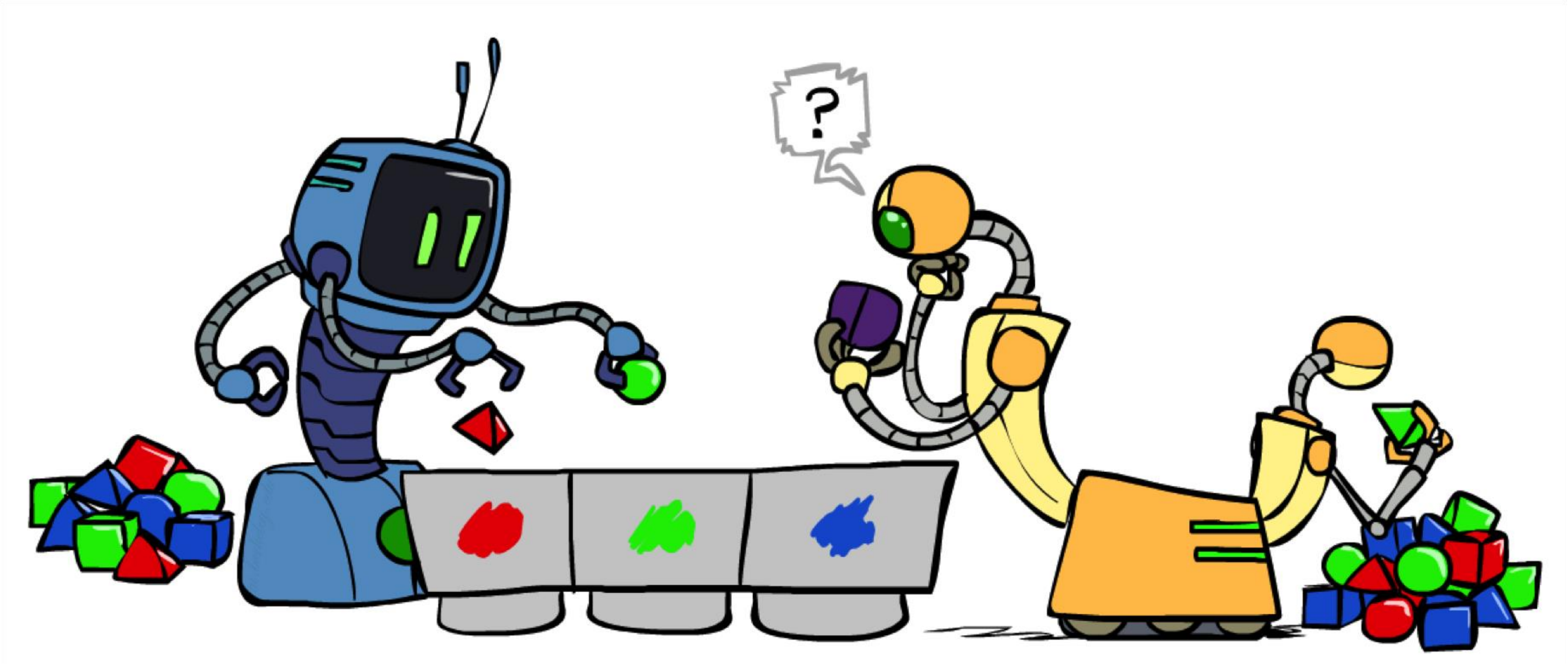
CSE 445: Machine Learning

Unsupervised Learning



Why use Unsupervised Learning?

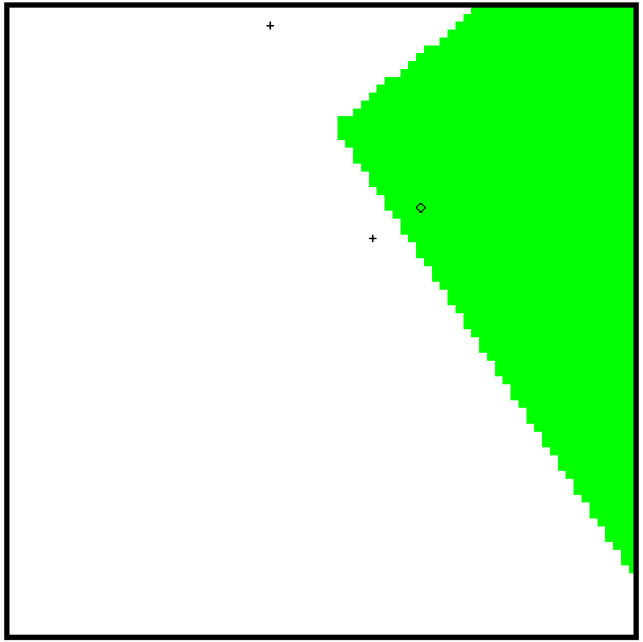
- Supervised Learning relies on large, labelled datasets to work!
- Data collection and labelling is:
  - Expensive
  - Slow
- Disconnect between how human learning operates
- Supervision → Learn to group objects into one class because someone tells us to
- Experience (Unsupervised) →Learn to group objects into one class by seeing many of them

With Supervision

No Supervision

Instance-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
  - Trade-offs: Small k gives relevant neighbors, Large k gives smoother functions



http://www.cs.cmu.edu/~zhuxj/courseproject/knndemo/KNN.html

Nearest-Neighbor Classification

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

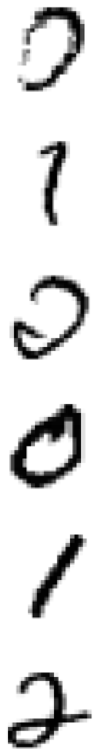
- Encoding: image is vector of intensities:

$$= \langle 0.0 \;\; 0.0 \;\; 0.3 \;\; 0.8 \;\; 0.7 \;\; 0.1 \ldots 0.0 \rangle$$

- 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?)



0

1

2

0

1

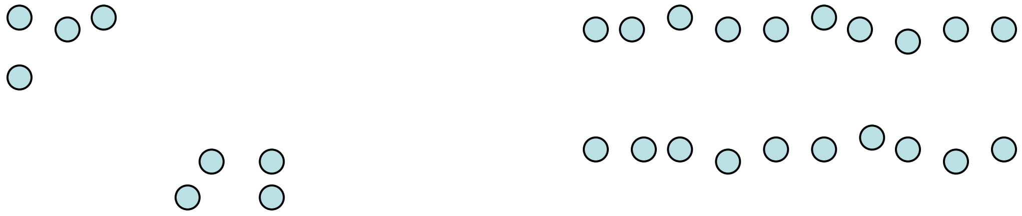
2

Clustering

- Unsupervised machine learning → no a priori labels available
- 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

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

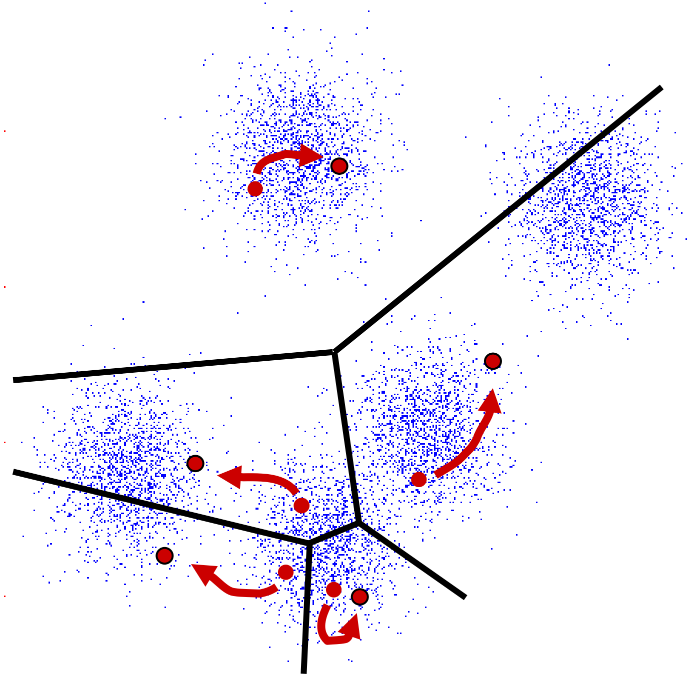


- What could "similar" mean?
  - One option: small (squared) Euclidean distance

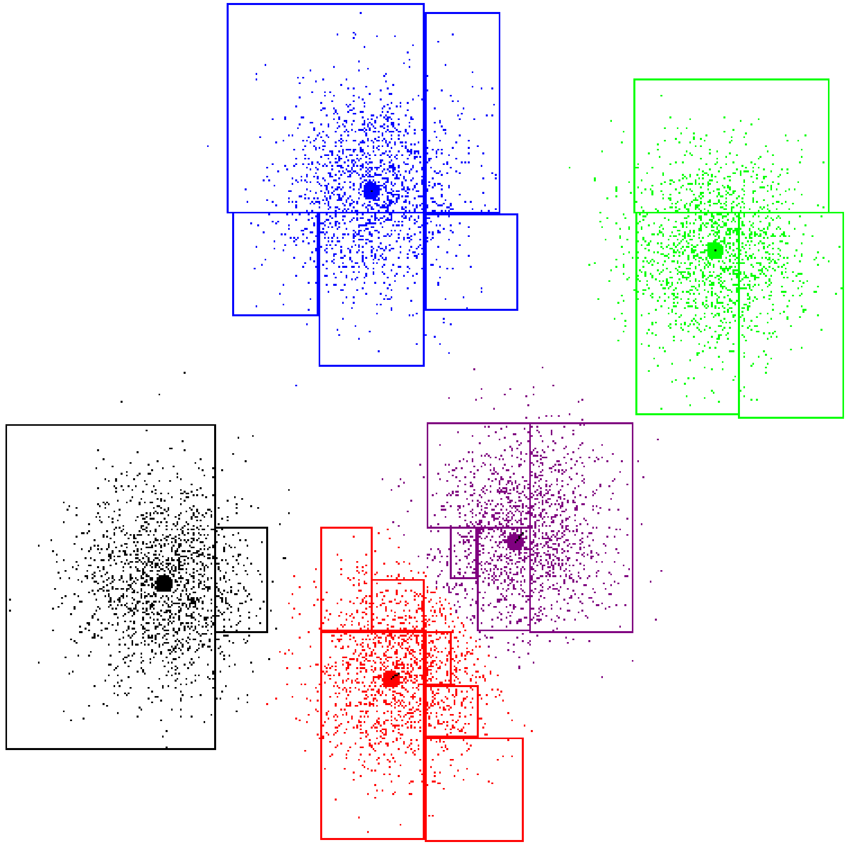
$$\text{dist}(x, y) = (x - y)^\top (x - y) = \sum_i (x_i - y_i)^2$$

K-Means

- 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 as Optimization

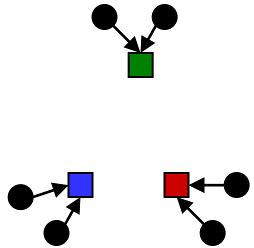
- Consider the total distance to the means:

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

points

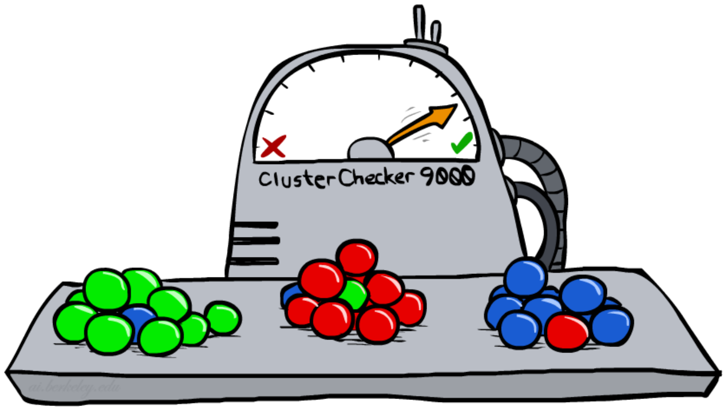
assignments

means



- Each iteration reduces phi

- Two stages each iteration:
  - Update assignments: fix means c, change assignments a
  - Update means: fix assignments a, change means c



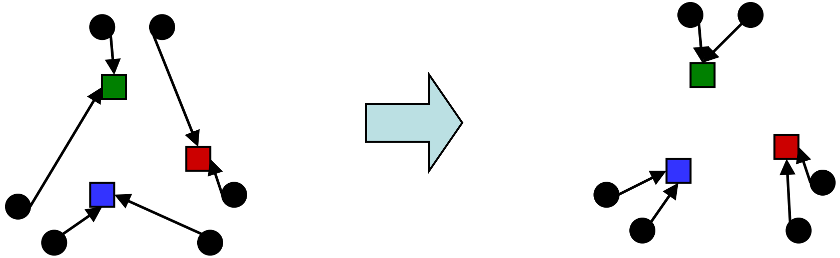
Phase I: Update Assignments

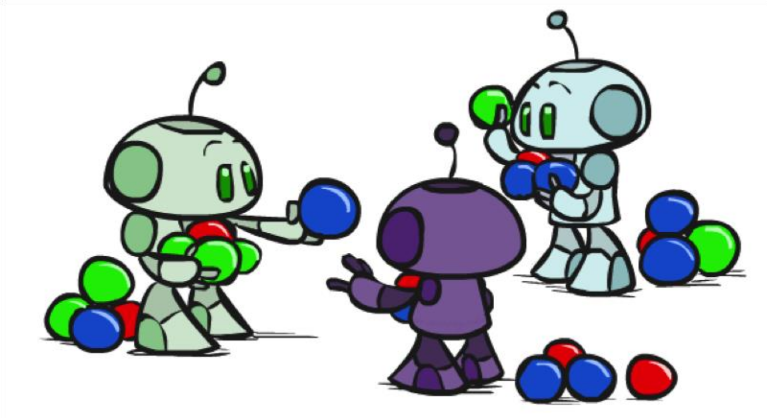
- For each point, re-assign to closest mean:

$$a_i = \underset{k}{\text{argmin}}\, \text{dist}(x_i, c_k)$$

- Can only decrease total distance phi!

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





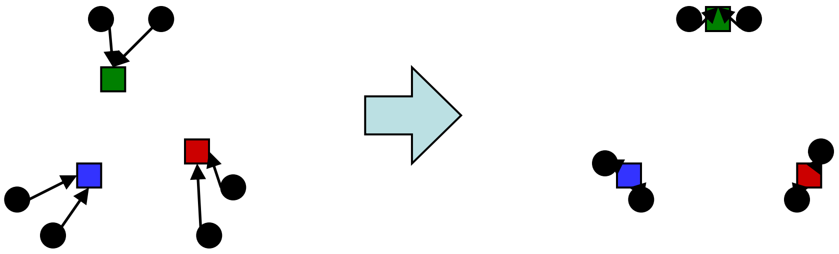
Phase II: Update Means

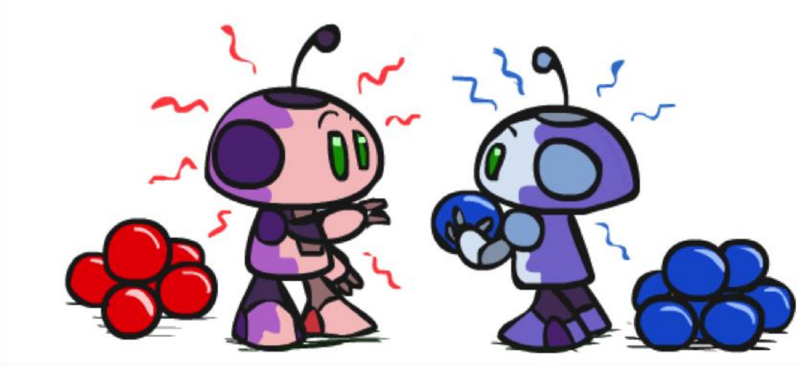
- Move each mean to the average of its assigned points:

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

- Also can only decrease total distance… (Why?)

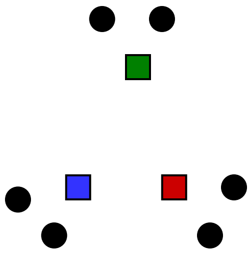
- Fun fact: the point y with minimum squared Euclidean distance to a set of points {x} is their mean

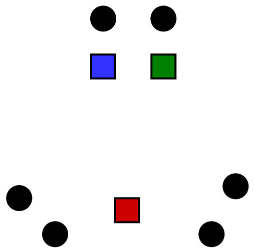


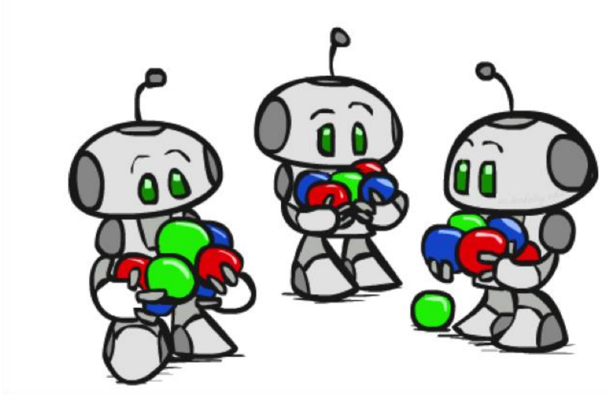


Initialization

- K-means is non-deterministic
  - Requires initial means
  - It does matter what you pick!
  - What can go wrong?



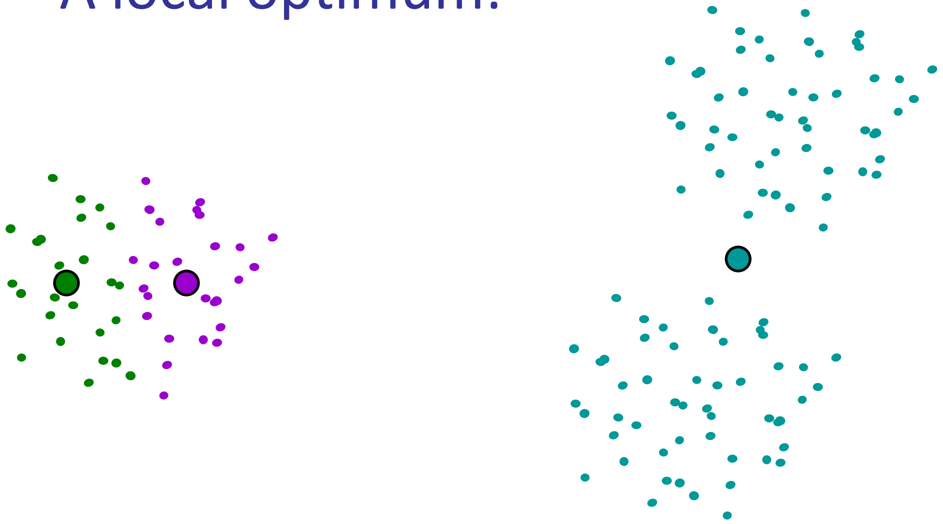




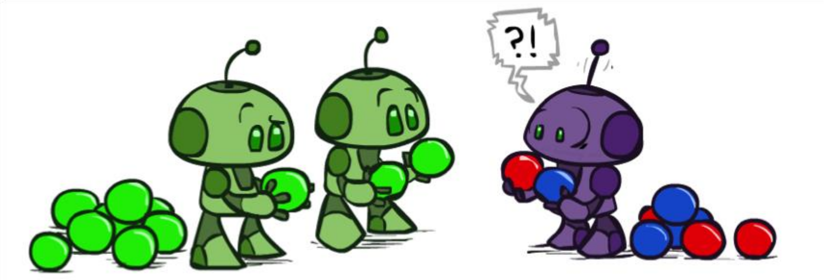
  - Various schemes for preventing this kind of thing: variance-based split / merge, initialization heuristics

K-Means Getting Stuck

- A local optimum:

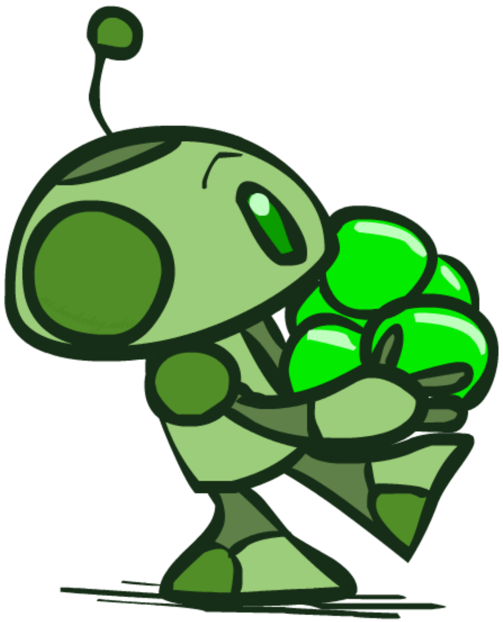


Why doesn't this work out like the earlier example, with the purple taking over half the blue?



K-Means Questions

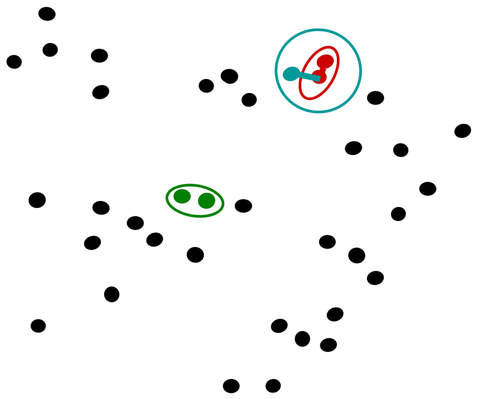
- Will K-means converge?
  - To a global optimum?
- Will it always find the true patterns in the data?
  - If the patterns are very very clear?
- Will it find something interesting?
- Do people ever use it?
- How many clusters to pick?



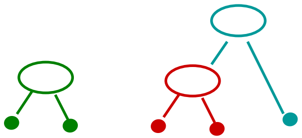
Agglomerative Clustering

- Agglomerative clustering:
  - 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