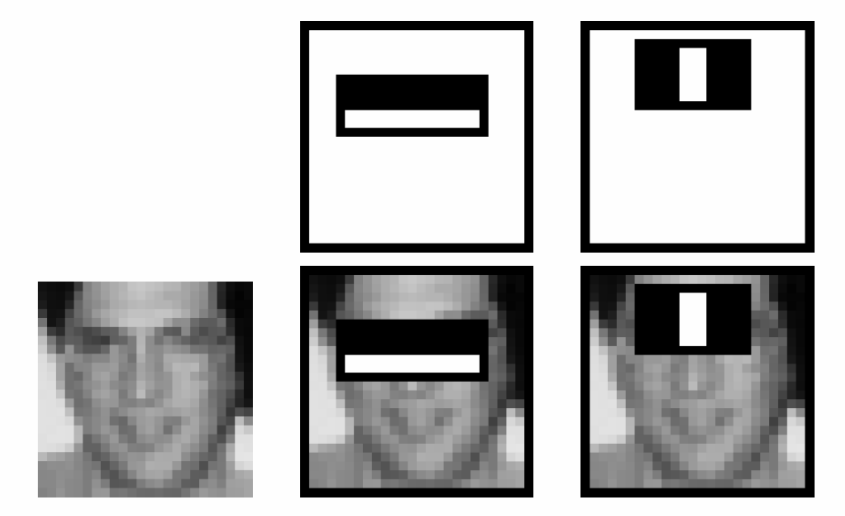
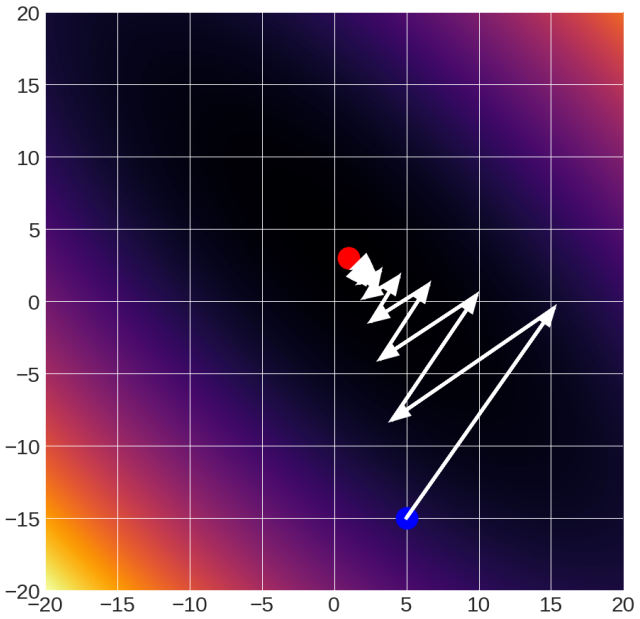
Lecture 9: Object classification

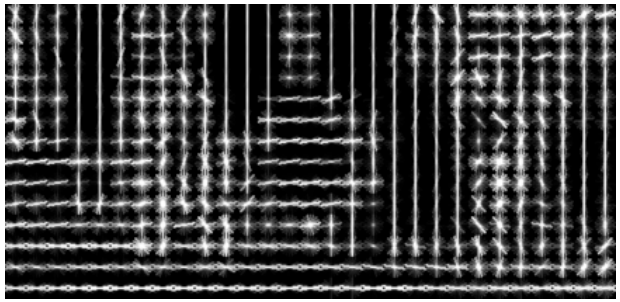
COS 429: Computer Vision



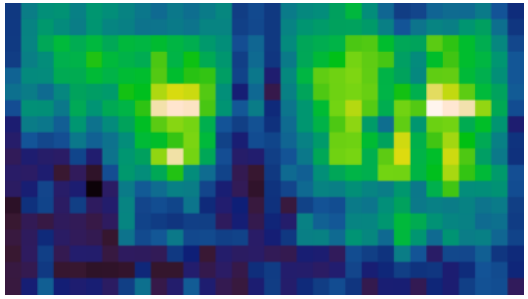
Last time







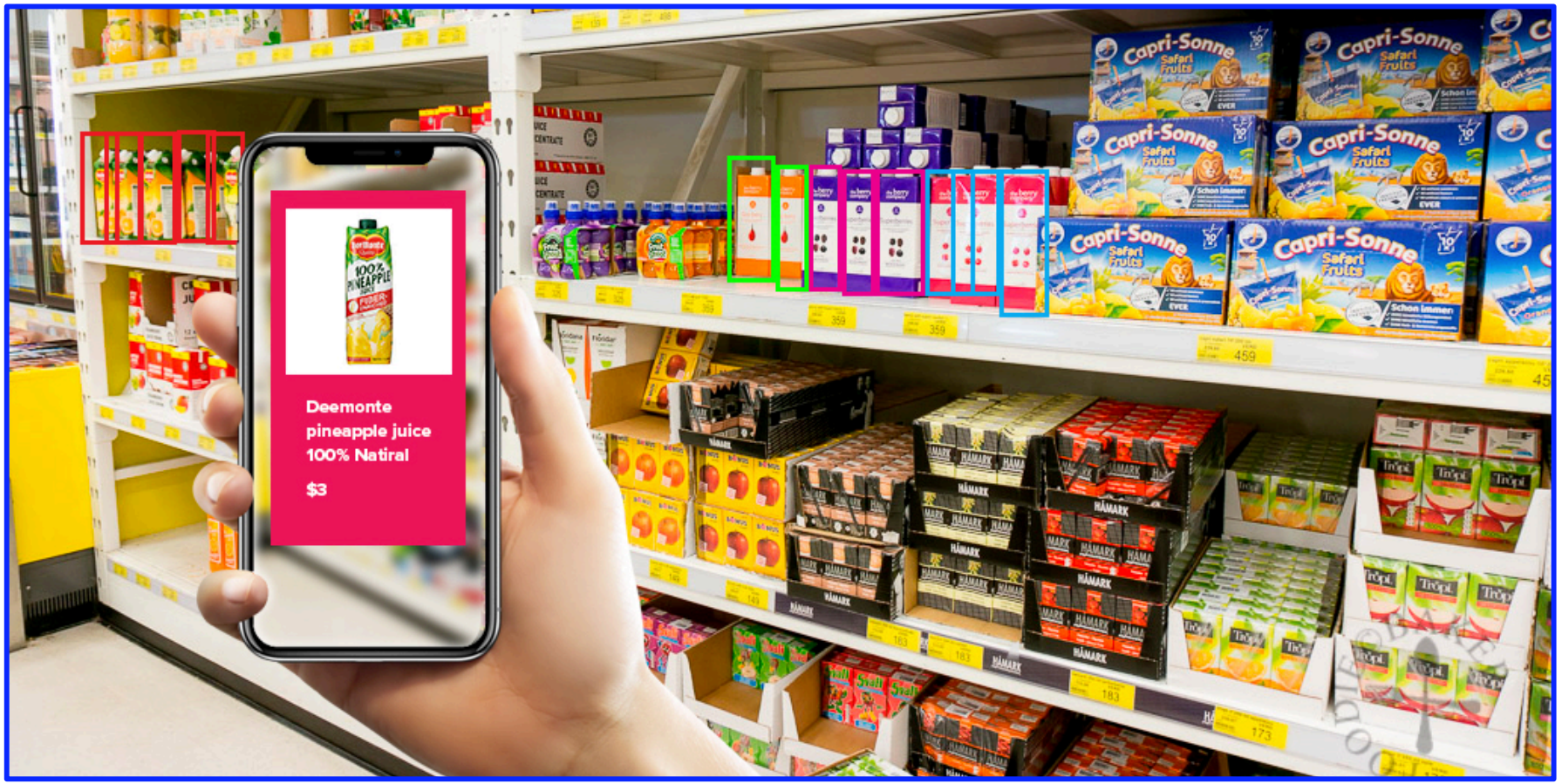




Key concepts

1. Bag of Words model / Spatial Pyramid Matching
2. K-means clustering
3. Mean Average Precision

Object classification use cases



https://infrrd.ai/blogs/image-recognition-product-and-shelf-monitoring-and-analysis

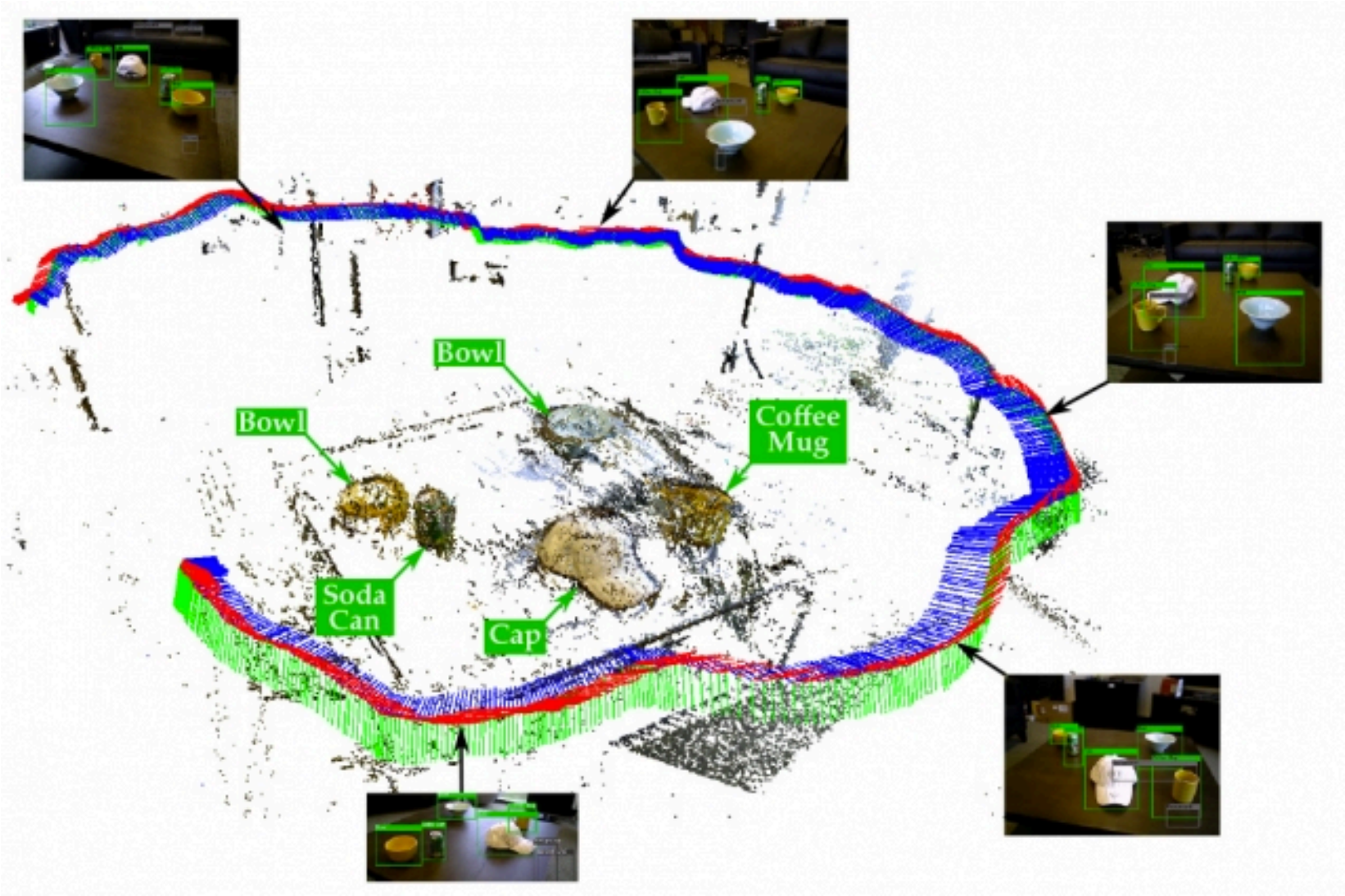
Object classification use cases



Assistance for the blind

https://vizwiz.org

Object classification use cases



http://news.mit.edu/2015/object-recognition-robots-0724

Grounding example and dataset: Caltech 101

COMPUTATIONAL VISION AT CALTECH

Caltech 101

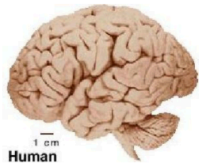
New Caltech256 New

[Description][Download][Discussion [Other Datasets]









Description

Pictures of objects belonging to 101 categories. About 40 to 800 images per category. Most categories have about 50 images. Collected in September 2003 by Fei-Fei Li, Marco Andreetto, and Marc 'Aurelio Ranzato. The size of each image is roughly 300 x 200 pixels.
We have carefully clicked outlines of each object in these pictures, these are included under the 'Annotations.tar'. There is also a matlab script to view the annotaitons, 'show_annotations.m'.

http://www.vision.caltech.edu/Image_Datasets/Caltech101/

Caltech 101: umbrellas





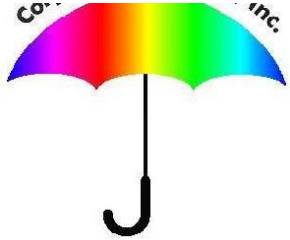
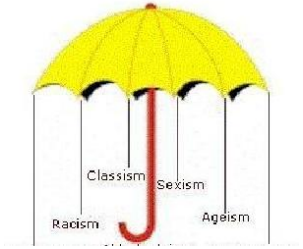
















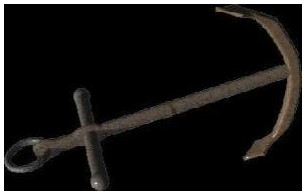


Haar features on Caltech 101

How well would Haar features work?

Some example object classes



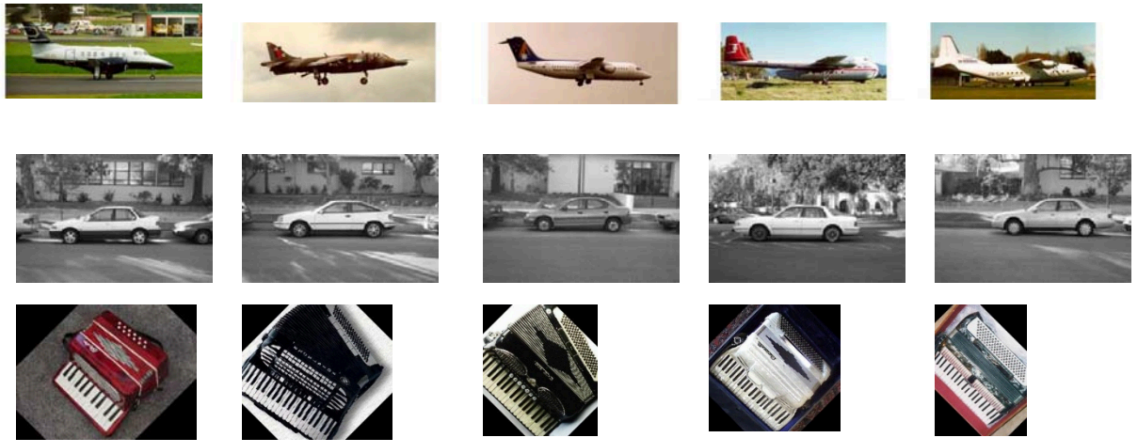




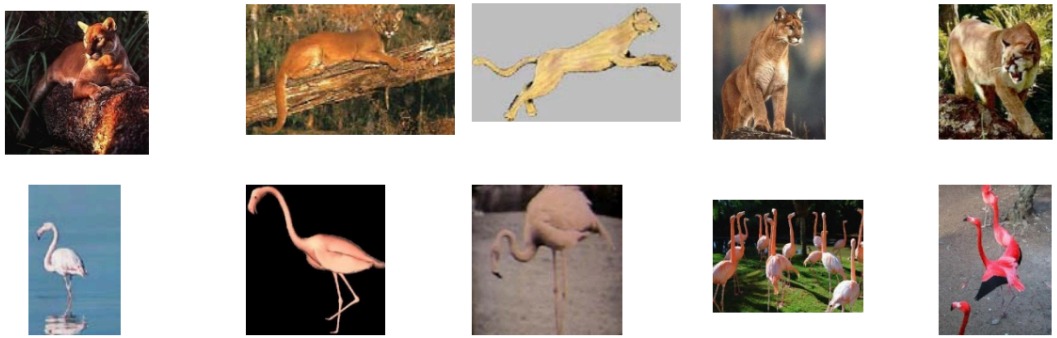
Haar features on Caltech 101

How well would Haar features work?

High accuracy categories



Low accuracy categories

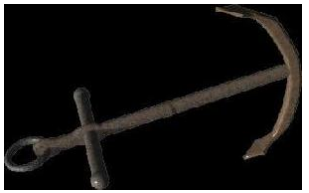


Credit: Kristen Grauman

Algorithms on Caltech 101

- How well would Haar features work?
- How well would HOG features work?
- **Need for a less shape-constrained model**



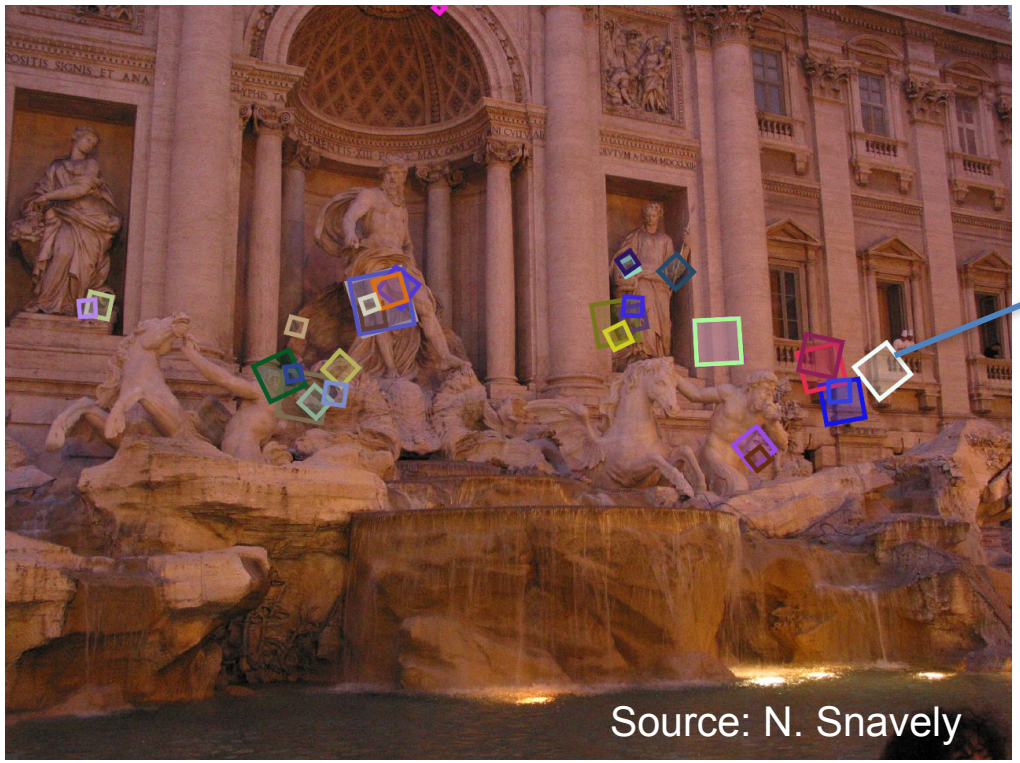







Bag of Words model

Had some preliminaries



128-dimensional
vector **x**

- Know how to find interesting locations in the image
- Know how to describe image areas (SIFT)

Category recognition with SIFT

- Extract features from set of images (either densely or at key points)

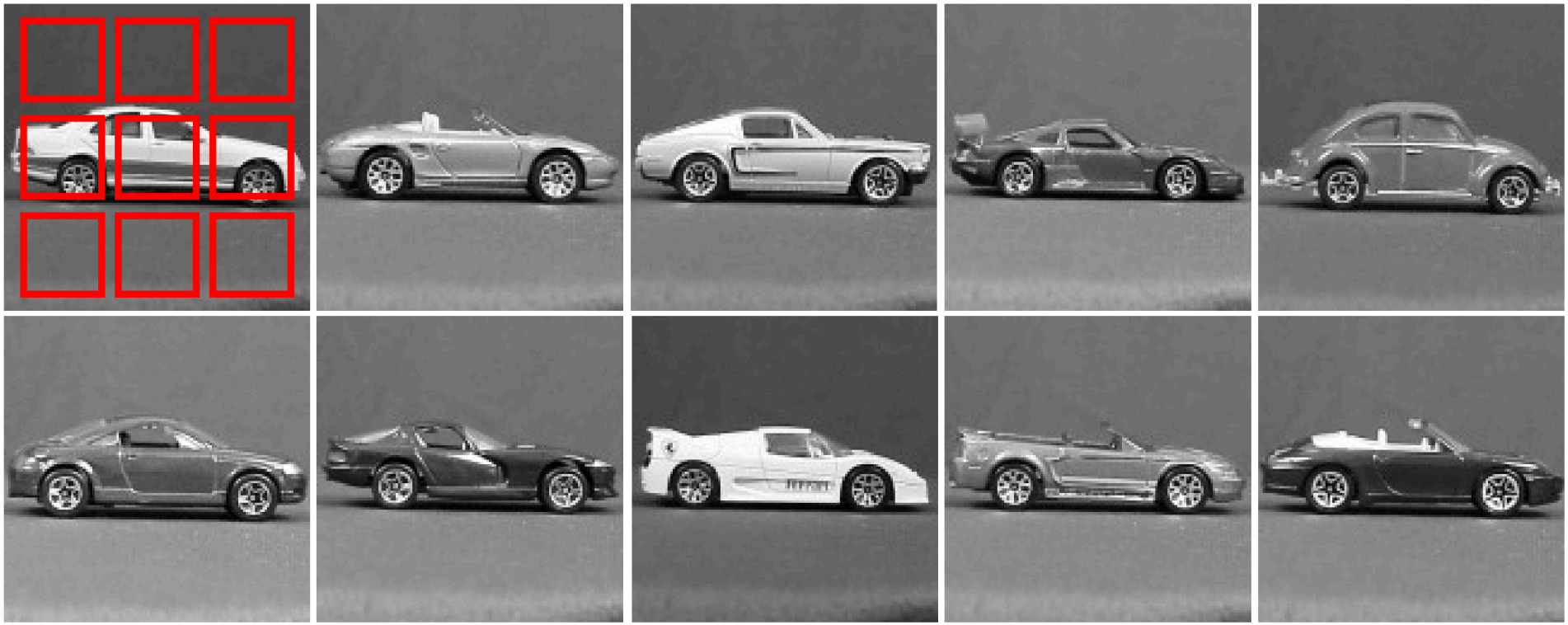


Figure: B. Liebe

Credit: David Fouhey

Category recognition with SIFT

- Build codebook of “concepts”

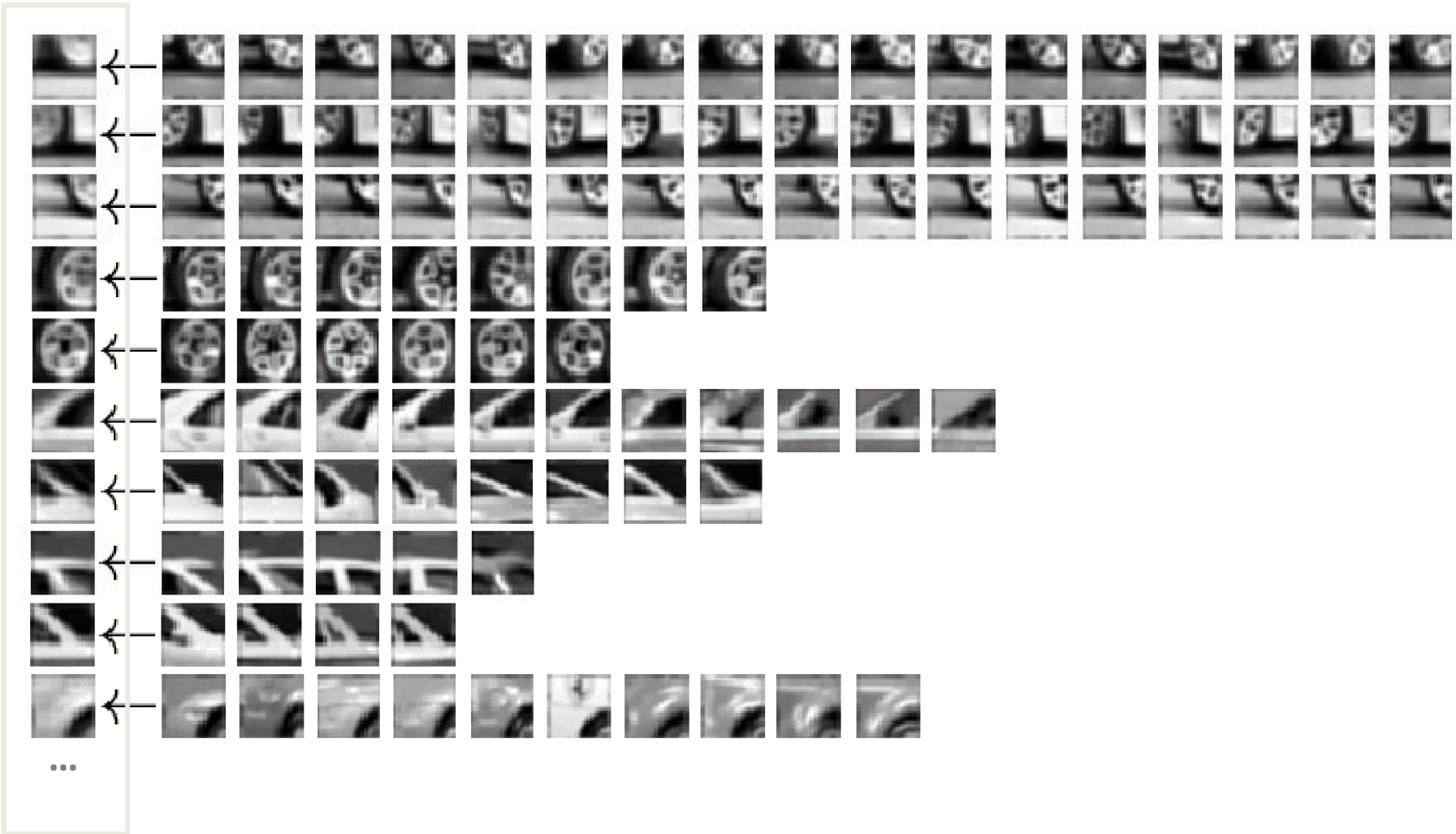
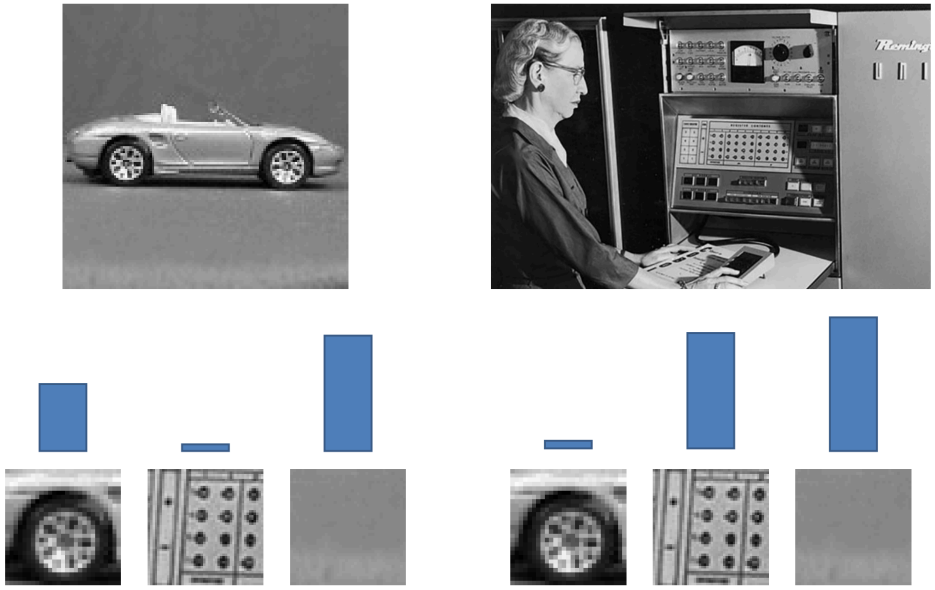


Figure: B. Liebe

Credit: David Fouhey

Category recognition with SIFT

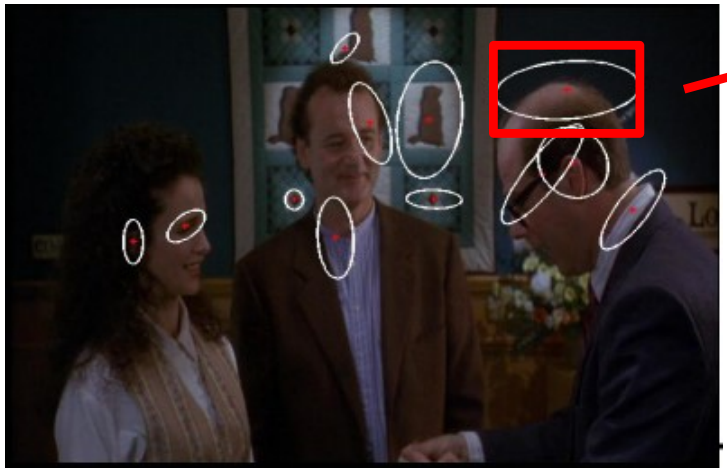
- Represent image as histogram of concepts



- Big pro: can have a different number of features extracted per image, but the resulting representation is still the same size

Credit: David Fouhey

Bag of words model



1. Resize patch to a fixed size (e.g. 32x32 pixels)
2. Extract SIFT/HOG on the patch (get 144 numbers)

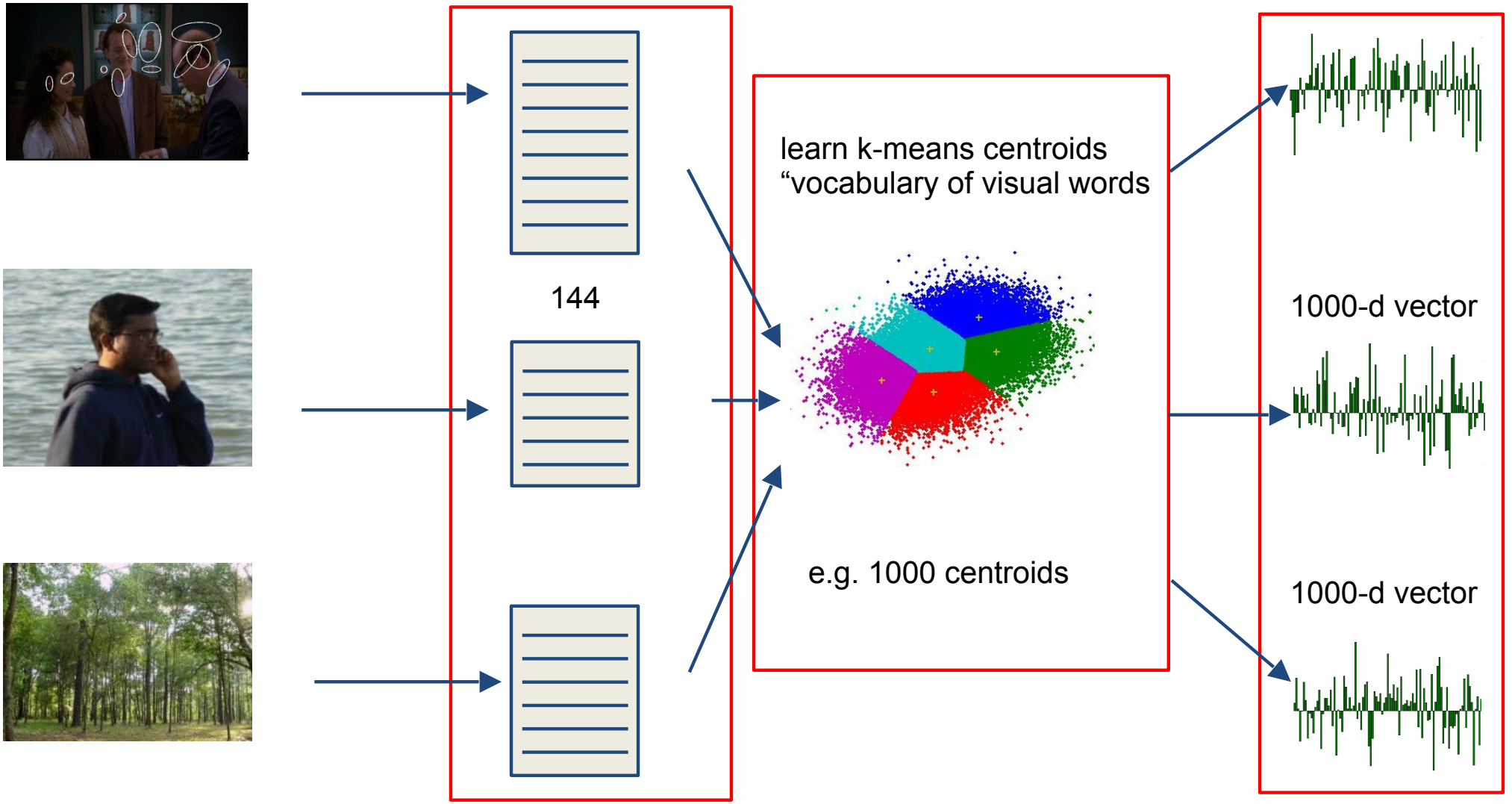
repeat for each detected feature



gives a matrix of size [number_of_features x 144]

Source: Andrej Karpathy & Fei-Fei Li

Bag of words model



Source: Andrej Karpathy & Fei-Fei Li

K-means clustering

Revisiting k-means

Most well-known and popular clustering algorithm:

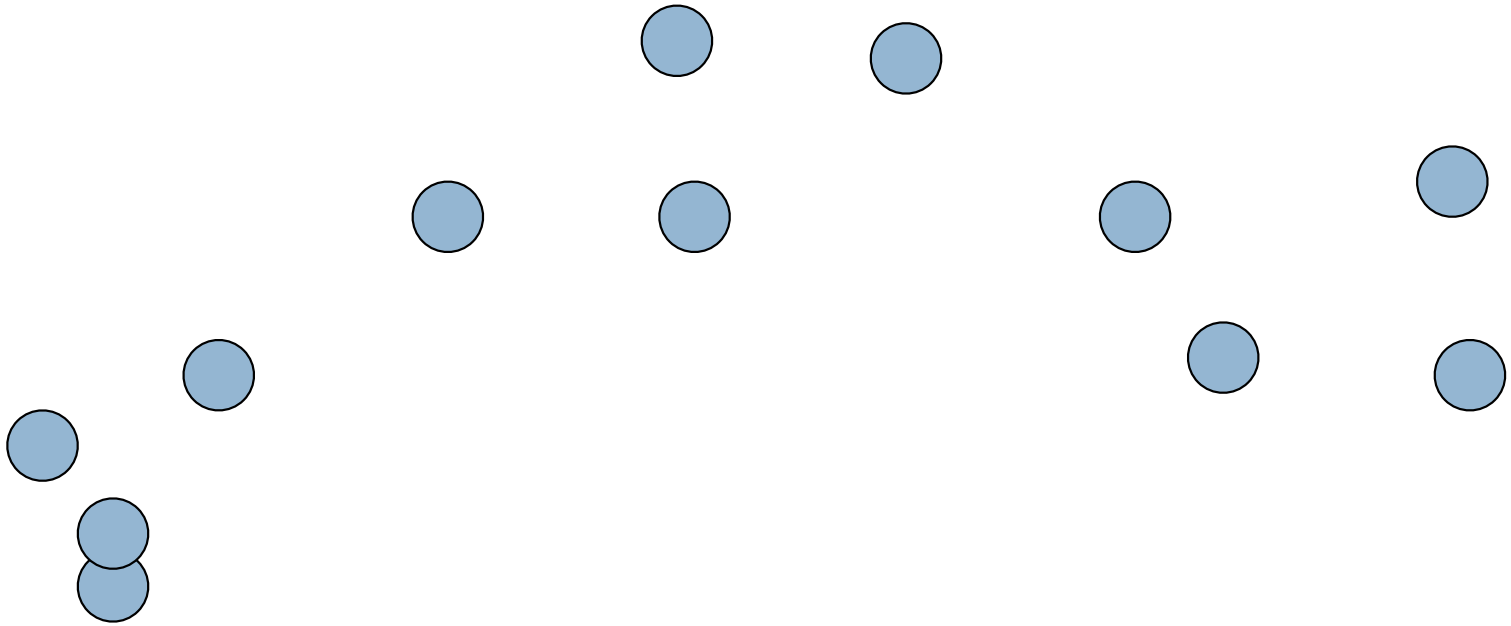
Start with some initial cluster centers

Iterate:

- Assign/cluster each example to closest center
- Recalculate centers as the mean of the points in a cluster

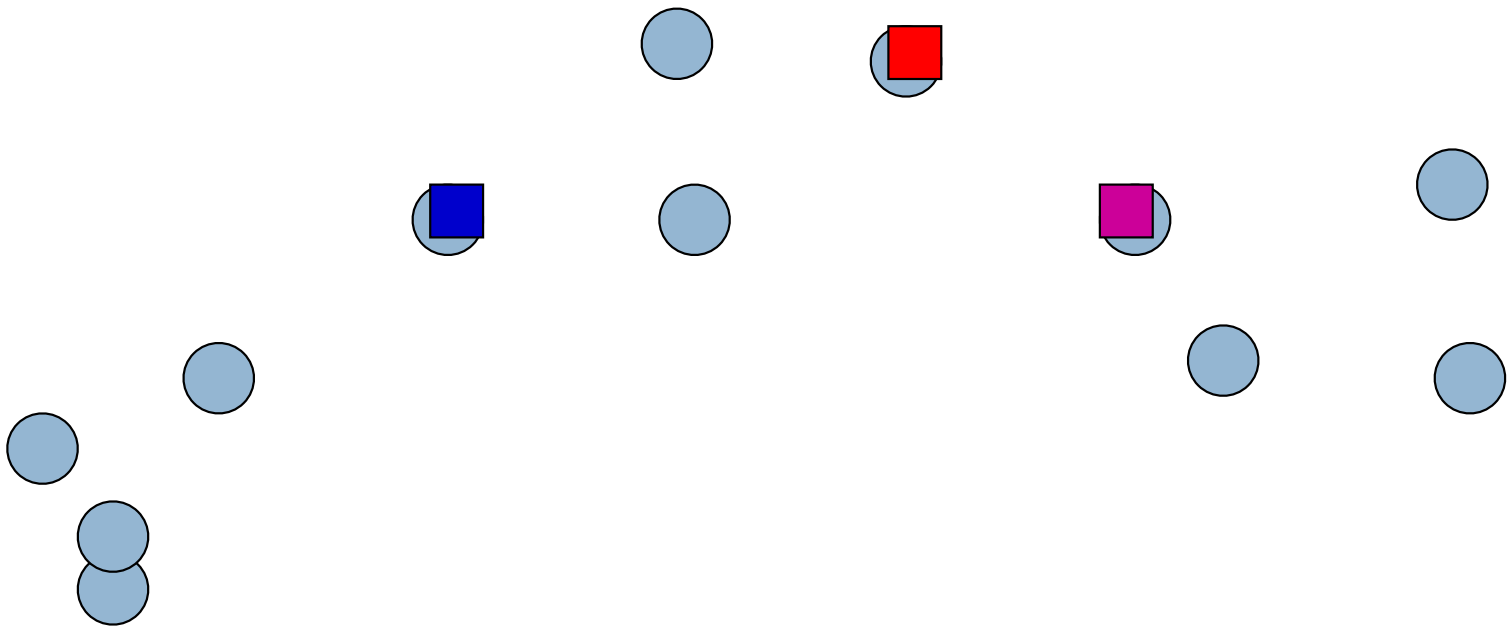
Credit: David Kauchak

K-means: an example



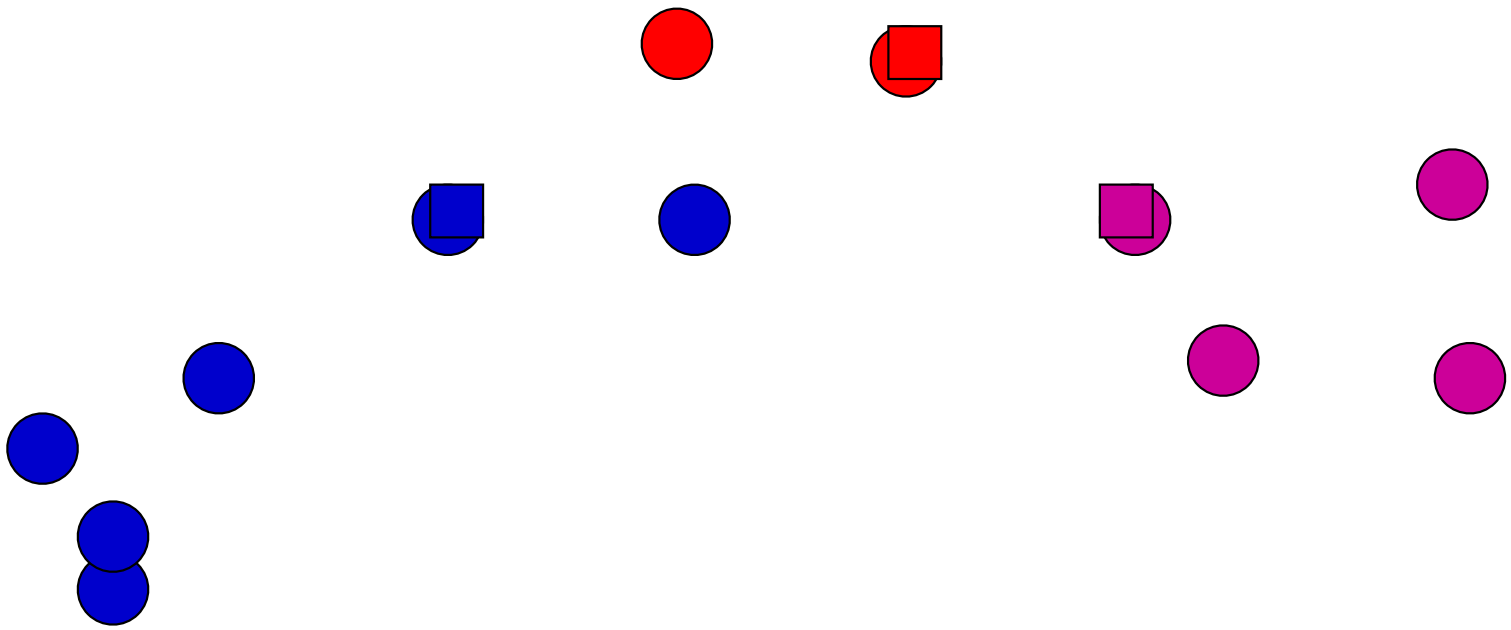
Credit: David Kauchak

K-means: Initialize centers randomly



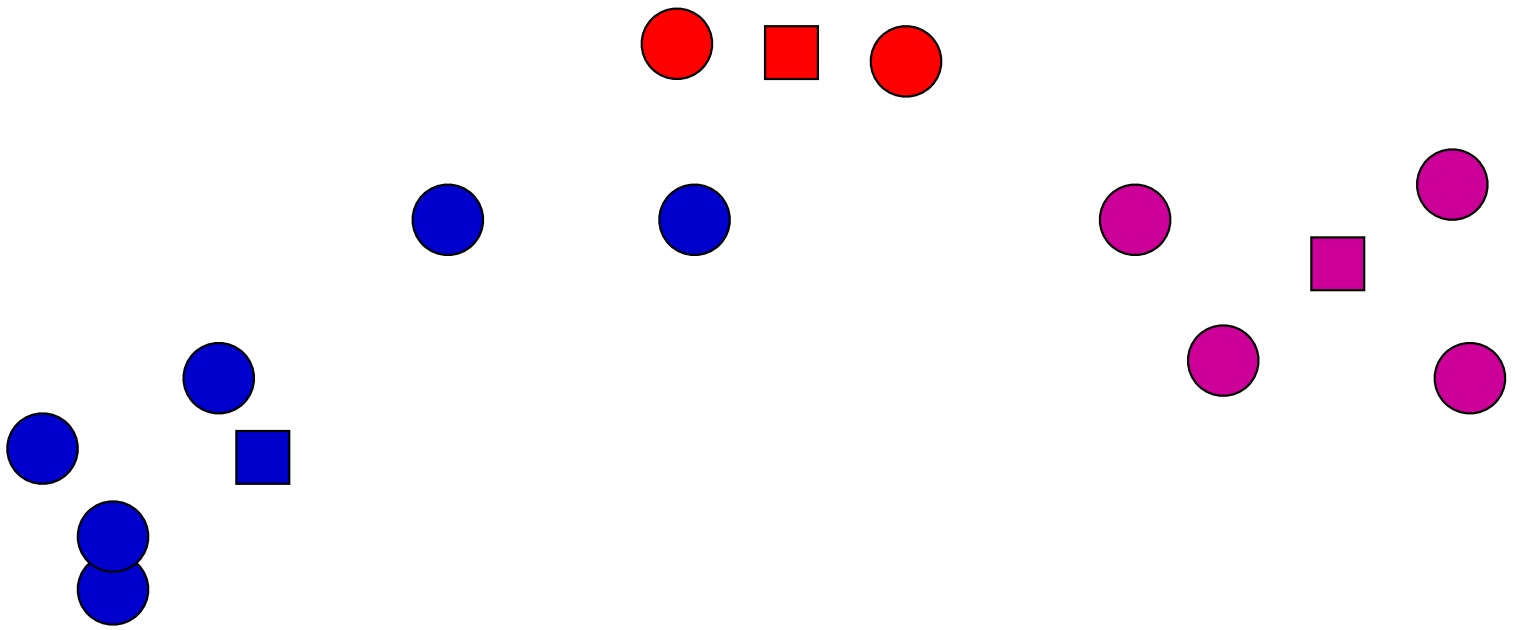
Credit: David Kauchak

K-means: assign points to nearest center



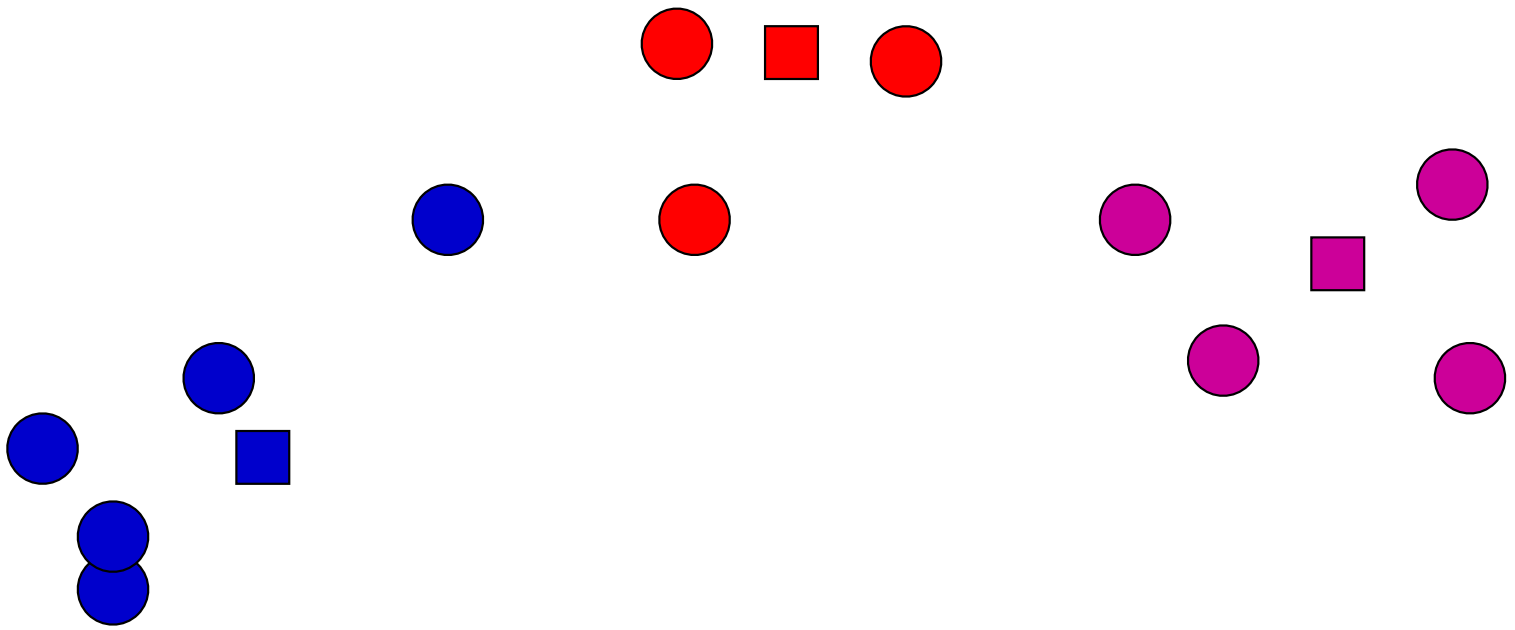
Credit: David Kauchak

K-means: readjust centers



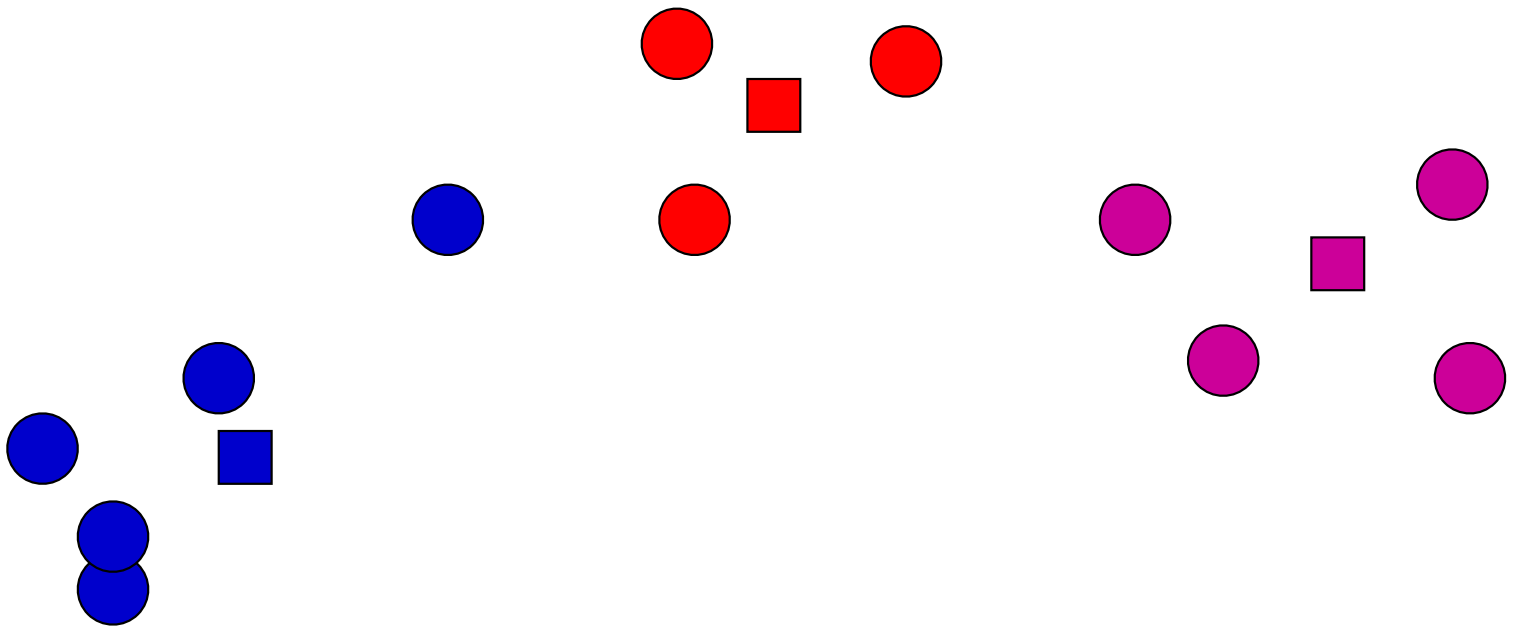
Credit: David Kauchak

K-means: assign points to nearest center



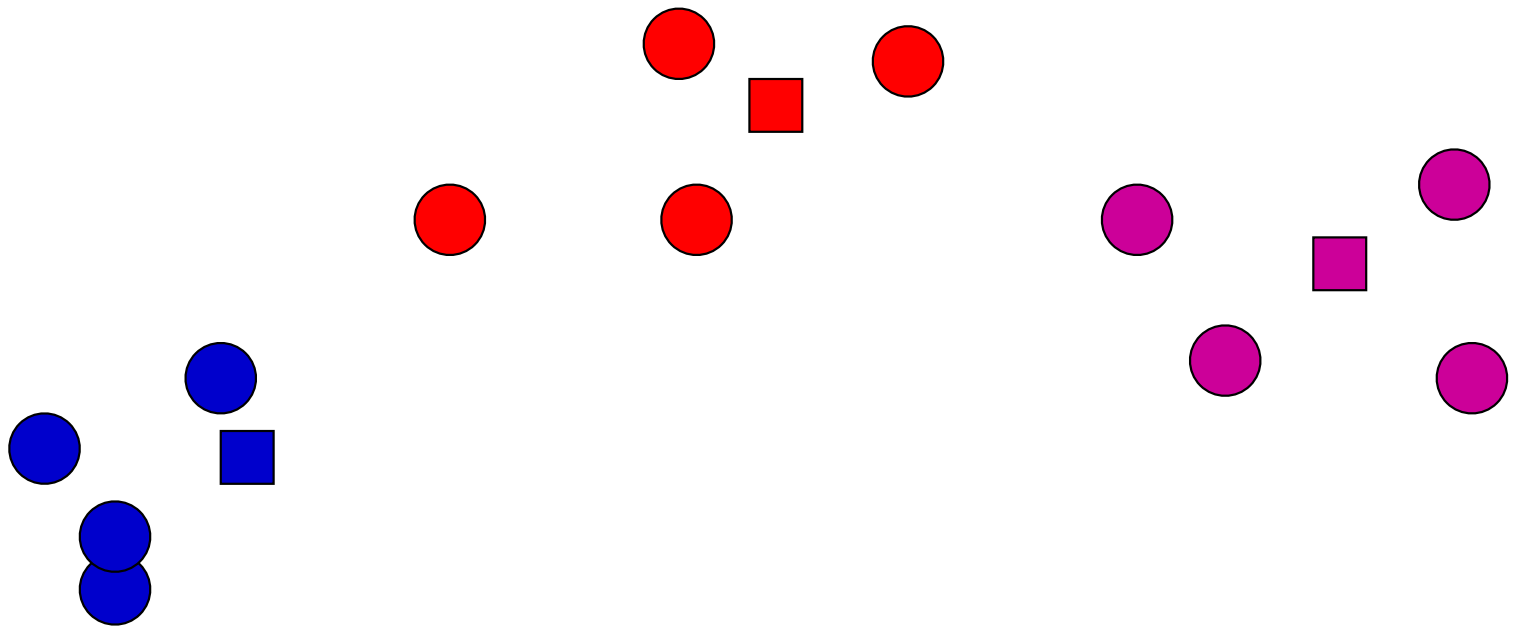
Credit: David Kauchak

K-means: readjust centers



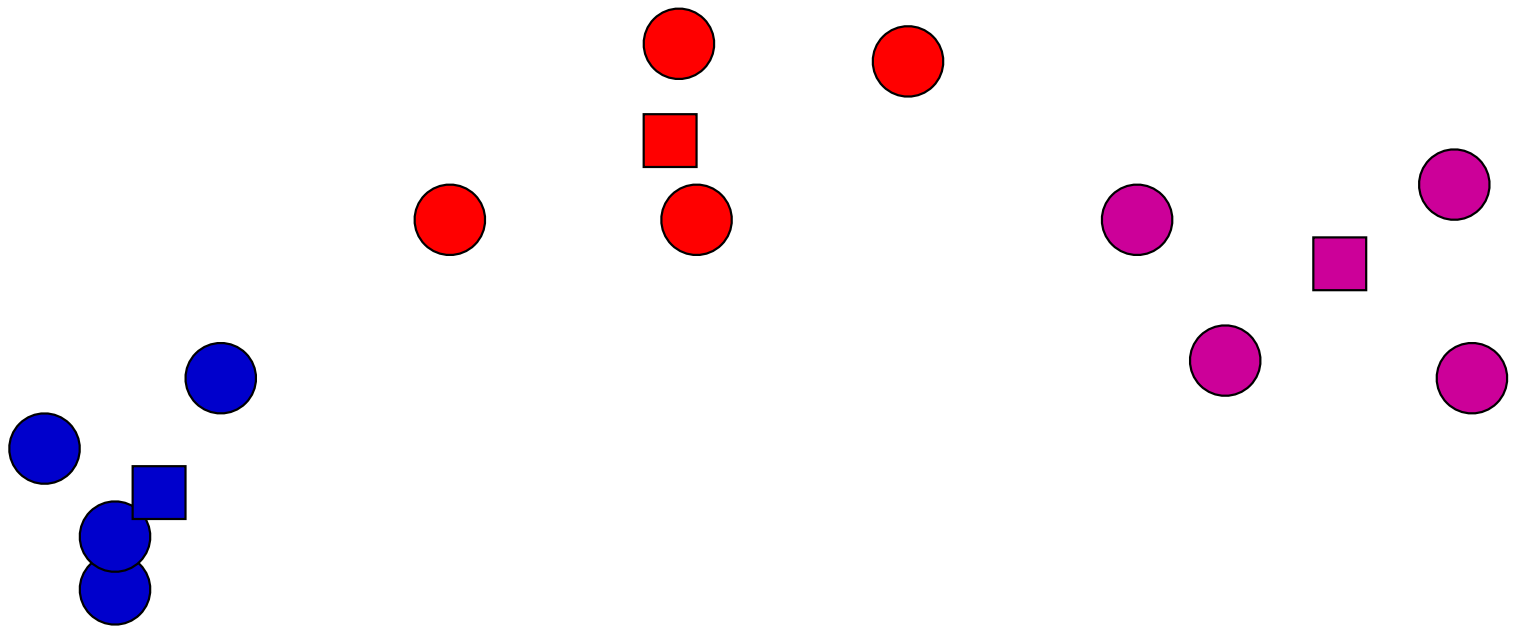
Credit: David Kauchak

K-means: assign points to nearest center



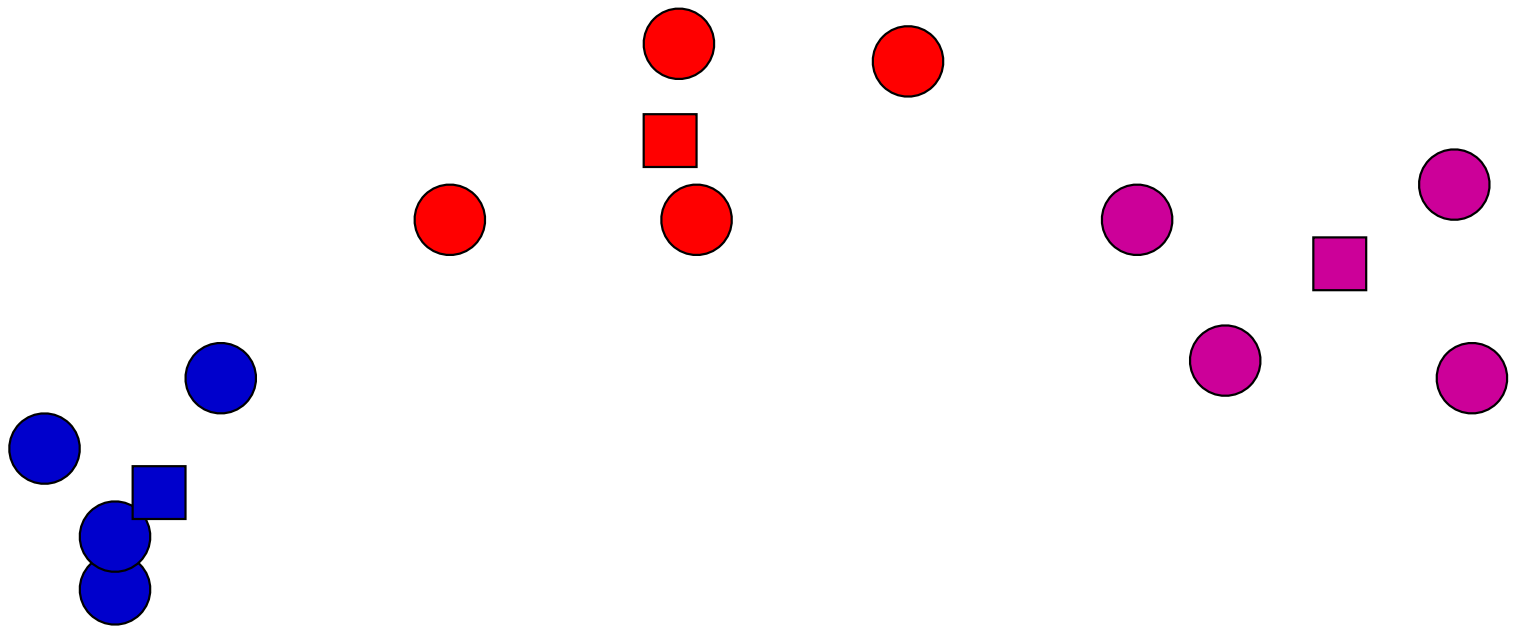
Credit: David Kauchak

K-means: readjust centers



Credit: David Kauchak

K-means: assign points to nearest center



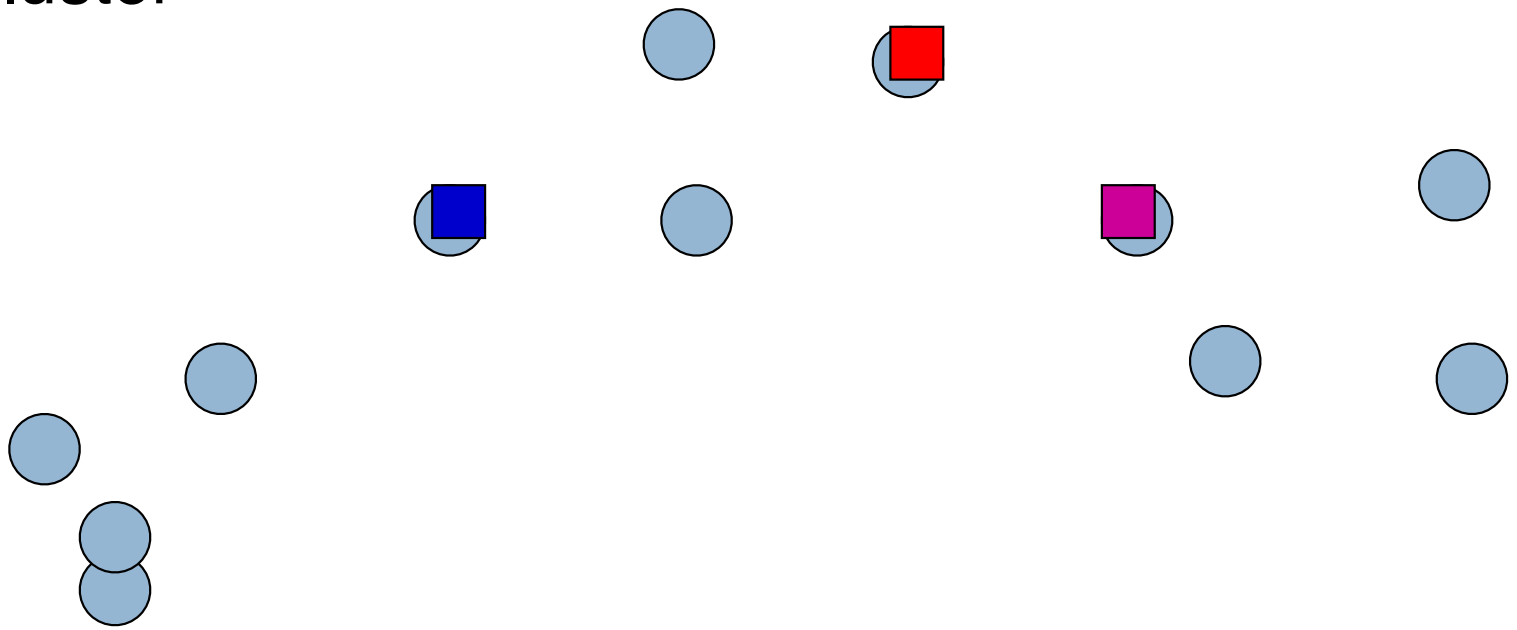
No changes: Done

Credit: David Kauchak

K-means

Iterate:

- **Assign/cluster each example to closest center**
- Recalculate centers as the mean of the points in a cluster



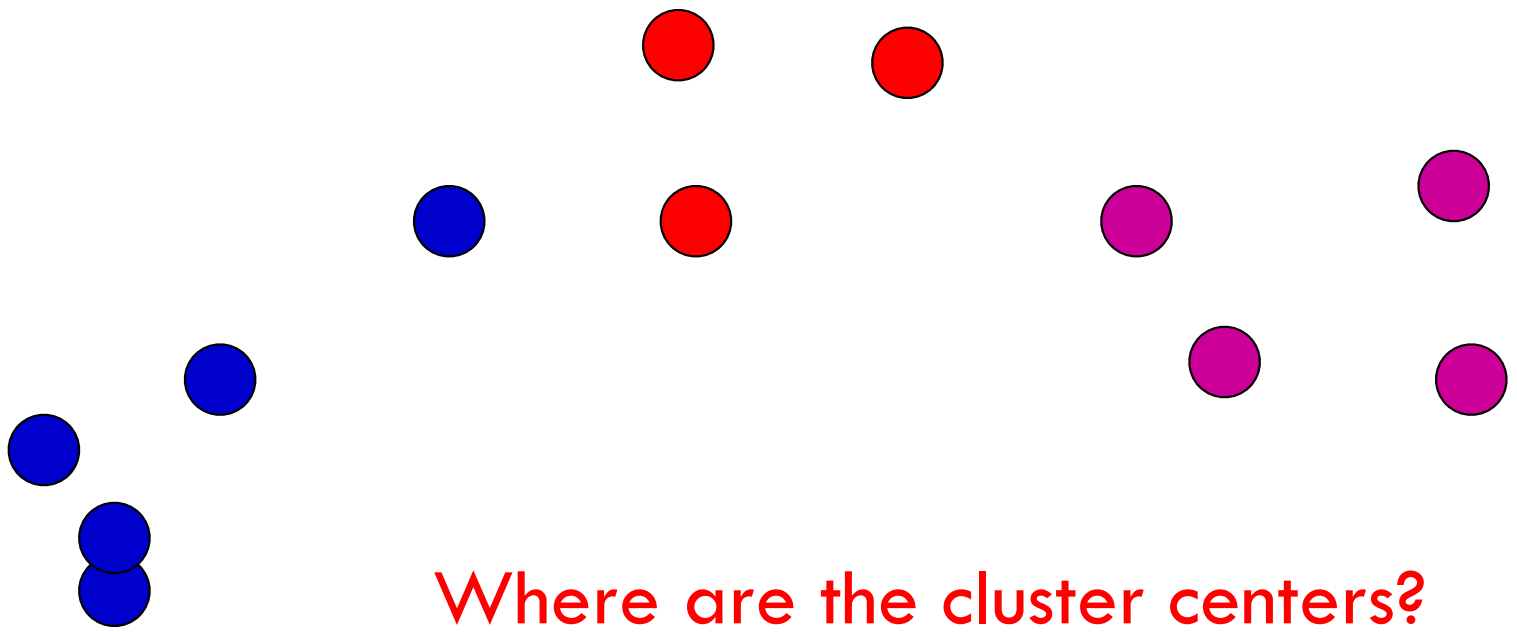
How do we do this?

Credit: David Kauchak

K-means

Iterate:

- Assign/cluster each example to closest center
- Recalculate centers as the mean of the points in a cluster



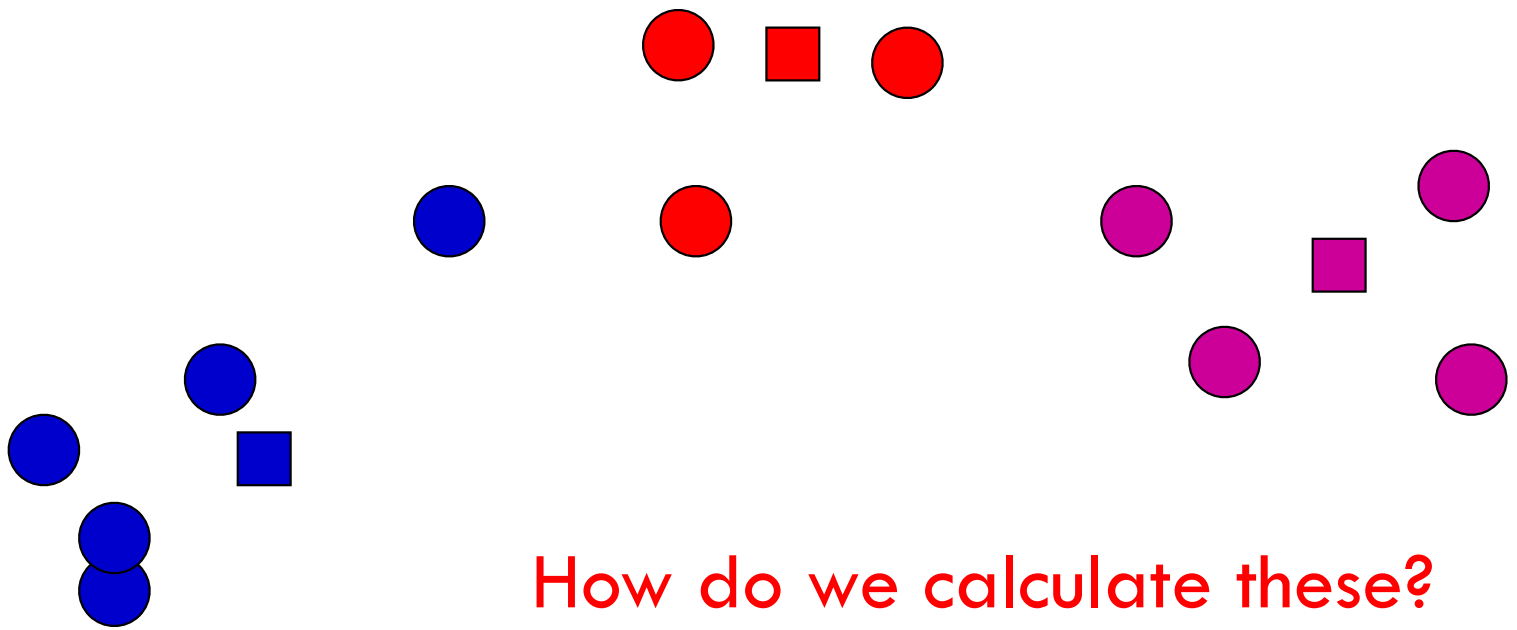
Where are the cluster centers?

Credit: David Kauchak

K-means

Iterate:

- Assign/cluster each example to closest center
- Recalculate centers as the mean of the points in a cluster



How do we calculate these?

Credit: David Kauchak

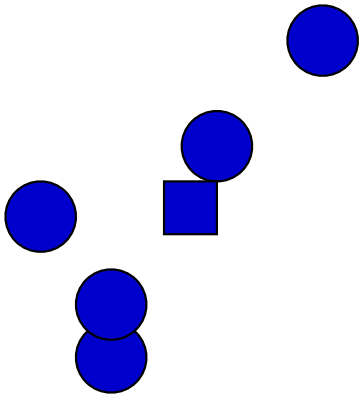
K-means

Iterate:

- Assign/cluster each example to closest center
- Recalculate centers as the mean of the points in a cluster

Mean of the points in the cluster:

$$\mu(C) = \frac{1}{|C|} \sum_{x \in C} x$$



Credit: David Kauchak

K-means loss function

K-means tries to minimize what is called the "k-means" loss function:

$$loss = \sum_{i=1}^{n} d(x_i, \mu_k)^2 \quad \text{where } \mu_k \text{ is cluster center for } x_i$$

that is, the sum of the squared distances from each point to the associated cluster center

Credit: David Kauchak

Minimizing k-means loss

Iterate:

1. Assign/cluster each example to closest center
2. Recalculate centers as the mean of the points in a cluster

---

$$loss = \sum_{i=1}^{n} d(x_i, \mu_k)^2 \quad \text{where } \mu_k \text{ is cluster center for } x_i$$

Does each step of k-means move towards reducing this loss function (or at least not increasing)?

Credit: David Kauchak

Minimizing k-means loss

Iterate:

1. Assign/cluster each example to closest center
2. Recalculate centers as the mean of the points in a cluster

---

$$loss = \sum_{i=1}^{n} d(x_i, \mu_k)^2 \quad \text{where } \mu_k \text{ is cluster center for } x_i$$

This isn't quite a complete proof/argument, but:

1. Any other assignment would end up in a larger loss

1. The mean of a set of values minimizes the squared error

Credit: David Kauchak

Minimizing k-means loss

Iterate:

1. Assign/cluster each example to closest center
2. Recalculate centers as the mean of the points in a cluster

---

$$loss = \sum_{i=1}^{n} d(x_i, \mu_k)^2 \quad \text{where } \mu_k \text{ is cluster center for } x_i$$

Does this mean that k-means will always find the minimum loss/clustering?

Credit: David Kauchak

Minimizing k-means loss

Iterate:

1. Assign/cluster each example to closest center
2. Recalculate centers as the mean of the points in a cluster

---

$$loss = \sum_{i=1}^{n} d(x_i, \mu_k)^2 \quad \text{where } \mu_k \text{ is cluster center for } x_i$$

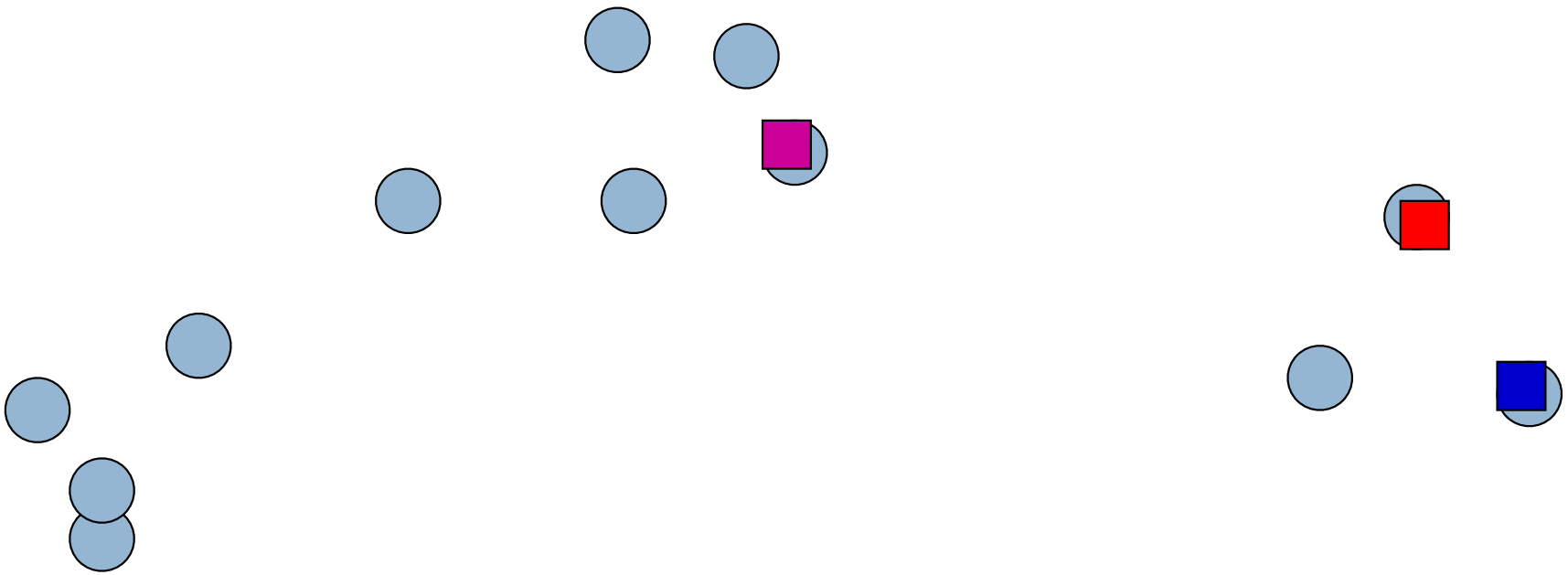
NO! It will find *a minimum*.

Unfortunately, the k-means loss function is generally not convex and for most problems has many, many minima

We're only guaranteed to find one of them

Credit: David Kauchak

K-means: Initialize centers randomly

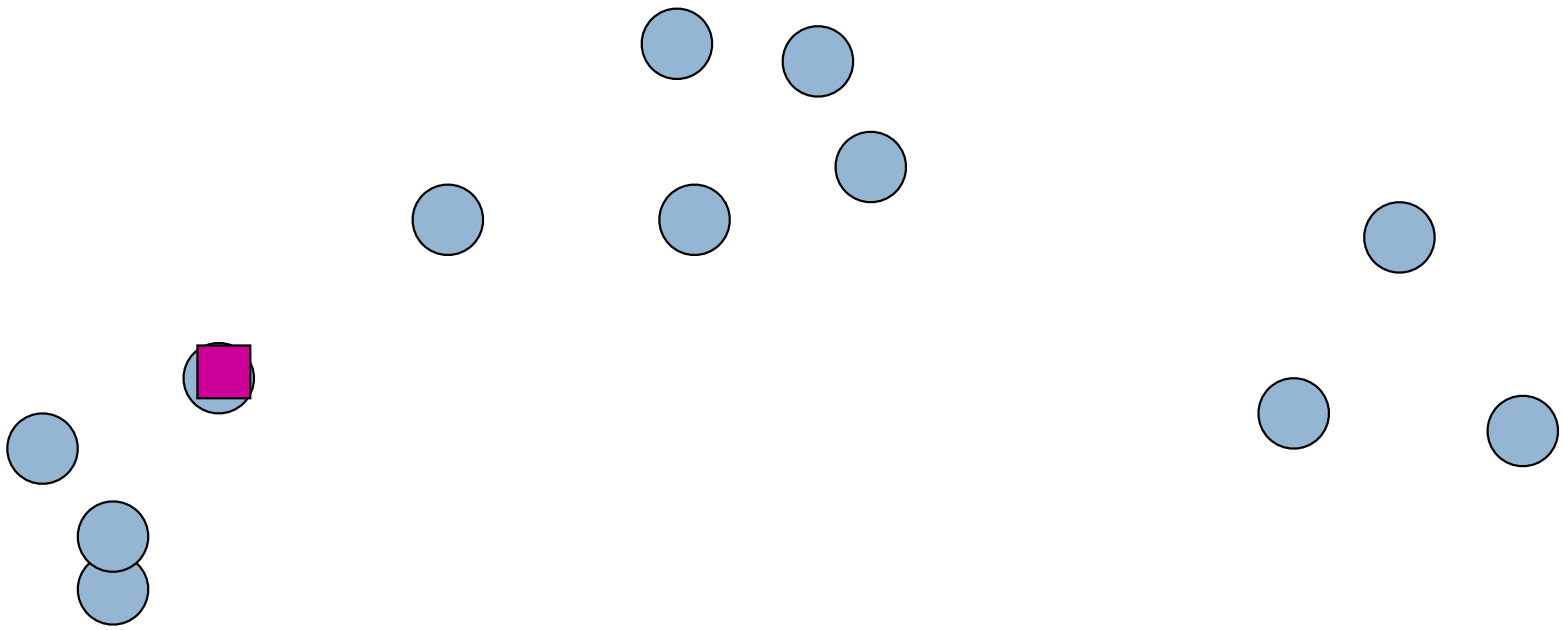


What would happen here?

Seed selection ideas?

Credit: David Kauchak

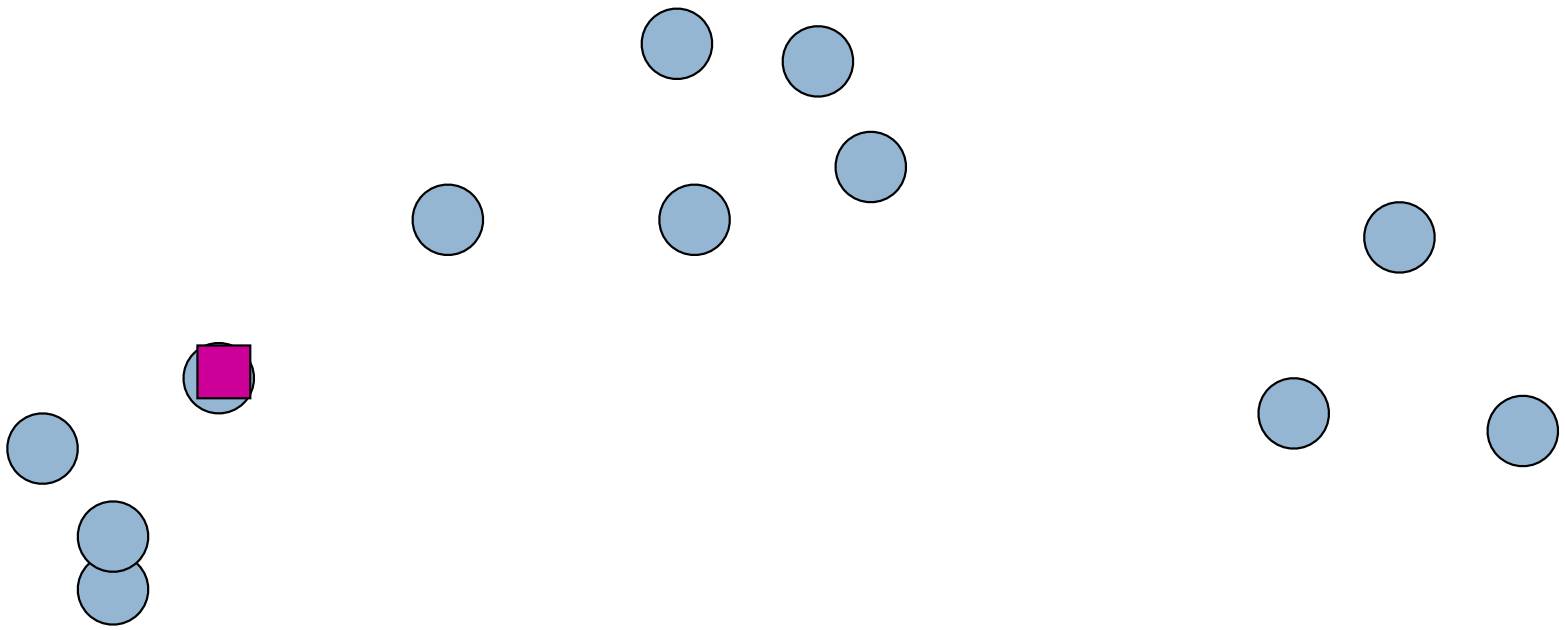
K-means: Initialize furthest from centers



Pick a random point for the first center

Credit: David Kauchak

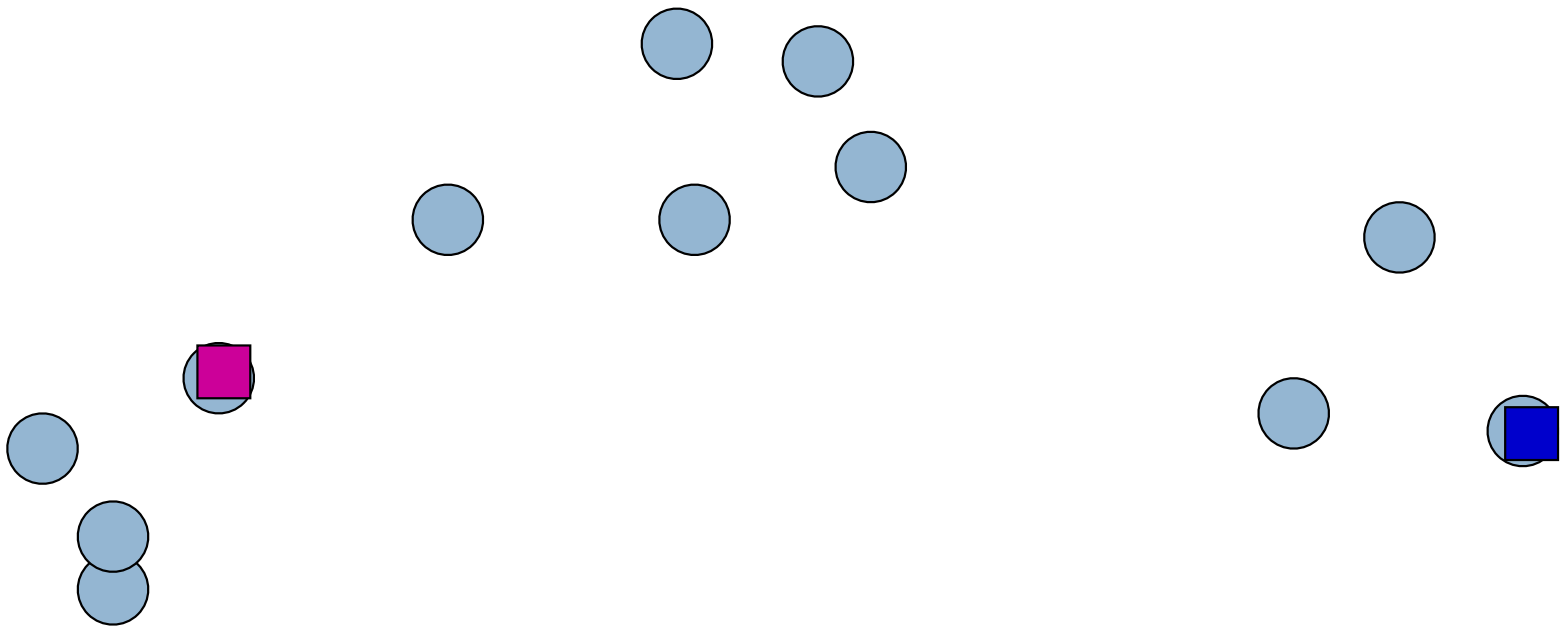
K-means: Initialize furthest from centers



What point will be chosen next?

Credit: David Kauchak

K-means: Initialize furthest from centers

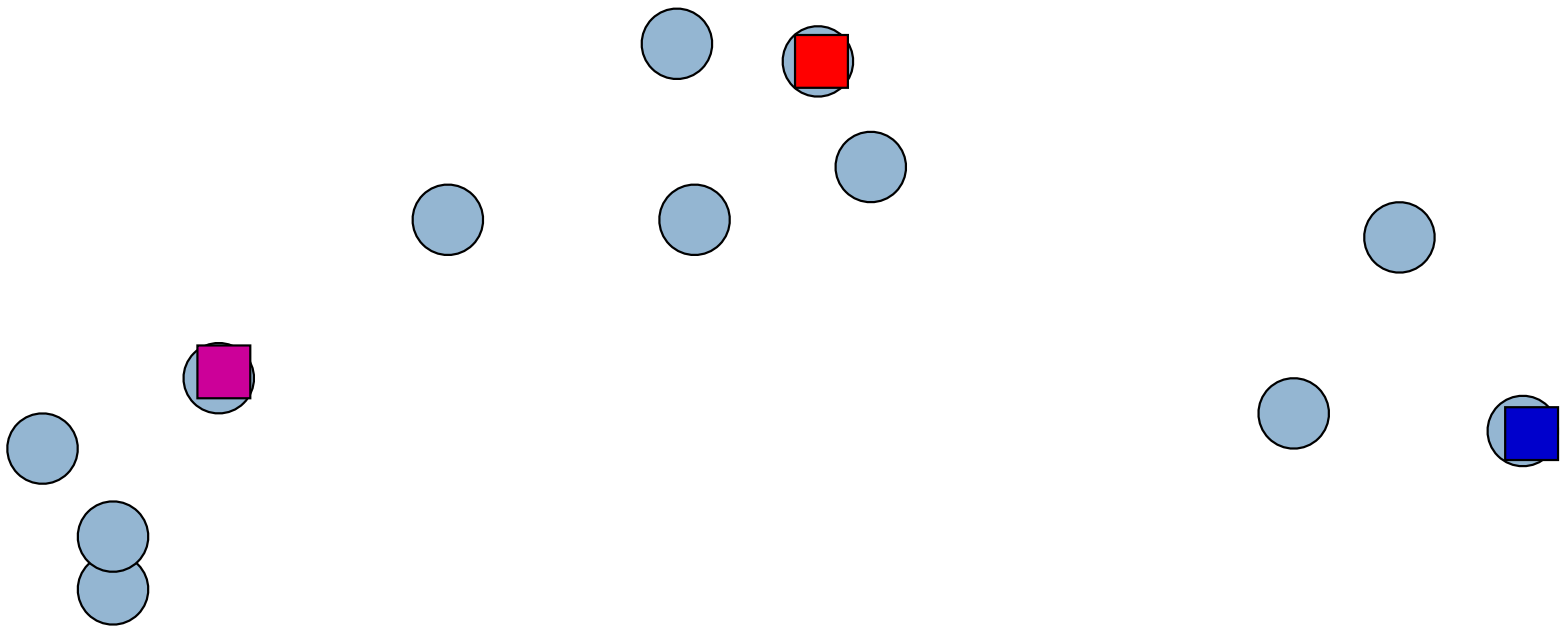


Furthest point from center

What point will be chosen next?

Credit: David Kauchak

K-means: Initialize furthest from centers

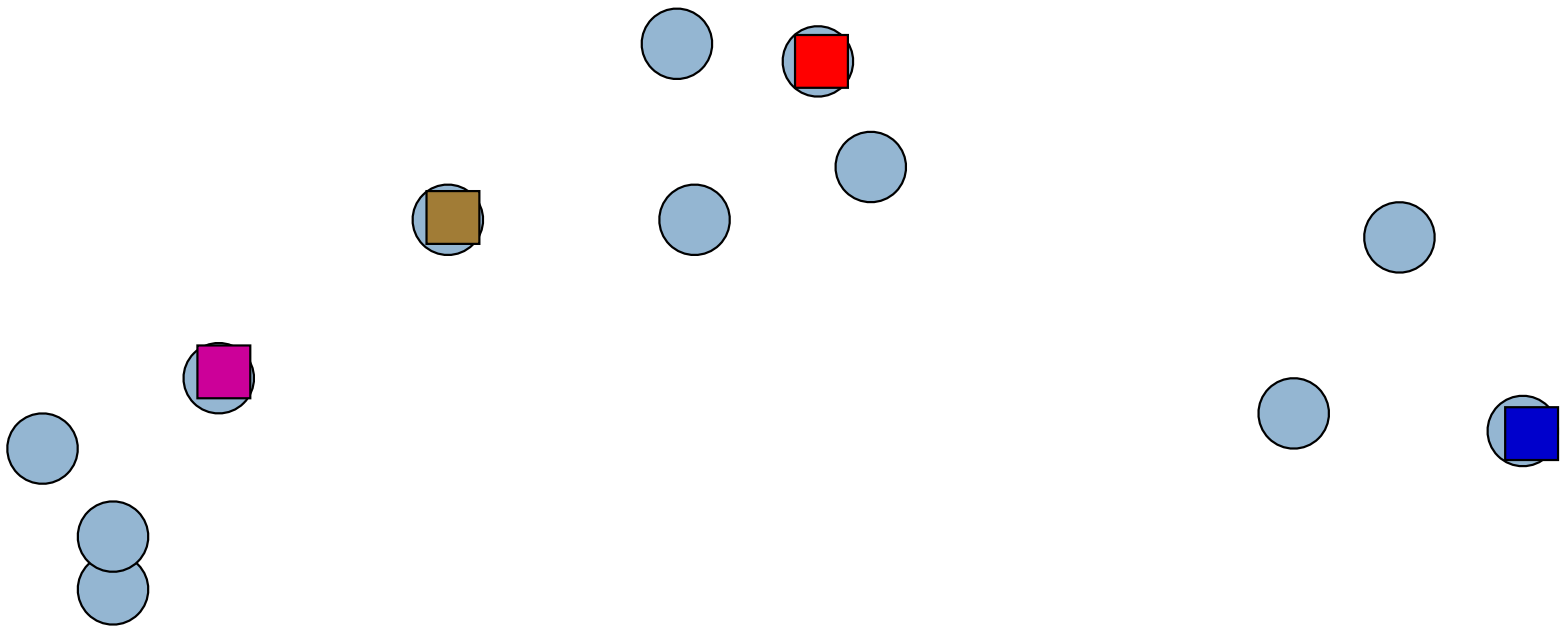


Furthest point from center

What point will be chosen next?

Credit: David Kauchak

K-means: Initialize furthest from centers

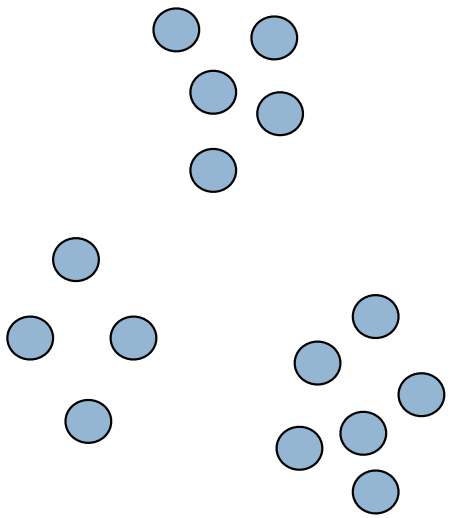


Furthest point from center

Any issues/concerns with this approach?

Credit: David Kauchak

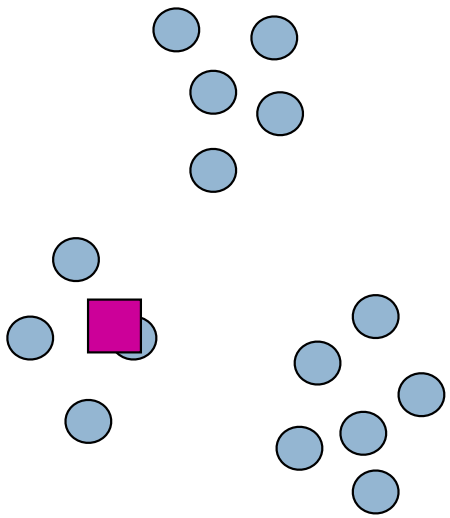
Furthest points concerns



If k = 4, which points will get chosen?

Credit: David Kauchak

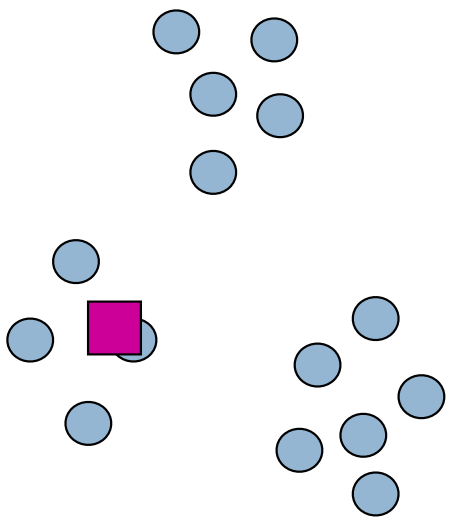
Furthest points concerns



If we do a number of trials, will we get different centers?

Credit: David Kauchak

Furthest points concerns



Doesn't deal well with outliers

Credit: David Kauchak

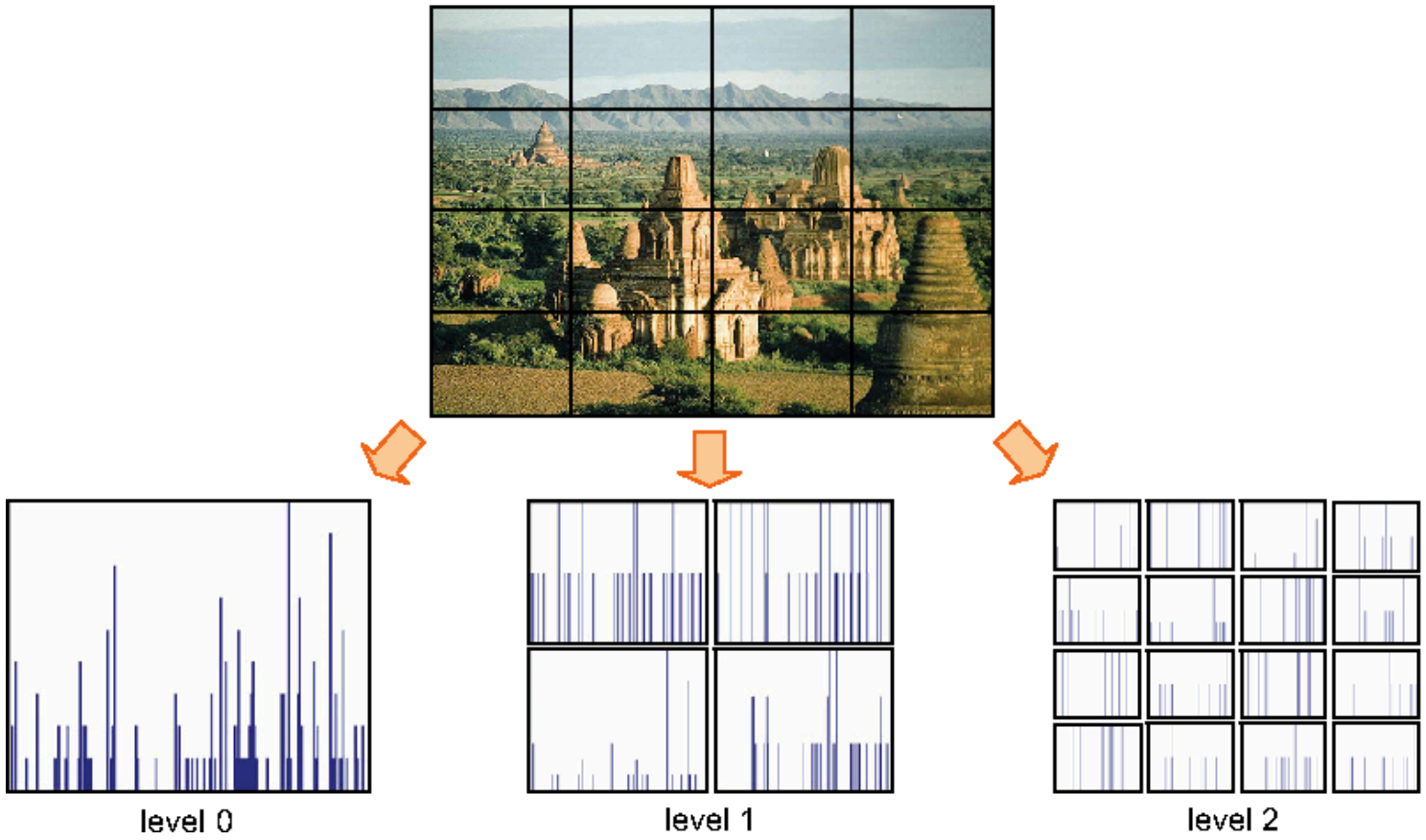
K-means

- But usually k-means works pretty well
  - Especially with large number of points and large number of centers k
- Variations: kmeans++, etc
- Alternatives: spectral clustering, hierarchical (bottom-up, agglomerative or top-down, divisive)

Coming back to image classification

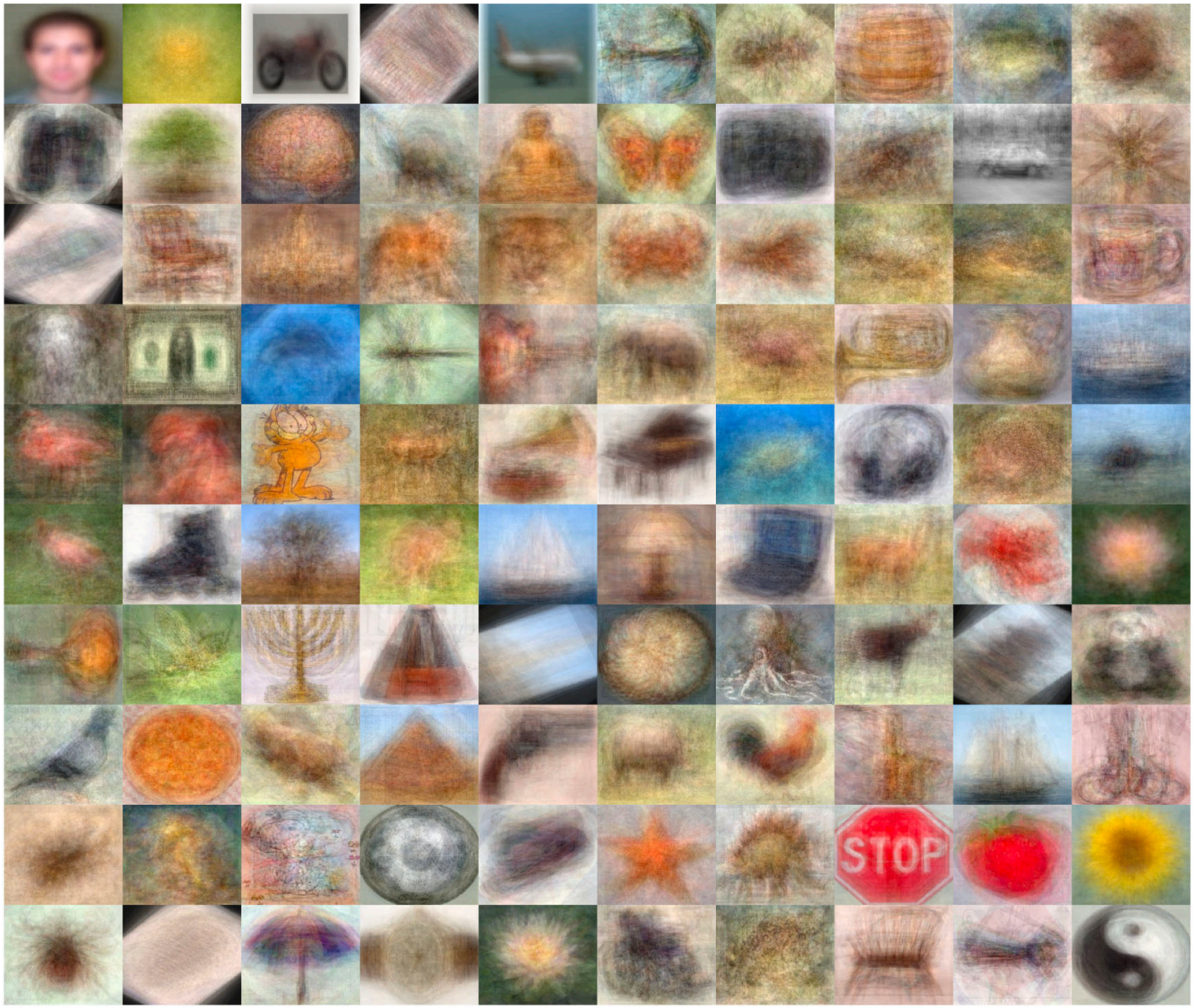
Spatial Pyramid Matching (SPM)

- Slight modification to Bag of Words:



Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories.
S. Lazebnik, C. Schmid, and J. Ponce. CVPR 2006.

Caltech 101 average image



Recognition algorithms

- Generally:
  - Feature matching & RANSAC is great for instance matching (very little variation)
  - HOG is great for recognizing shape-consistent objects (pedestrians)
  - BoW/SPM is good for fairly shape-flexible categories
- Will come back to this next time when talking about generic object detection

Evaluation

Accuracy




- Is this an umbrella or a strawberry?
- assign 1 of N class labels to an image (N = 101 in Caltech 101)

$$Accuracy = \frac{1}{NumImages} \sum_{i=1}^{NumImages} 1[correct\ on\ image\ i]$$

Good:
- clean and simple

Bad:
- What if there are multiple objects in the image?
- Sensitive to class priors

Mean per-class accuracy

$$Accuracy = \frac{1}{Nclasses} \sum_{c=1}^{Nclasses} \frac{1}{Nimages_c} \sum_{i=1}^{Nimages_c} 1[correct\ on\ image\ i]$$

- Better:
  - Less sensitive to class priors
- However:
  - What if want to consider false positives and false negatives separately? (e.g., positive vs negatives in face/pedestrian detection?)
  - Requires selecting one class per image, conflating the classifier's ability to trade off between different classes with how well each class is modeled

Evaluation of individual classes

- Consider and evaluate each object classifier independently

Model for recognizing an umbrella



Gives an "umbrellaness" score to each test image:



0.95 0.74 0.55 0.92 0.85 …

- Intuitively, this is a "good" classifier if the positive images get assigned high scores and the negative images get assigned low scores

Precision and recall

- Supposing we have a classification threshold

Classify as umbrella
if score > 0.75



0.95 0.74 0.55 0.92 0.85 …

\+ — — \+ \+

- Can evaluate using accuracy, or using precision and recall:

$$Precision = \frac{NumTruePositives}{NumPredictions} = \frac{2}{3}$$

$$Recall = \frac{NumTruePositives}{NumPositives} = \frac{2}{4}$$

(supposing there are 4 umbrellas in the test set in this example)

F-score

- Can reconcile precision and recall using F1-score (harmonic mean):

$$Fscore = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}} = \frac{2}{\frac{1}{\frac{2}{3}} + \frac{1}{\frac{2}{4}}} = \frac{4}{7} = 0.57$$

- But still begs the question of how to set the classification threshold, and puts the load on the model
  - Even though arguably we care about evaluating the **decision boundary**, not the threshold

Varying the threshold

- Consider **all** possible thresholds:

$$Pr = \frac{NumTruePositives}{NumPredictions}$$

$$R = \frac{NumTruePositives}{NumPositives}$$



Model scores:	0.95	0.74	0.55	0.92	0.85	…	
Threshold of 0.96	−	−	−	−	−		Pr = 1; R = 0
Threshold of 0.93	+	−	−	−	−		Pr = 1; R = 0.25
Threshold of 0.86	+	−	−	+	−		Pr = 1; R = 0.5
Threshold of 0.75	+	−	−	+	+		Pr = 0.67; R = 0.5
Threshold of 0.6	+	+	−	+	+		Pr = 0.5; R = 0.5
Threshold of 0.5	+	+	+	+	+		Pr = 0.6; R = 0.75
Threshold of 0							Pr = 4/N; R = 1

(supposing there are 4 umbrellas in the test set in this example)

Varying the threshold

- Consider **all** possible thresholds:

$$Pr = \frac{NumTruePositives}{NumPredictions}$$

$$R = \frac{NumTruePositives}{NumPositives}$$



Model scores: 0.95 0.74 0.55 0.92 0.85 …

Threshold of 0.96 Pr = 1; R = 0

Threshold of 0.93 Pr = 1; R = 0.25

Threshold of 0.86 Pr = 1; R = 0.5

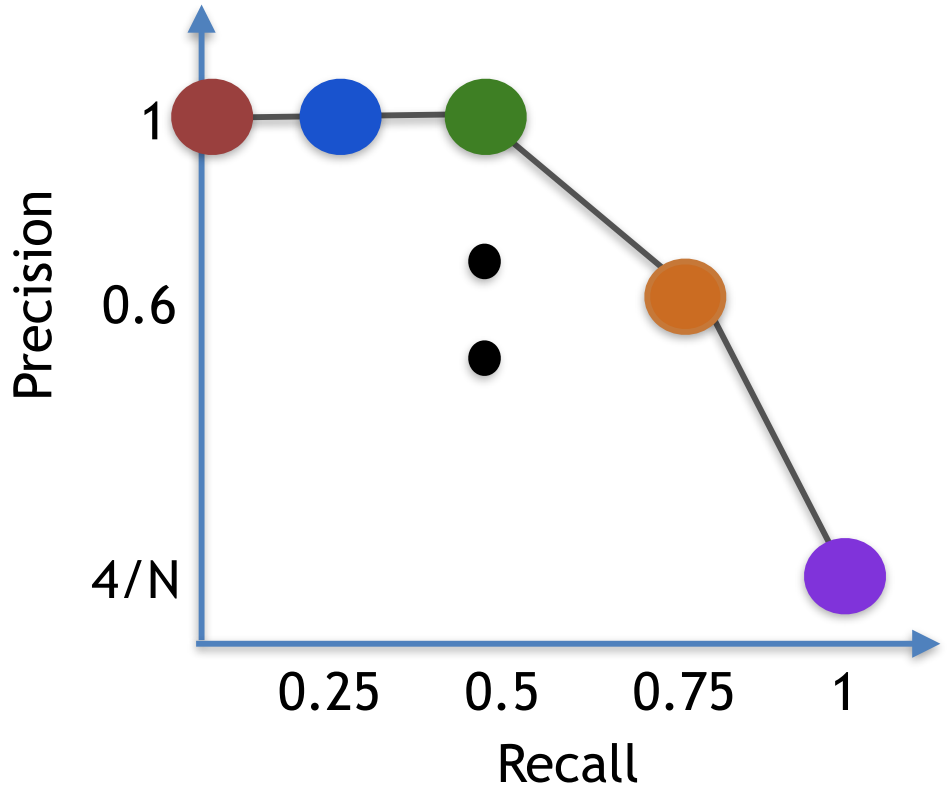
Threshold of 0.75 Pr = 0.67; R = 0.5

Threshold of 0.6 Pr = 0.5; R = 0.5

Threshold of 0.5 Pr = 0.6; R = 0.75

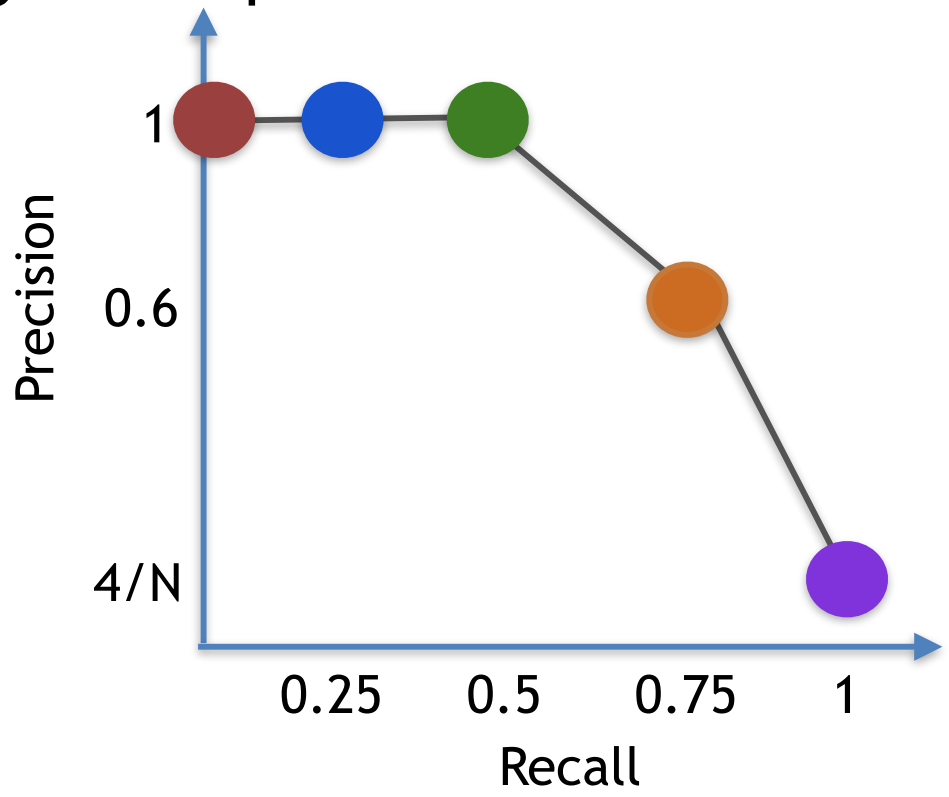
Threshold of 0 Pr = 4/N; R = 1

(supposing there are 4 umbrellas in the test set in this example)



Average Precision

- Area under this curve is a direct measure of the quality of the classifier
  - "Average Precision"
- Can compute the area using the trapezoid rule
- This is a measure of quality of a single class model
  - Mean Average Precision (mAP) is used to evaluate across all classes



Summary: key concepts

1. Bag of Words model / Spatial Pyramid Matching
2. K-means clustering
3. Mean Average Precision

Next class: object detection

