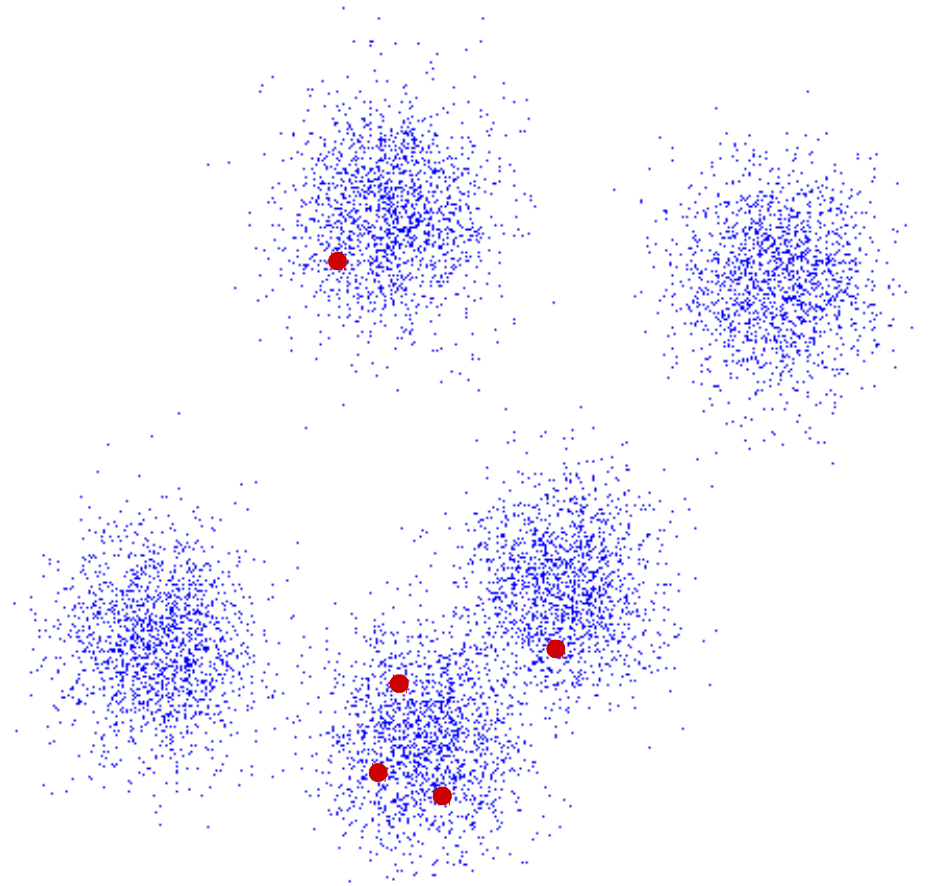
$$\text{dist}(x, y) = (x - y)^{\top} (x - y) = \sum_i (x_i - y_i)^2$$

K-Means



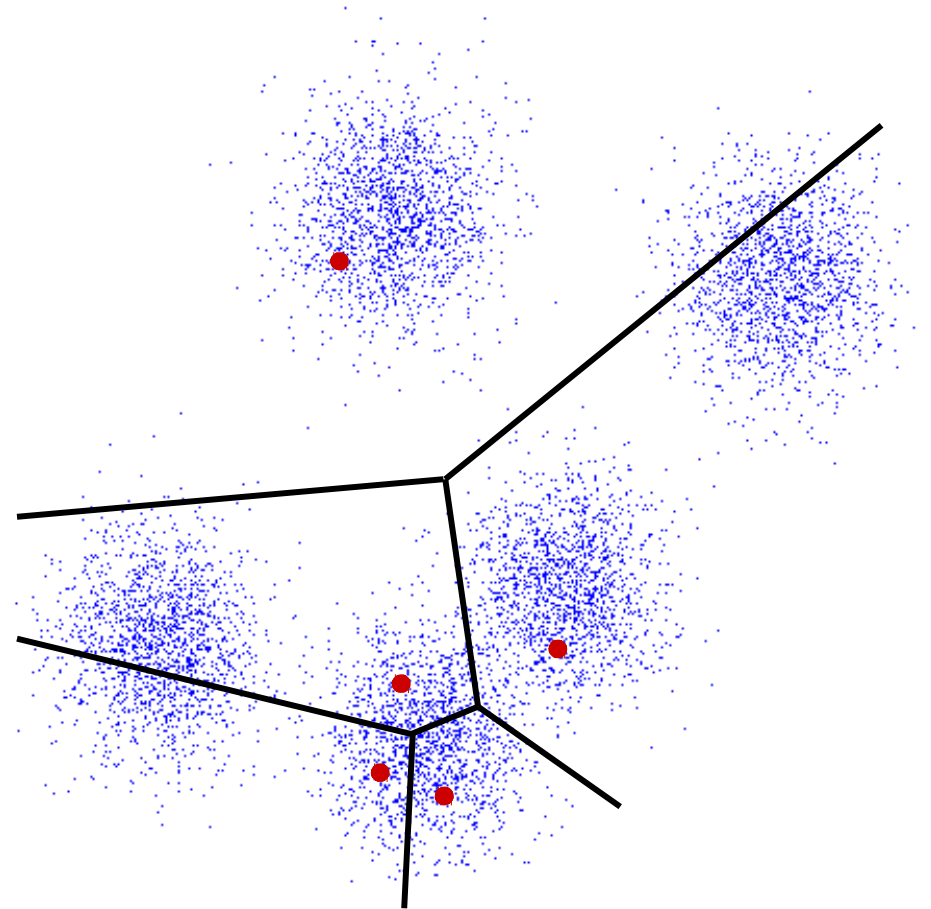
K-Means

- An iterative clustering algorithm
 - Pick K random points as cluster centers (means)
 - Alternate:
 - Assign data instances to closest mean
 - Assign each mean to the average of its assigned points
 - Stop when no points' assignments change



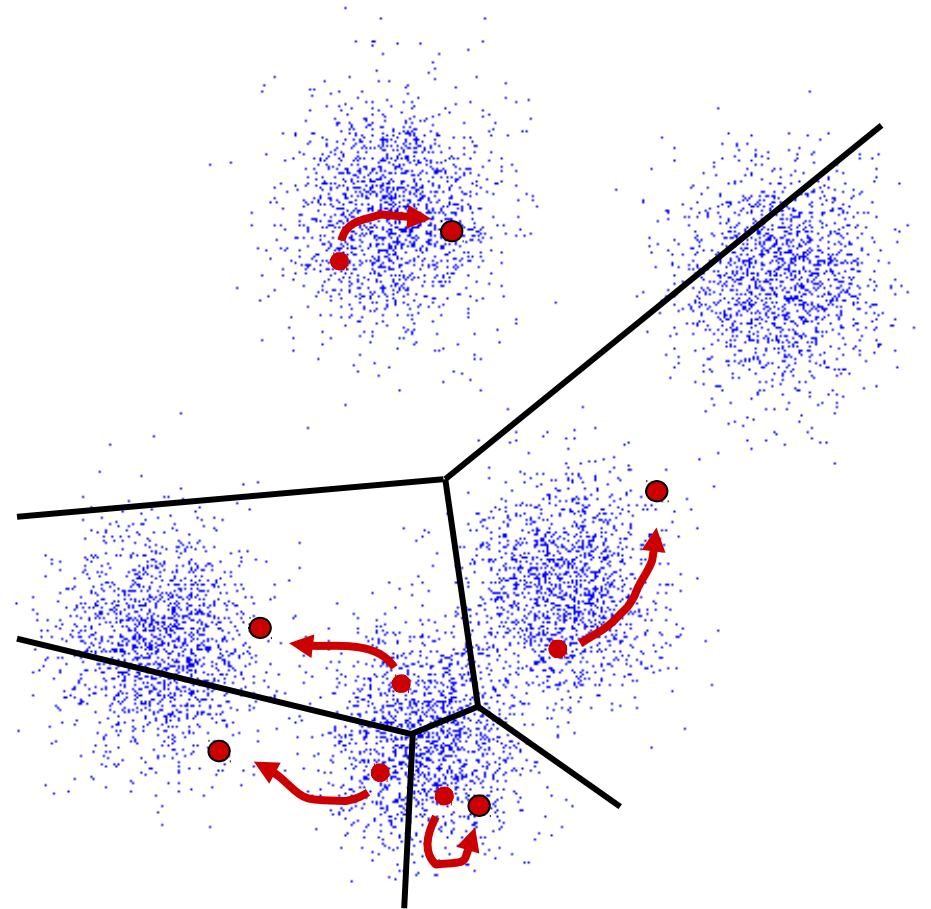
K-Means

- An iterative clustering algorithm
 - Pick K random points as cluster centers (means)
 - Alternate:
 - Assign data instances to closest mean
 - Assign each mean to the average of its assigned points
 - Stop when no points' assignments change

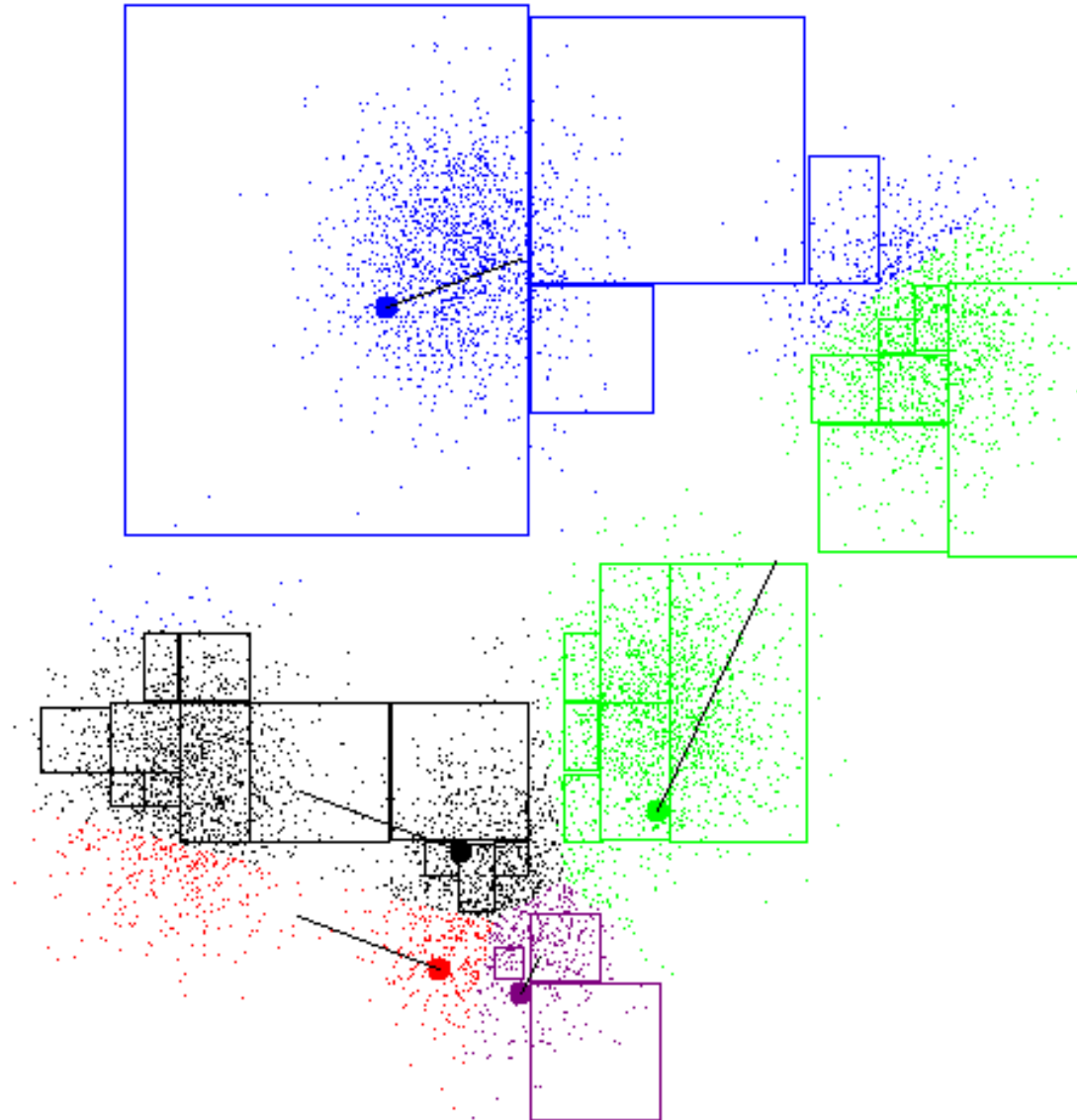


K-Means

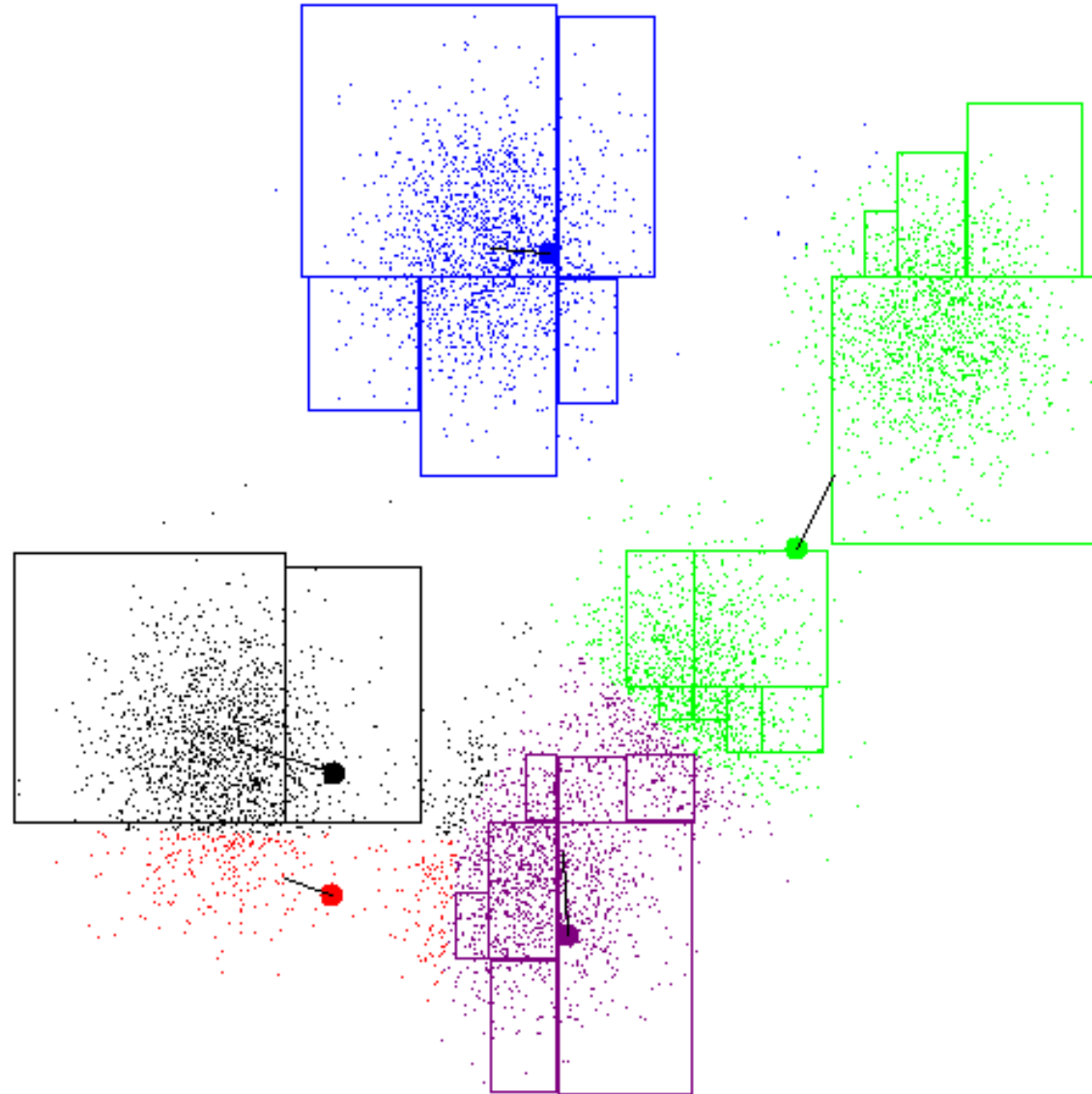
- An iterative clustering algorithm
 - Pick K random points as cluster centers (means)
 - Alternate:
 - Assign data instances to closest mean
 - Assign each mean to the average of its assigned points
 - Stop when no points' assignments change



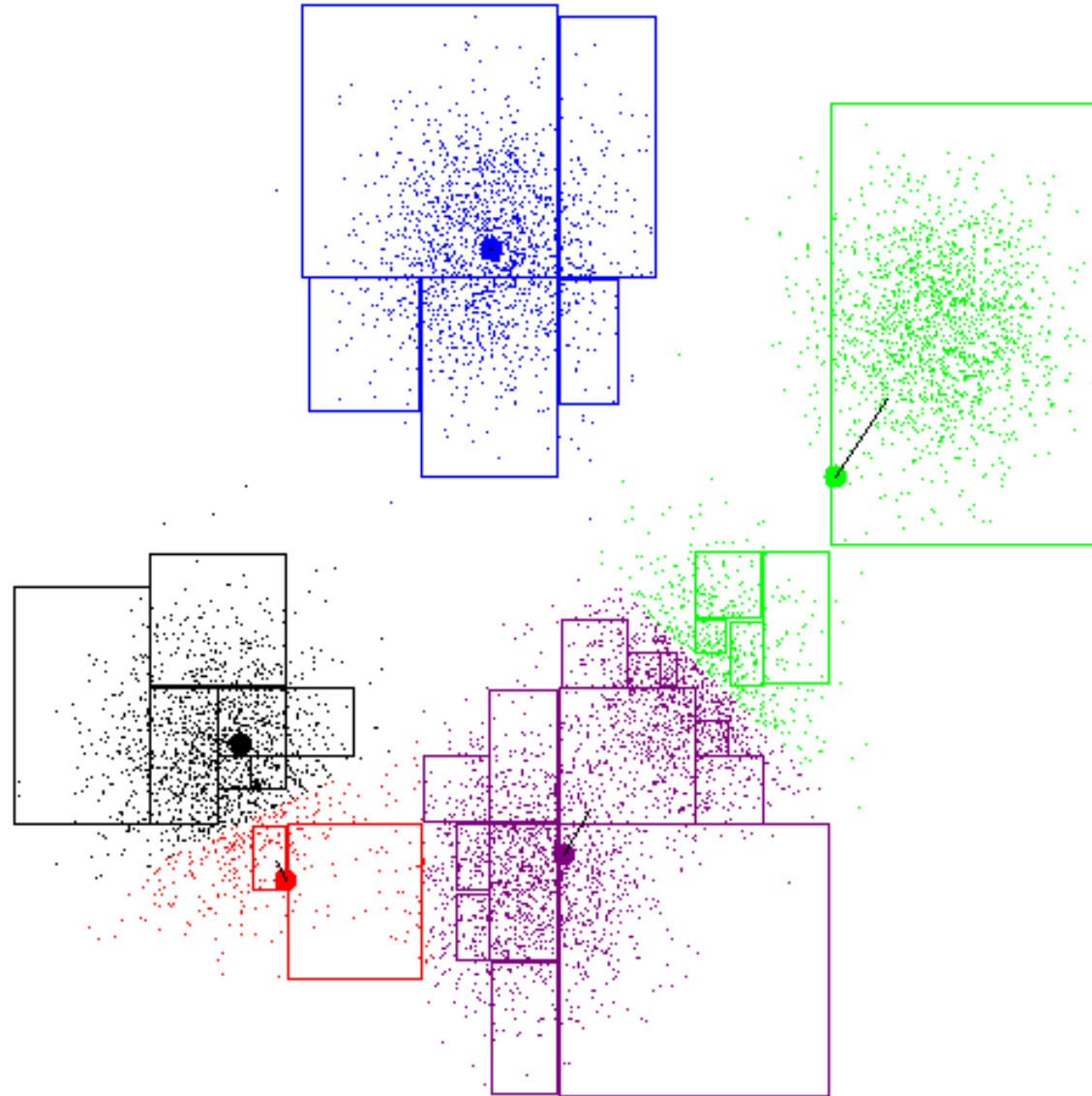
K-Means Example



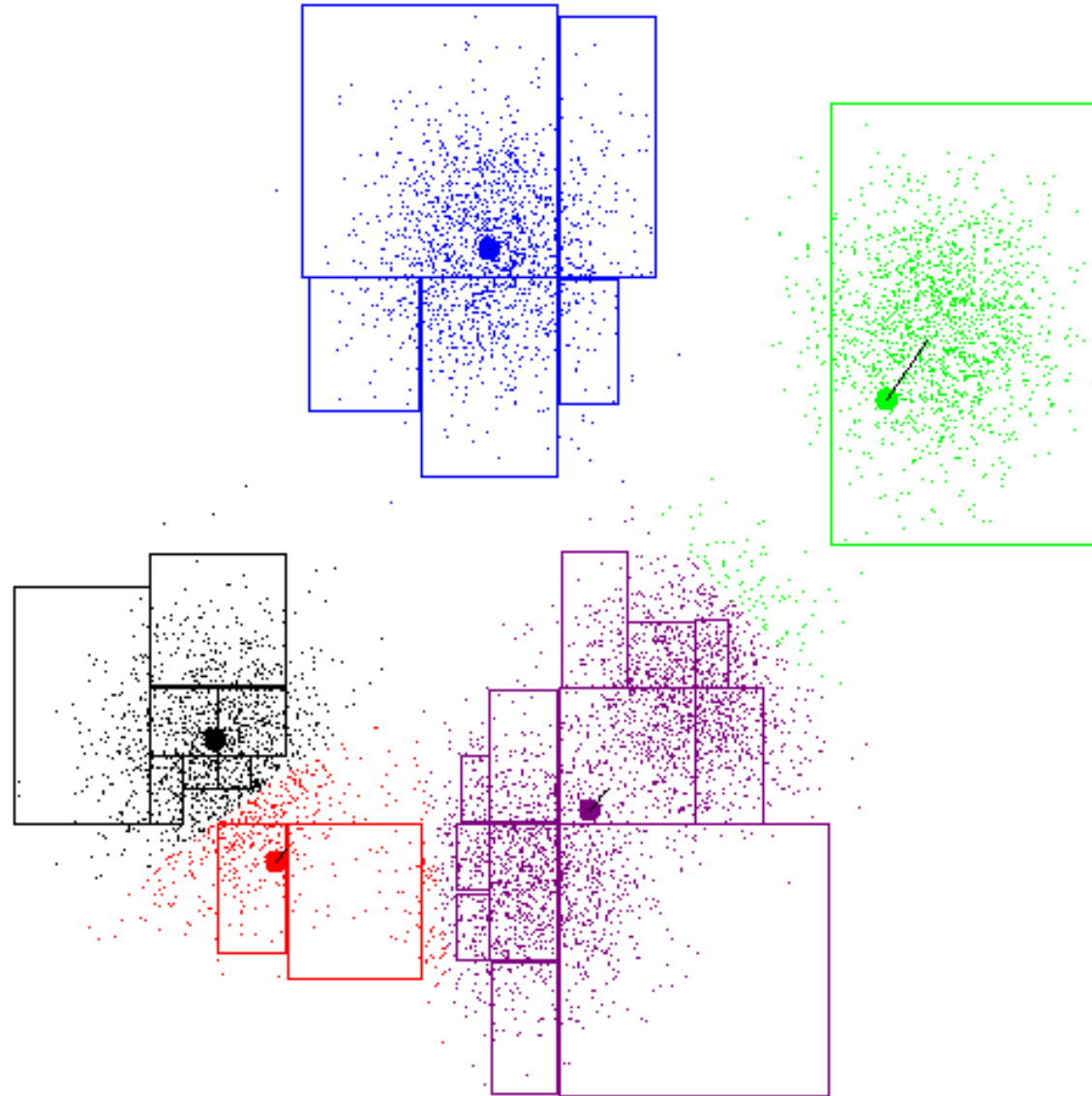
K-Means Example



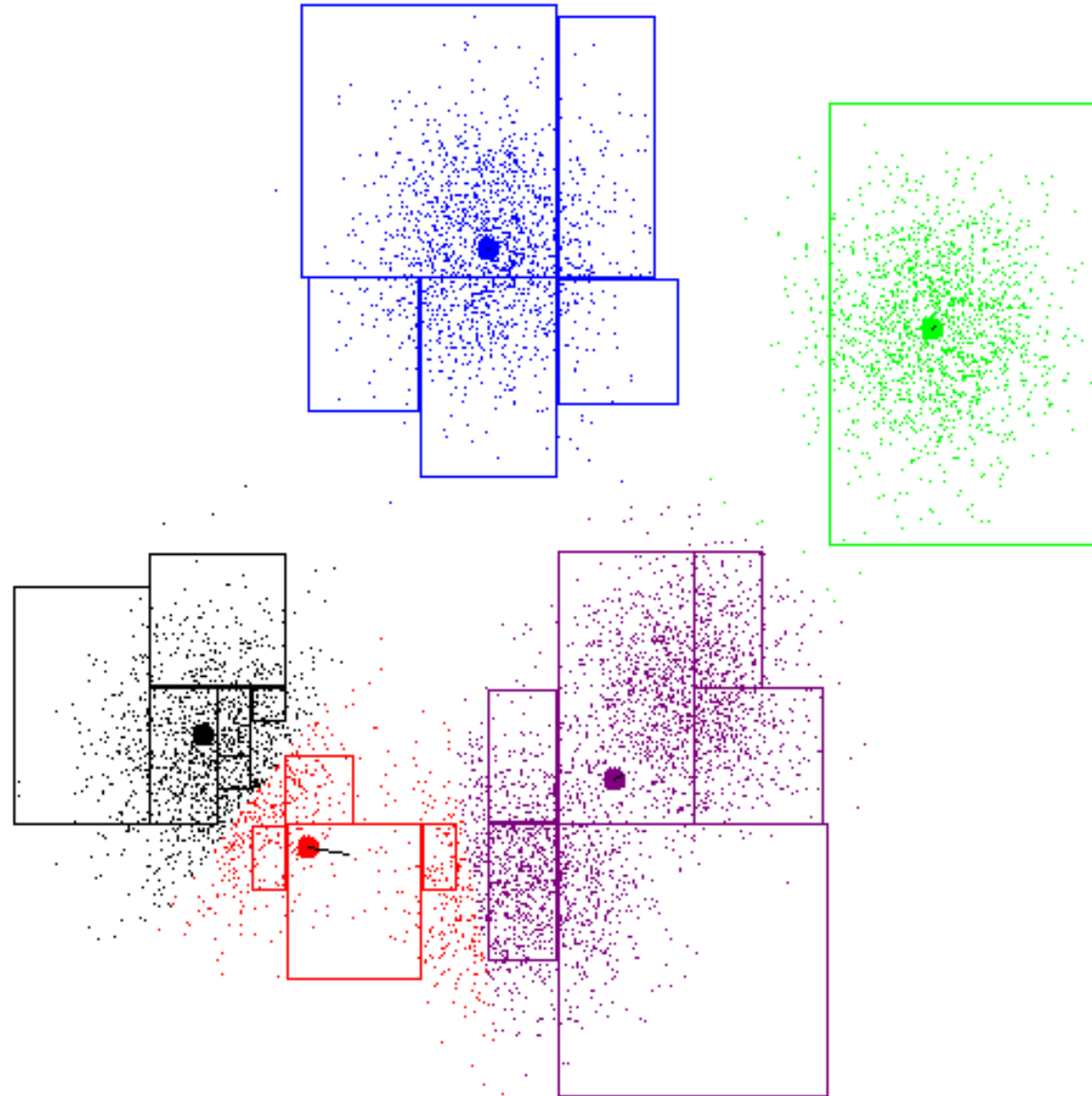
K-Means Example



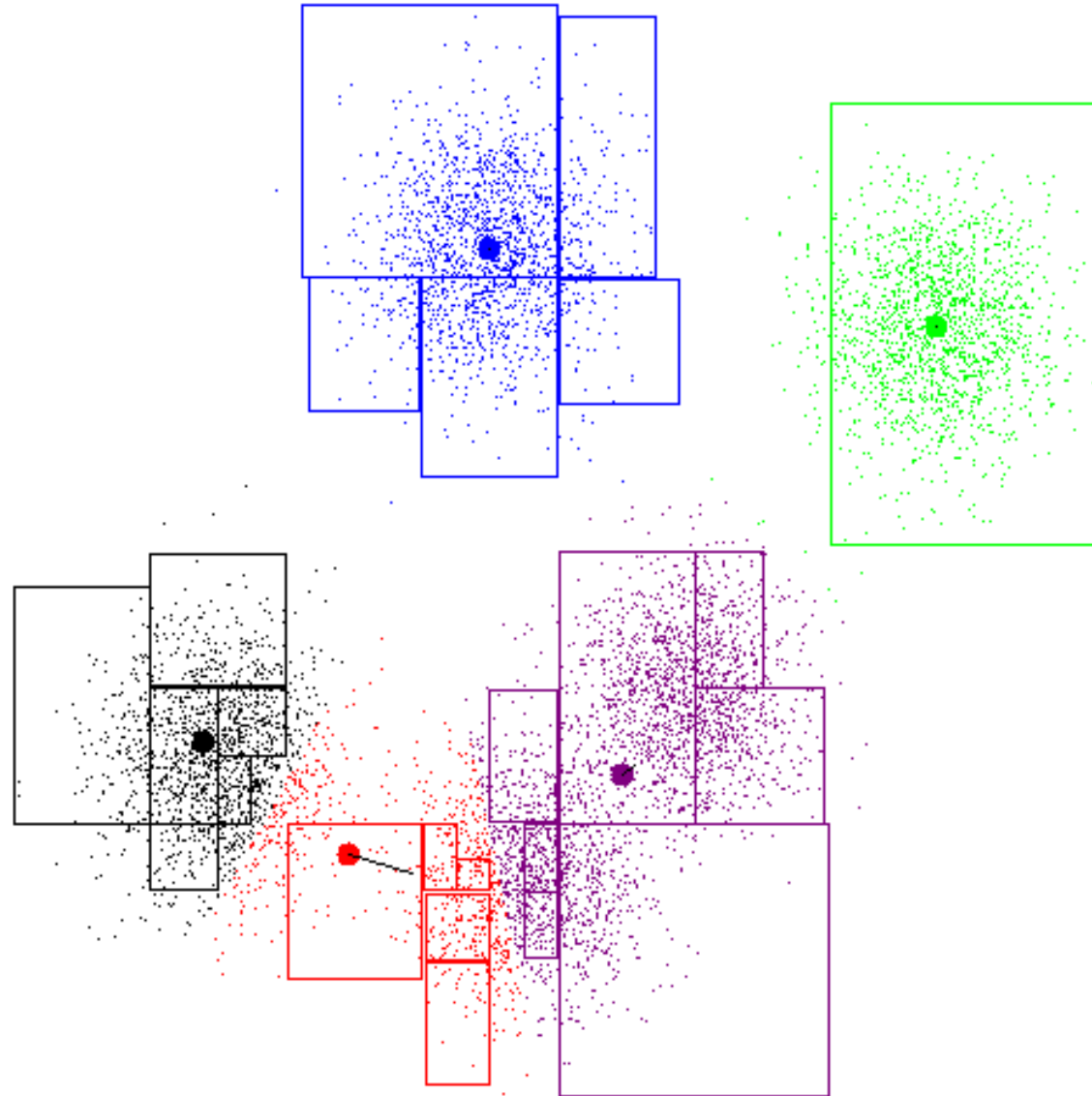
K-Means Example



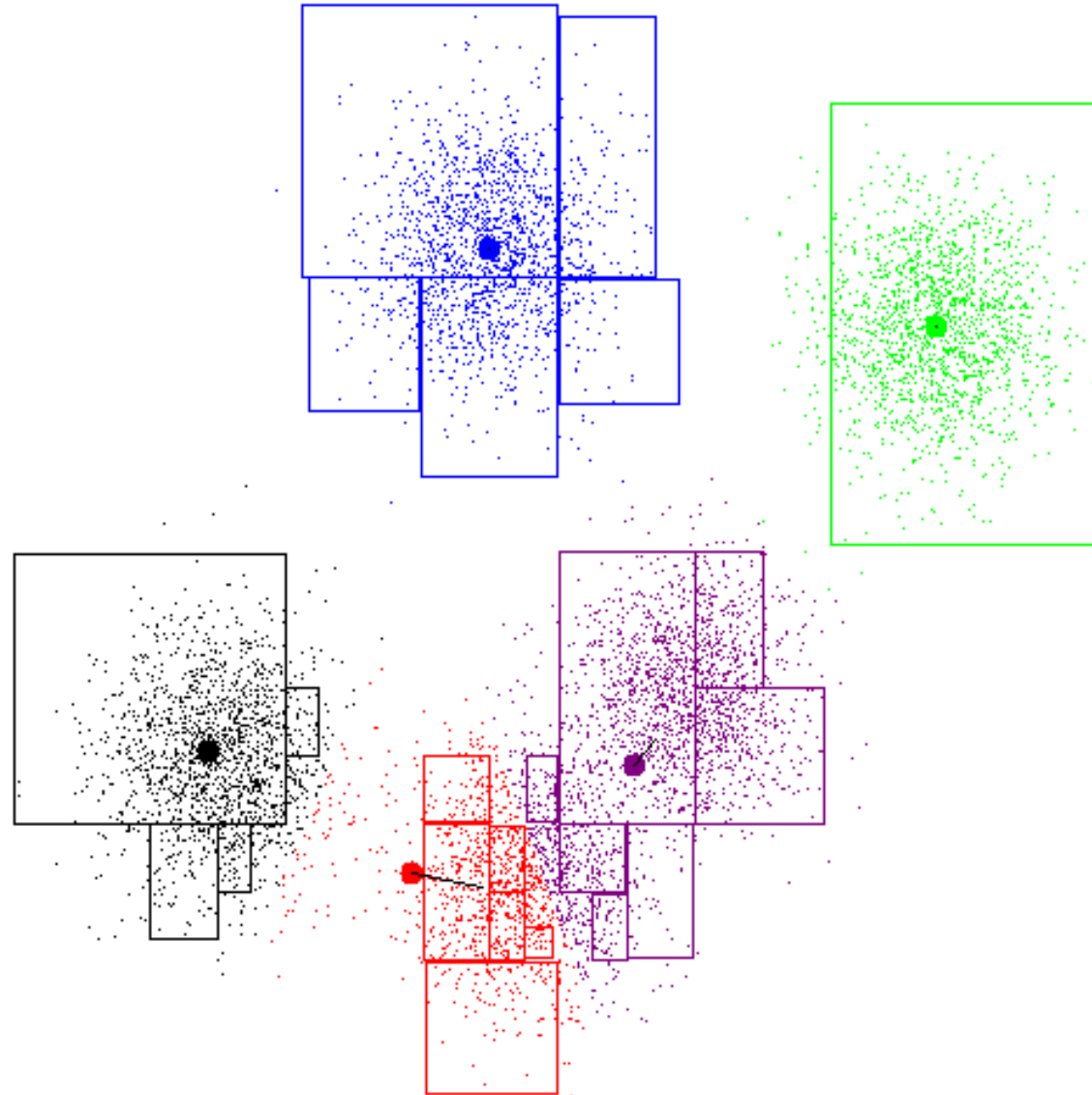
K-Means Example



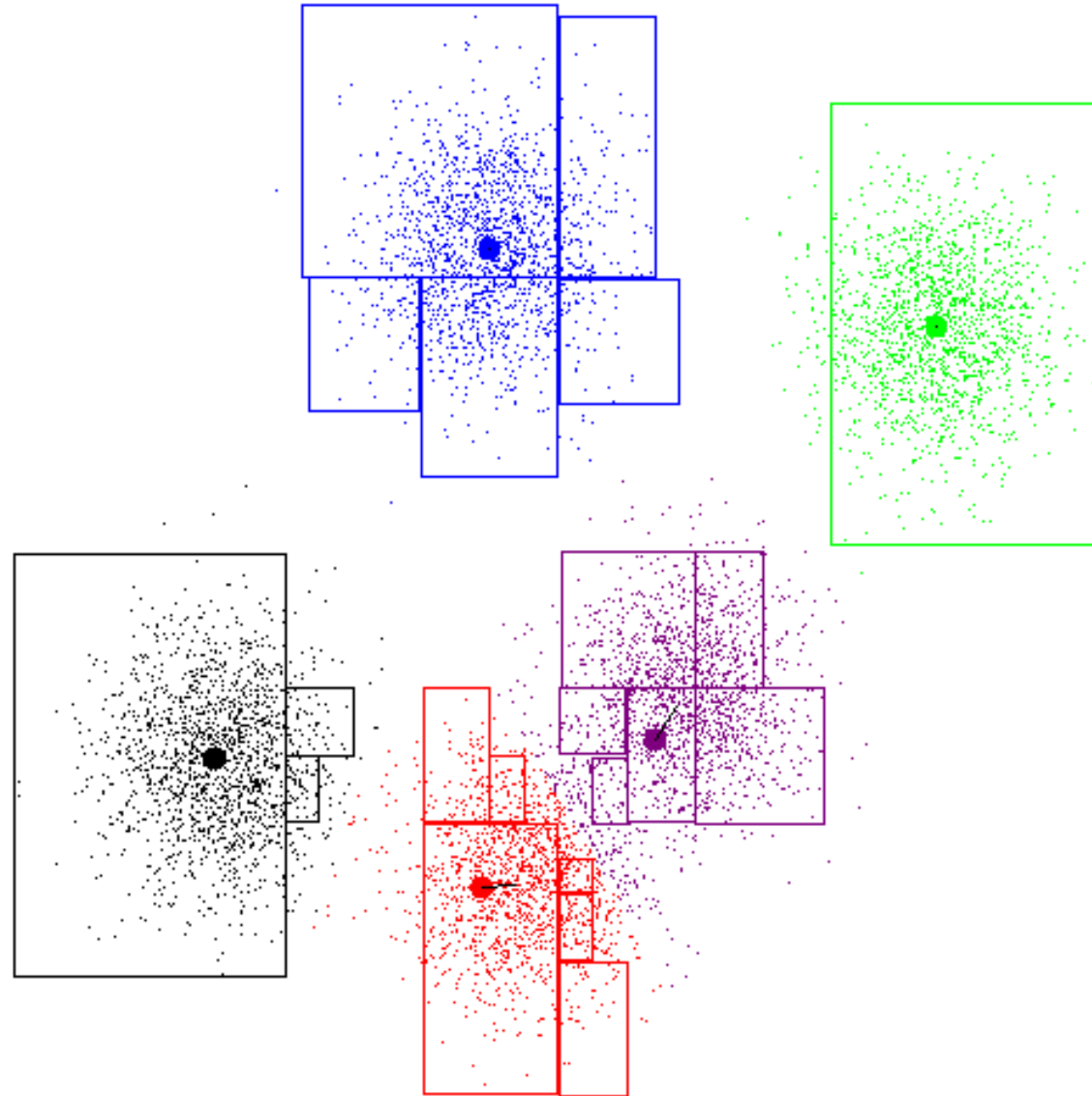
K-Means Example



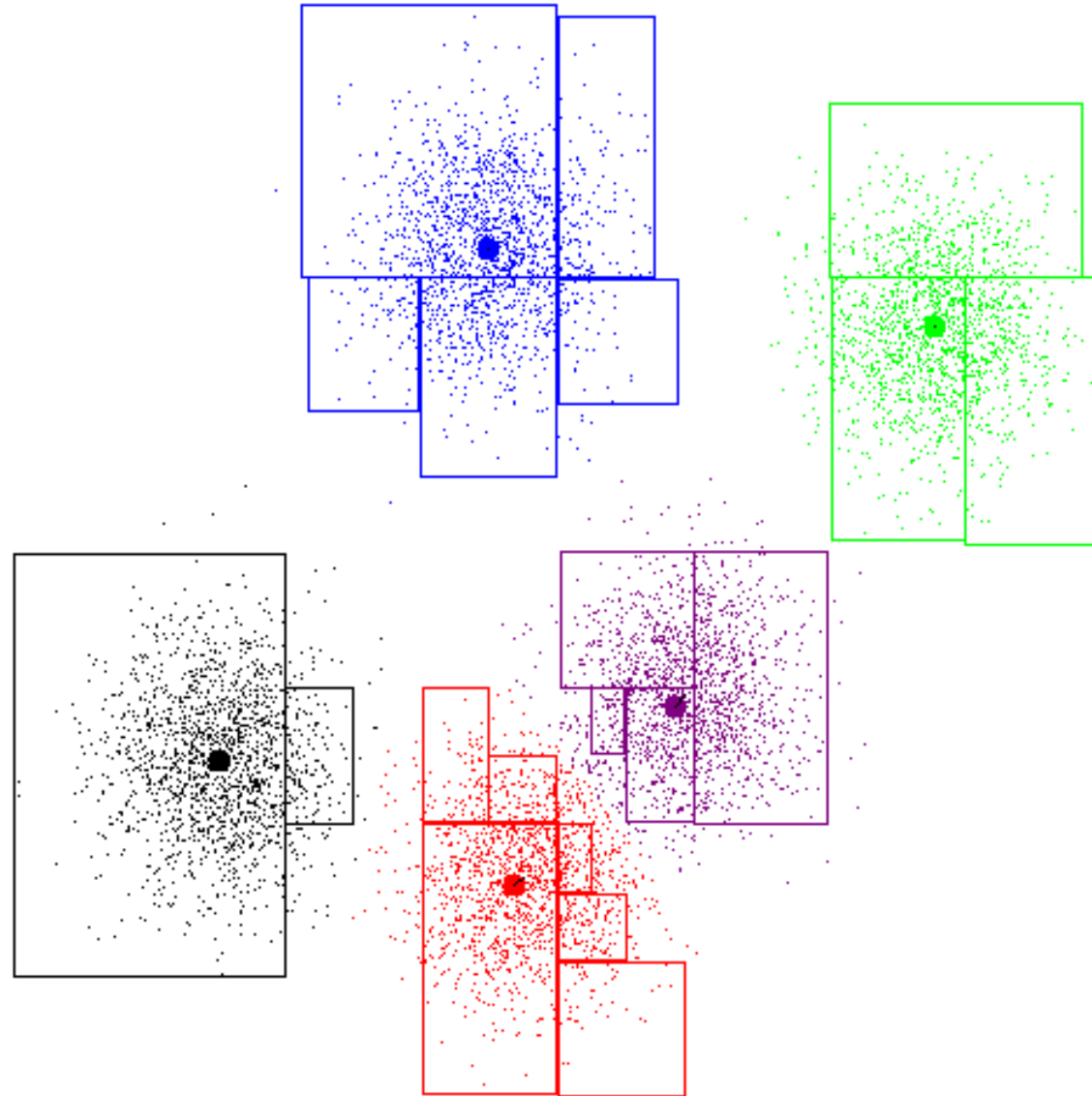
K-Means Example



K-Means Example



K-Means Example



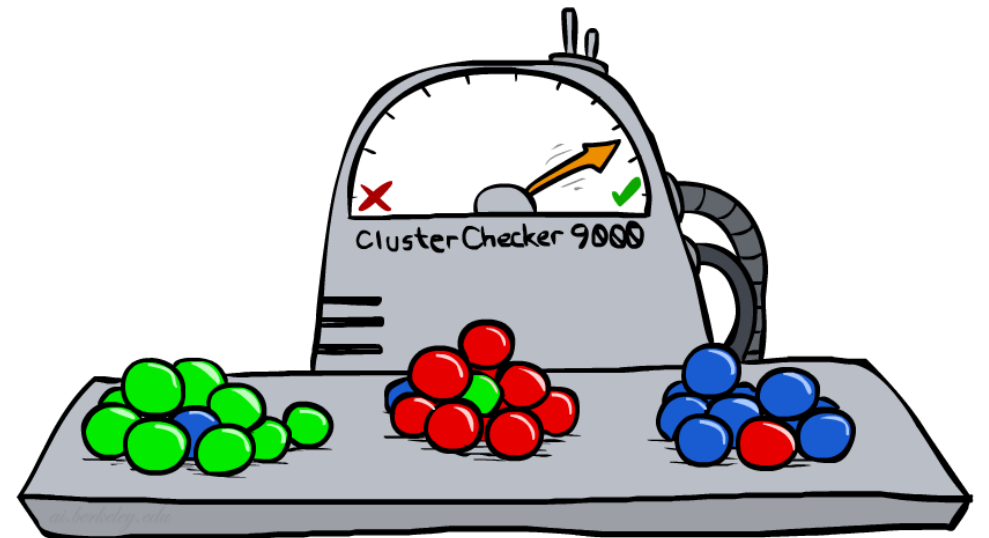
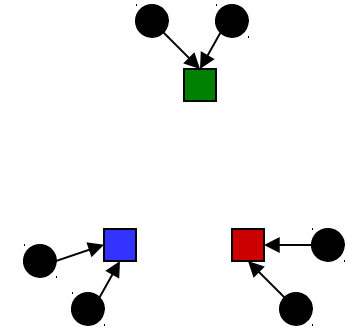
K-Means as Optimization

- Consider the total distance to the means:

$$\phi(\{x_i\}, \{a_i\}, \{c_k\}) = \sum_i \text{dist}(x_i, c_{a_i})$$

points assignments means

- Each iteration reduces phi
- Two stages each iteration:
 - Update assignments: fix means c, change assi
 - Update means: fix assignments a, change mea



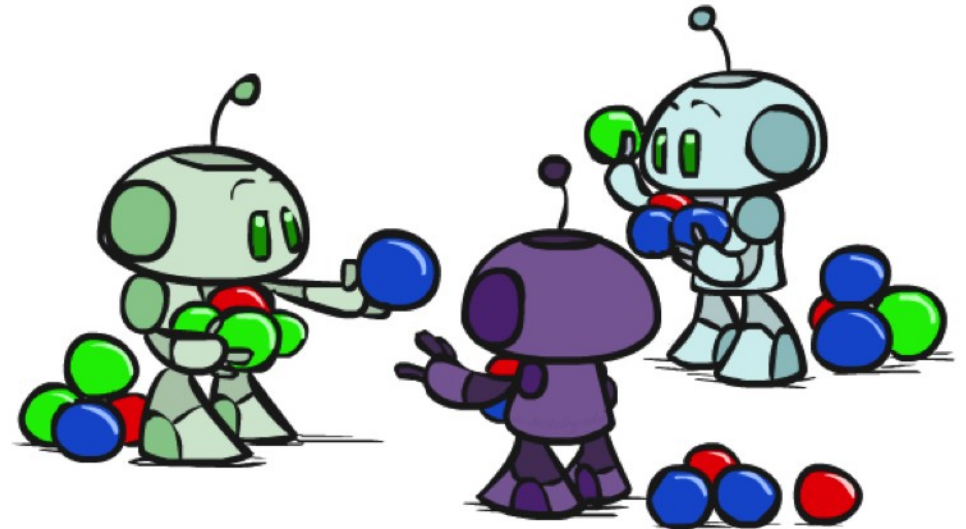
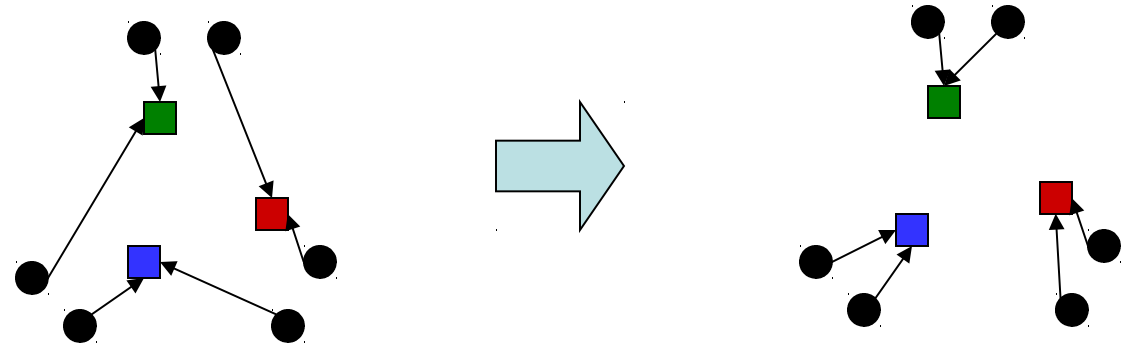
Phase I: Update Assignments

- For each point, re-assign to closest mean:

$$a_i = \operatorname{argmin}_k \operatorname{dist}(x_i, c_k)$$

- Can only decrease total distance ϕ !

$$\phi(\{x_i\}, \{a_i\}, \{c_k\}) = \sum_i \operatorname{dist}(x_i, c_{a_i})$$

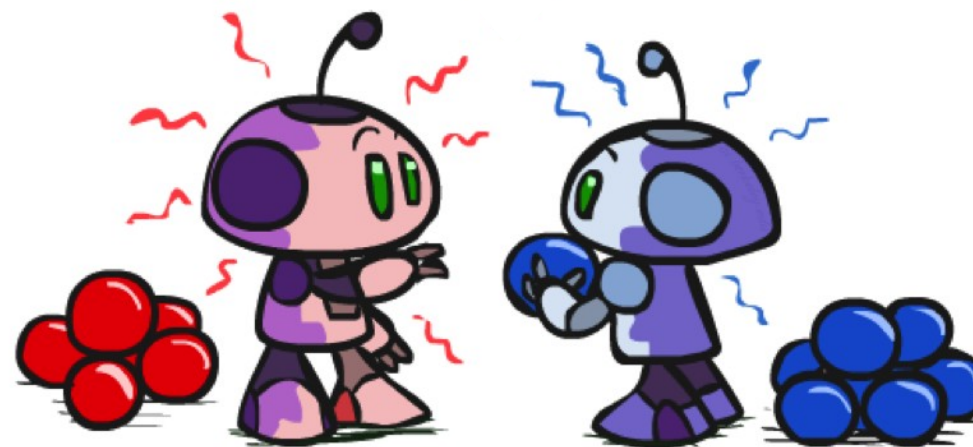
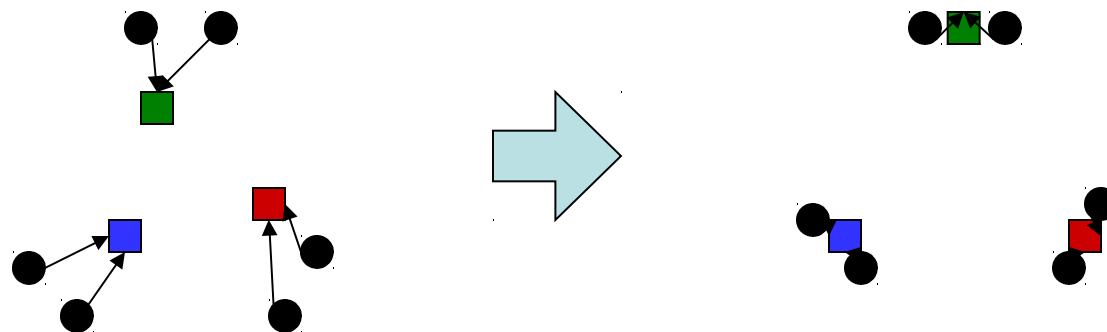


Phase II: Update Means

- Move each mean to the average of its assigned points:

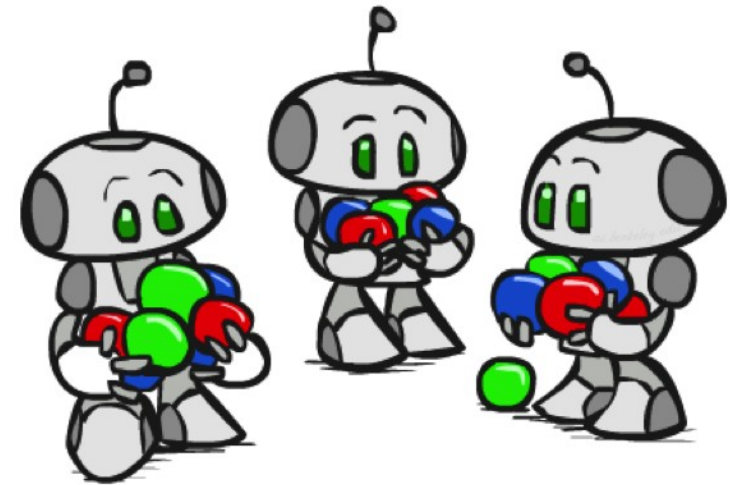
$$c_k = \frac{1}{|\{i : a_i = k\}|} \sum_{i:a_i=k} x_i$$

- Also can only decrease total distance... (Why?)
- Fun fact: the point y with minimum squared Euclidean distance to a set of points $\{x\}$ is their mean



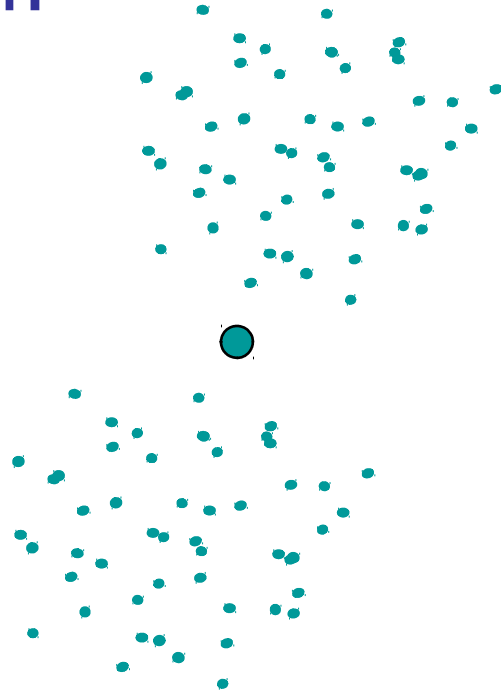
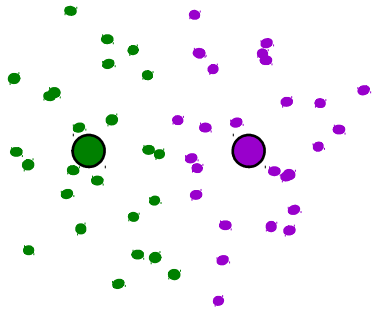
Initialization

- K-means is non-deterministic
 - Requires initial means
 - It does matter what you pick!
 - What can go wrong?
- Various schemes for preventing this kind of thing: variance-based split / merge, initialization heuristics

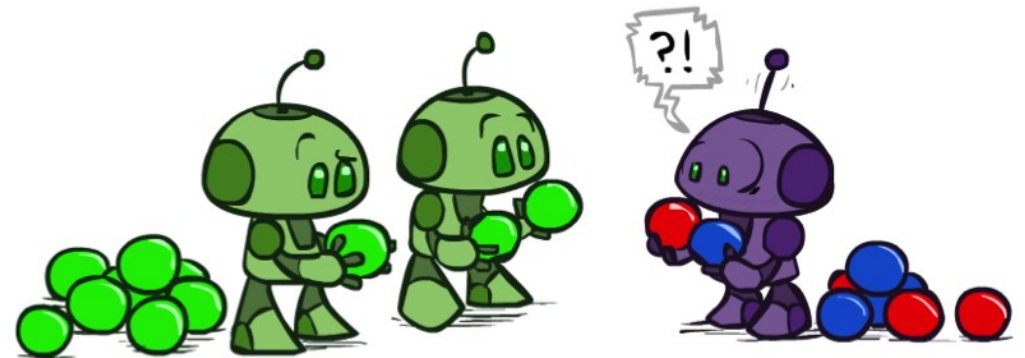


K-Means Getting Stuck

- A local optimum:

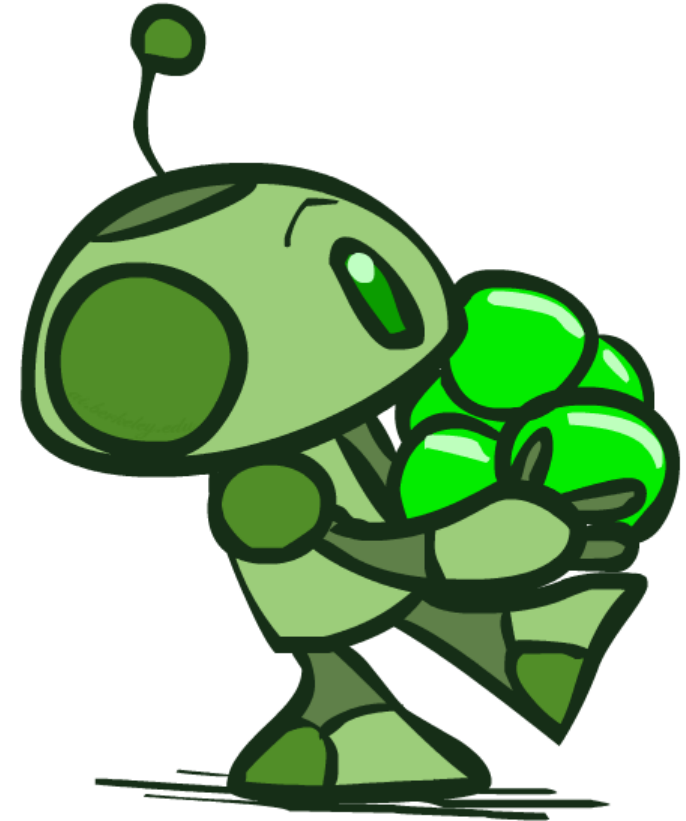


Why doesn't this work out like the earlier example, with the purple taking over half the blue?

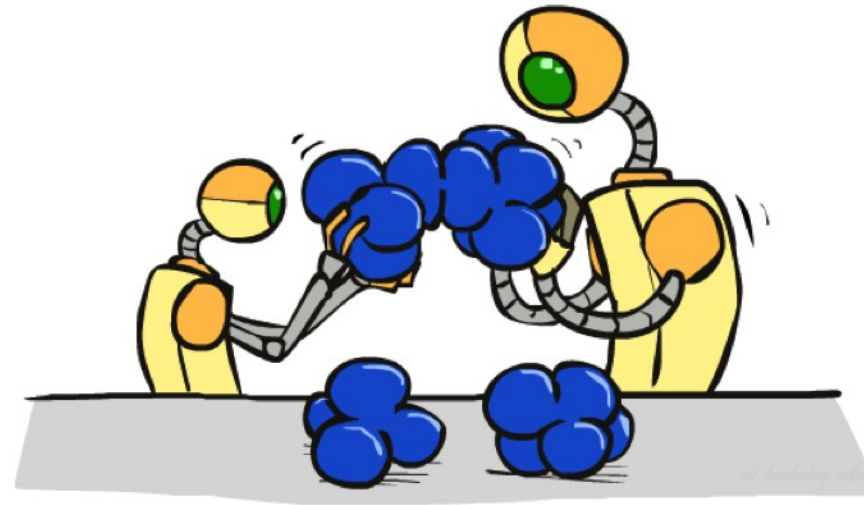
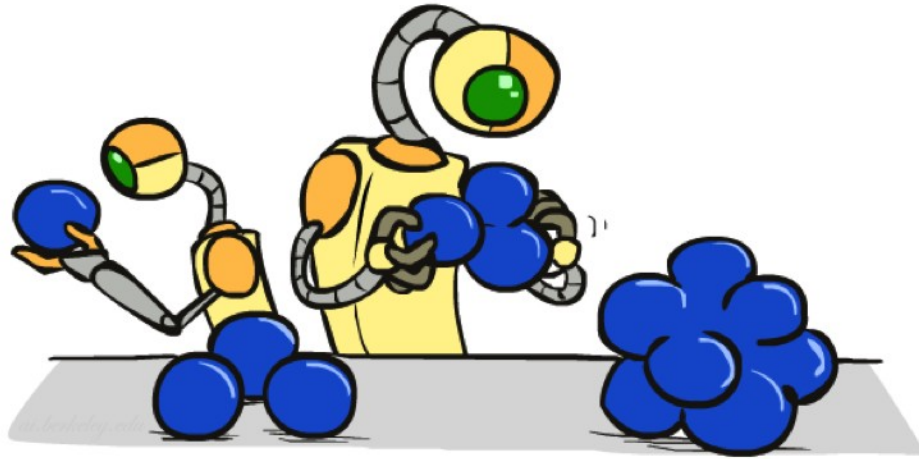


K-Means Questions

- Will K-means converge?
 - To a global optimum?
- Will it always find the true patterns in the data?
 - If the patterns are very very clear?
- Will it find something interesting?
- Do people ever use it?
- How many clusters to pick?

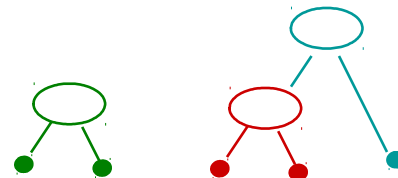
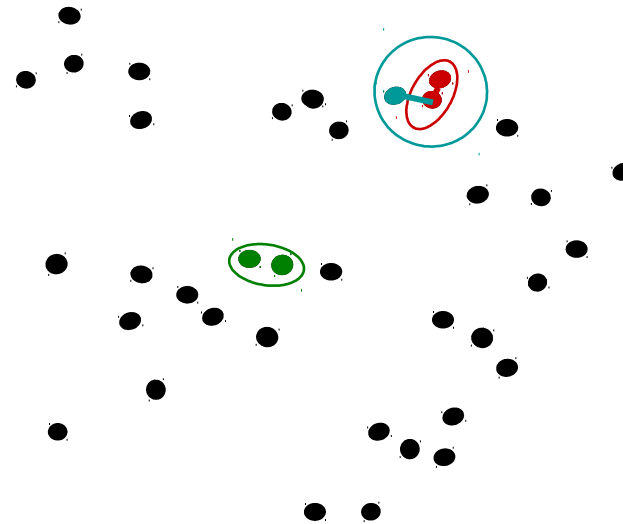


Agglomerative Clustering



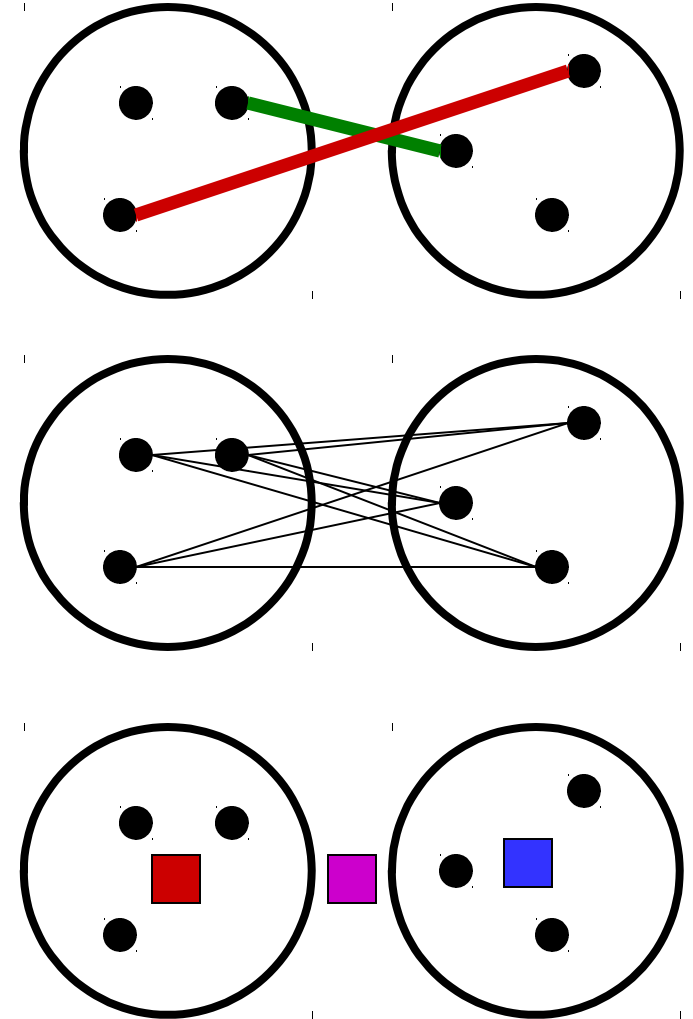
Agglomerative Clustering

- **Agglomerative clustering:**
 - First merge very similar instances
 - Incrementally build larger clusters out of smaller clusters
- **Algorithm:**
 - Maintain a set of clusters
 - Initially, each instance in its own cluster
 - Repeat:
 - Pick the two **closest** clusters
 - Merge them into a new cluster
 - Stop when there's only one cluster left
- Produces not one clustering, but a family of clusterings represented by a **dendrogram**



Agglomerative Clustering

- How should we define “closest” for clusters with multiple elements?
- Many options
 - Closest pair (single-link clustering)
 - Farthest pair (complete-link clustering)
 - Average of all pairs
 - Ward’s method (min variance, like k-means)
- Different choices create different clustering behaviors



Example: Google News

The screenshot shows the Google News homepage. At the top, the Google logo is followed by a search bar and links for "Search News", "Search the Web", "Advanced news search", and "Preferences". Below the search bar, a red banner reads "Search and browse 25,000 news sources updated continuously." The page is divided into sections for "World" and "U.S.". The "World" section is circled in black and contains three news items: "Heavy Fighting Continues As Pakistan Army Battles Taliban" (Voice of America, 10 hours ago), "Sri Lanka admits bombing safe haven" (guardian.co.uk, 3 hours ago), and "Buffett Calls Investment Candidates' 2008 Performance Subpar" (Bloomberg, 2 hours ago). The "U.S." section contains three news items: "Weekend Opinionator: Souter, Specter and the Future of the GOP" (New York Times, 48 minutes ago), "Joe Biden, the Flu and You" (New York Times, 48 minutes ago), and "Chrysler's Fall May Help Administration Reshape GM" (New York Times, 5 hours ago). The "Chrysler's Fall..." item is circled in red. Small thumbnail images are placed next to several news items, such as "ABC News" for the Pakistan article, "WA today" for the Sri Lanka article, and "FOXNews" for the Souter article. The "Chrysler's Fall..." item has a thumbnail from "guardian.co.uk".

Top-level categories:
supervised classification

Story groupings:
unsupervised clustering