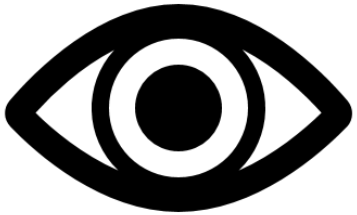
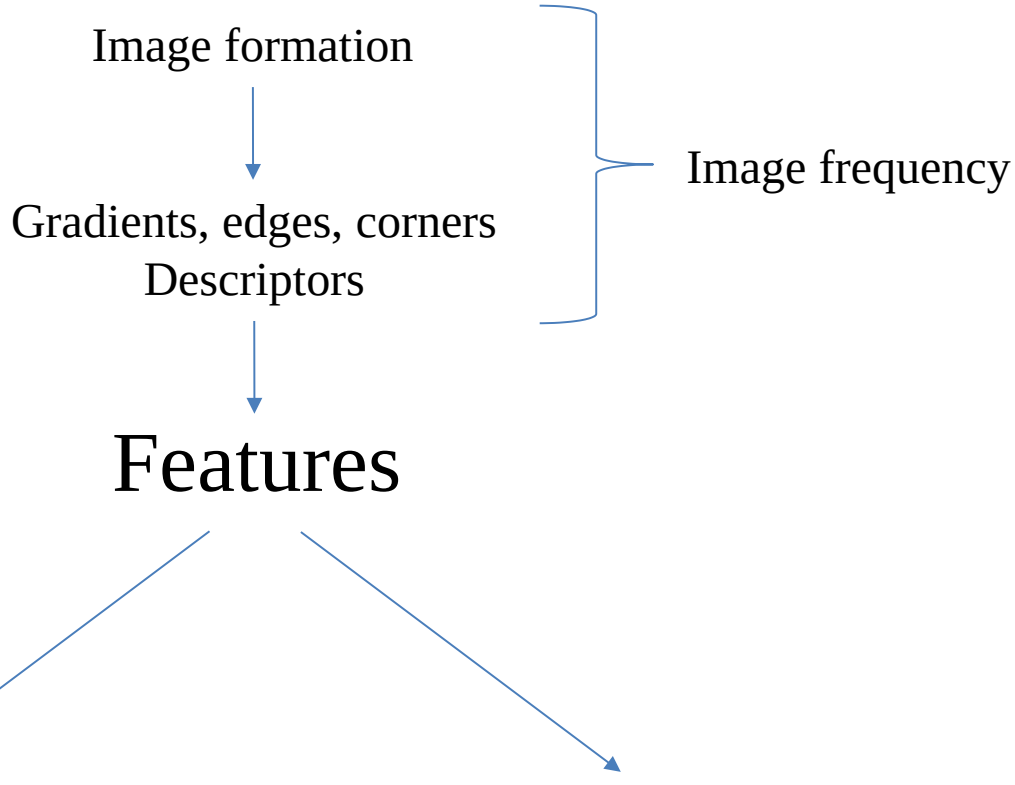


Advanced Image Processing

ML: Unsupervised Learning

SYDE 671

Where to go from our basic building block?



Recognition

Scenes, places, objects,

~ 5 weeks (inc. CNNs)

Reconstruction

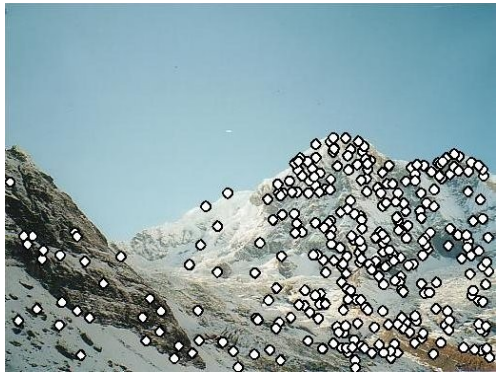
Geometric understanding

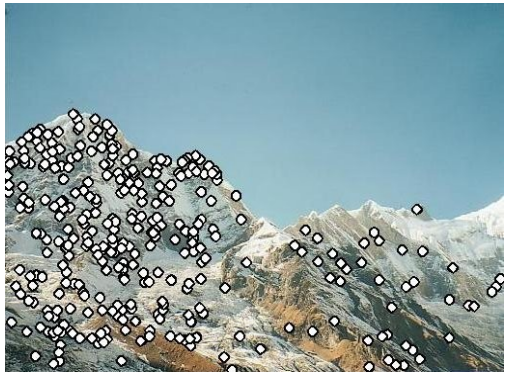
Last topic in class

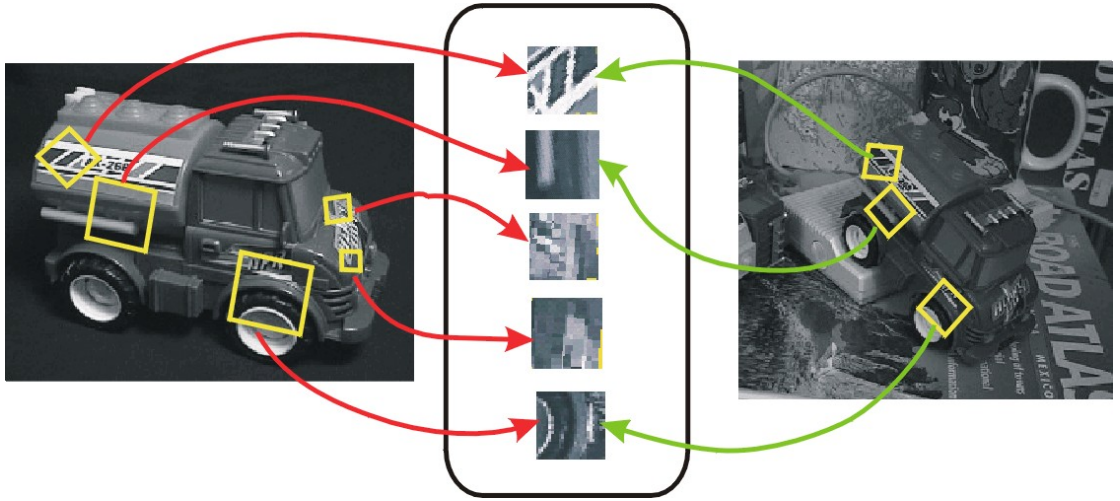
Panorama stitching / instance recognition

Needs more geometric understanding…

…but we'll see it later on.



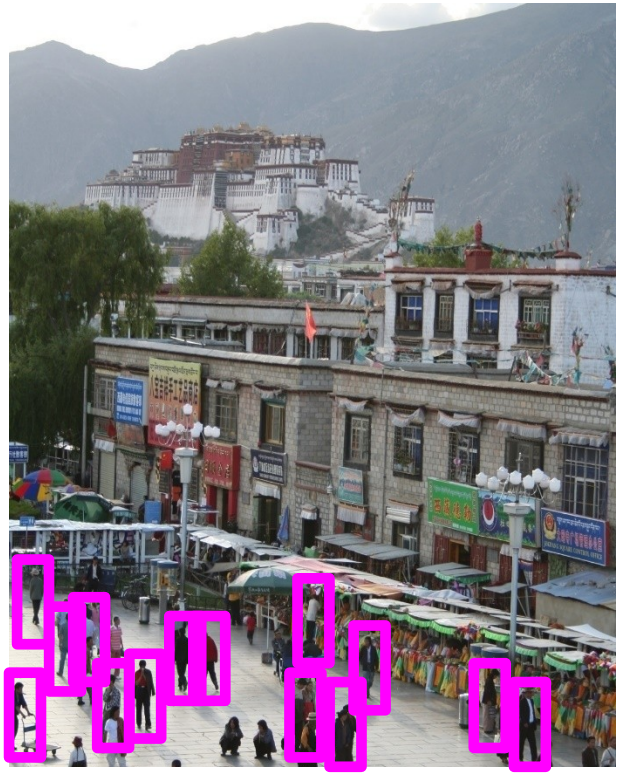




Recognition

Often needs machine learning
for compact descriptions of the visual world.

Scene recognition
- City/forest/factory/…



Find pedestrians

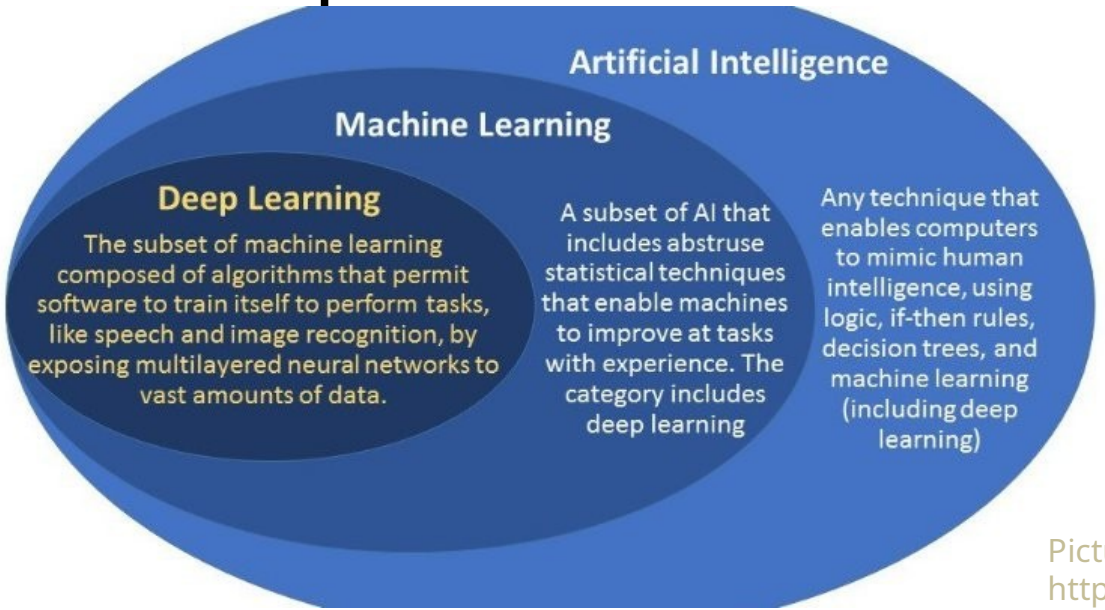
ML CRASH COURSE

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. Alternatively there are quite a few good reference books on the topic.

Machine Learning

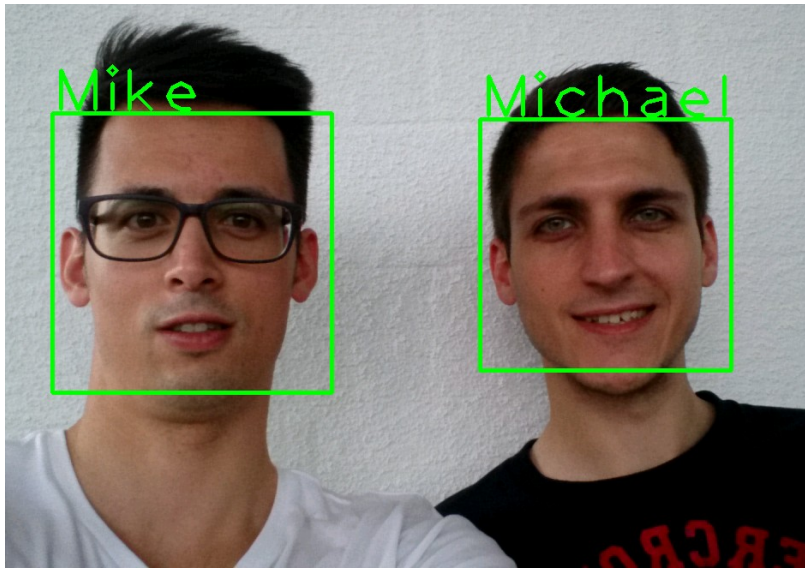
- Learn from and make predictions on data.
- Arguably the greatest export from computing to other scientific fields.
- Statisticians might disagree with Computer Scientists on the true

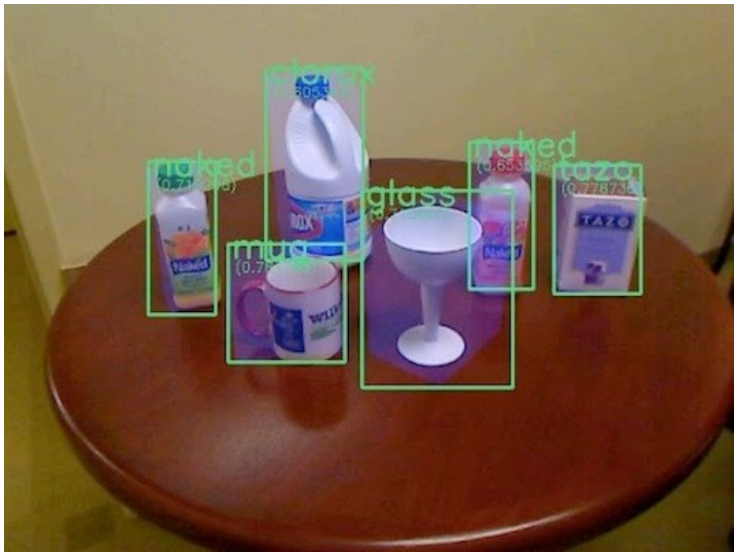


Picture credit:
https://twitter.com/deeplearn007/status/952833643386370

ML for Computer Vision

- Face Recognition
- Object Classification
- Scene Segmentation





Data, data, data!

Peter Norvig – “The Unreasonable Effectiveness of Data” (IEEE Intelligent Systems, 2009)

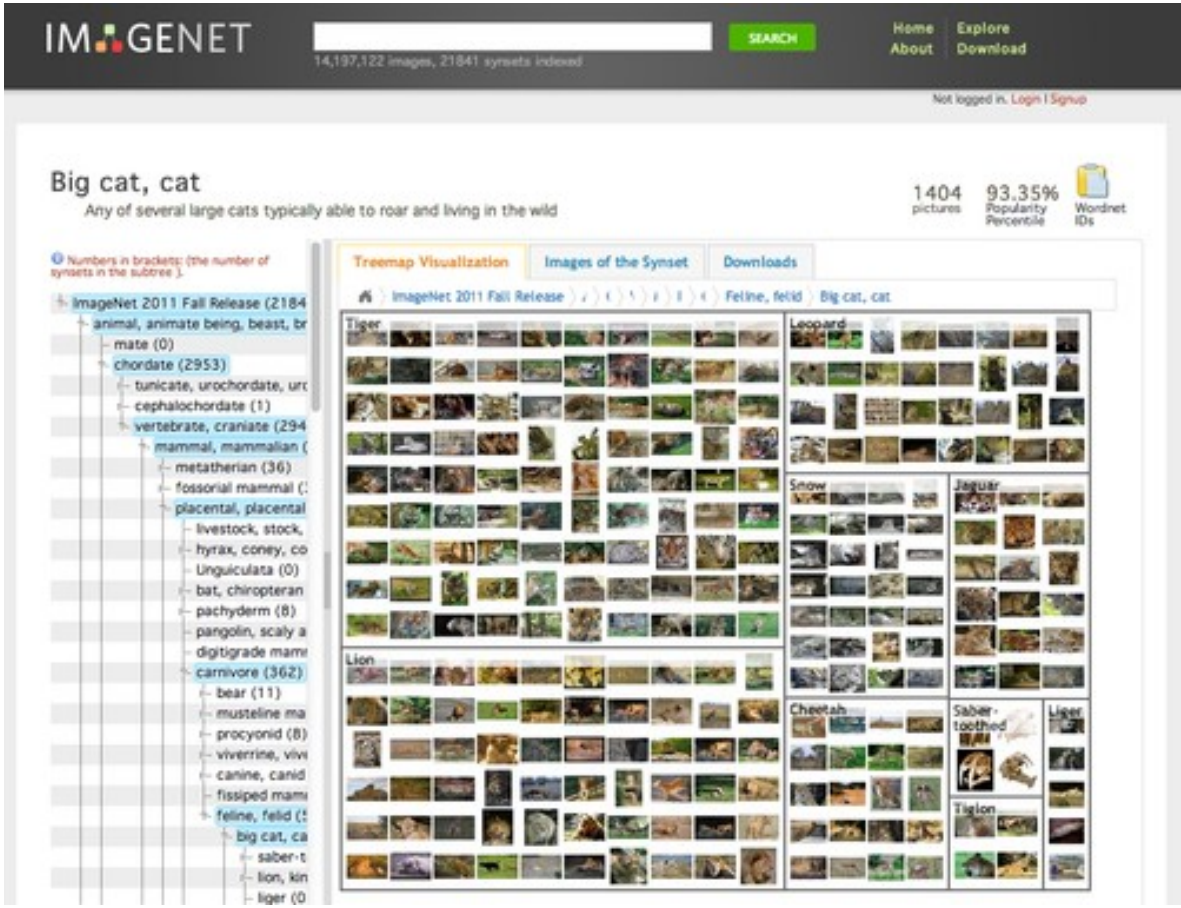
- “… invariably, simple models and a lot of data trump more elaborate models based on less data”

[Norvig is Director of Research @ Google, and a Brown APMA alum!]

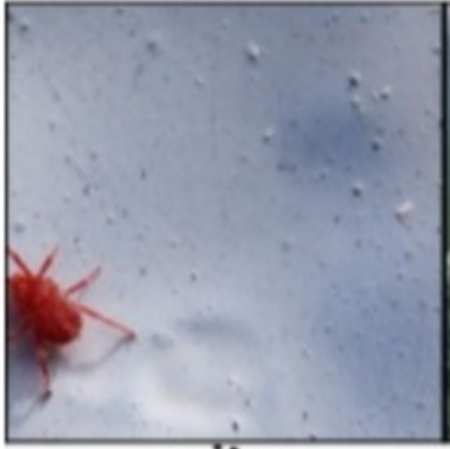
ImageNet

- Images for nouns in WordNet
- 1000 classes
- 1.2mil images
- 100k test

- Top 5 error



imagenet



mite

mite
black widow
cockroach
tick
starfish

container ship

container ship
lifeboat
amphibian
fireboat
drilling platform

motor scooter

motor scooter
go-kart
moped
bumper car
golfcart

leopard

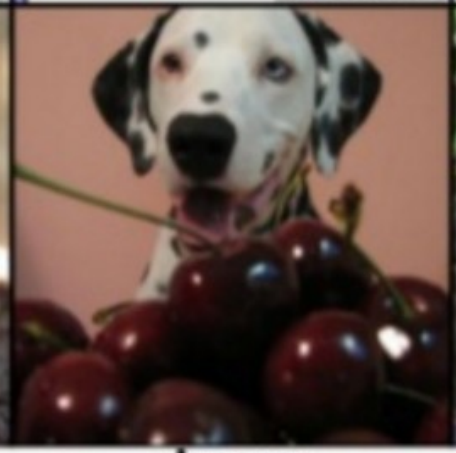
leopard
jaguar
cheetah
snow leopard
Egyptian cat

grille

convertible
grille
pickup
beach wagon
fire engine

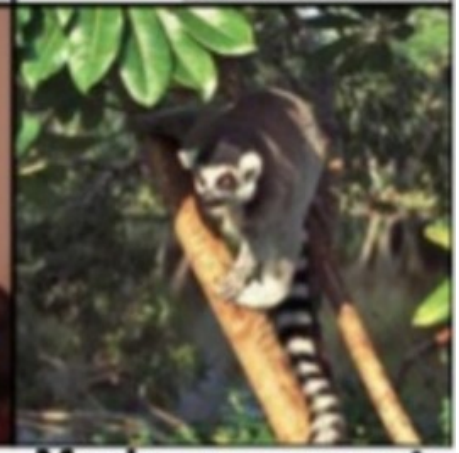
mushroom

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



cherry

dalmatian
grape
elderberry
ffordshire bullterrier
currant

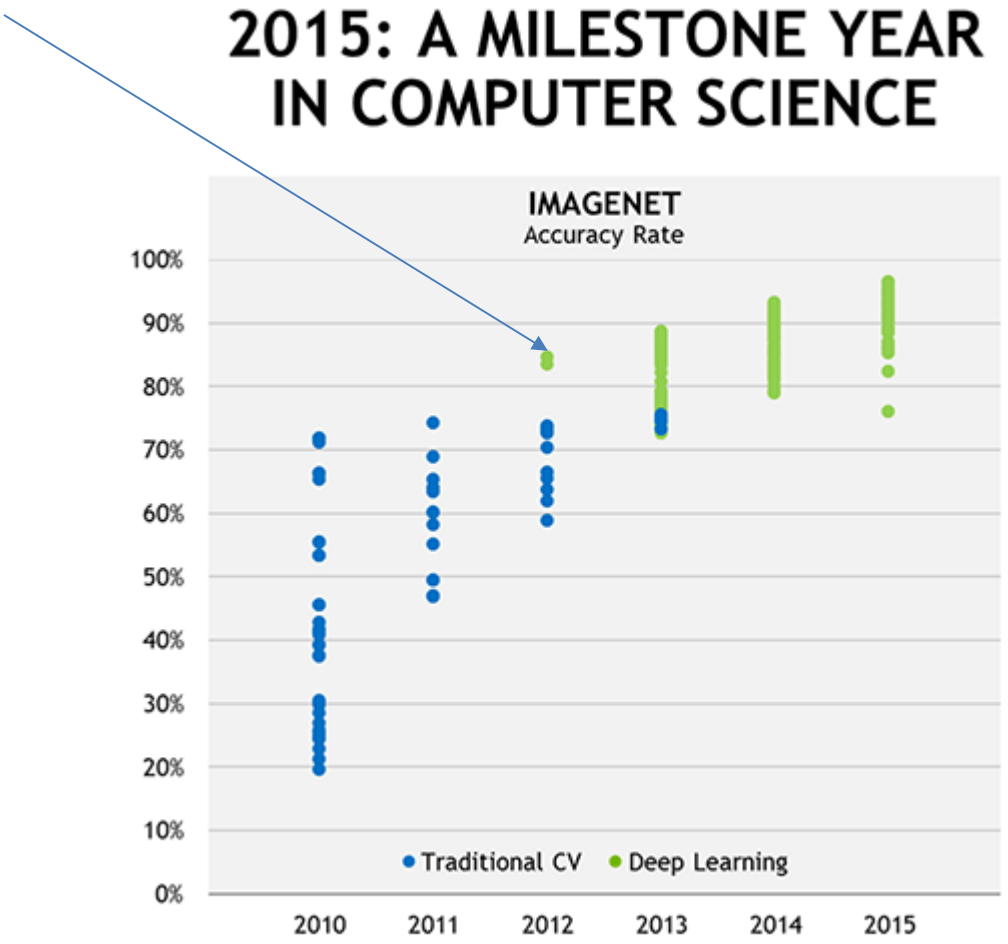


Madagascar cat

squirrel monkey
spider monkey
titi
indri
howler monkey

ImageNet Competition

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



Machine Learning Problems

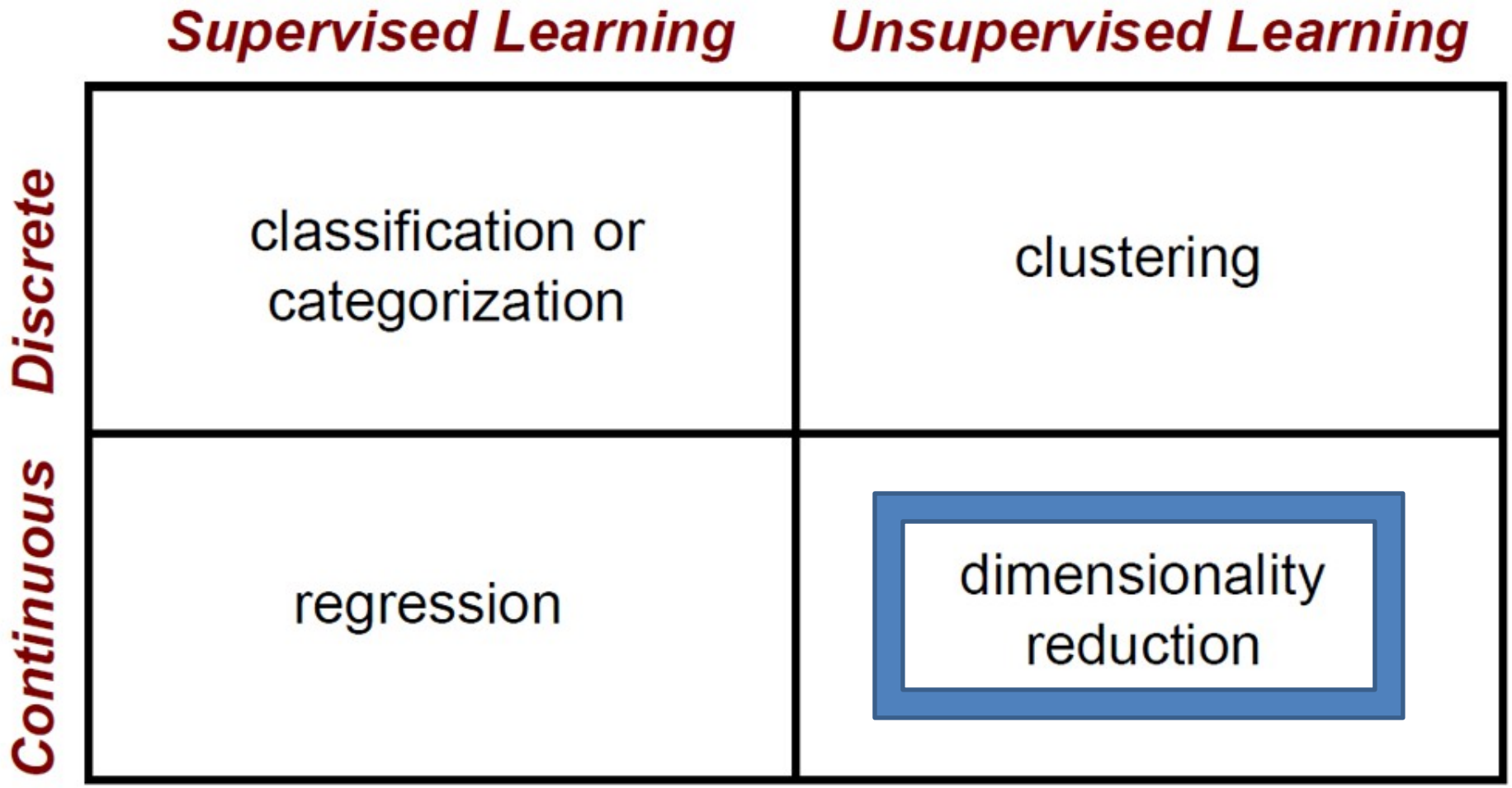
	Supervised Learning	*Unsupervised Learning*
Discrete	classification or categorization	clustering
Continuous	regression	dimensionality reduction

... But not exactly

Supervised vs Unsupervised

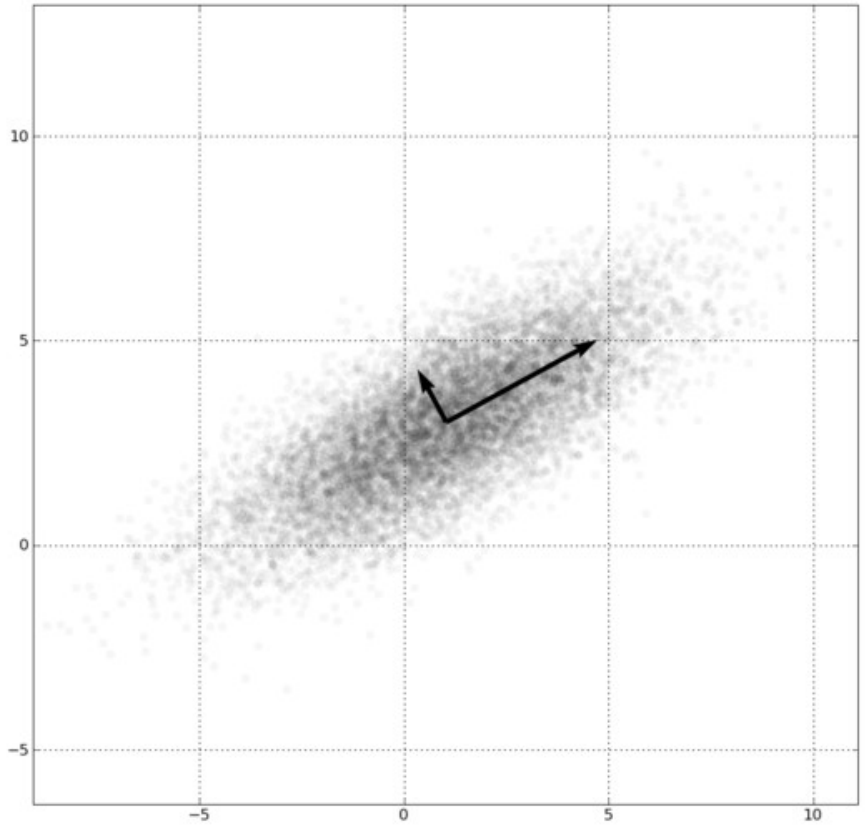
- Supervised
  - Input: labelled training data
  - finds relationship between input and output to predict a label/value for new data
- Unsupervised
  - Input: data
  - finds meaningful pattern in the data
- Semi-supervised?

Machine Learning Problems

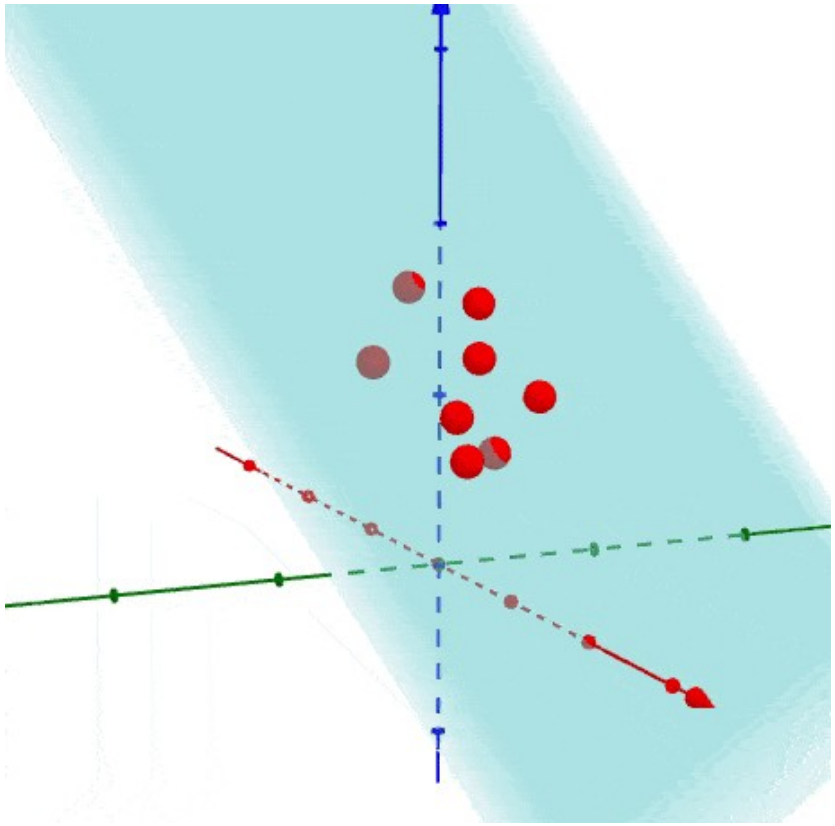


Dimensionality Reduction

- **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.



PCA



PCA pseudocode

1) Compute data mean.

for i in feature dimensions:

    inMean(i) = np.mean(featureVector[i,:])

2) Optional(but not really optional): align data so that mean = 0 and variance = 1

3) Compute covariance matrix: (square matrix with size FeatSizeXFeatSize

    Covar = np.cov(FeatVec[0,:], FeatVec[1,:] , …)

4) Compute eigen value decomposition

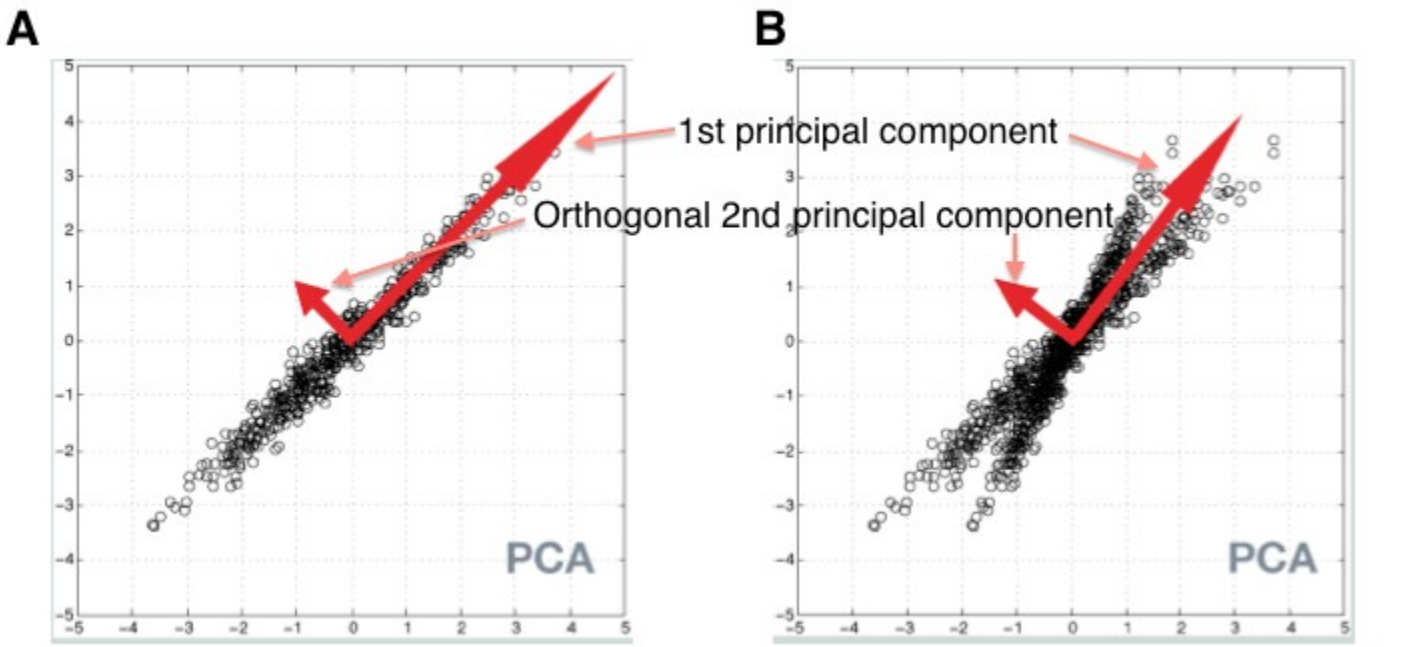
    eigVal, eigVec = np.linalg.eig(cov_mat)

5) Sort eigVec by decreasing order of eigVal

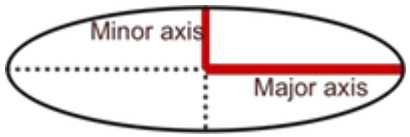
6) Form a transformation matrix ‘w’ by keeping the first K eigVec such that it’s a matrix of size: d x K

7) Transform the data using:

PCA



(Figure adapted from C. Beckmann, Oxford FMRIB)



PCA

- Using sklearn:

```
from sklearn.decomposition import PCA
sklearn_pca = sklearnPCA(k) # k=nbr dim to keep
Sklearn_tranf = sklearn.pca.fit_transform(data)
```

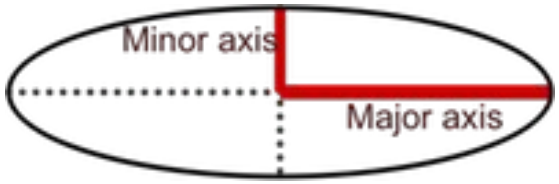
- Eigen Decomposition trouble:

$\Sigma v = \lambda v$

Σ=Covariance matrix

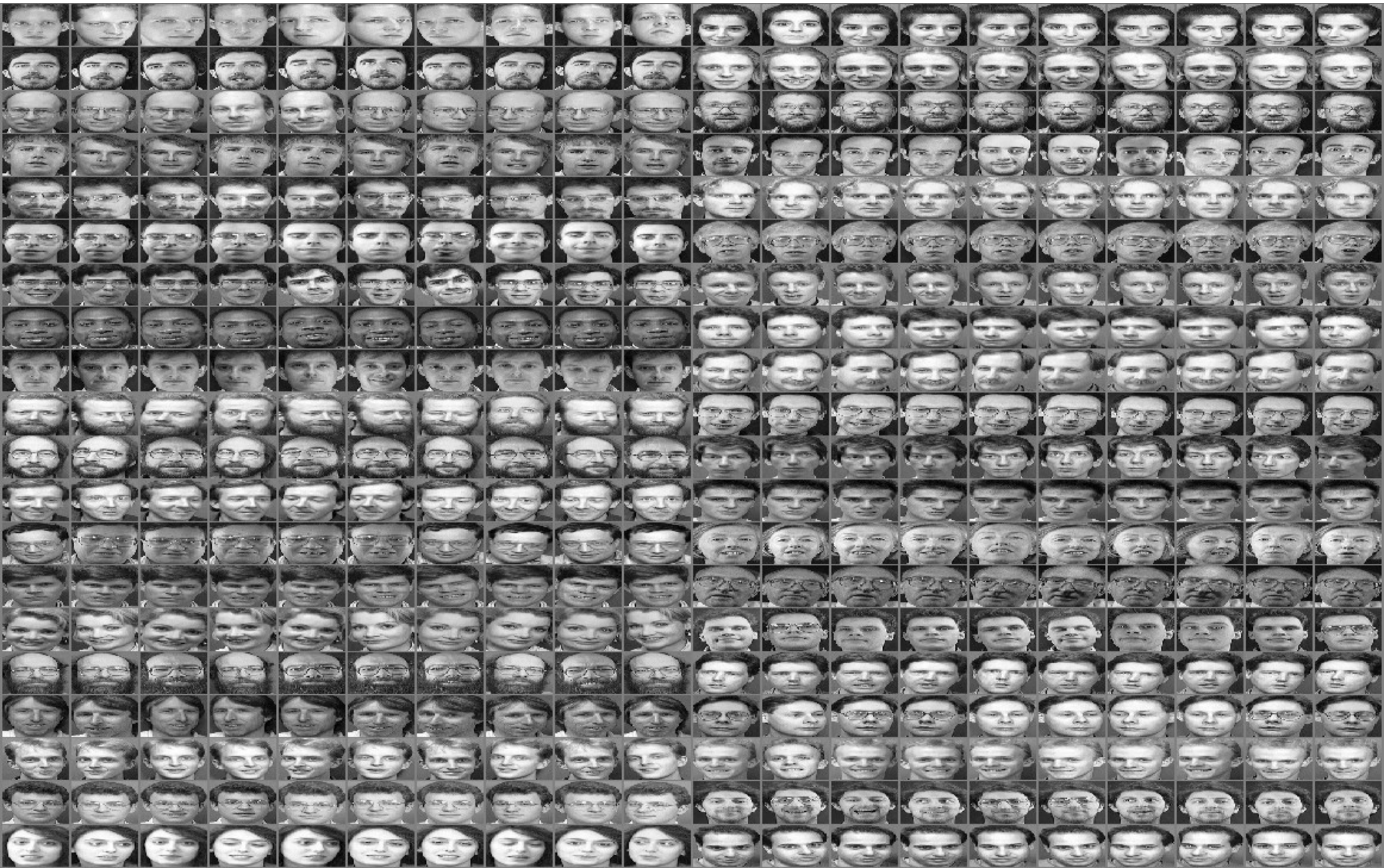
v=Eigenvector

λ=Eigenvalue



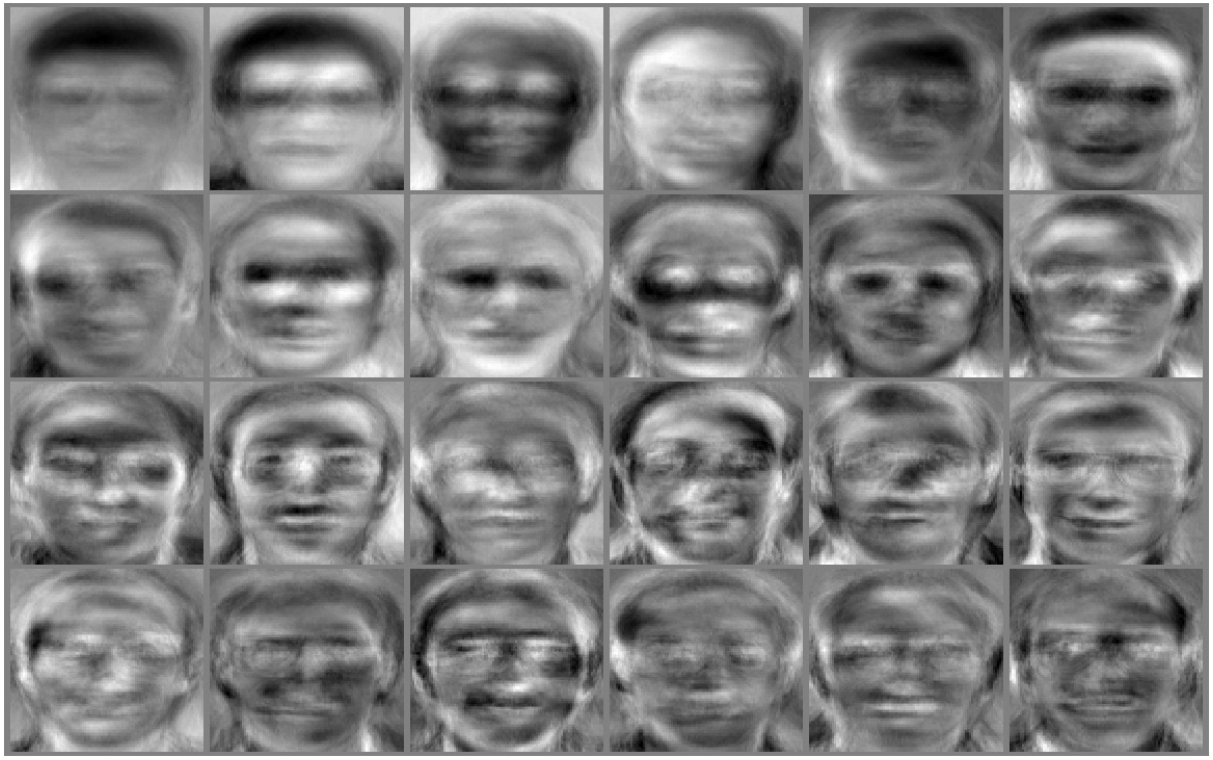
Eigenfaces

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



Eigenfaces

Mean face

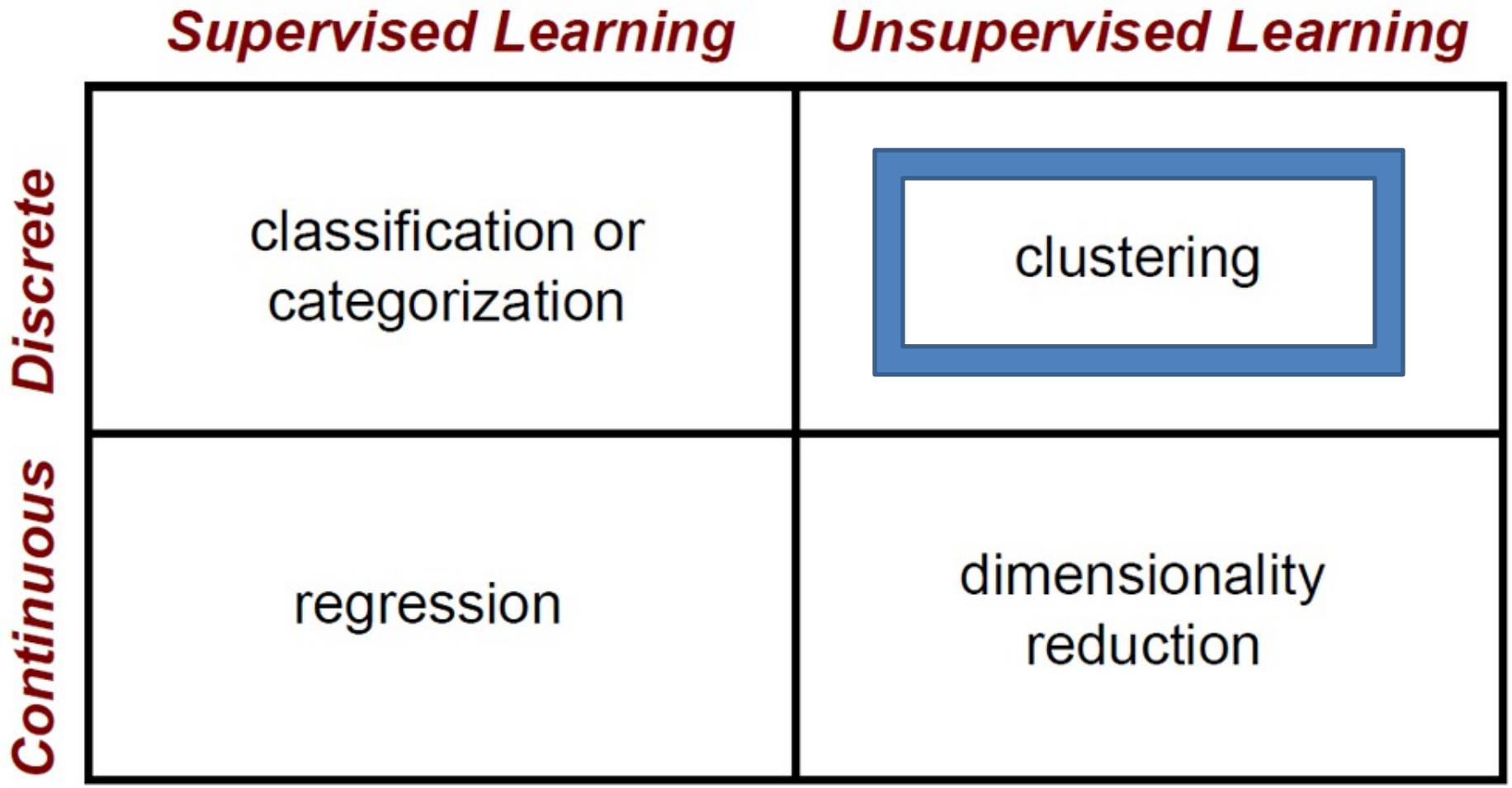


Basis of variance (eigenvectors)

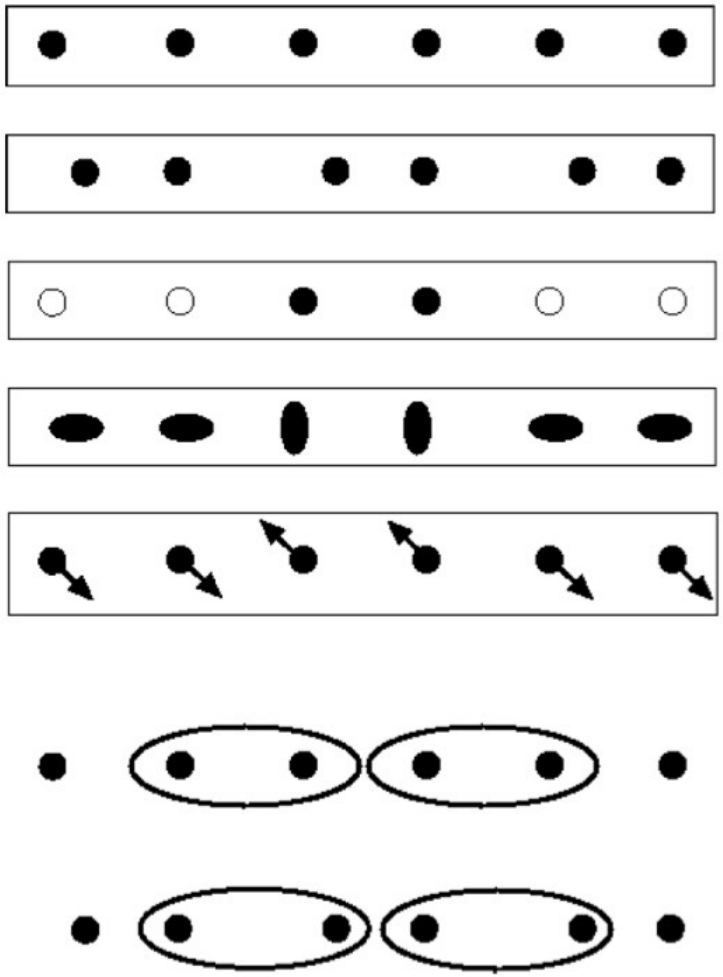
M. Turk; A. Pentland (1991). "Face recognition using eigenfaces" (PDF). Proc. IEEE Conference on Computer Vision and Pattern Recognition. pp. 586–591.

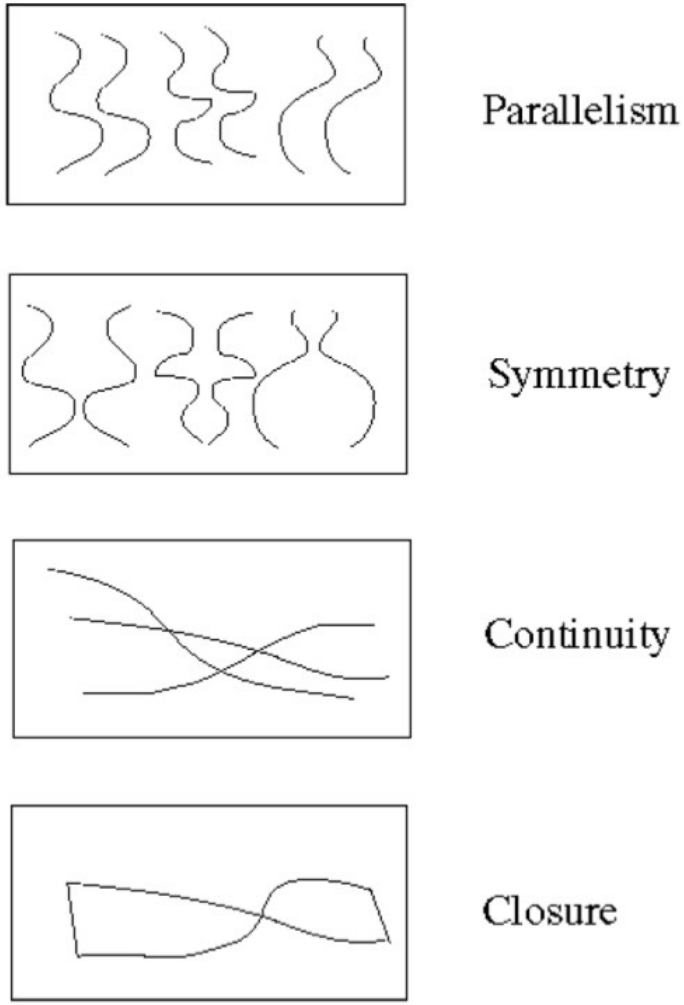
R.P.W.

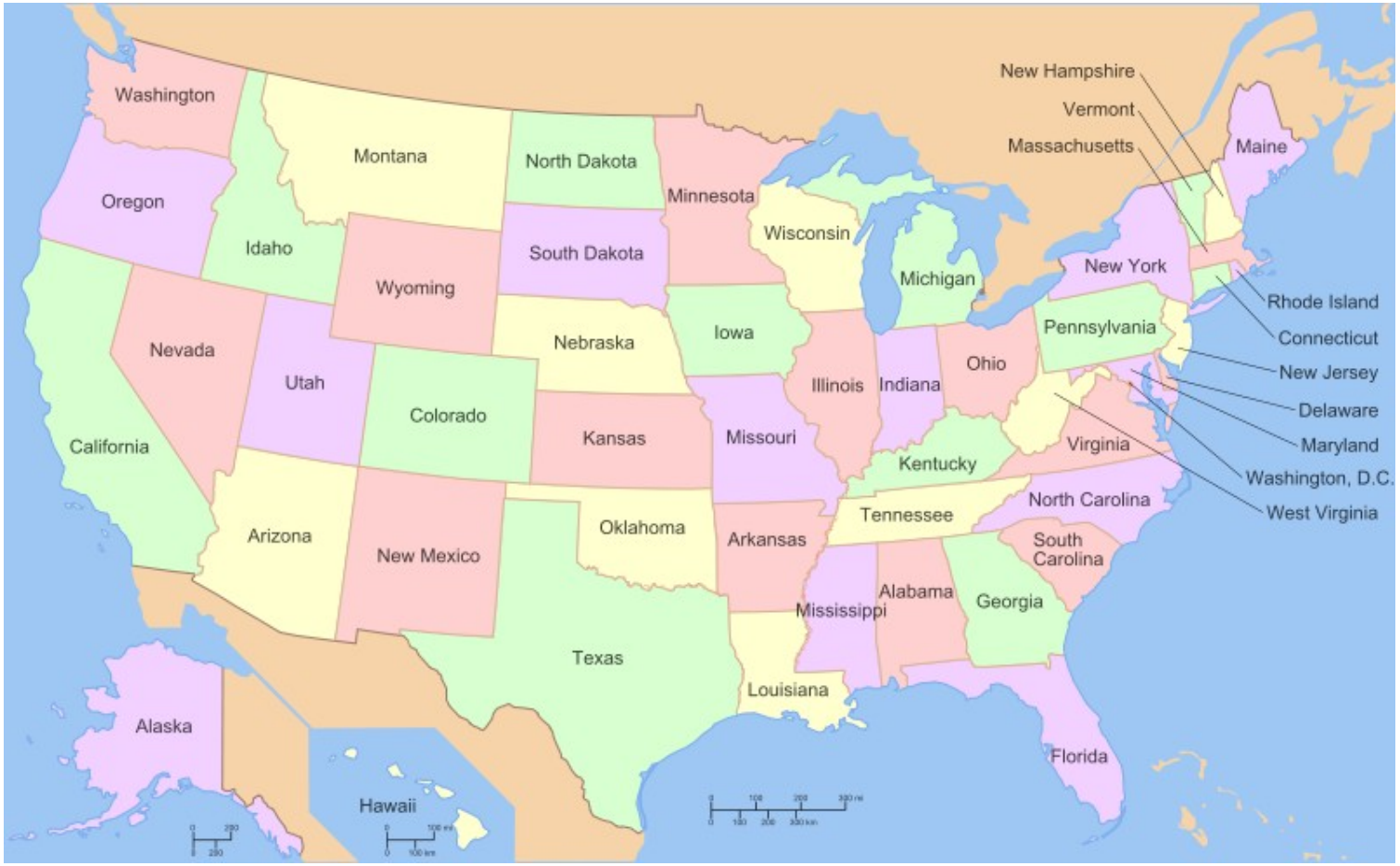
Machine Learning Problems



Unsupervised learning Gestalt factors






Washington
Oregon
California
Idaho
Nevada
Montana
Wyoming
Utah
Arizona
Colorado
New Mexico
North Dakota
South Dakota
Nebraska
Kansas
Oklahoma
Texas
Minnesota
Iowa
Missouri
Arkansas
Louisiana
Wisconsin
Illinois
Michigan
Indiana
Ohio
Kentucky
Tennessee
Mississippi
Alabama
Georgia
Florida
South Carolina
North Carolina
Virginia
West Virginia
Pennsylvania
New York
Maryland
Delaware
New Jersey
Connecticut
Rhode Island
Massachusetts
Vermont
New Hampshire
Maine
Washington, D.C.
Alaska
Hawaii

The United States *redrawn as* Fifty States *with* Equal Population


Yukon
Whitehorse
Northwest Territories
Yellowknife
Nunavut
Iqaluit
Newfoundland and Labrador
St John's
British Columbia
Victoria
Alberta
Edmonton
Saskatchewan
Regina
Manitoba
Winnipeg
Ontario
Toronto
Quebec
Quebec
Ottawa
Charlottetown
Fredericton
Halifax
Prince Edward Island
Nova Scotia
New Brunswick
CANADA – Political
International border
Provincial/territorial border
Alberta Province/territory
Ottawa National capital
Regina Provincial/territorial capital
Scale
0 250 500 750 1000 km
0 250 500 750 miles

Source: alasdairgunn

Clustering example: image segmentation

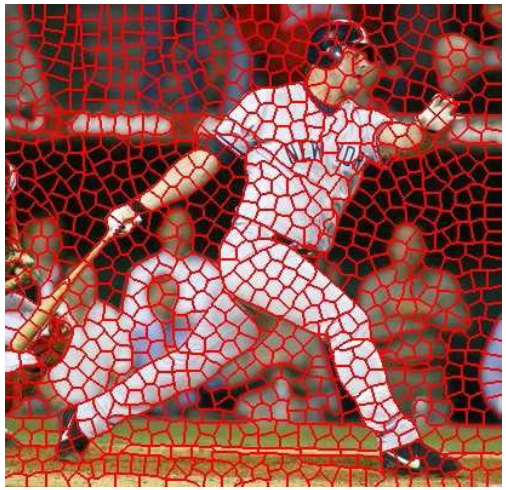
Goal: Break up the image into meaningful or perceptually similar regions

Segmentation for feature support or efficiency

50x50 Patch

50x50 Patch

[Felzenszwalb and Huttenlocher 2004]

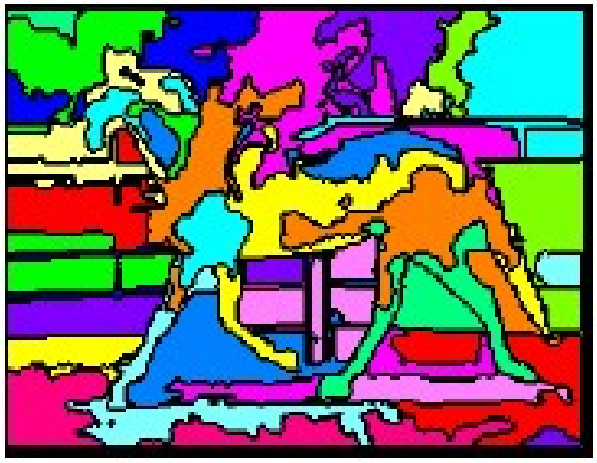


Superpixels!

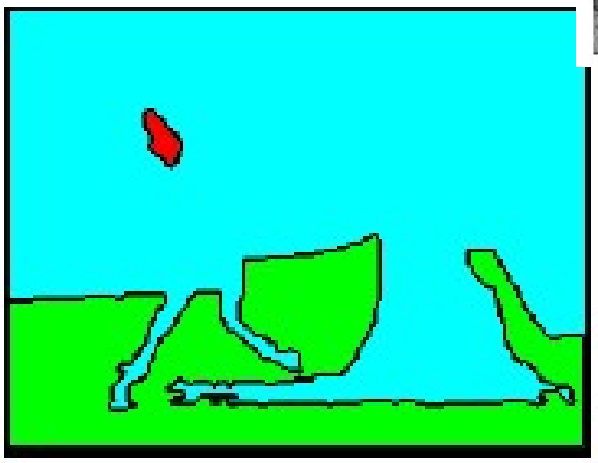
[Shi and Malik 2001]

[Hoiem et al. 2005, Mori 2005]

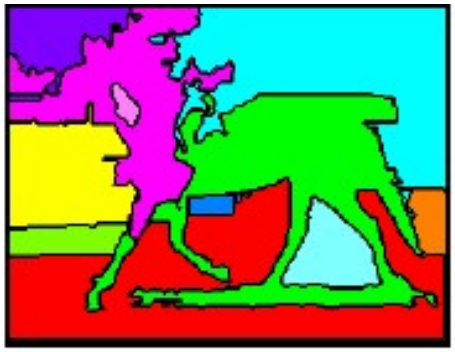
Types of Segmentations

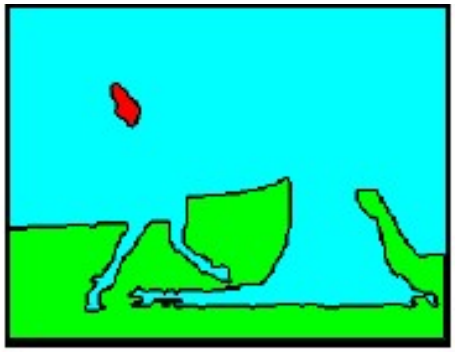


Oversegmentation



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?

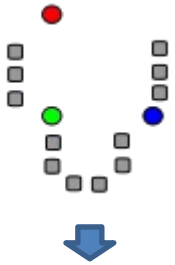
- **Summarizing data**
  - Look at large amounts of data
  - Patch-based compression or denoising
  - Represent a large continuous vector with the cluster number
- **Counting**
  - Histograms of texture, color, SIFT vectors
- **Segmentation**
  - Separate the image into different regions
- **Prediction**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

How do we cluster?

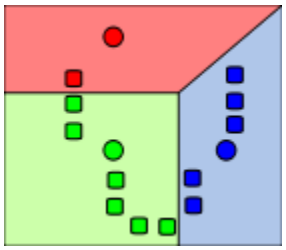
- K-means
  - Iteratively re-assign points to the nearest cluster center
- Agglomerative clustering
  - Start with each point as its own cluster and iteratively merge the closest clusters
- Mean-shift clustering
  - Estimate modes of pdf
- Spectral clustering
  - Split the nodes in a graph based on assigned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

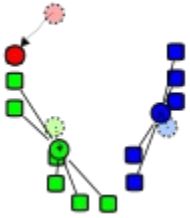
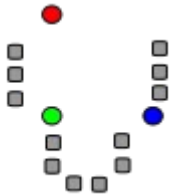


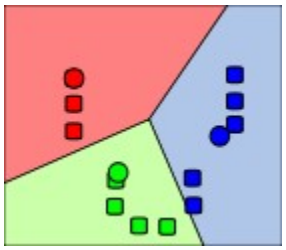
Illustration: http://en.wikipedia.org/wiki/K-means_clustering

K-means algorithm

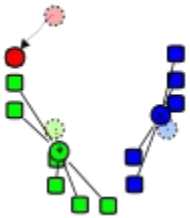
1. Randomly select K centers



2. Assign each point to nearest center



3. Compute new center (mean) for each cluster



Back to 2

Illustration: http://en.wikipedia.org/wiki/K-means_clustering

K-means

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

2. Assign each point to the closest center

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

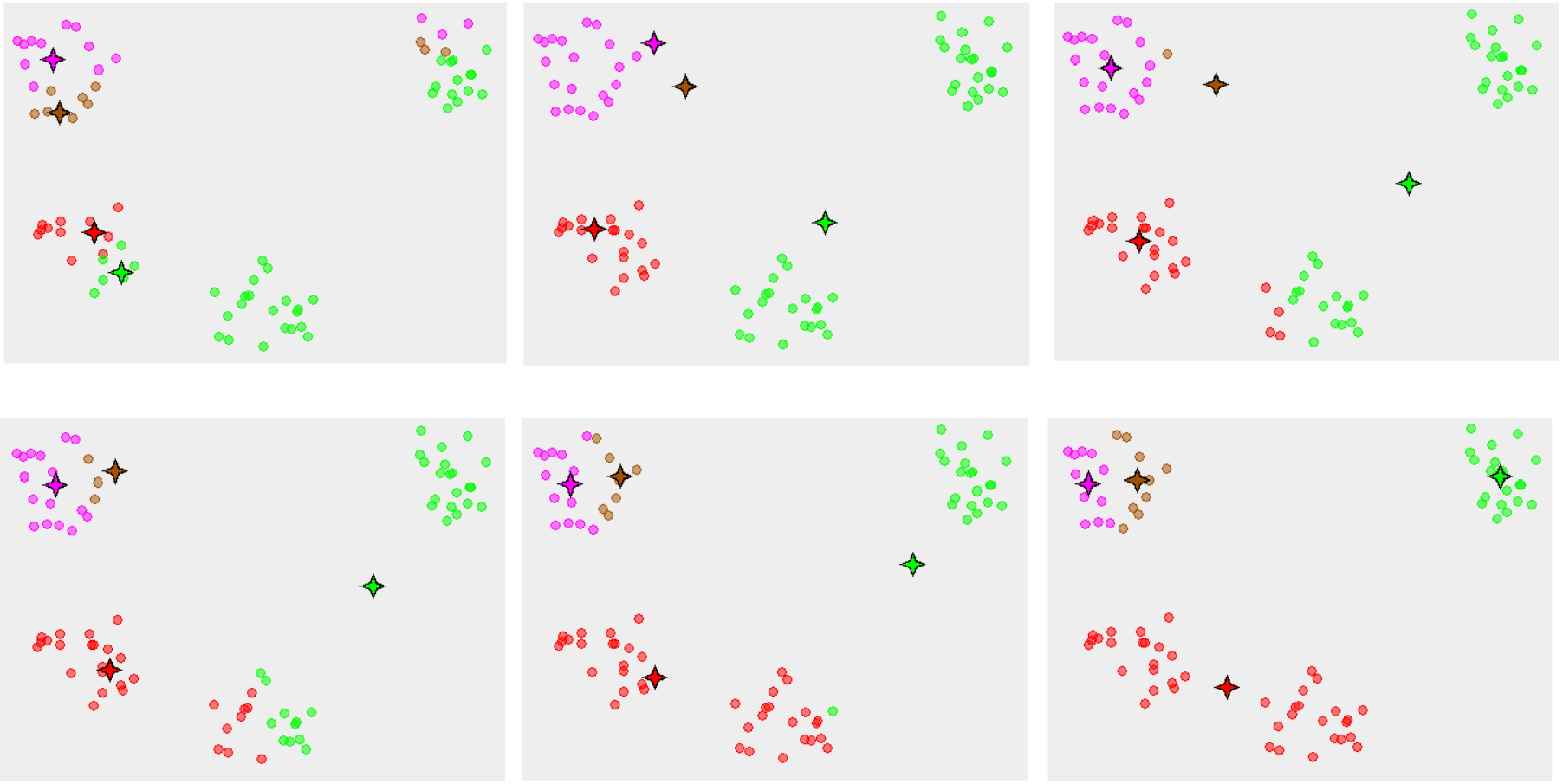
3. Update cluster centers as the mean of the points

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

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

Slide: Derek Hoiem

K-means convergence



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*

K-means clustering using intensity or color

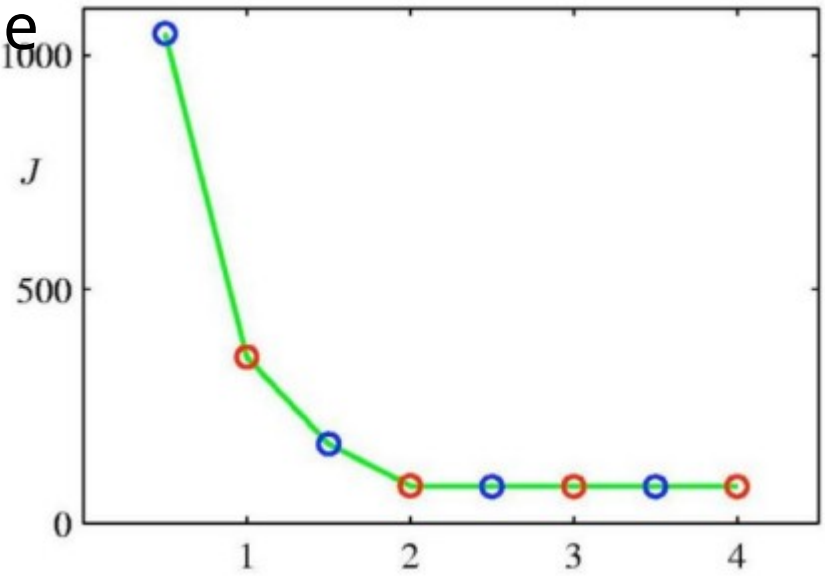
Image

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.
- Plot k-means objective vs. K, and pick k at the elbow (in this example its k=2)
  - J here is the total sum of the distances of all points to their associated centers

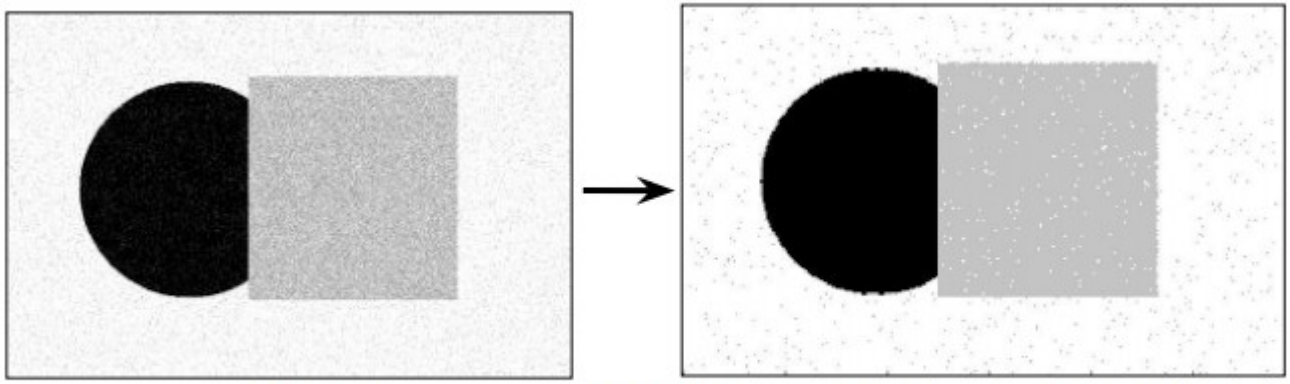


Distance measures and termination

- Choice of “distance” measures”
  - Euclidean (commonly used)
  - Cosine similarity
  - Non-linear (called kernel k-means)

- Termination criteria:
  - Maximum number of iterations reaches
  - No change during the re-assignment step (convergence)

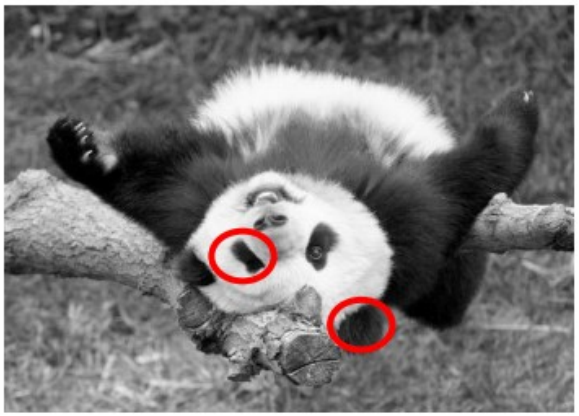
Smoothing cluster assignments

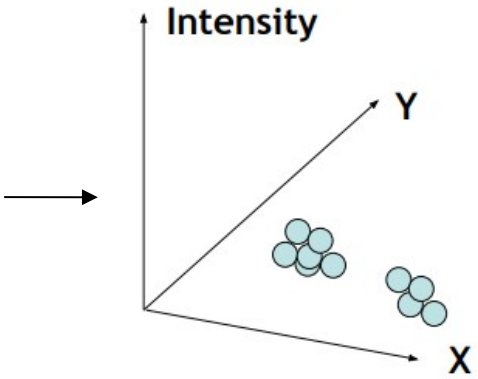


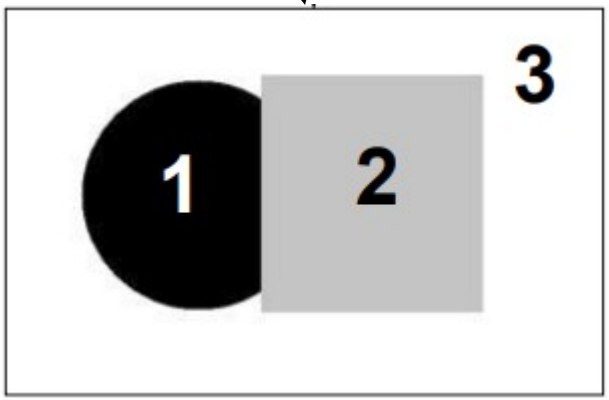
Original

Labeled by cluster center's intensity

Group pixels based on intensity + position







How to initialize the clusters?

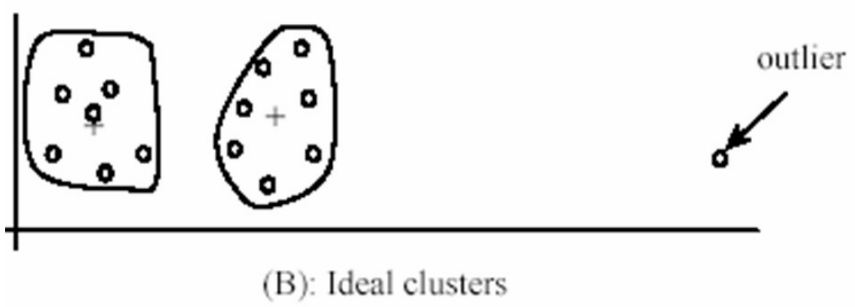
- k-means++ initialization
  - Make the initial cluster centers span the space

1. Choose one center uniformly at random from all data points.
2. For each point *x*, compute the distance D(*x*) between *x* and the nearest center that has already been chosen.
3. Choose one new data point at random as a new center, using a weighted probability distribution where a point *x* is chosen with probability proportional to $D(x)^2$.
4. Repeat Steps 2 and 3 until *k* centers have been chosen.
5. Proceed using standard *k*-means clustering.

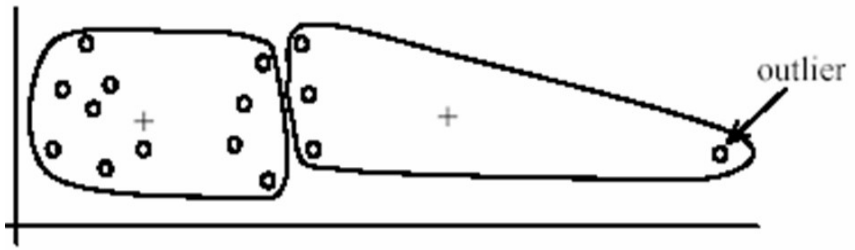
Wikipedia / Arthur and Vassilvitskii

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
  - Cluster features to build visual dictionaries



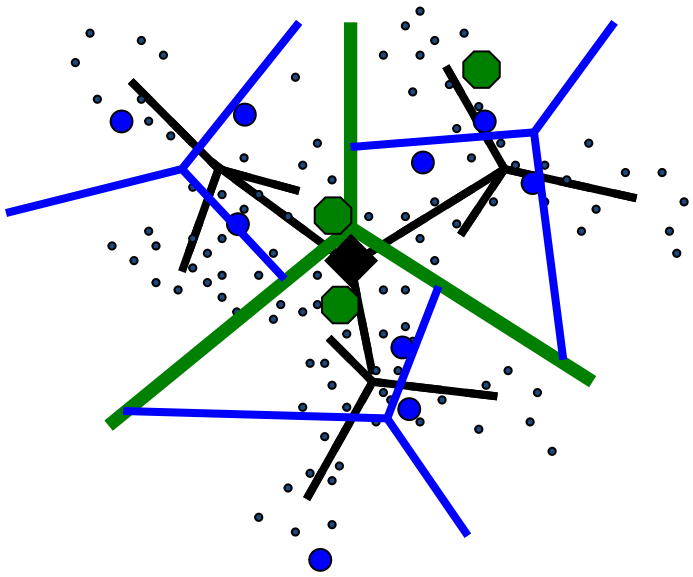
(B): Ideal clusters



Building Visual Dictionaries

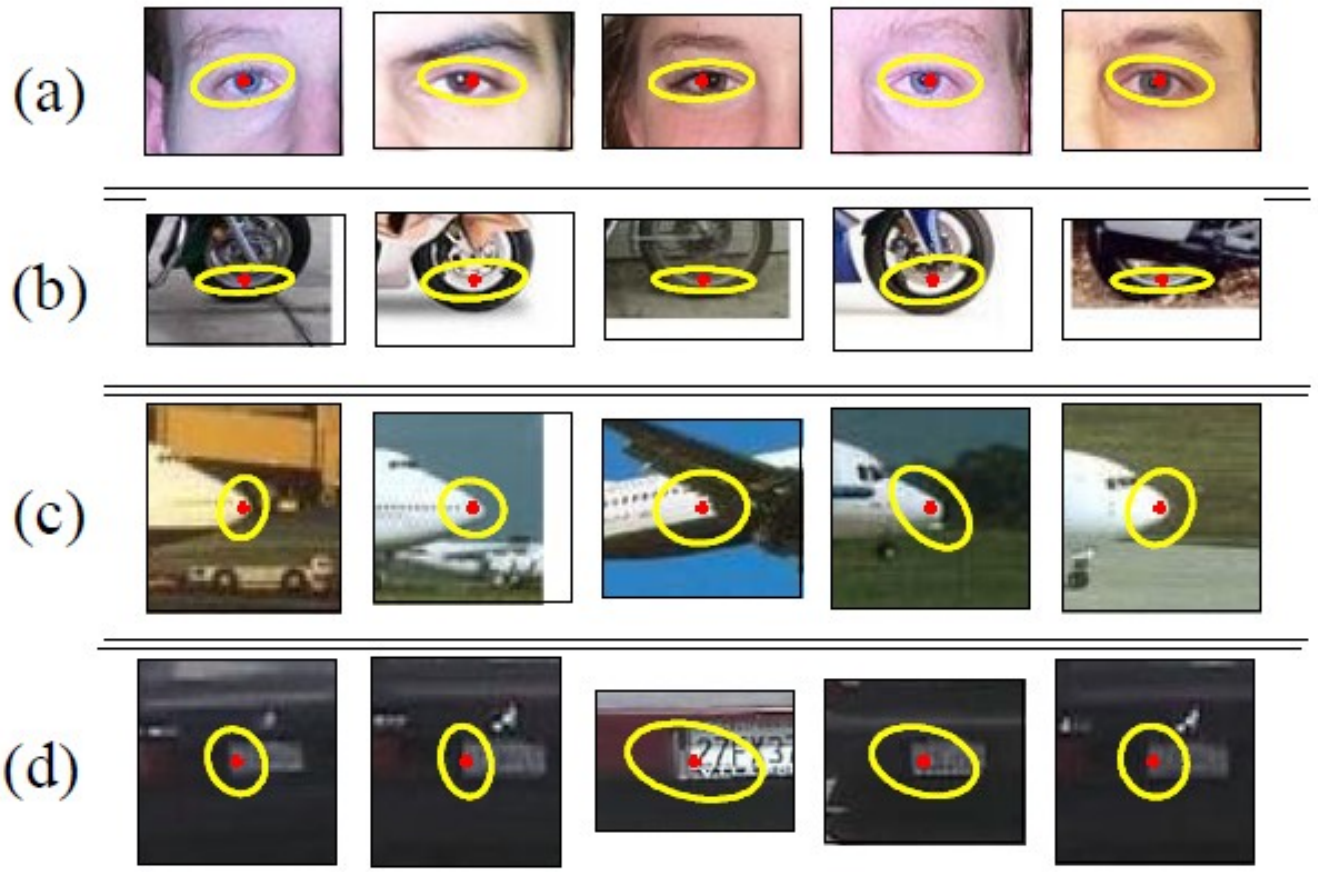
1. Sample features from a database
   - E.g., 128 dimensional SIFT vectors

2. Cluster to build dictionary
   - Cluster centers are the dictionary words



3. To match new features, assign to the nearest cluster to save rebuilding dictionary

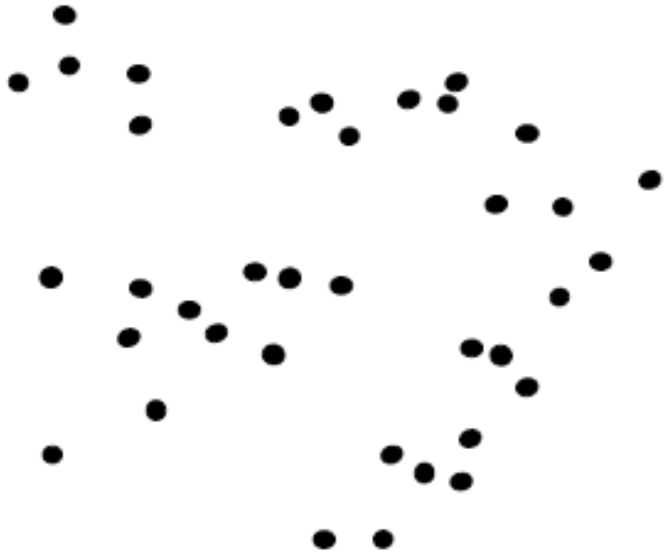
Examples of learned codewords



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

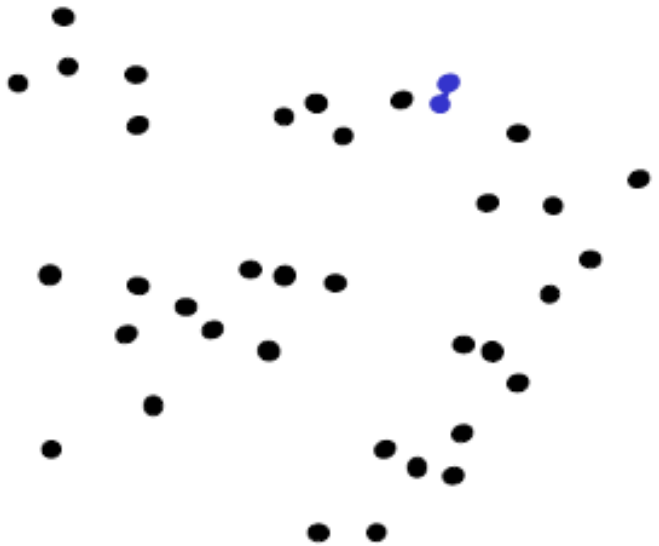
http://www.robots.ox.ac.uk/~vgg/publications/papers/sivic05b.pdf Sivic et al. ICCV 2005

Agglomerative clustering



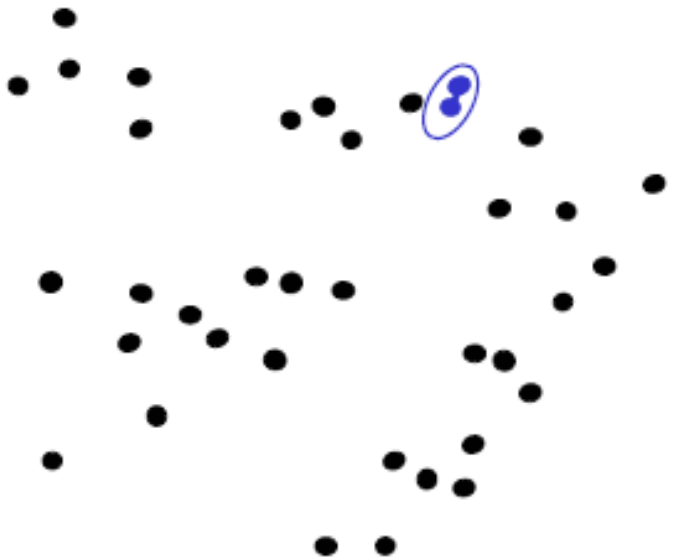
1. Say "Every point is its own cluster"

Agglomerative clustering



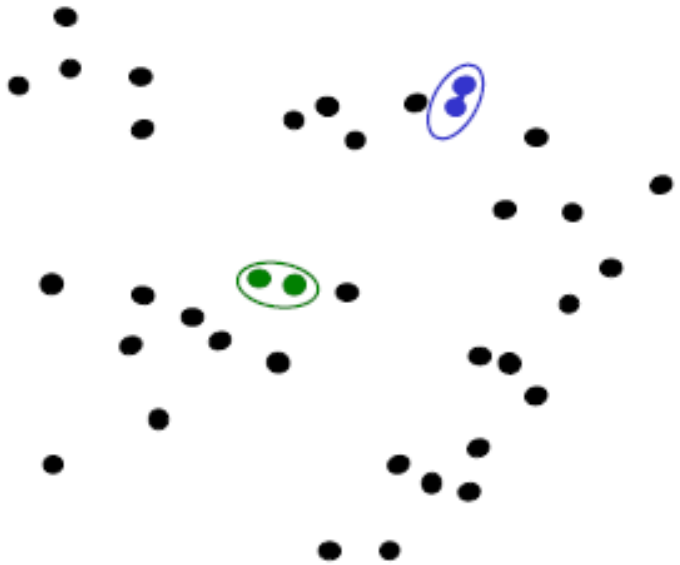
1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters

Agglomerative clustering



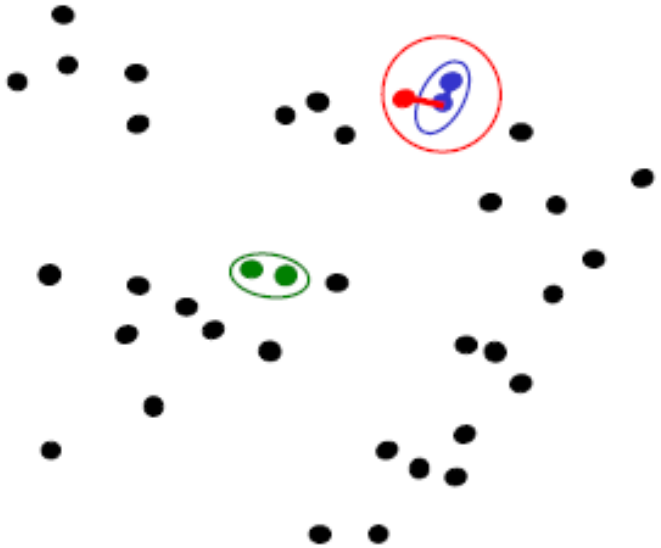
1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster

Agglomerative clustering

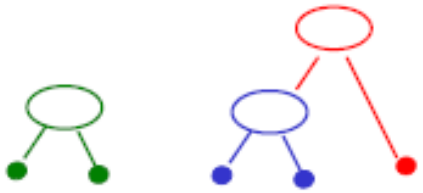


1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster
4. Repeat

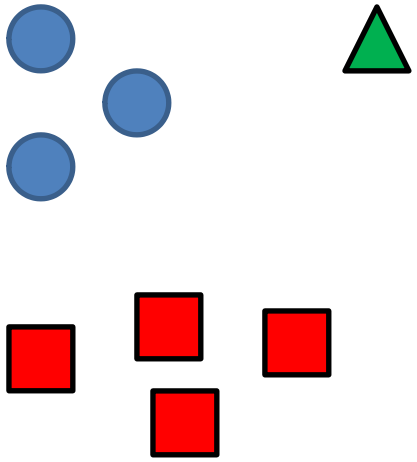
Agglomerative clustering



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

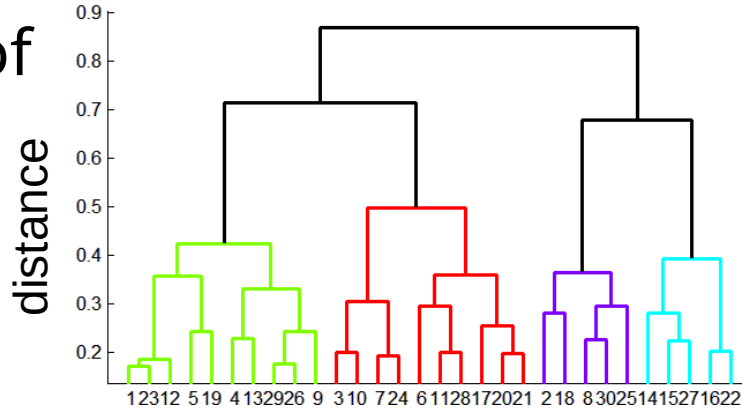


How to define cluster similarity?

- 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

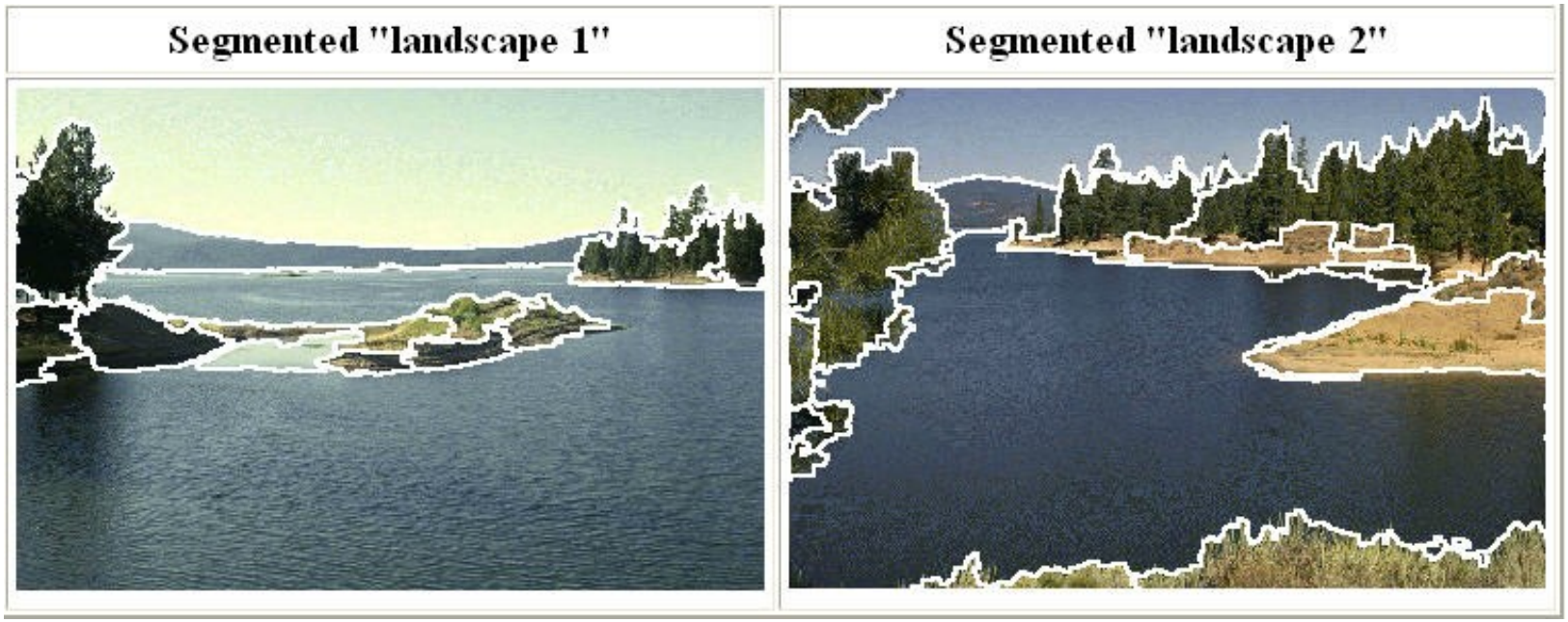
Bad

- May have imbalanced clusters
- Still have to choose number of clusters or threshold
- Need to use an “ultrametric” to get a meaningful hierarchy
- Can be very slow $O(n^2d + n^3)$

Mean shift segmentation

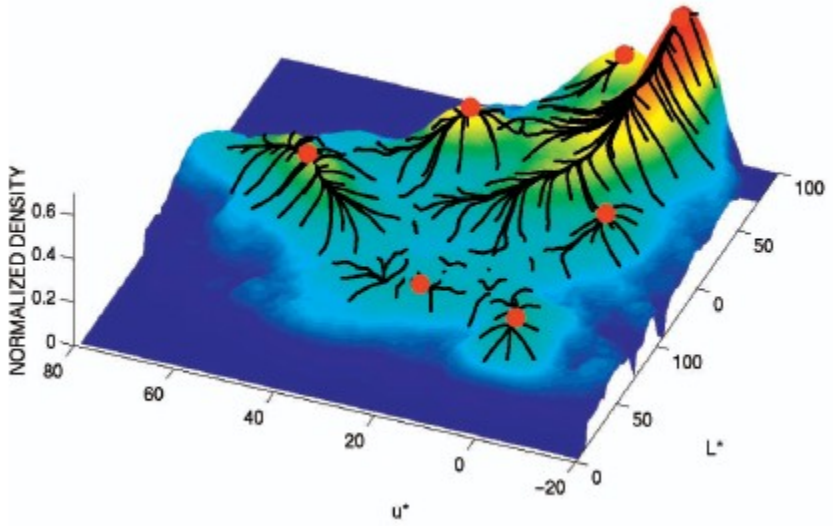
D. Comaniciu and P. Meer, Mean Shift: A Robust Approach toward Feature Space Analysis, PAMI 2002.

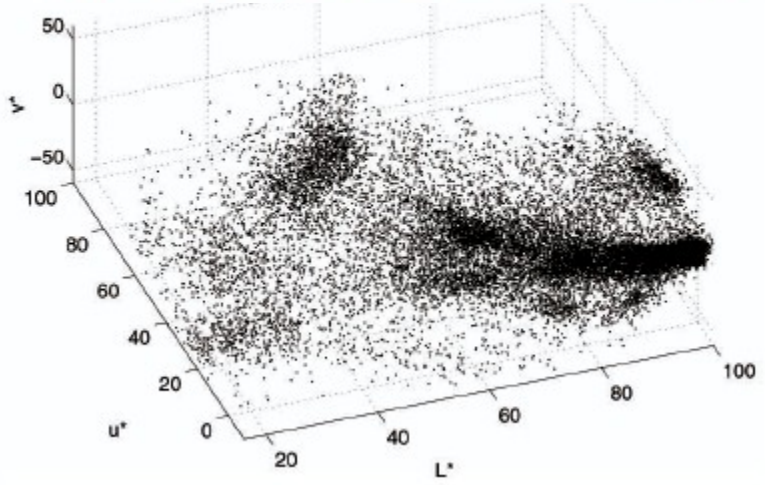
- Versatile technique for clustering-based segmentation



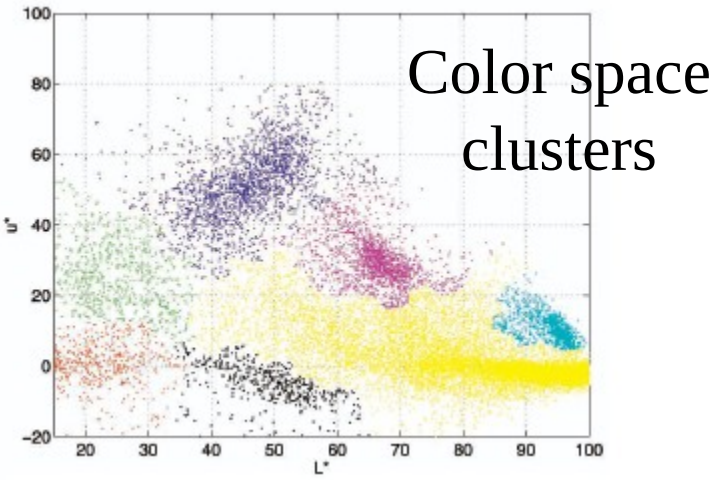
Mean shift algorithm

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





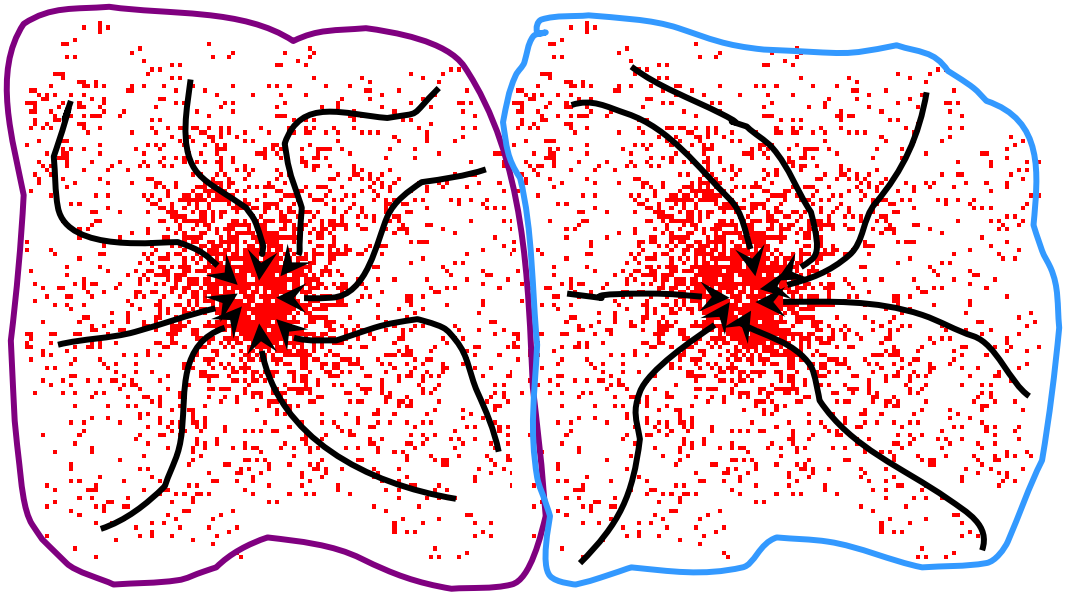
Color space



Color space clusters

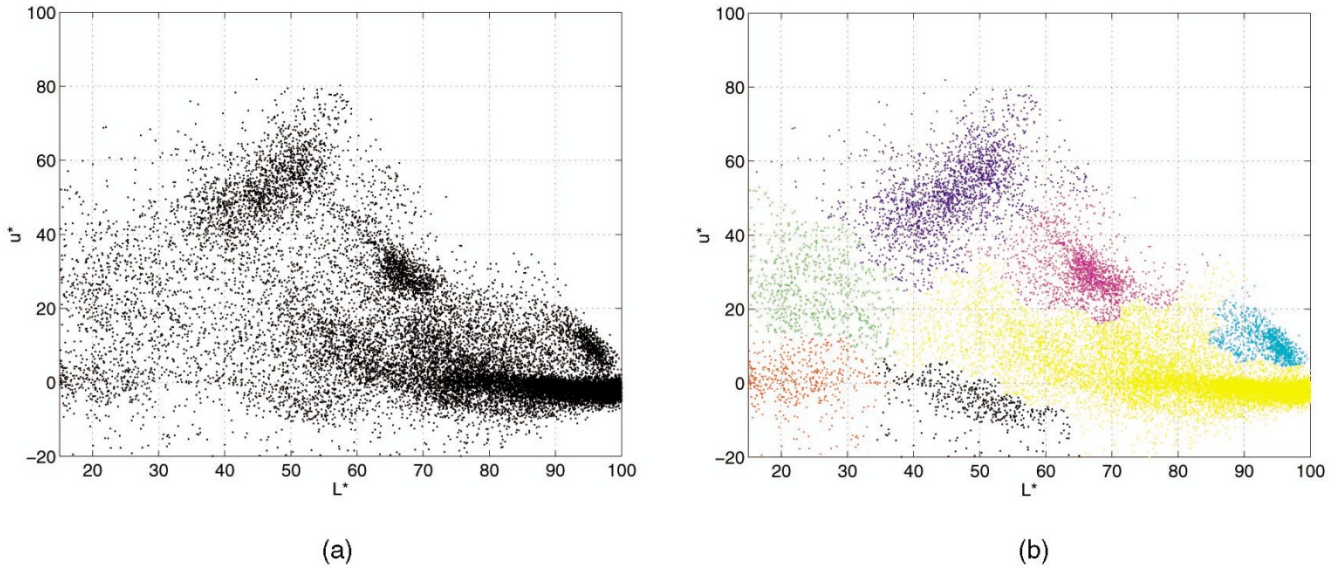
Attraction basin

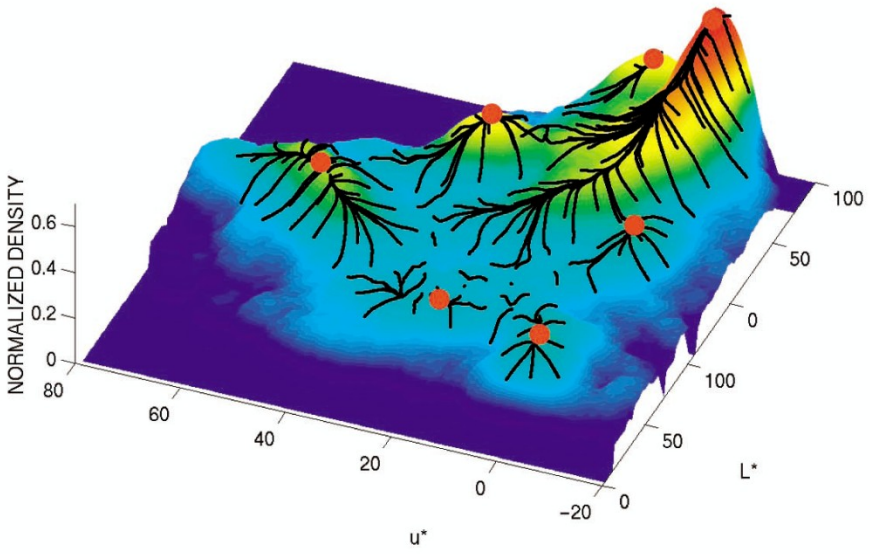
- Attraction basin: the region for which all trajectories lead to the same mode
- Cluster: all data points in the attraction basin of a mode



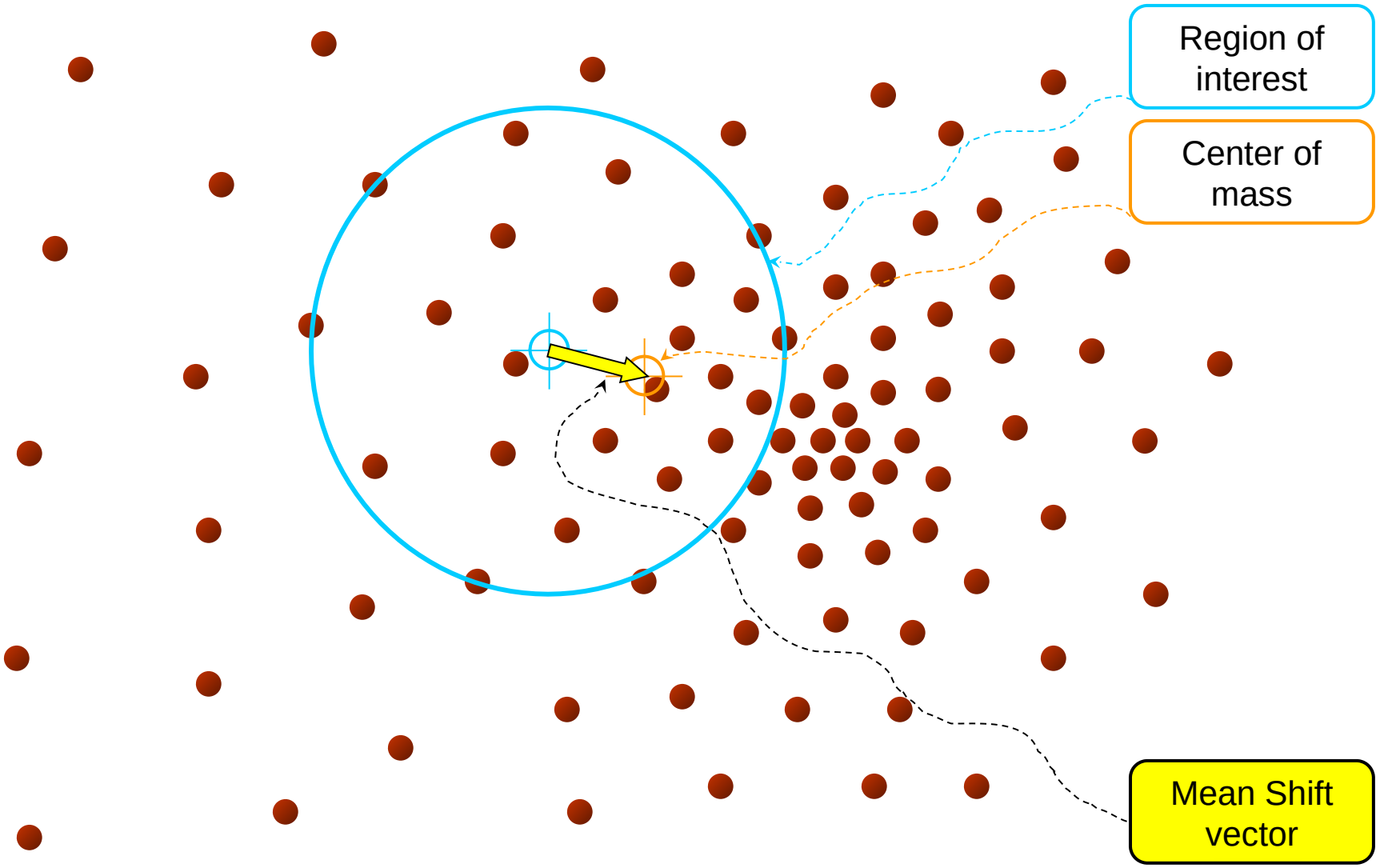
Slide by Y. Ukrainitz & B. Sarel

Attraction basin



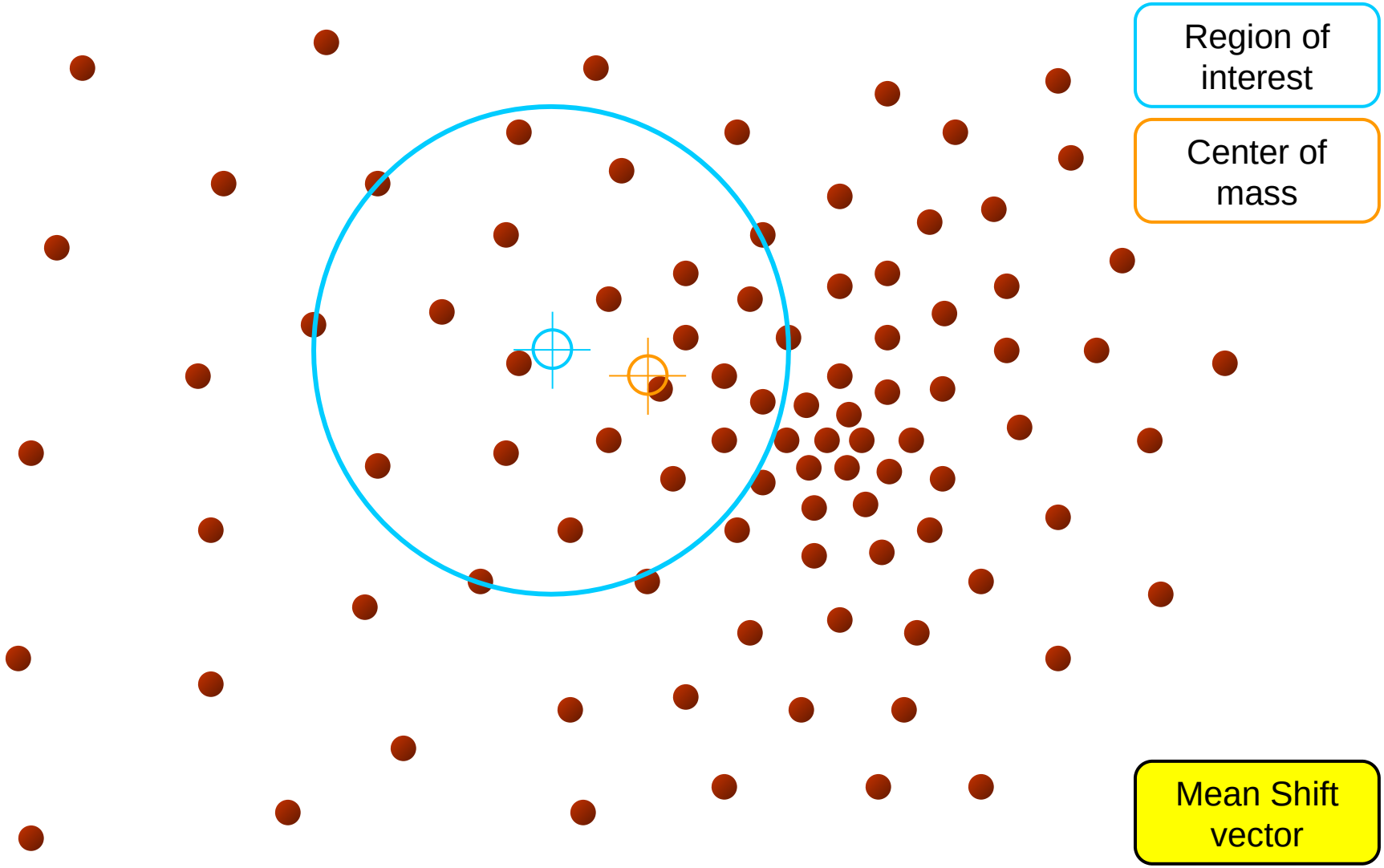


Mean shift



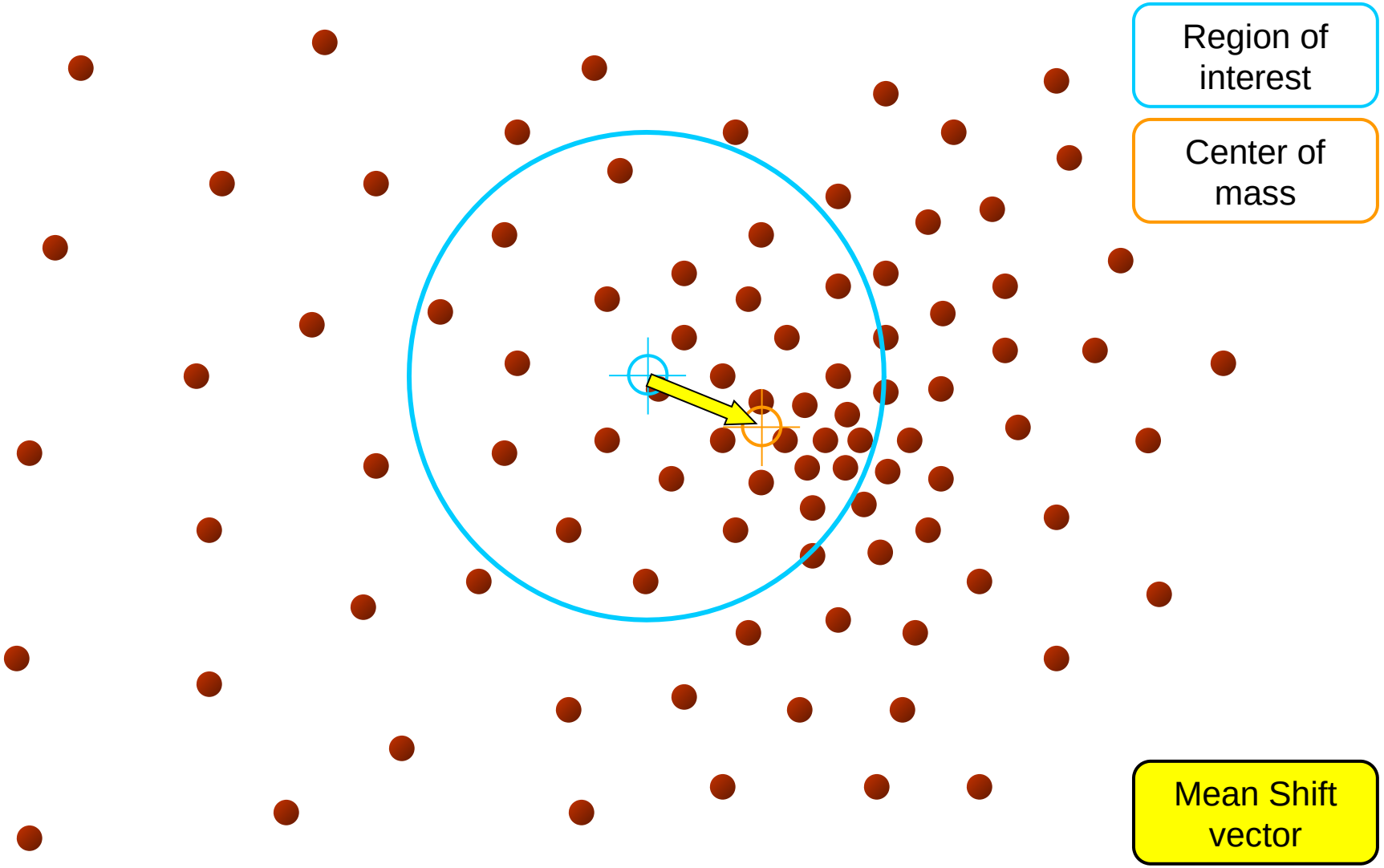
Slide by Y. Ukrainitz & B. Sarel

Mean shift



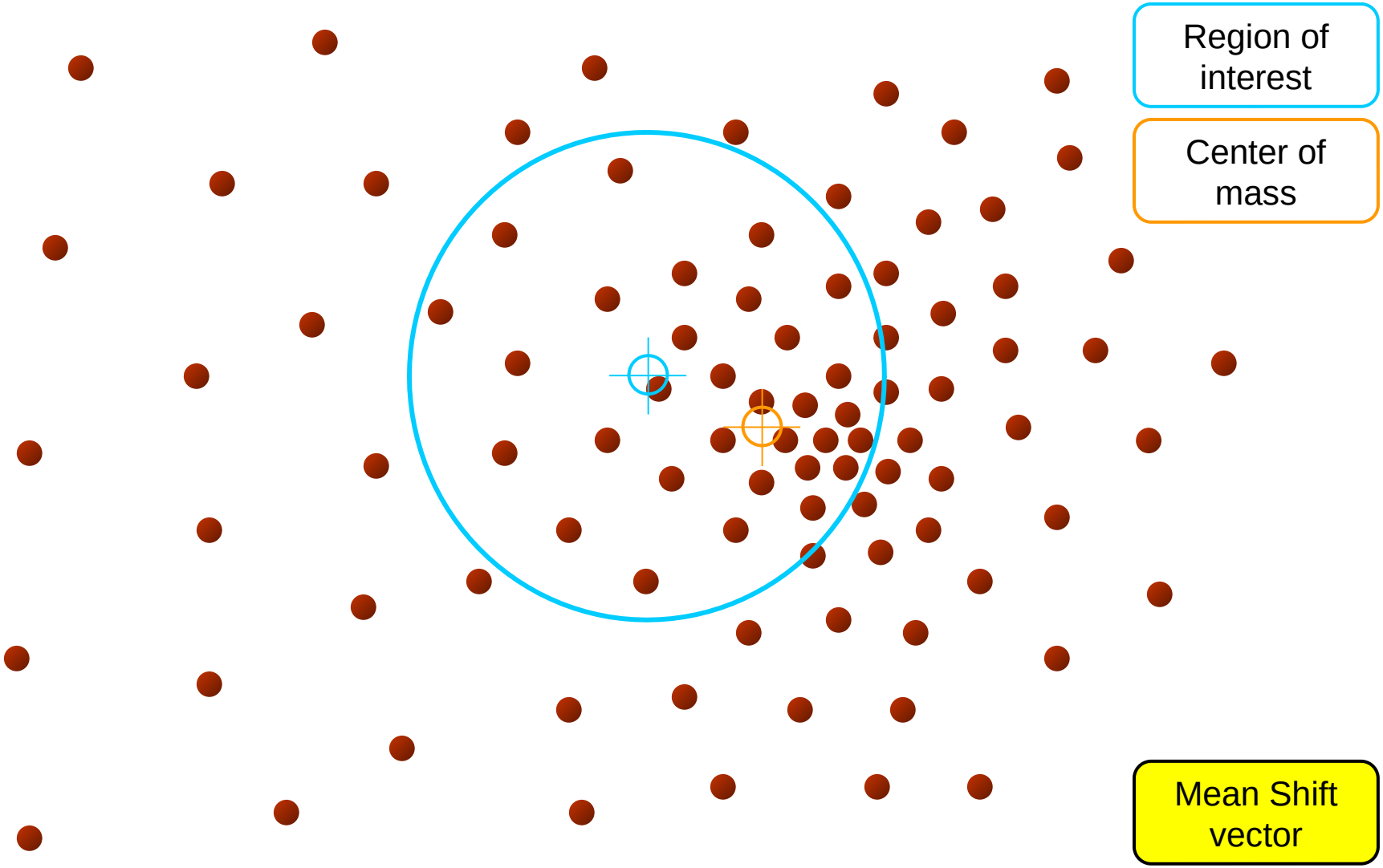
Slide by Y. Ukrainitz & B. Sarel

Mean shift



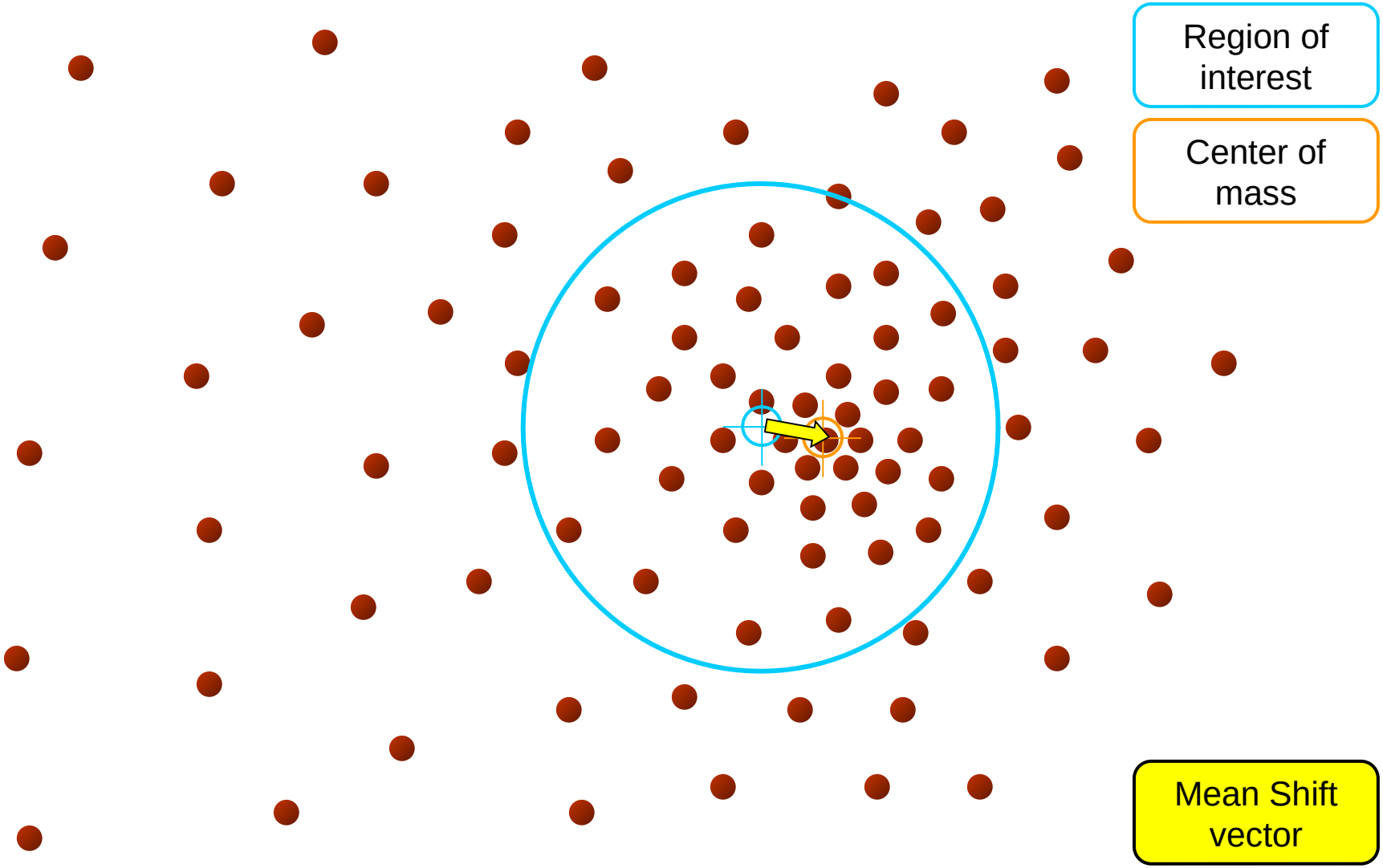
Slide by Y. Ukrainitz & B. Sarel

Mean shift



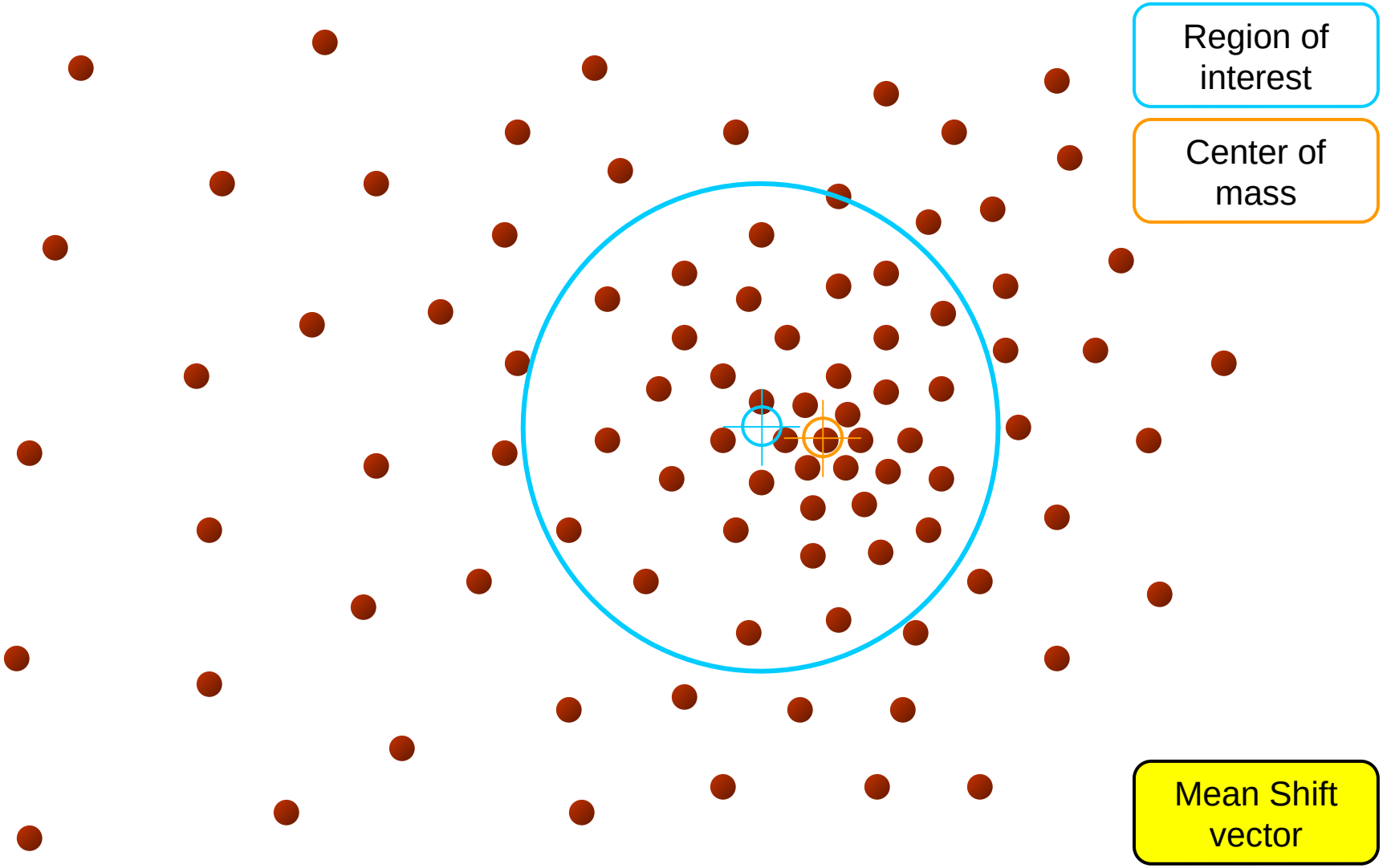
Slide by Y. Ukrainitz & B. Sarel

Mean shift



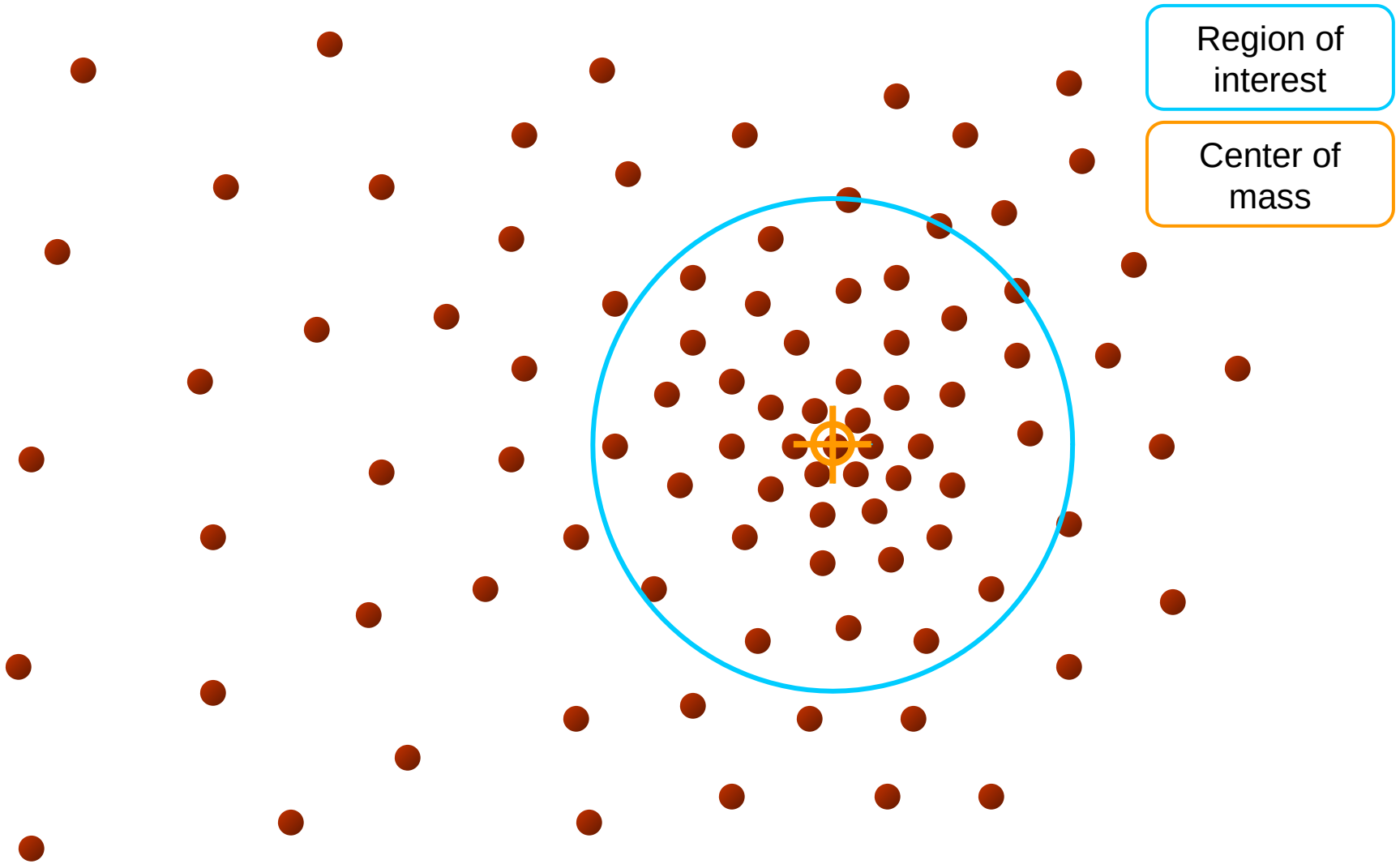
Slide by Y. Ukrainitz & B. Sarel

Mean shift



Slide by Y. Ukrainitz & B. Sarel

Mean shift



Slide by Y. Ukrainitz & B. Sarel

Kernel density estimation

Kernel density estimation function

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

n = number of $x_i \in R^d$

h = radius of the kernel (bandwidth parameter)

Gaussian kernel

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

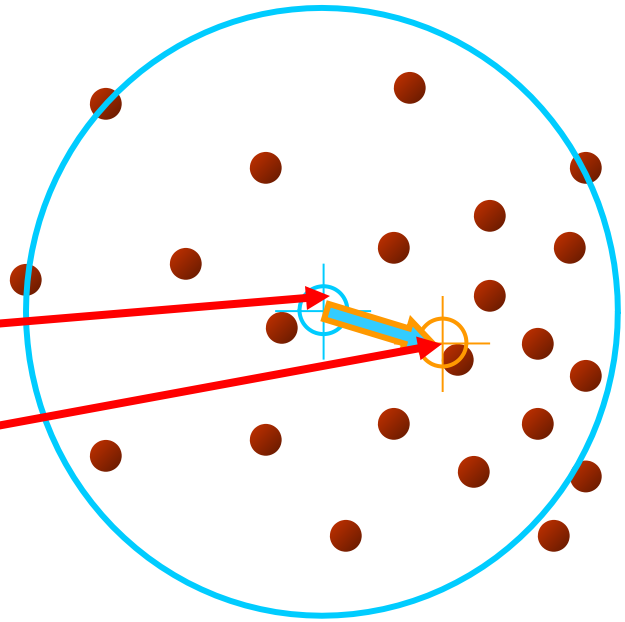
g(x) = -K′(x)

Computing the Mean Shift

Simple Mean Shift procedure:

- Compute mean shift vector **m(x)**
- Iteratively translate the kernel window by **m(x)** until convergence.

$$\mathbf{m}(\mathbf{x}) = \left[\frac{\sum_{i=1}^{n} \mathbf{x}_i g\left(\frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{h} \right)}{\sum_{i=1}^{n} g\left(\frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{h} \right)} - \mathbf{x} \right]$$



$$g(\mathbf{x}) = -k'(\mathbf{x})$$

Slide by Y. Ukrainitz & B. Sarel

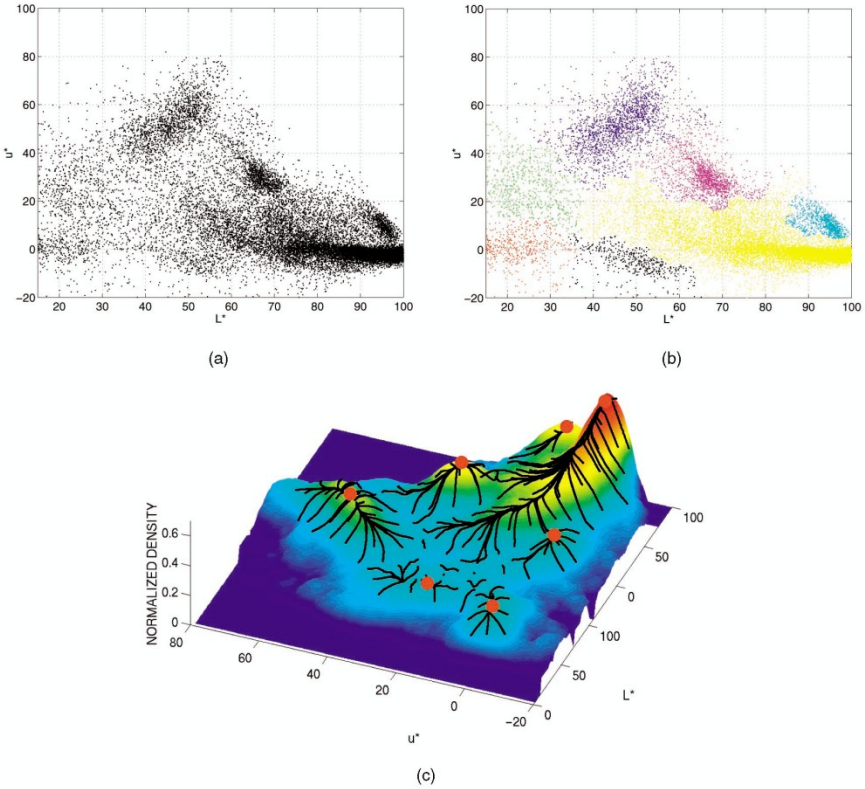
Mean shift clustering

The mean shift algorithm seeks *modes* of the given set of points

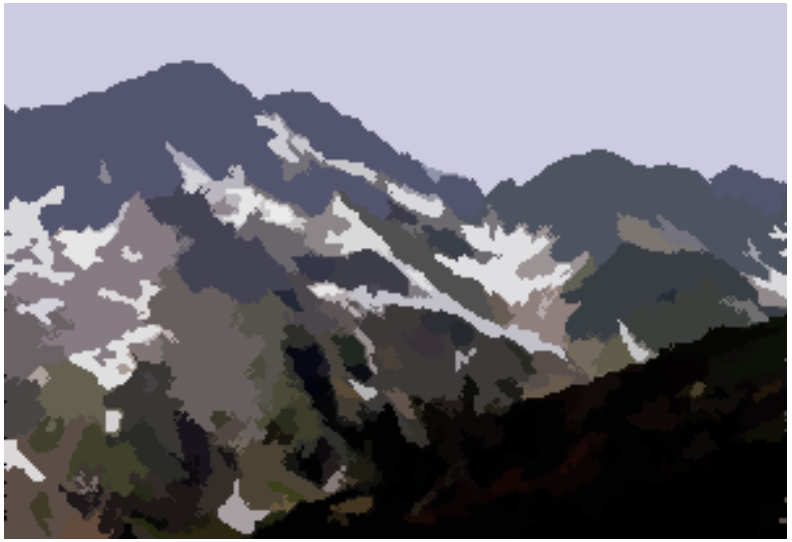
1. Choose kernel and bandwidth
2. For each point:
   a) Center a window on that point
   b) Compute the mean of the data in the search window
   c) Center the search window at the new mean location
   d) Repeat (b,c) until convergence
3. Assign points that lead to nearby modes to the same cluster

Segmentation by Mean Shift

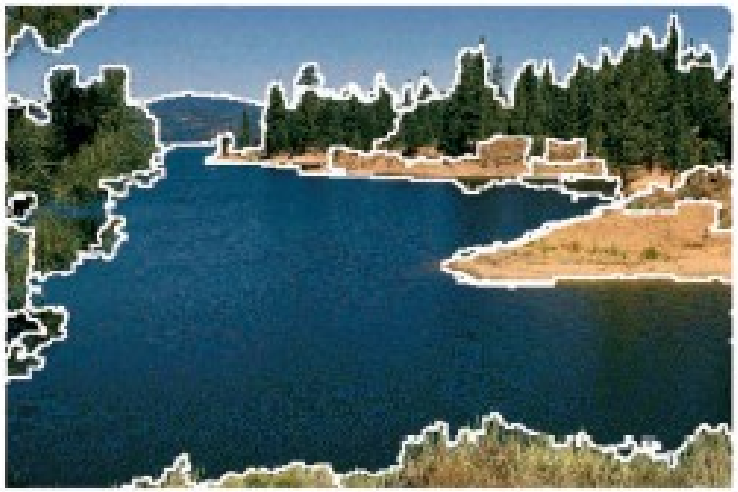
- Compute features for each pixel (color, gradients, texture, etc.).
- Set kernel size for features K_f and position K_s.
- Initialize windows at individual pixel locations.
- Perform mean shift for each window until convergence.
- Merge windows that are within width of K_f and K_s.

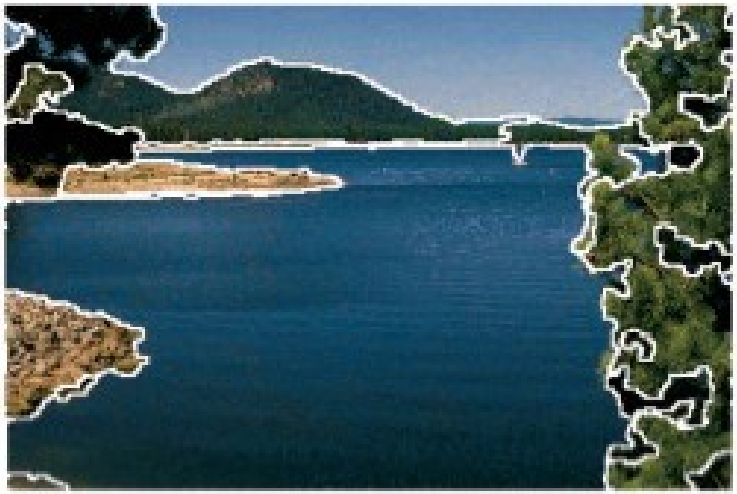


Mean shift segmentation results



Comaniciu and Meer 2002





Comaniciu and Meer 2002

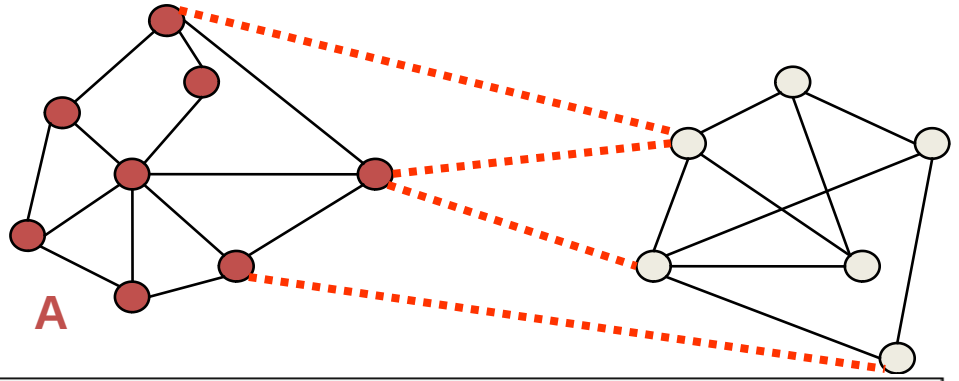
Mean shift pros and cons

- 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 graph structure & edge costs.

Captures “neighborhood-ness” or local smoothness.



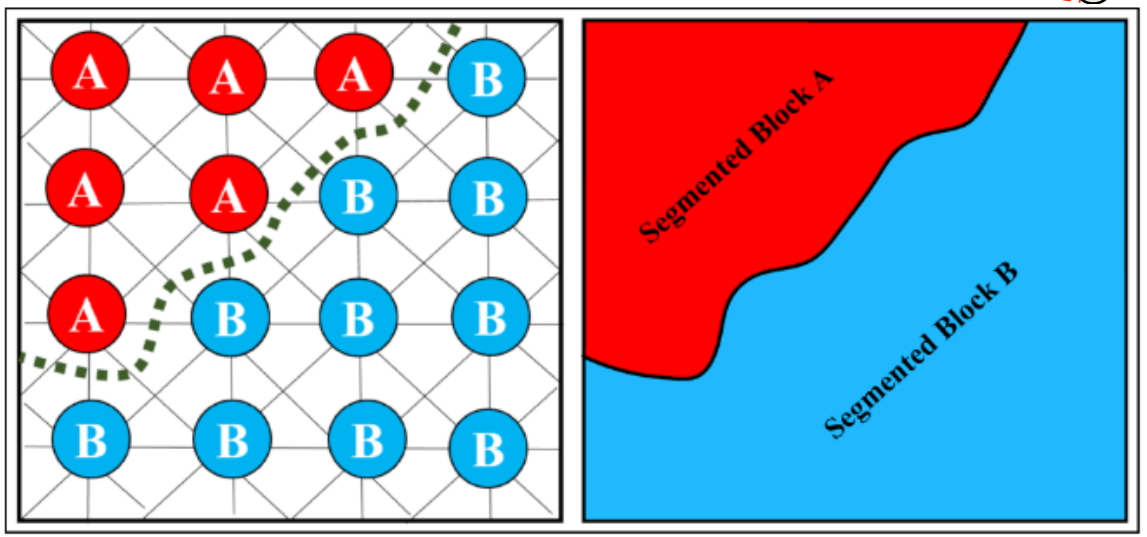
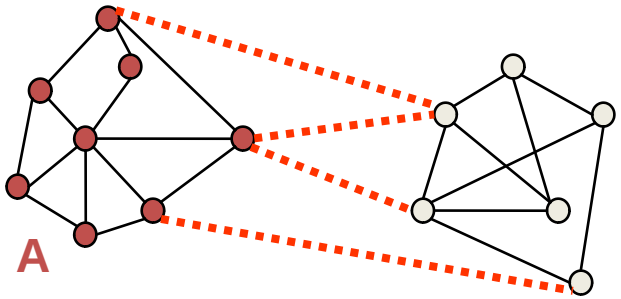
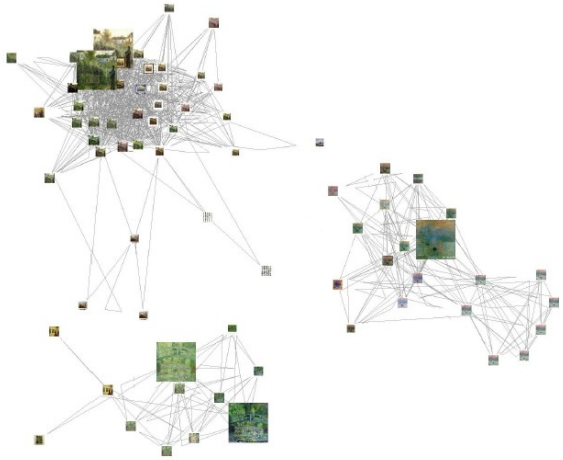


Image: Hassan et al.

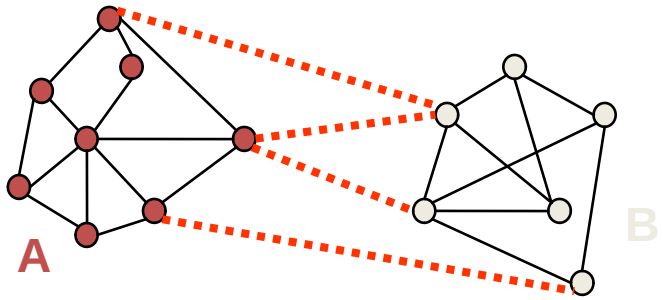
Spectral clustering

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

Source: Seitz

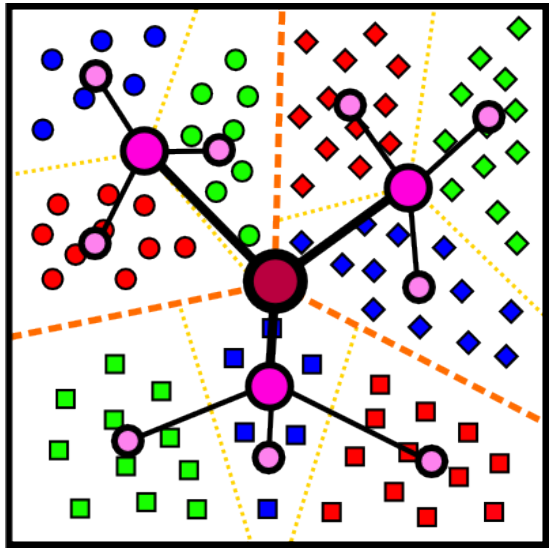
Normalized cuts for segmentation

Segmentation as a result

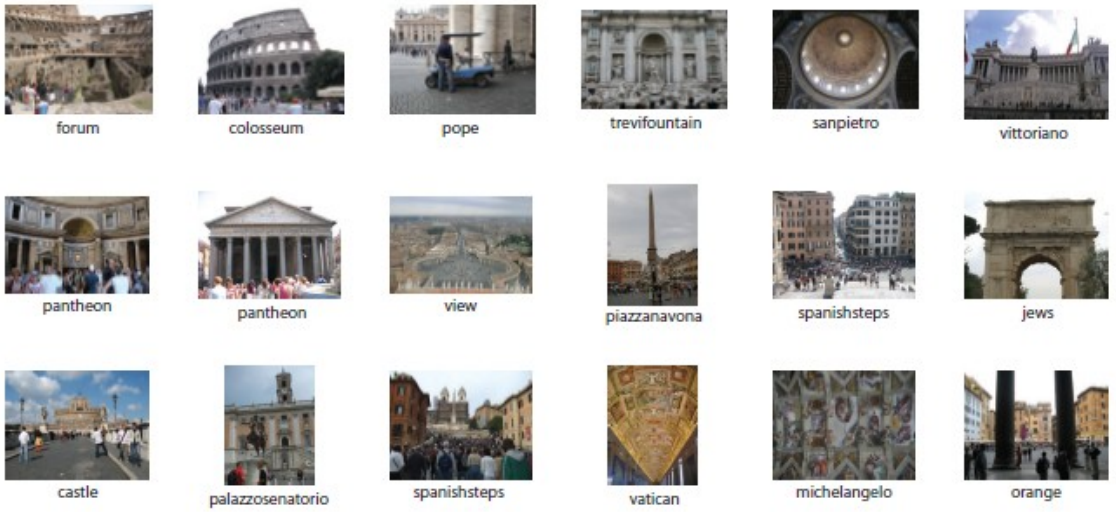
GrabCut, Rother et al. 2004

Which algorithm to use?

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



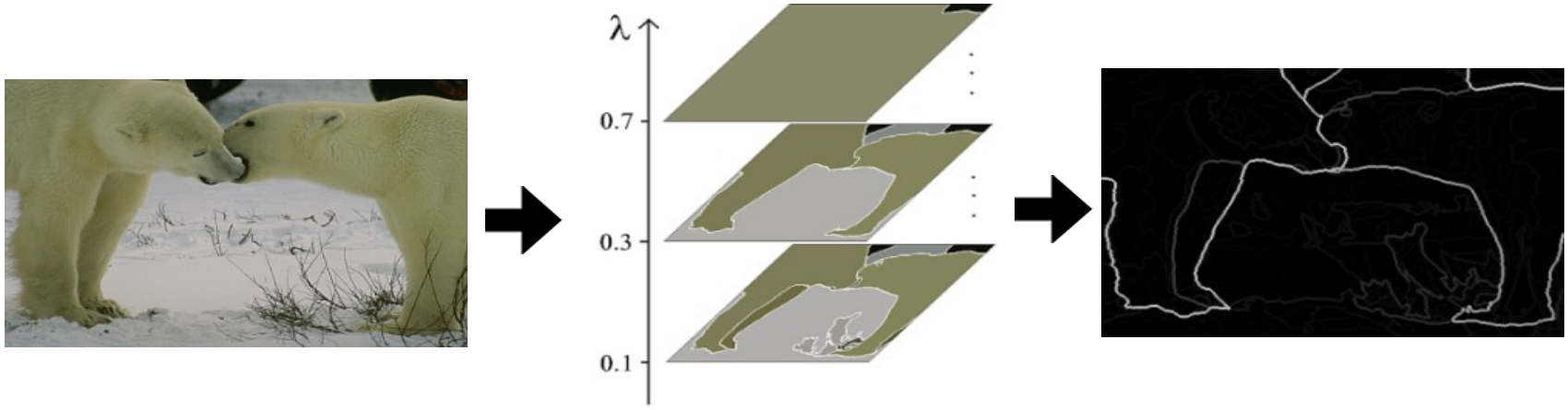
Quantization for computing histograms



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



http://www.cs.berkeley.edu/~arbelaez/UCM.html