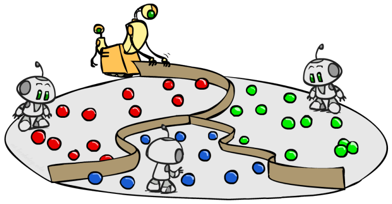
# CS 188: Artificial Intelligence

kNN and Clustering



Instructor: Marco Alvarez --- University of Rhode Island

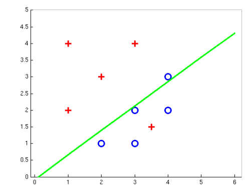
[These slides were created by Dan Klein and Pieter Abbeel for CS188 Intro to AI at UC Berkeley. All CS188 materials are available at http://ai.berkeley.edu.]

1

## Case-Based Learning

2

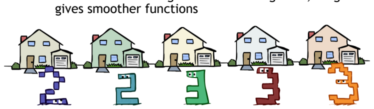
## Non-Separable Data

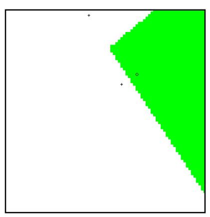


3

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



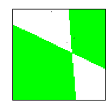


http://www.cs.cmu.edu/~zhuxj/courseproject/knndemo

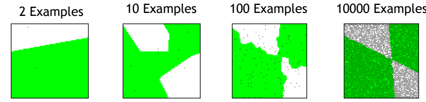
4

## 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




5

## 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:

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

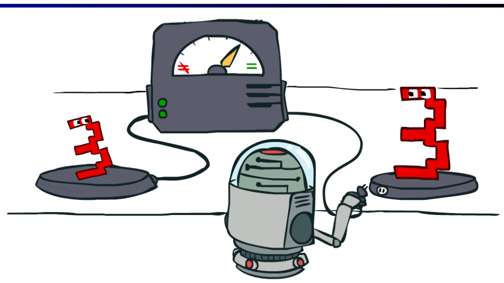
- What's the similarity function?
  - Dot product of two images vectors?

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

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


6

## Similarity Functions



7

## 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:

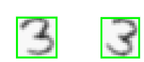
$$\text{sim}(x, x') = x \cdot x' = \sum_i x_i x'_i$$

- Note: not all similarities are of this form

8

## 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:

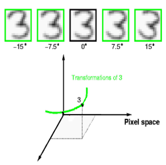


    - 16 x 16 = 256 pixels; a point in 256-dim space
    - These points have small similarity in $R^{256}$ (why?)
  - How can we incorporate such invariances into our similarities?

This and next few slides adapted from Xiao Hu, UIUC

9

## Rotation Invariant Metrics




- Each example is now a curve in $R^{256}$
- Rotation invariant similarity:

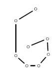
  s'=max s( r(  ), r(  ))

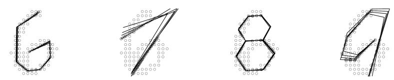
- E.g. highest similarity between images' rotation lines

10

## 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





Examples from [Hastie 94

11

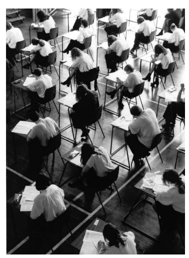
## A Tale of Two Approaches…

- 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!

12

## Recap: Classification

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



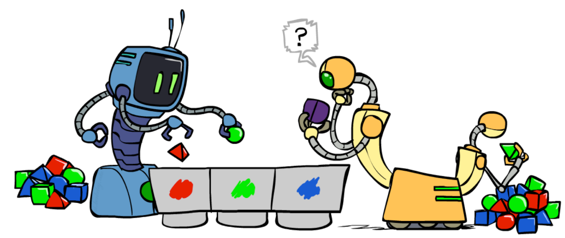
13

## 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

14

## Clustering




15

## Clustering

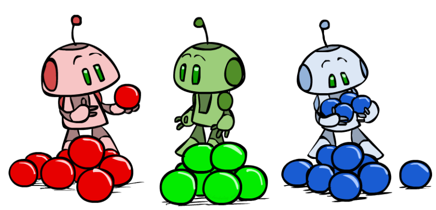
- Basic idea: group together similar instances
- Example: 2D point patterns

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

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

16

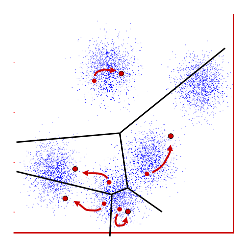
## K-Means



17

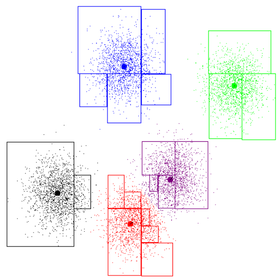
## K-Means

- 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



18

## K-Means Example



19

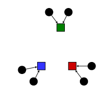
## K-Means as Optimization

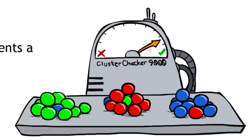
- Consider the total distance to the means:

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

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






20

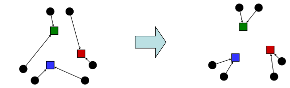
## Phase I: Update Assignments

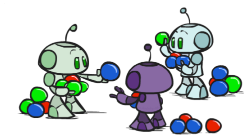
- For each point, re-assign to closest mean:

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

- Can only decrease total distance phi!

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





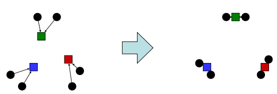
21

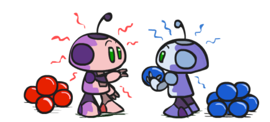
## Phase II: Update Means

- 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



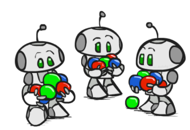


22

## Initialization

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

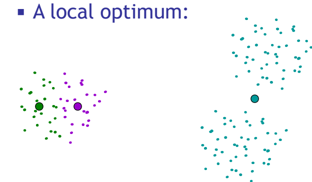
  - Various schemes for preventing this kind of thing: variance-based split / merge, initialization heuristics



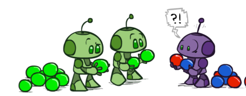
23

## K-Means Getting Stuck

- A local optimum:



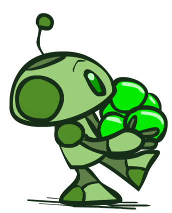
*Why doesn't this work out like the earlier example, with the purple taking over half the blue?*




24

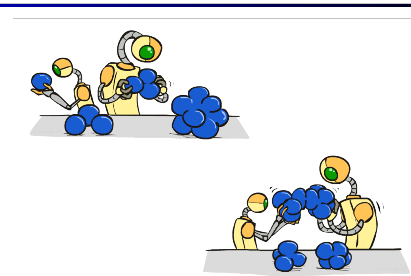
## K-Means Questions

- 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?



25

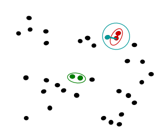
## Agglomerative Clustering

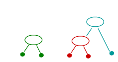


26

## 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

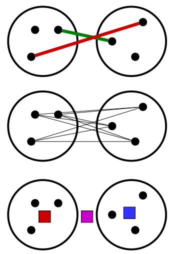




27

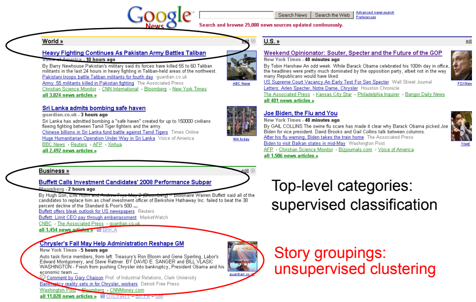
# Agglomerative Clustering

- 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




Top-level categories: supervised classification

Story groupings: unsupervised clustering