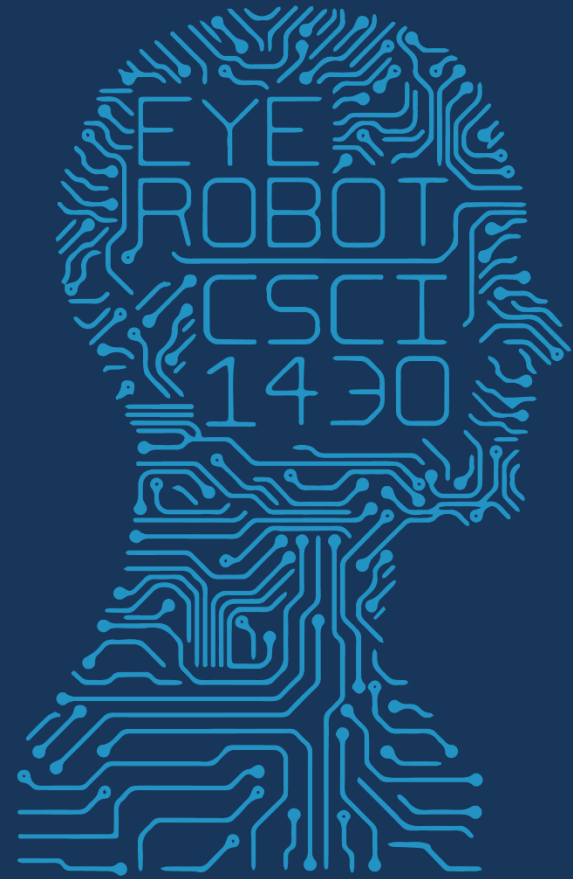


1950

FUTURE VISION



2017 MWF 1PM 368

COMPUTER VISION



Photo: CMU Machine Learning Department Protests G20

Slides: James Hays, Isabelle Guyon, Erik Sudderth, Mark Johnson, Derek Hoiem

Our approach

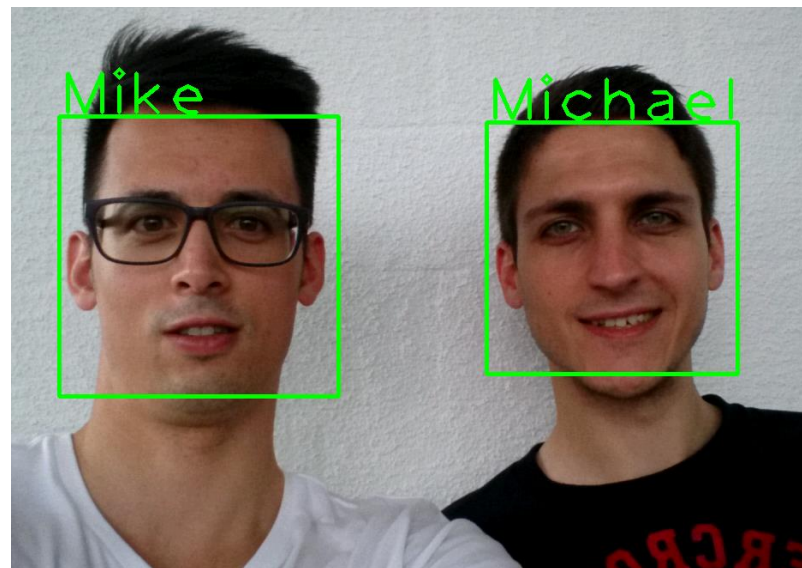
- We will look at ML as a tool. We will not detail the underpinnings of each learning method.
- Please take a machine learning course if you want to know more!

Machine Learning

- Learn from and make predictions on data.
- Arguably the greatest export from computing to other scientific fields.
- Statisticians might disagree with CompScis on the true origins...

ML for Computer Vision

- Face Recognition
- Object Classification
- Scene Segmentation

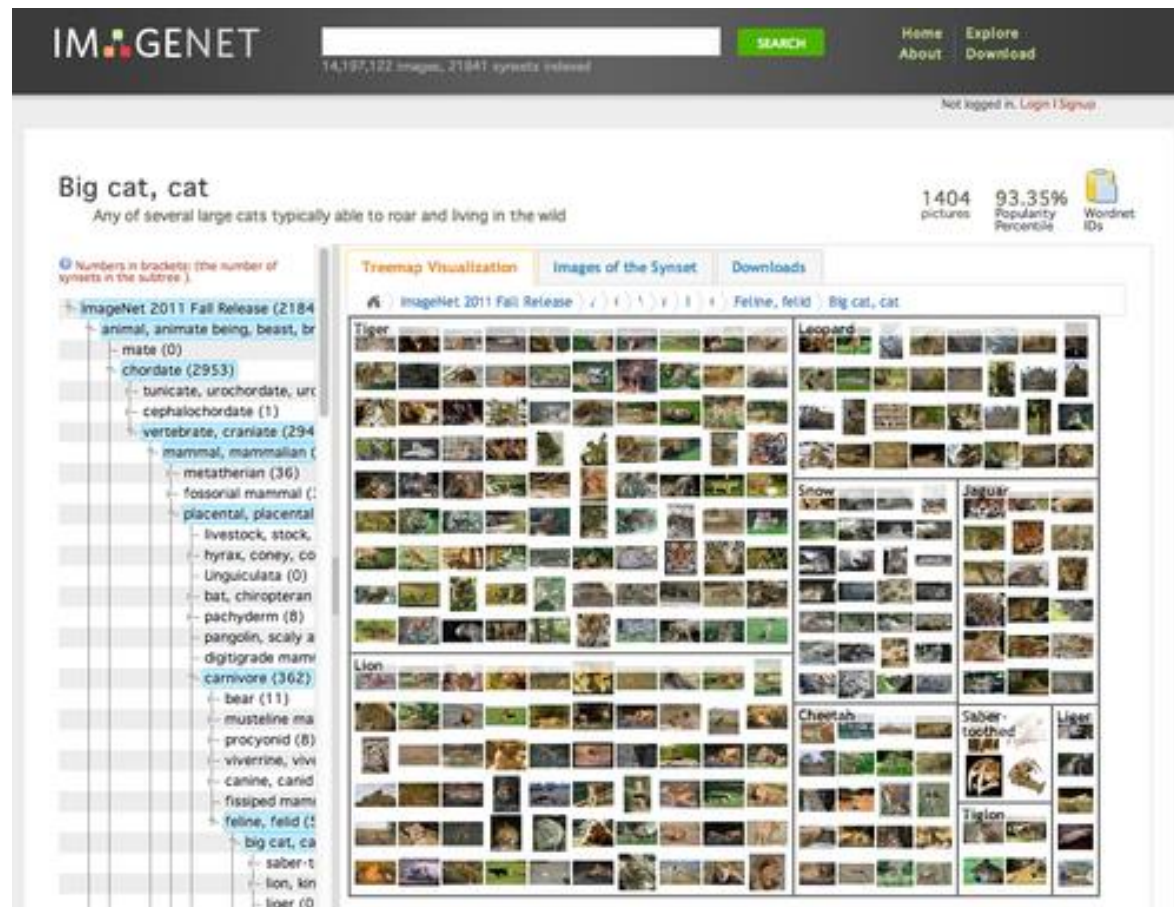


Data, data, data!

- Norvig – “The Unreasonable Effectiveness of Data” (IEEE Intelligent Systems, 2009)
 - “... invariably, simple models and a lot of data trump more elaborate models based on less data”

ImageNet

- Images for each category of WordNet
- 1000 classes
- 1.2mil images
- 100k test
- Top 5 error



IMGENET





mite



container ship



motor scooter



leopard

	mite
	black widow
	cockroach
	tick
	starfish

	container ship
	lifeboat
	amphibian
	fireboat
	drilling platform

	motor scooter
	go-kart
	moped
	bumper car
	golfcart

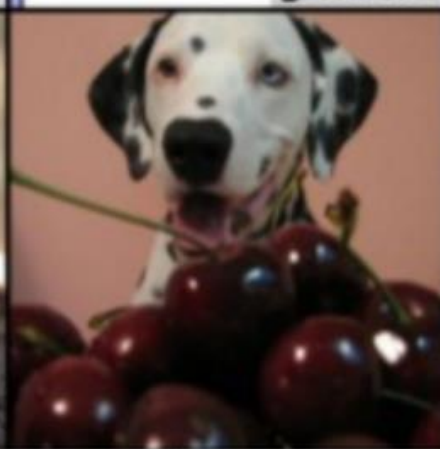
	leopard
	jaguar
	cheetah
	snow leopard
	Egyptian cat



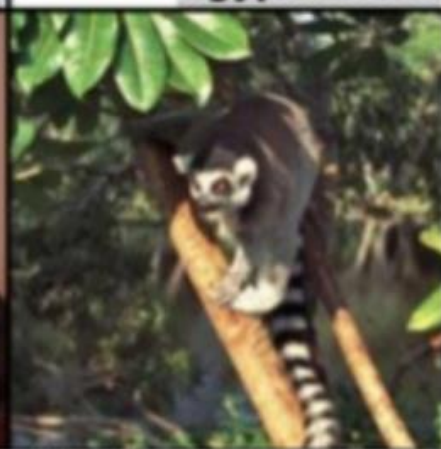
grille



mushroom



cherry



Madagascar cat

	convertible
	grille
	pickup
	beach wagon
	fire engine

	agaric
	mushroom
	jelly fungus
	gill fungus
	dead-man's-fingers

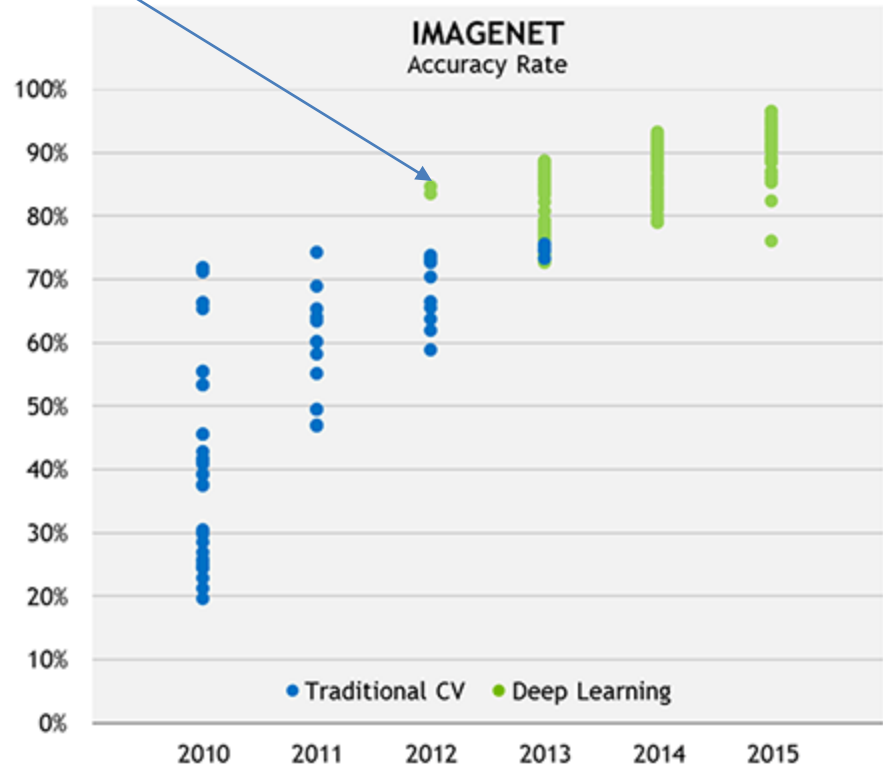
	dalmatian
	grape
	elderberry
	ffordshire bullterrier
	currant

	squirrel monkey
	spider monkey
	titi
	indri
	howler monkey

ImageNet Competition

- Krizhevsky, 2012
- Google, Microsoft 2015
 - Beat the best human score in the ImageNet challenge.

2015: A MILESTONE YEAR IN COMPUTER SCIENCE



Machine Learning Problems

Supervised Learning

Unsupervised Learning

Discrete

classification or
categorization

clustering

Continuous

regression

dimensionality
reduction

Machine Learning Problems

Supervised Learning

Unsupervised Learning

Discrete
Continuous

classification or
categorization

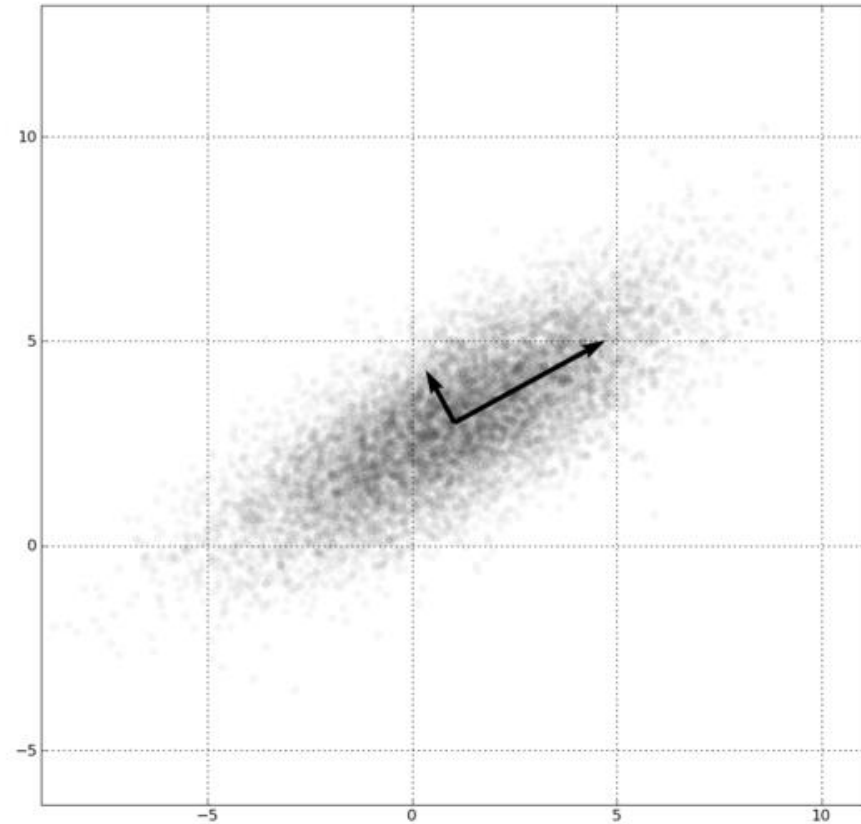
clustering

regression

dimensionality
reduction

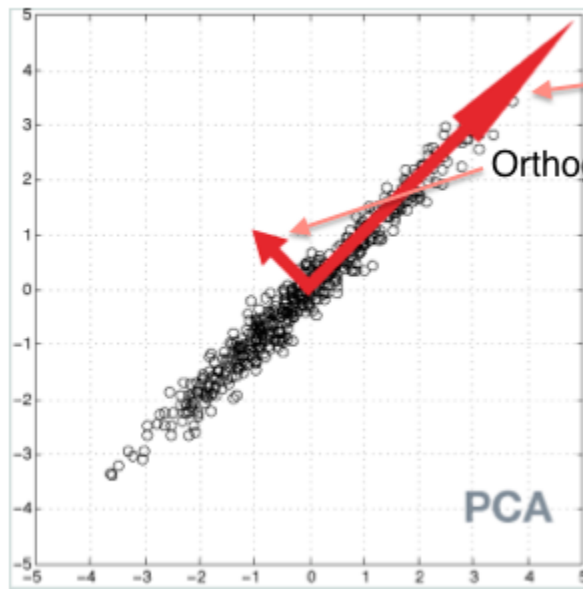
Dimensionality Reduction

- **PCA, ICA, LLE, Isomap**
- Principal component analysis
 - Creates a basis where the axes represent the dimensions of variance, from high to low.
 - Finds correlations in data dimensions to produce *best possible* lower-dimensional representation based on linear projections.
 - Used for dimensionality reduction.

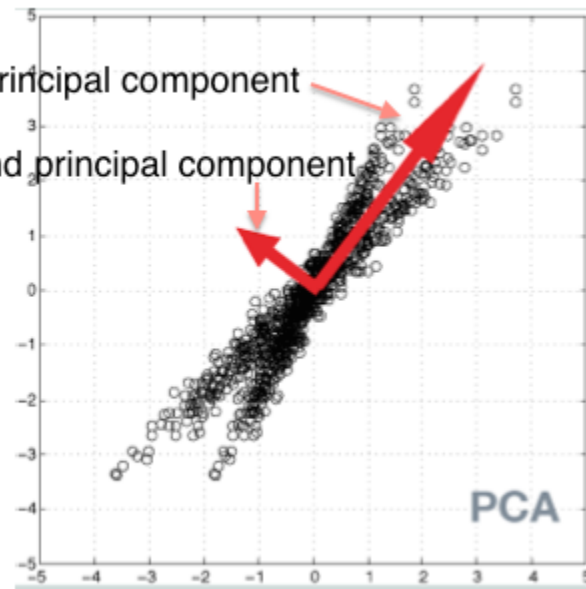


PCA

A

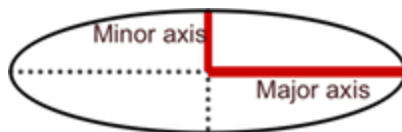


B



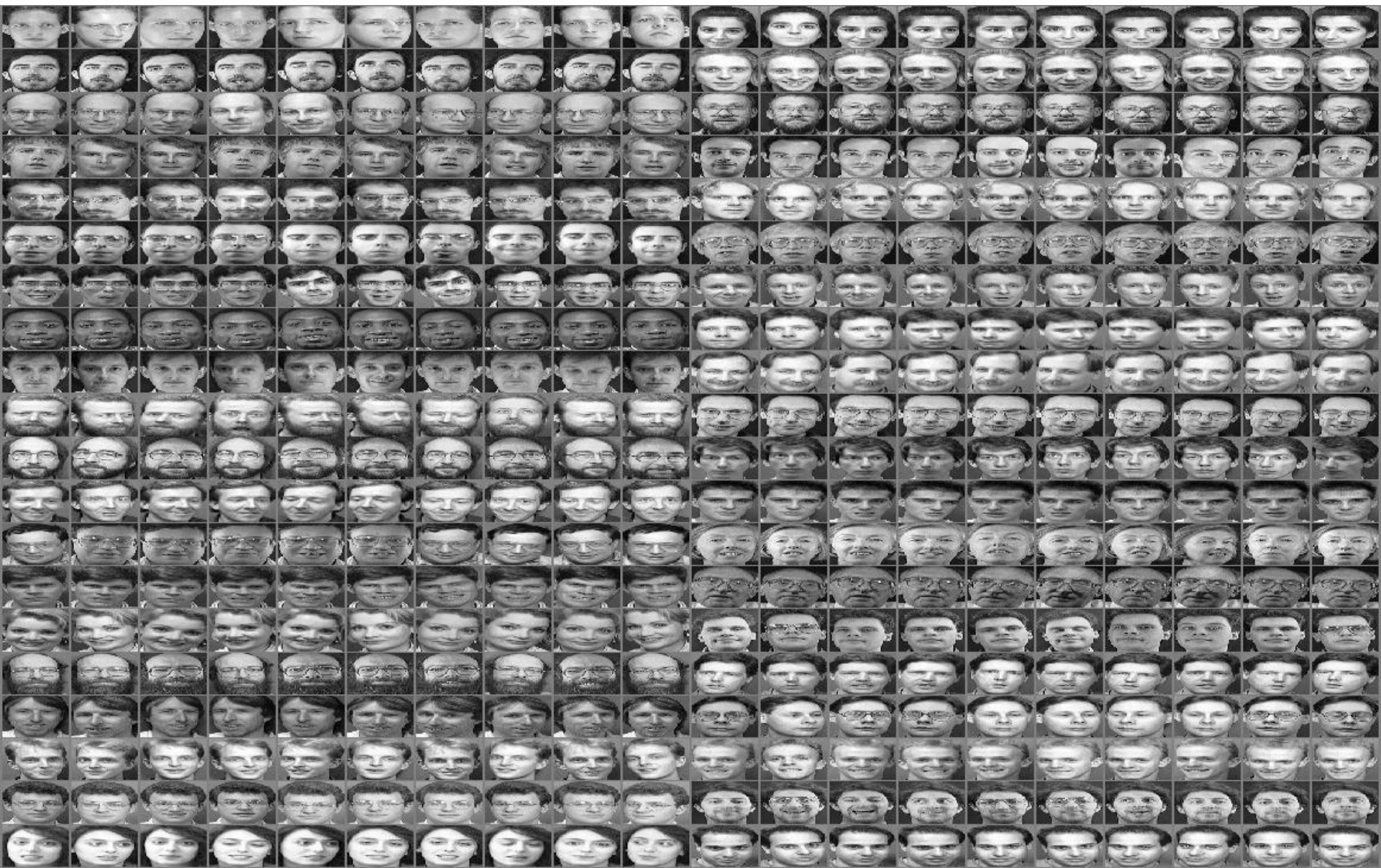
1st principal component
Orthogonal 2nd principal component

(Figure adapted from C. Beckmann, Oxford FMRIB)



Eigenfaces

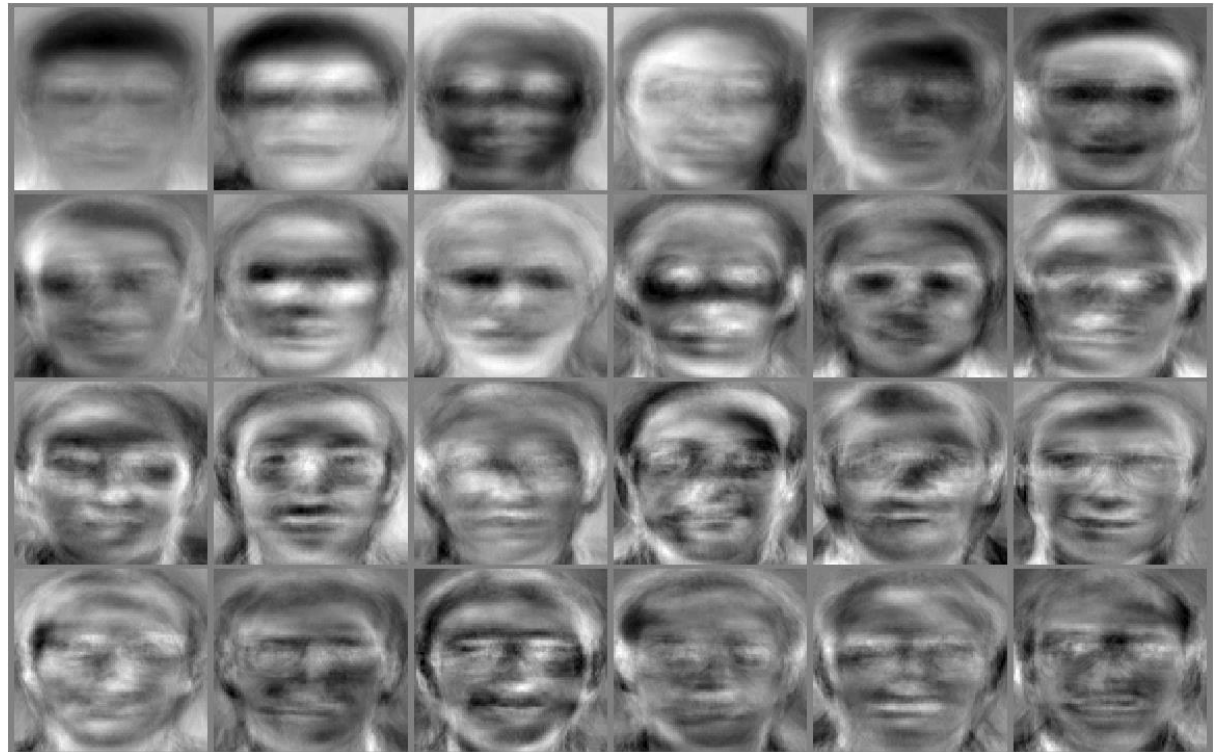
*The ATT face database (formerly the ORL database),
10 pictures of 40 subjects each*



Eigenfaces



Mean face



Basis of variance (eigenvectors)

Machine Learning Problems

Supervised Learning

Unsupervised Learning

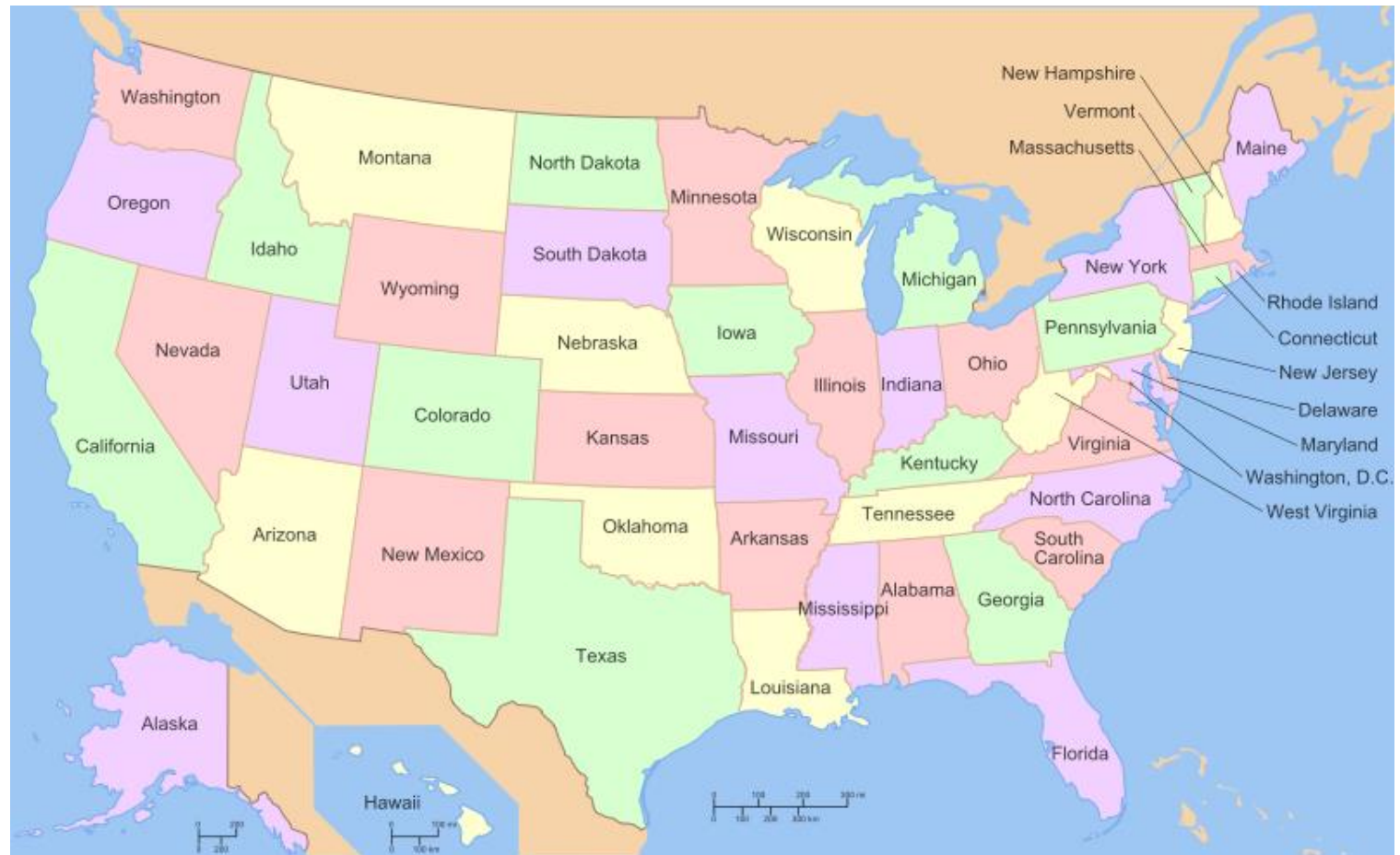
Discrete
Continuous

classification or
categorization

clustering

regression

dimensionality
reduction





The United States redrawn as Fifty States with Equal Population

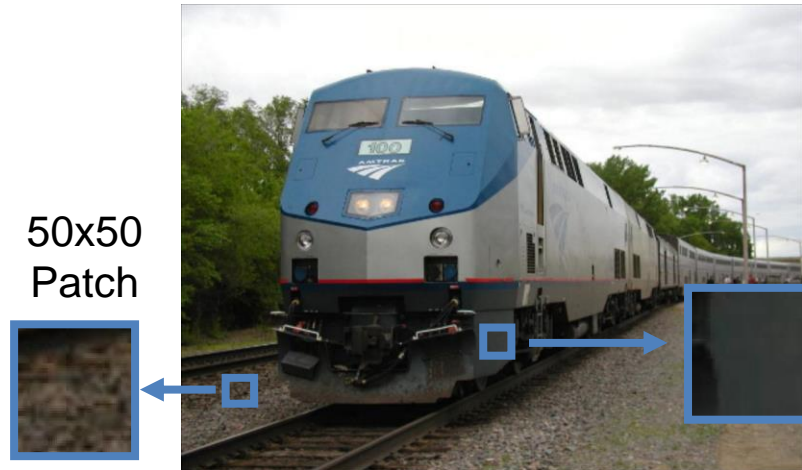


Clustering example: image segmentation

Goal: Break up the image into meaningful or perceptually similar regions



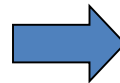
Segmentation for feature support or efficiency



50x50 Patch



[Felzenszwalb and Huttenlocher 2004]



Superpixels!

[Shi and Malik 2001]

[Hoiem et al. 2005, Mori 2005]

Derek Hoiem

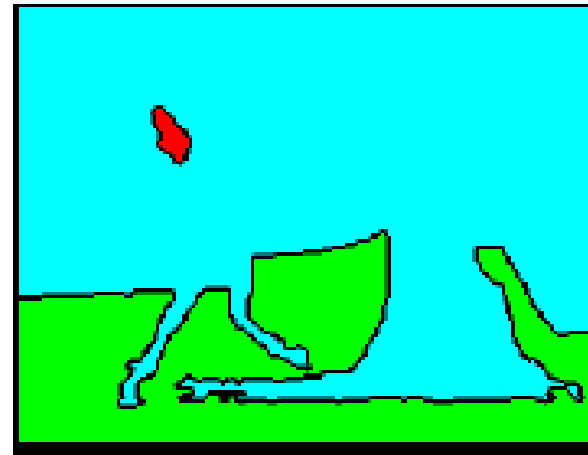
Segmentation as a result



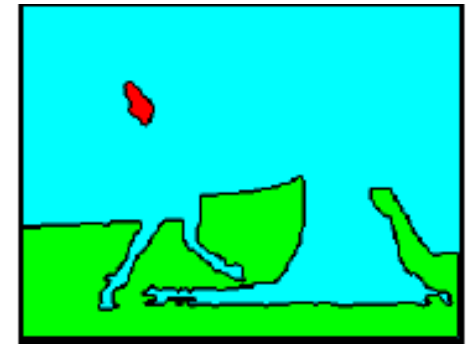
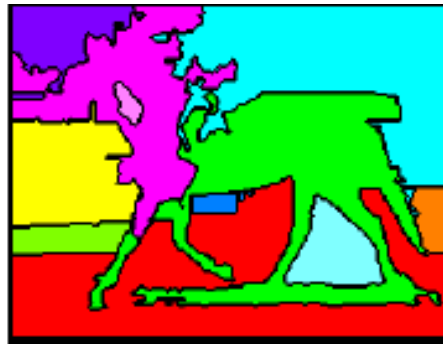
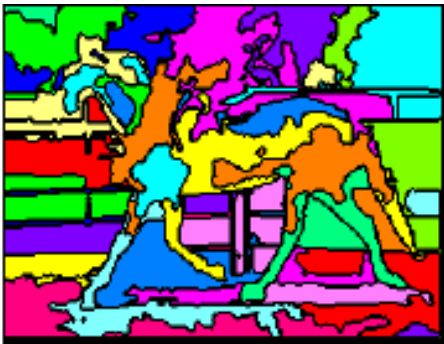
Types of segmentations



Oversegmentation



Undersegmentation



Hierarchical Segmentations

Clustering

Group together similar 'points' and represent them with a single token.

Key Challenges:

- 1) What makes two points/images/patches similar?
- 2) How do we compute an overall grouping from pairwise similarities?

Why do we cluster?

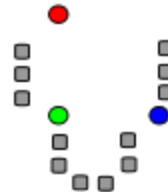
- **Summarizing data**
 - Look at large amounts of data
 - Patch-based compression or denoising
 - Represent a large continuous vector with the cluster number
- **Counting**
 - Histograms of texture, color, SIFT vectors
- **Segmentation**
 - Separate the image into different regions
- **Prediction**
 - Images in the same cluster may have the same labels

How do we cluster?

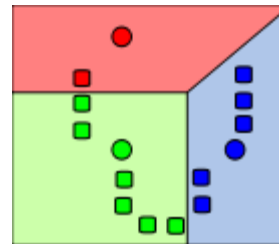
- K-means
 - Iteratively re-assign points to the nearest cluster center
- Agglomerative clustering
 - Start with each point as its own cluster and iteratively merge the closest clusters
- Mean-shift clustering
 - Estimate modes of pdf
- Spectral clustering
 - Split the nodes in a graph based on assigned links with similarity weights

K-means algorithm

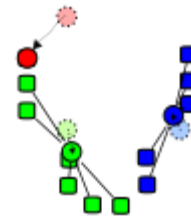
1. Randomly select K centers



2. Assign each point to nearest center

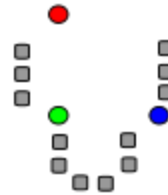


3. Compute new center (mean) for each cluster



K-means algorithm

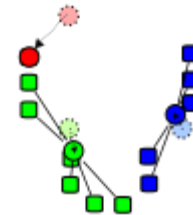
1. Randomly select K centers



2. Assign each point to nearest center



3. Compute new center (mean) for each cluster



Back to 2



K-means

1. Initialize cluster centers: \mathbf{c}^0 ; $t=0$

2. Assign each point to the closest center

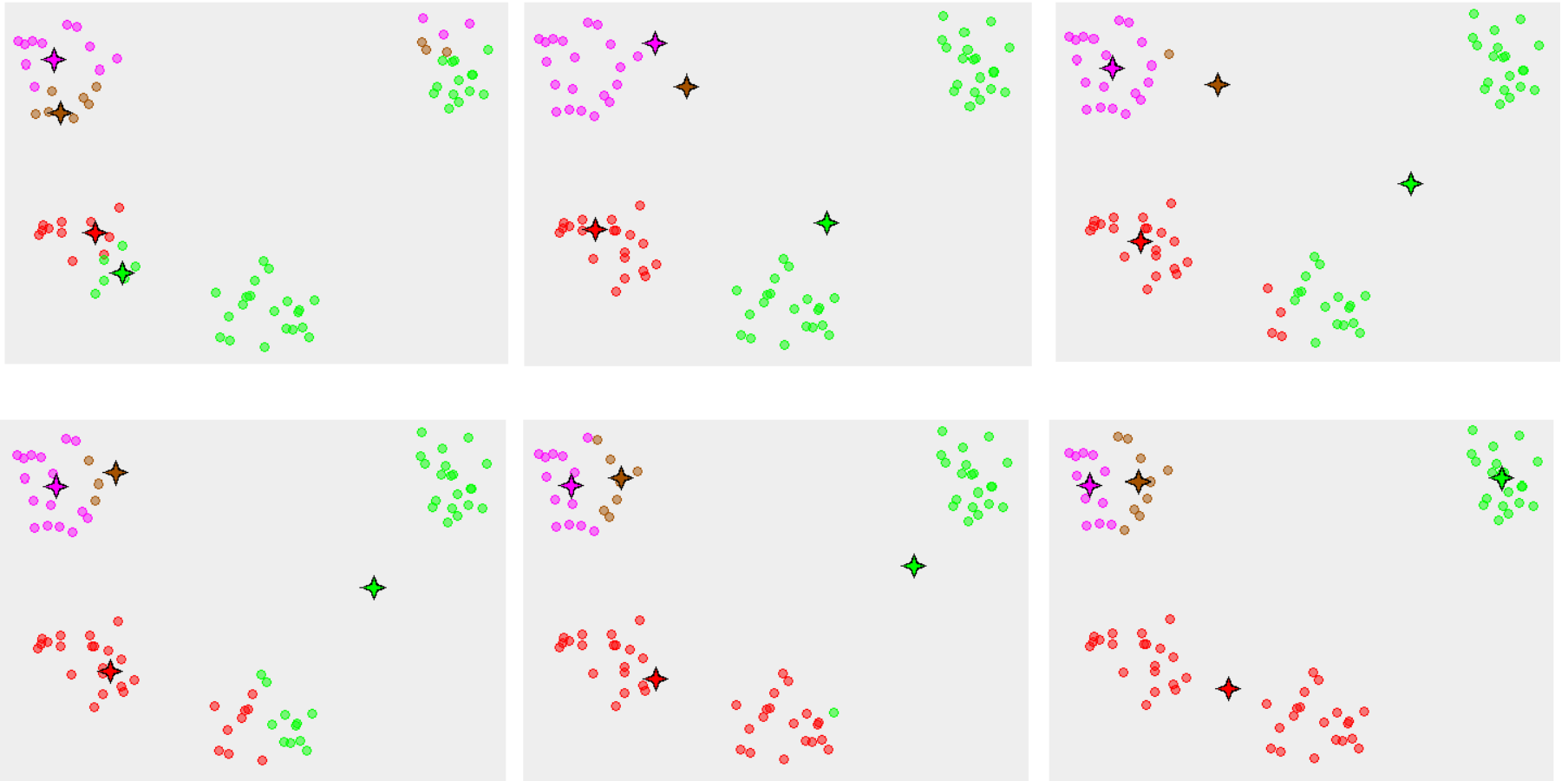
$$\delta^t = \underset{\delta}{\operatorname{argmin}} \frac{1}{N} \sum_j^N \sum_i^K \delta_{ij} \left(\mathbf{c}_i^{t-1} - \mathbf{x}_j \right)^2$$

3. Update cluster centers as the mean of the points

$$\mathbf{c}^t = \underset{\mathbf{c}}{\operatorname{argmin}} \frac{1}{N} \sum_j^N \sum_i^K \delta_{ij}^t \left(\mathbf{c}_i - \mathbf{x}_j \right)^2$$

4. Repeat 2-3 until no points are re-assigned ($t=t+1$)

K-means converges to a local minimum



K-means: design choices

- Initialization
 - Randomly select K points as initial cluster center
 - Or greedily choose K points to minimize residual
- Distance measures
 - Traditionally Euclidean, could be others
- Optimization
 - Will converge to a *local minimum*
 - May want to perform multiple restarts

K-means clustering using intensity or color

Image



Clusters on intensity



Clusters on color

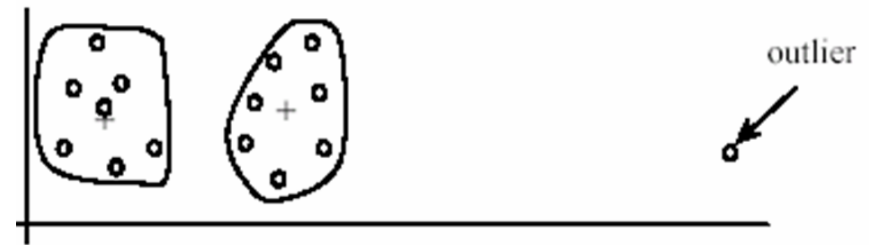


How to choose the number of clusters?

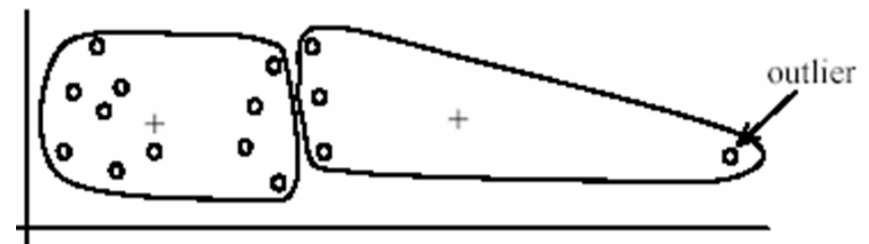
- Validation set
 - Try different numbers of clusters and look at performance
 - When building dictionaries (discussed later), more clusters typically work better

K-Means pros and cons

- Pros
 - Finds cluster centers that minimize conditional variance (good representation of data)
 - Simple and fast*
 - Easy to implement
- Cons
 - Need to choose K
 - Sensitive to outliers
 - Prone to local minima
 - All clusters have the same parameters (e.g., distance measure is non-adaptive)
 - *Can be slow: each iteration is $O(KNd)$ for N d-dimensional points
- Usage
 - Rarely used for pixel segmentation

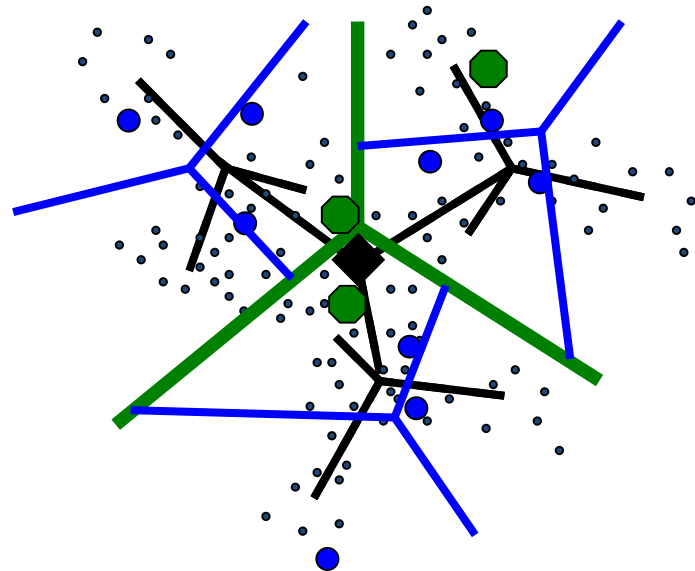


(B): Ideal clusters

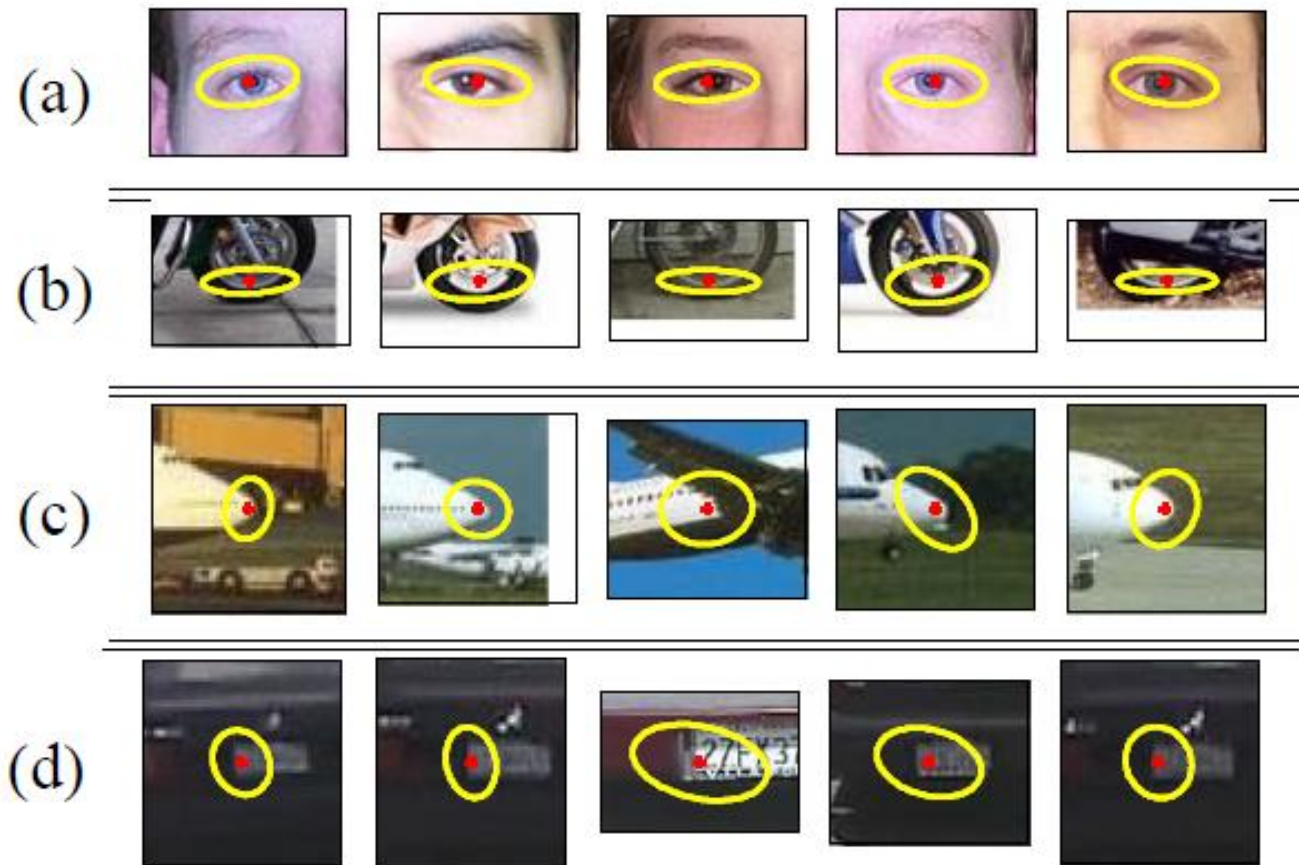


Building Visual Dictionaries

1. Sample patches from a database
 - E.g., 128 dimensional SIFT vectors
2. Cluster to build dictionary
 - Cluster centers are the dictionary words
3. For new patches, assign to the nearest cluster to save rebuilding dictionary

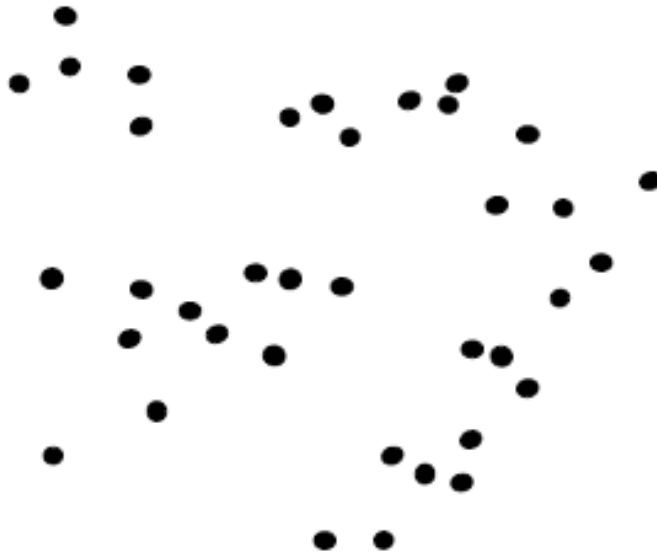


Examples of learned codewords



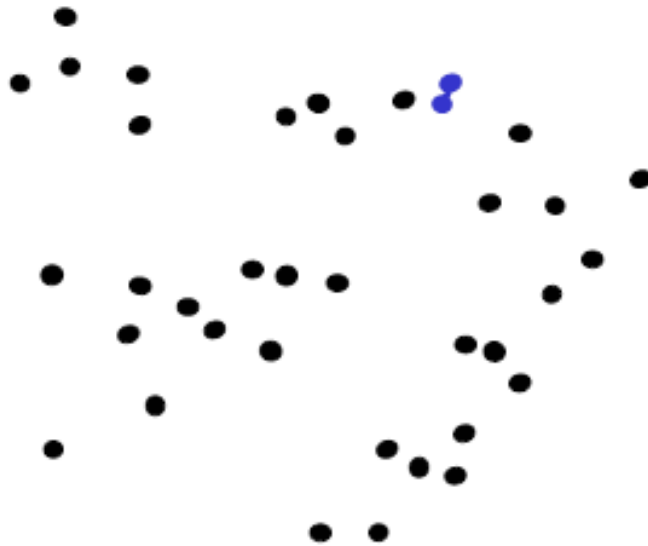
Most likely codewords for 4 learned “topics”
EM with multinomial (problem 3) to get topics

Agglomerative clustering



1. Say "Every point is its own cluster"

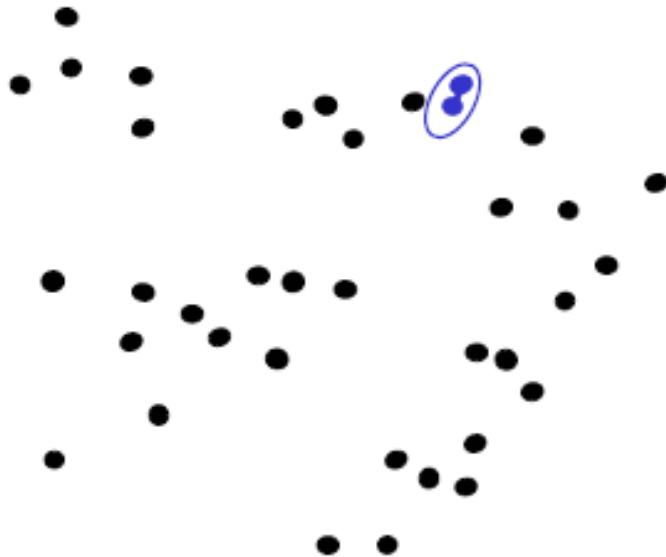
Agglomerative clustering



1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters



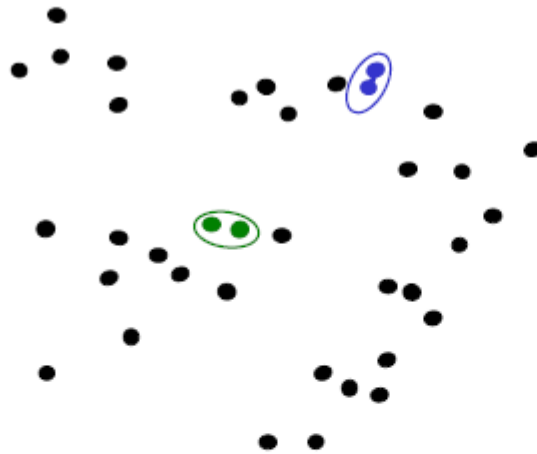
Agglomerative clustering



1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster



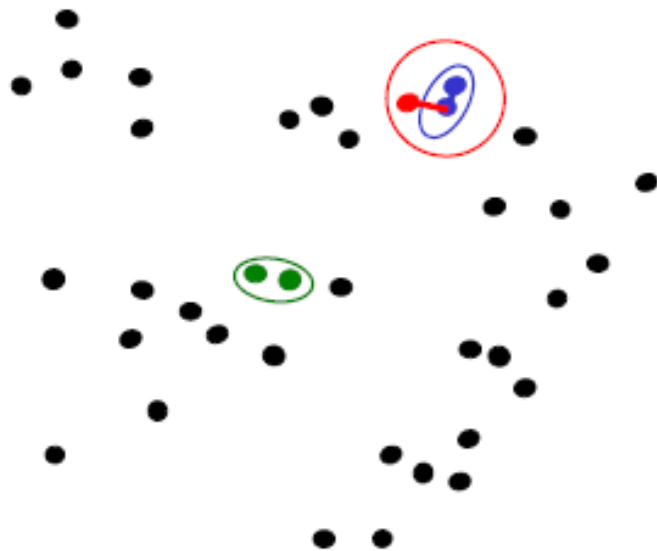
Agglomerative clustering



1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster
4. Repeat



Agglomerative clustering



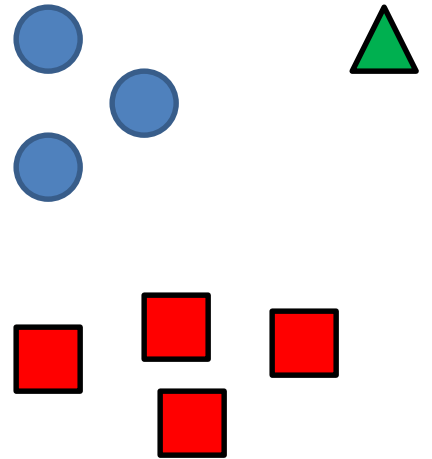
1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster
4. Repeat



Agglomerative clustering

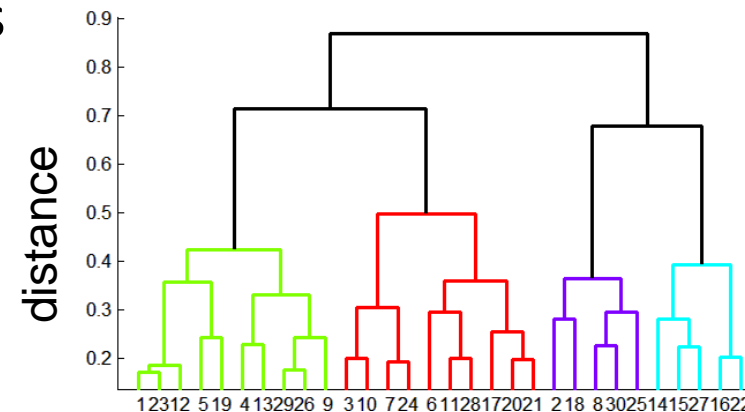
How to define cluster similarity?

- Average distance between points, maximum distance, minimum distance
- Distance between means or medoids



How many clusters?

- Clustering creates a dendrogram (a tree)
- Threshold based on max number of clusters or based on distance between merges



Conclusions: Agglomerative Clustering

Good

- Simple to implement, widespread application
- Clusters have adaptive shapes
- Provides a hierarchy of clusters

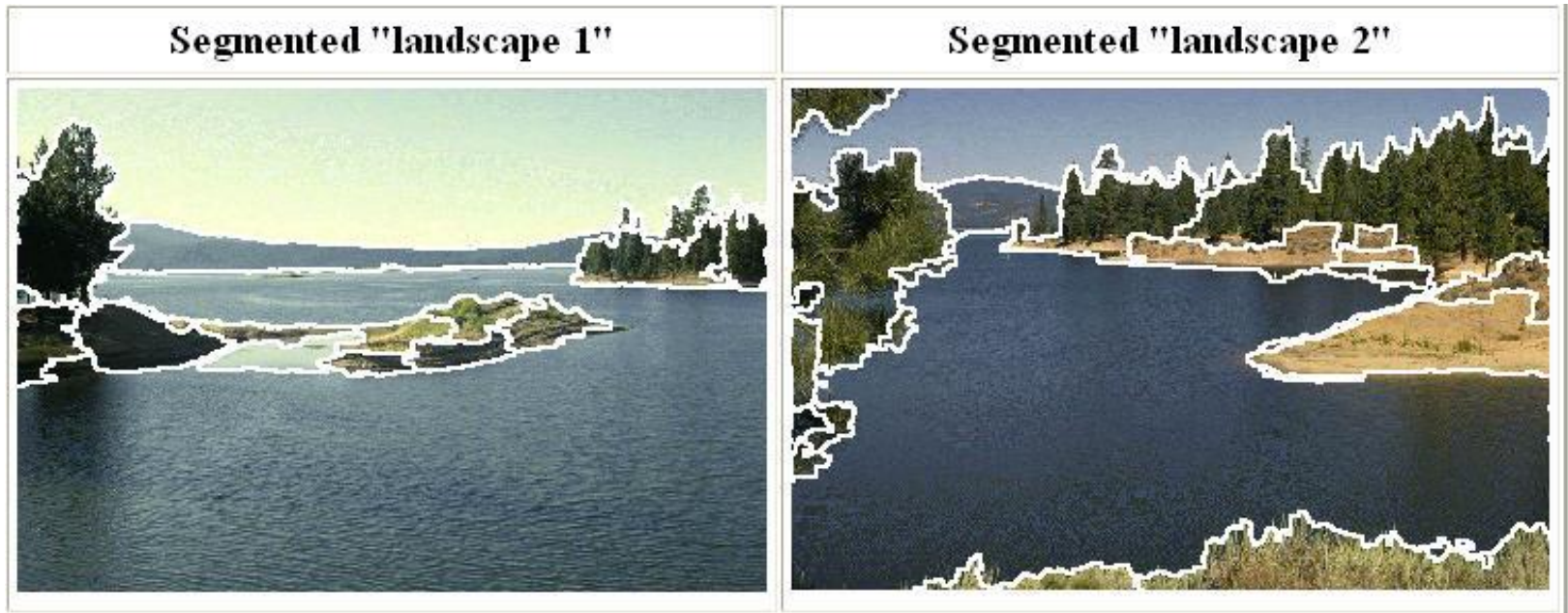
Bad

- May have imbalanced clusters
- Still have to choose number of clusters or threshold
- Need to use an “ultrametric” to get a meaningful hierarchy

Mean shift segmentation

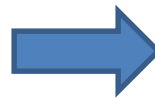
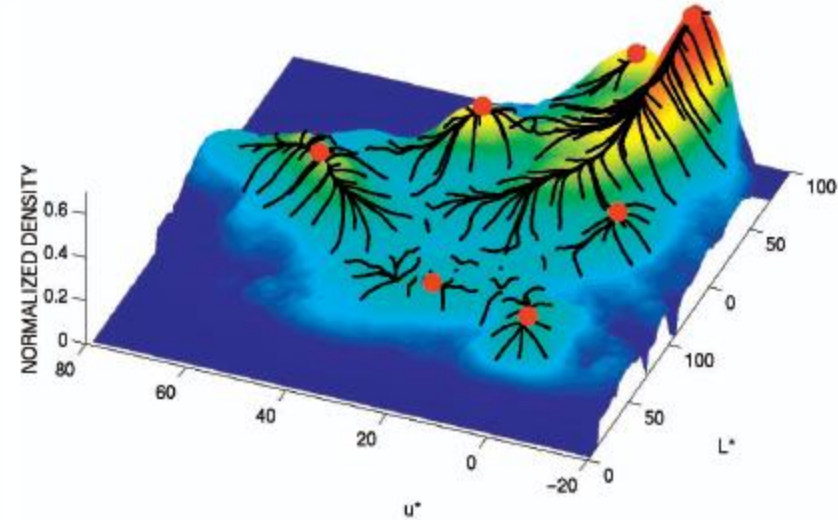
D. Comaniciu and P. Meer, Mean Shift: A Robust Approach toward Feature Space Analysis, PAMI 2002.

- Versatile technique for clustering-based segmentation

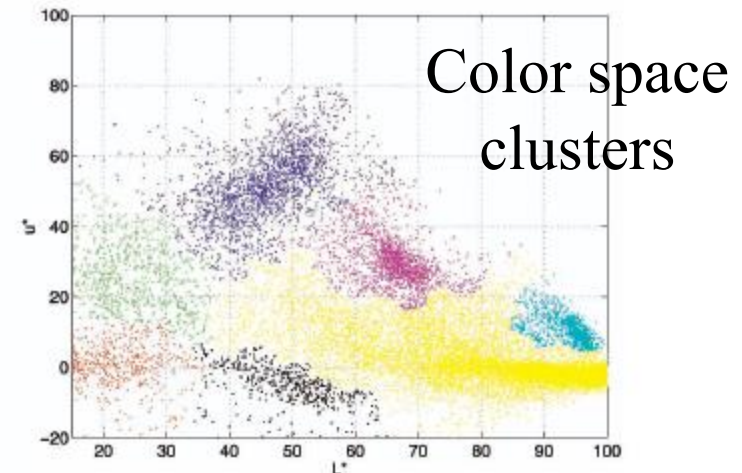
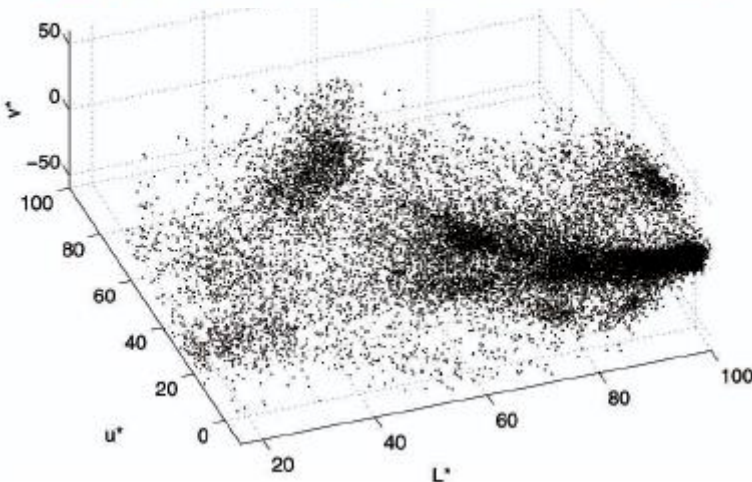


Mean shift algorithm

Try to find *modes* of a non-parametric density.

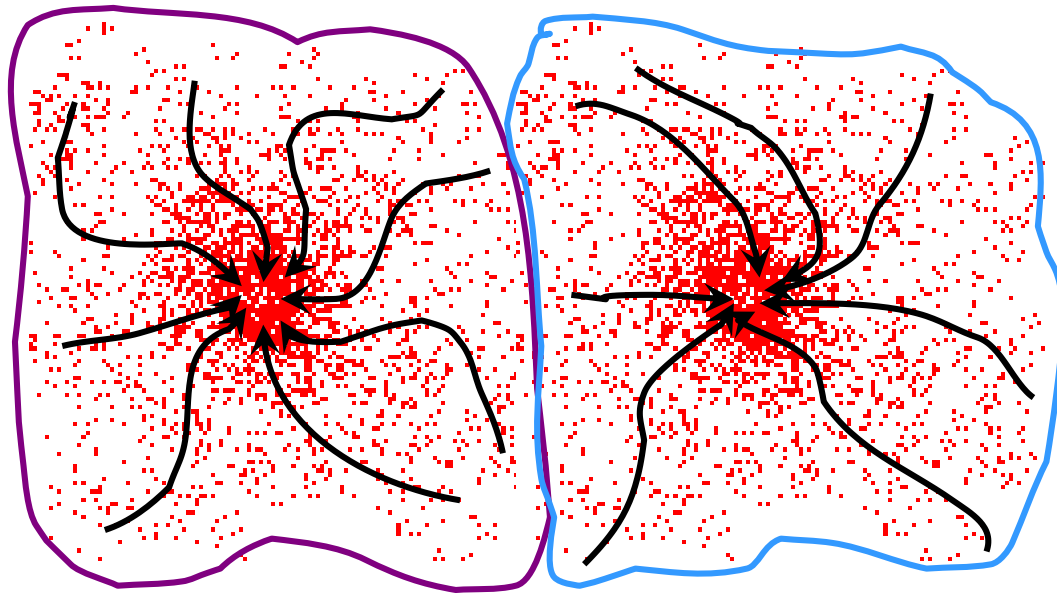


Color
space

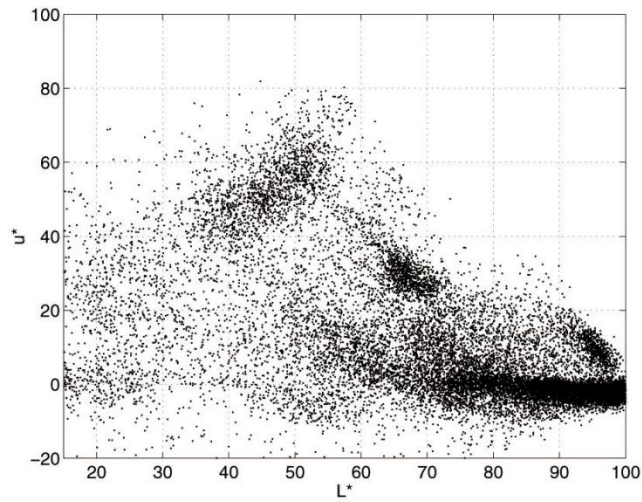


Attraction basin

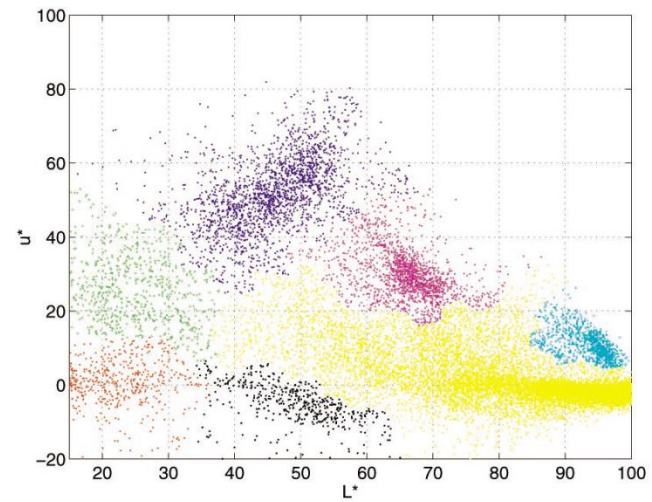
- **Attraction basin:** the region for which all trajectories lead to the same mode
- **Cluster:** all data points in the attraction basin of a mode



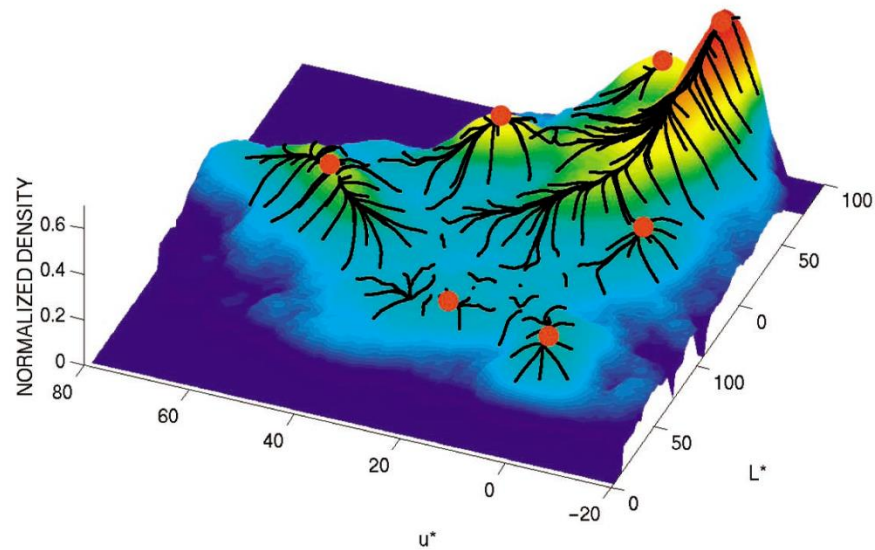
Attraction basin



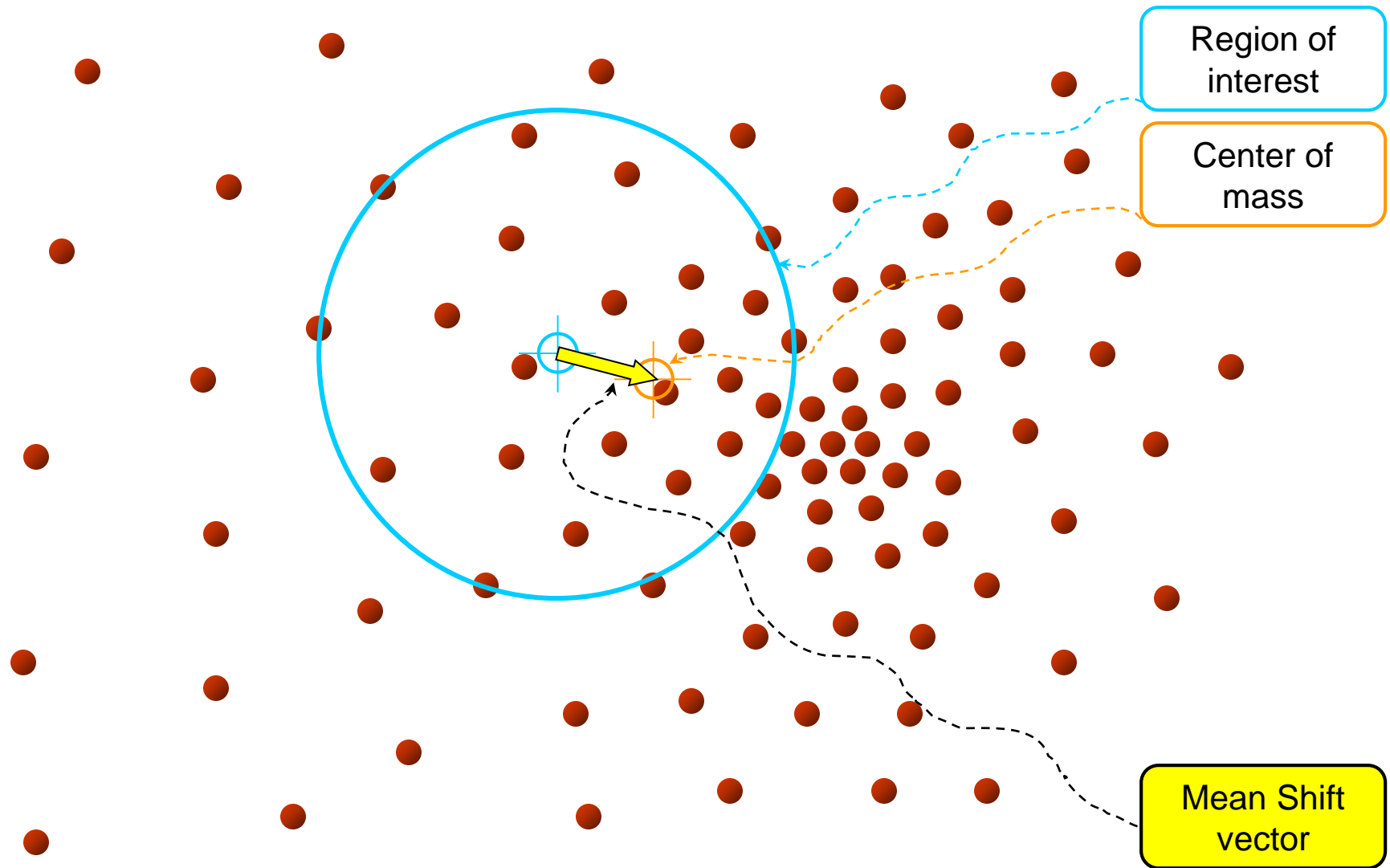
(a)



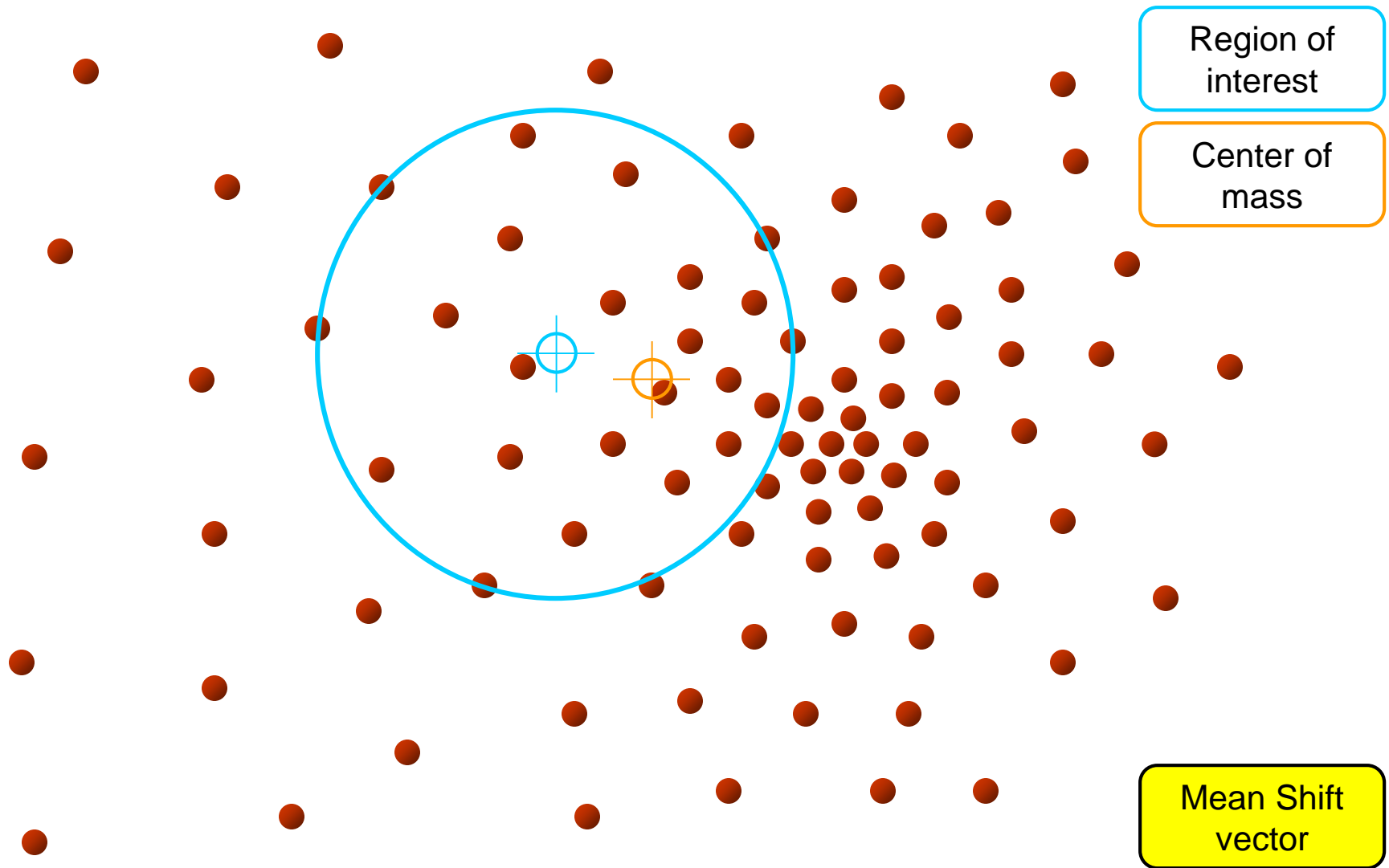
(b)



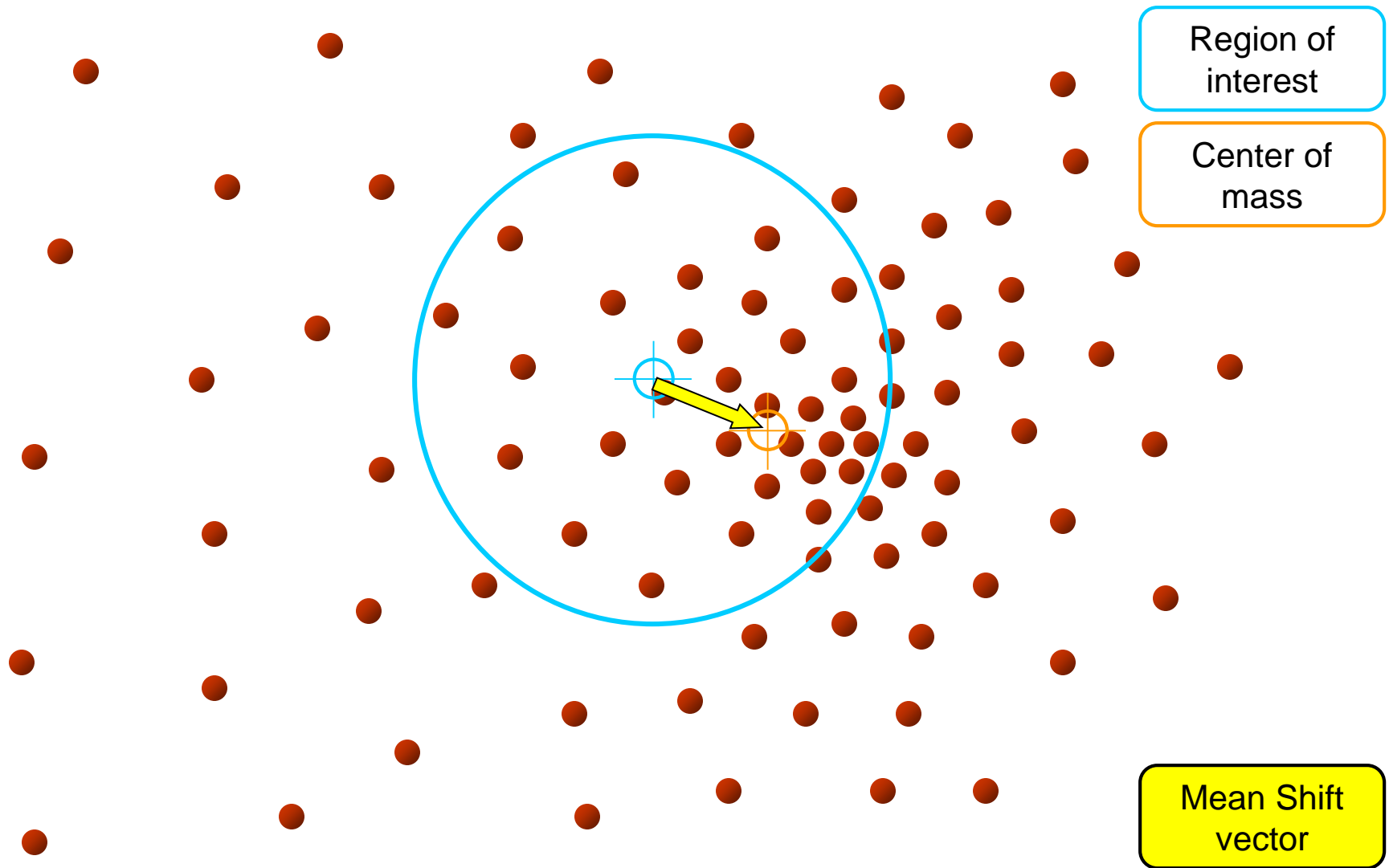
Mean shift



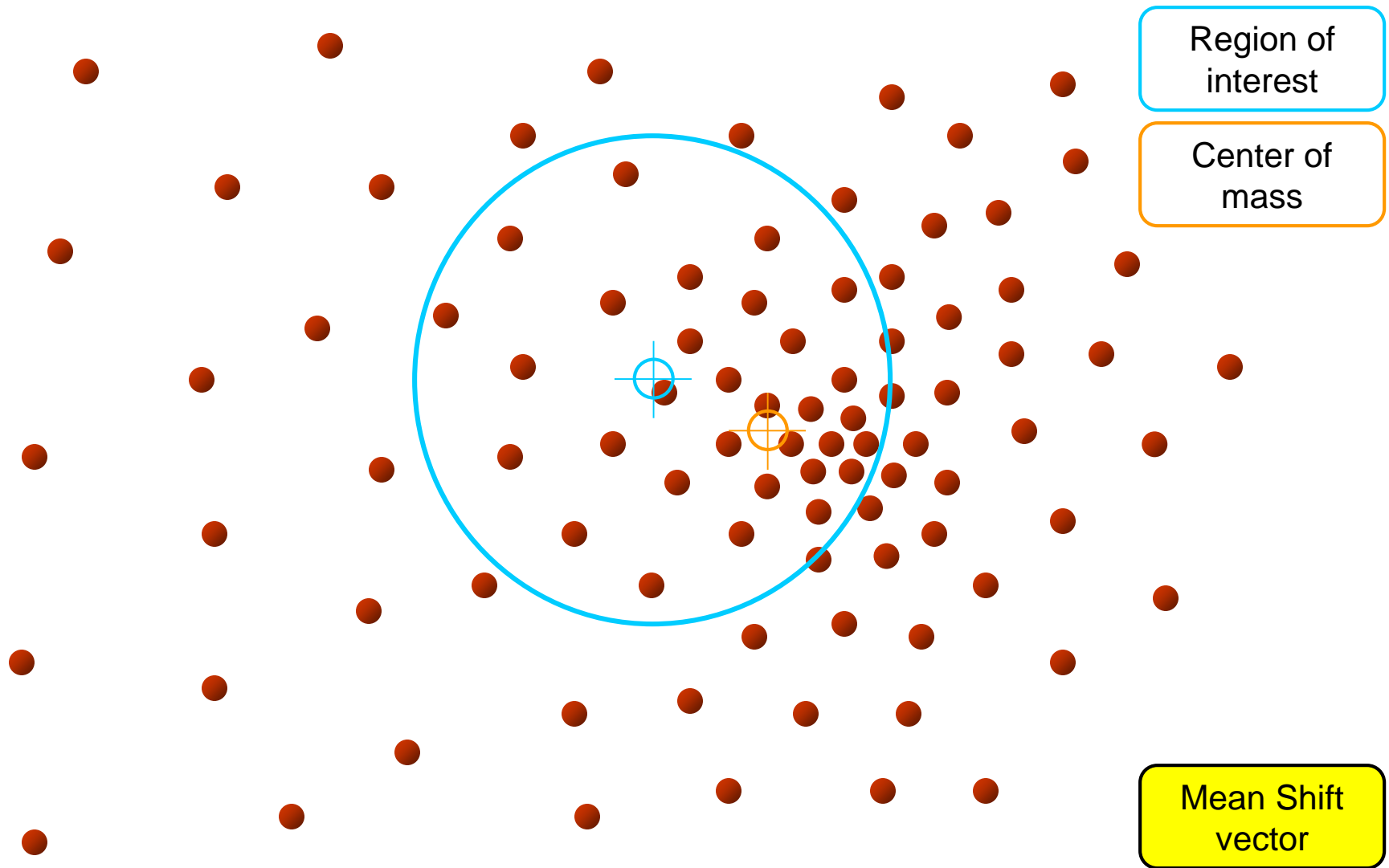
Mean shift



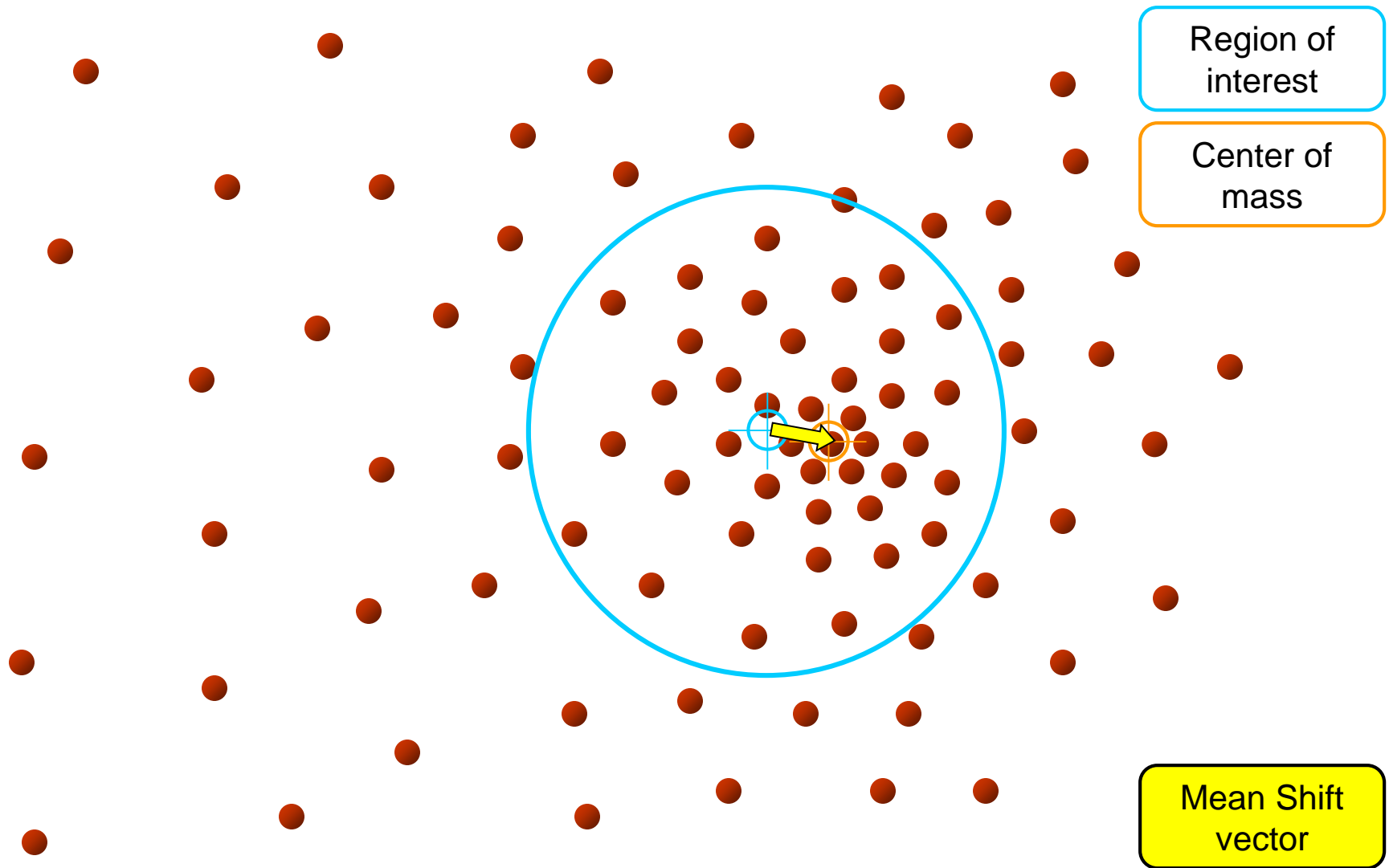
Mean shift



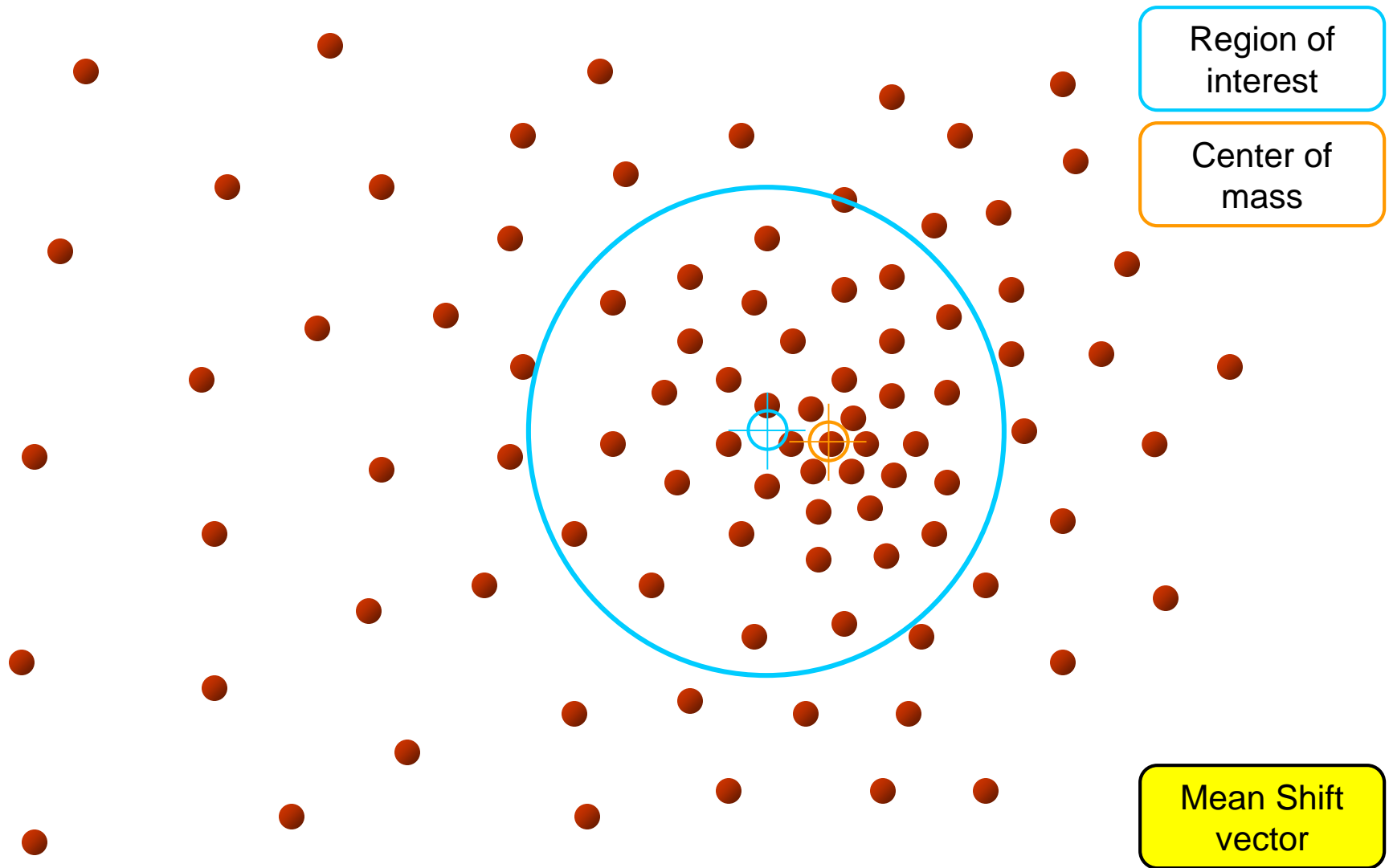
Mean shift



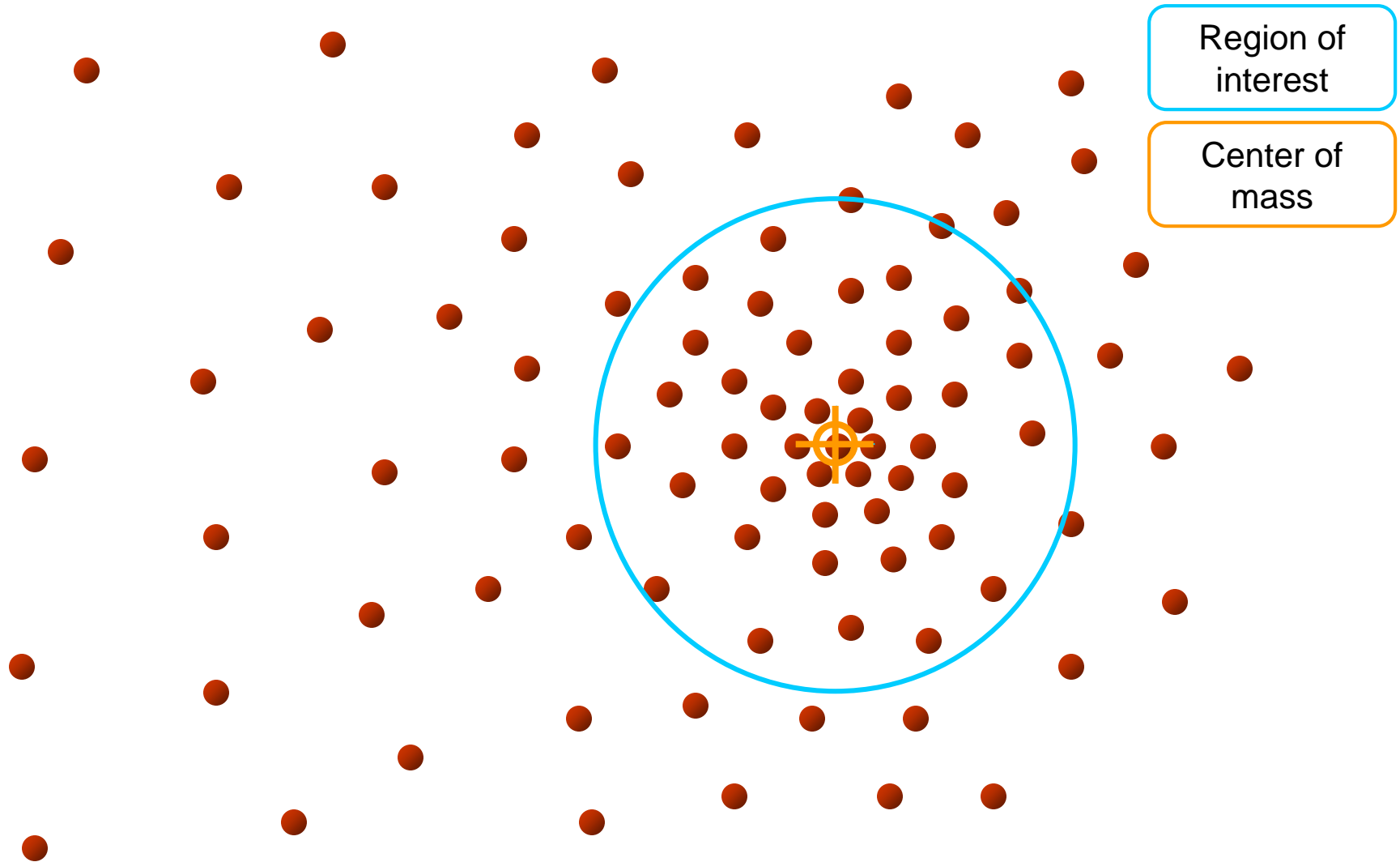
Mean shift



Mean shift



Mean shift



Kernel density estimation

Kernel density estimation function

$$\hat{f}_h(x) = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{x - x_i}{h}\right)$$

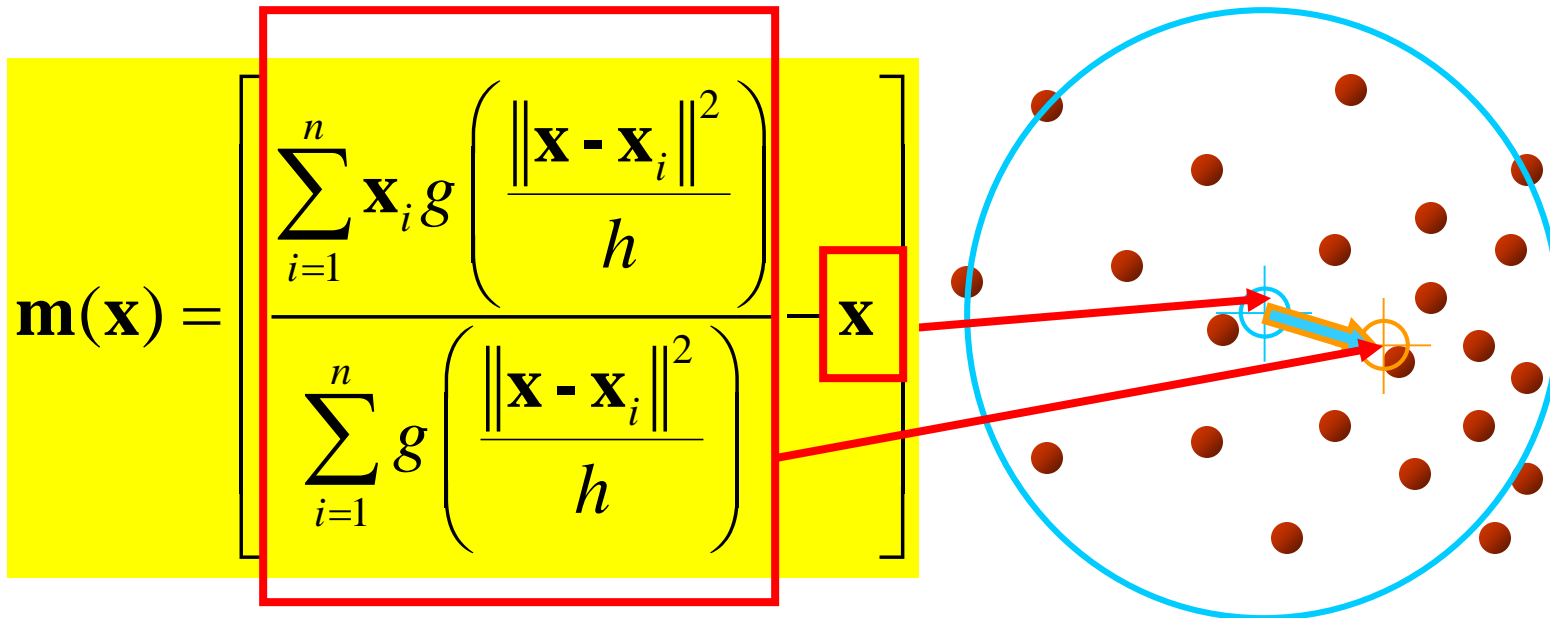
Gaussian kernel

$$K\left(\frac{x - x_i}{h}\right) = \frac{1}{\sqrt{2\pi}} e^{-\frac{(x - x_i)^2}{2h^2}}.$$

Computing the Mean Shift

Simple Mean Shift procedure:

- Compute mean shift vector
- Translate the Kernel window by $\mathbf{m}(\mathbf{x})$

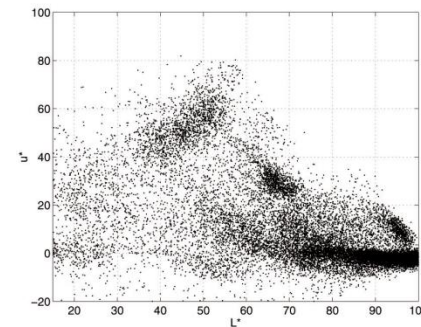


Mean shift clustering

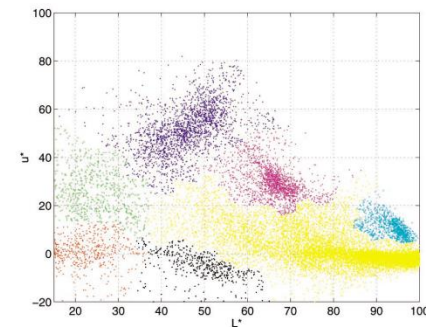
- The mean shift algorithm seeks *modes* of the given set of points
 1. Choose kernel and bandwidth
 2. For each point:
 - a) Center a window on that point
 - b) Compute the mean of the data in the search window
 - c) Center the search window at the new mean location
 - d) Repeat (b,c) until convergence
 3. Assign points that lead to nearby modes to the same cluster

Segmentation by Mean Shift

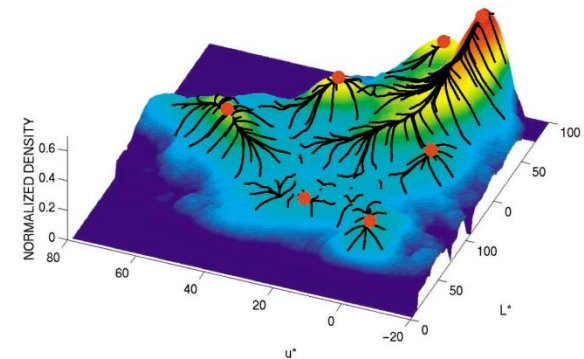
- Compute features for each pixel (color, gradients, texture, etc.).
- Set kernel size for features K_f and position K_s .
- Initialize windows at individual pixel locations.
- Perform mean shift for each window until convergence.
- Merge windows that are within width of K_f and K_s .



(a)

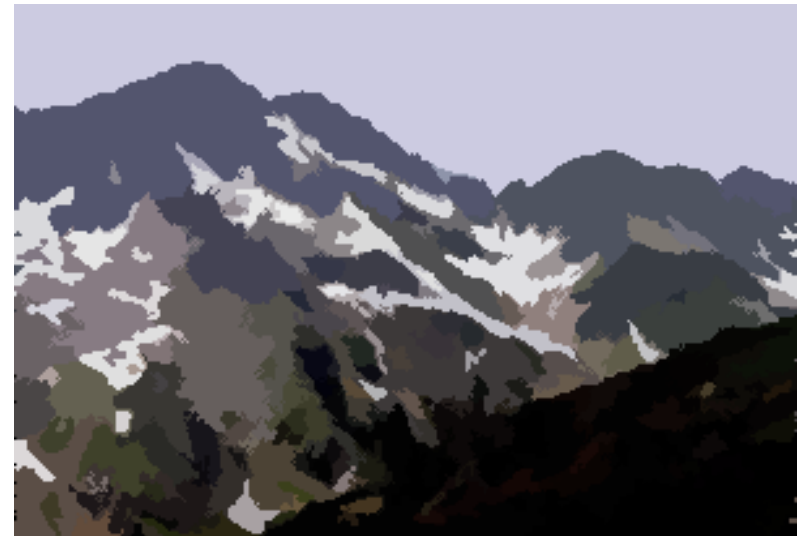


(b)



(c)

Mean shift segmentation results



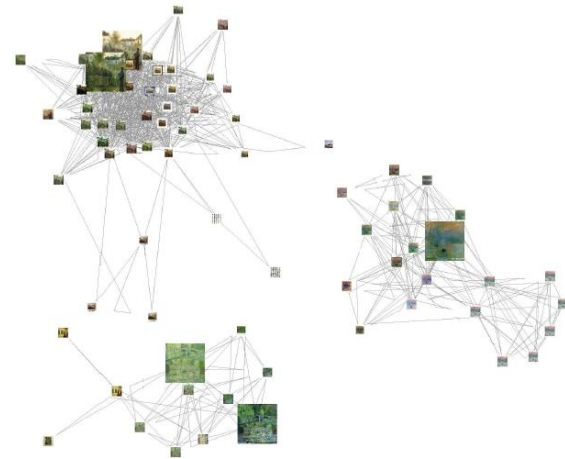
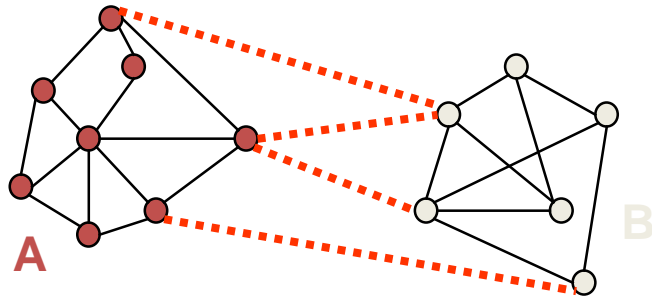


Mean shift pros and cons

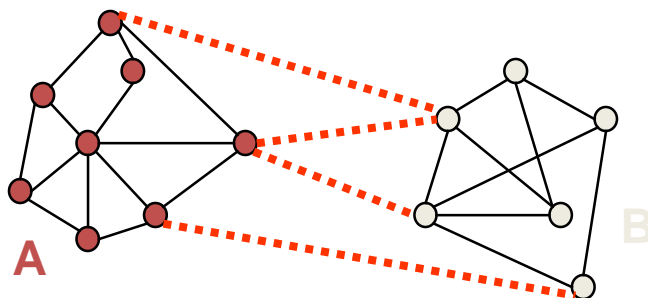
- Pros
 - Good general-practice segmentation
 - Flexible in number and shape of regions
 - Robust to outliers
- Cons
 - Have to choose kernel size in advance
 - Not suitable for high-dimensional features
- When to use it
 - Oversegmentation
 - Multiple segmentations
 - Tracking, clustering, filtering applications

Spectral clustering

Group points based on links in a graph



Cuts in a graph



Normalized Cut

- a cut penalizes large segments
- fix by normalizing for size of segments

$$Ncut(A, B) = \frac{cut(A, B)}{volume(A)} + \frac{cut(A, B)}{volume(B)}$$

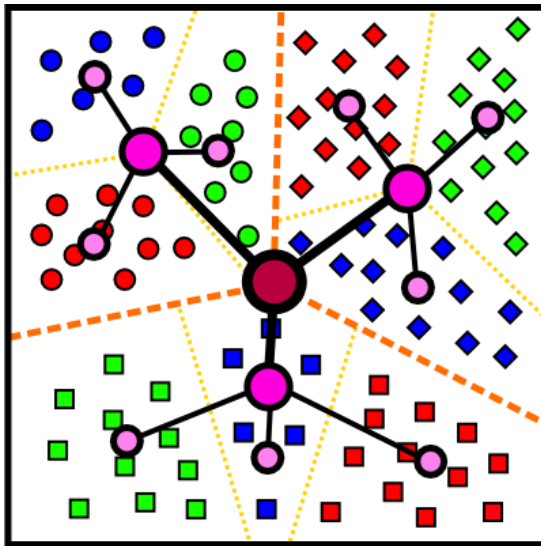
- $volume(A)$ = sum of costs of all edges that touch A

Normalized cuts for segmentation

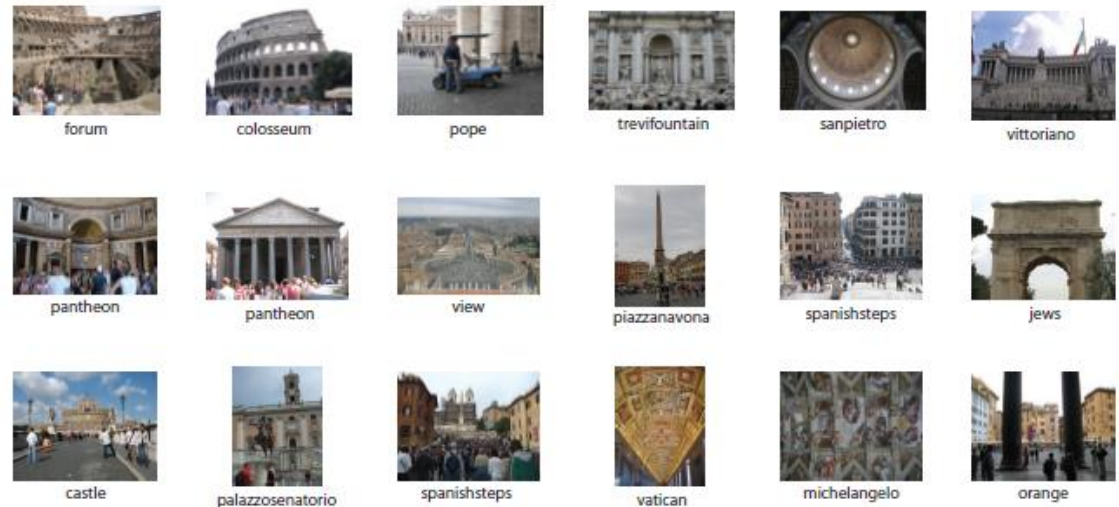


Which algorithm to use?

- Quantization/Summarization: K-means
 - Aims to preserve variance of original data
 - Can easily assign new point to a cluster



Quantization for
computing histograms

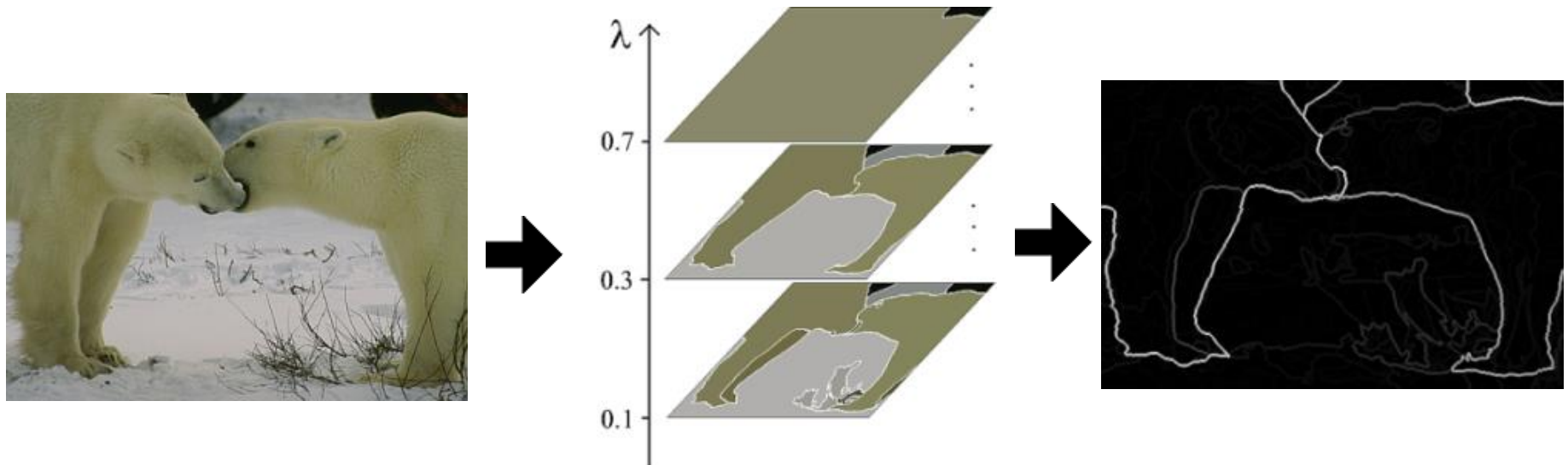


Summary of 20,000 photos of Rome using
“greedy k-means”

<http://grail.cs.washington.edu/projects/canonview/>

Which algorithm to use?

- Image segmentation: agglomerative clustering
 - More flexible with distance measures (e.g., can be based on boundary prediction)
 - Adapts better to specific data
 - Hierarchy can be useful



Machine Learning Problems

