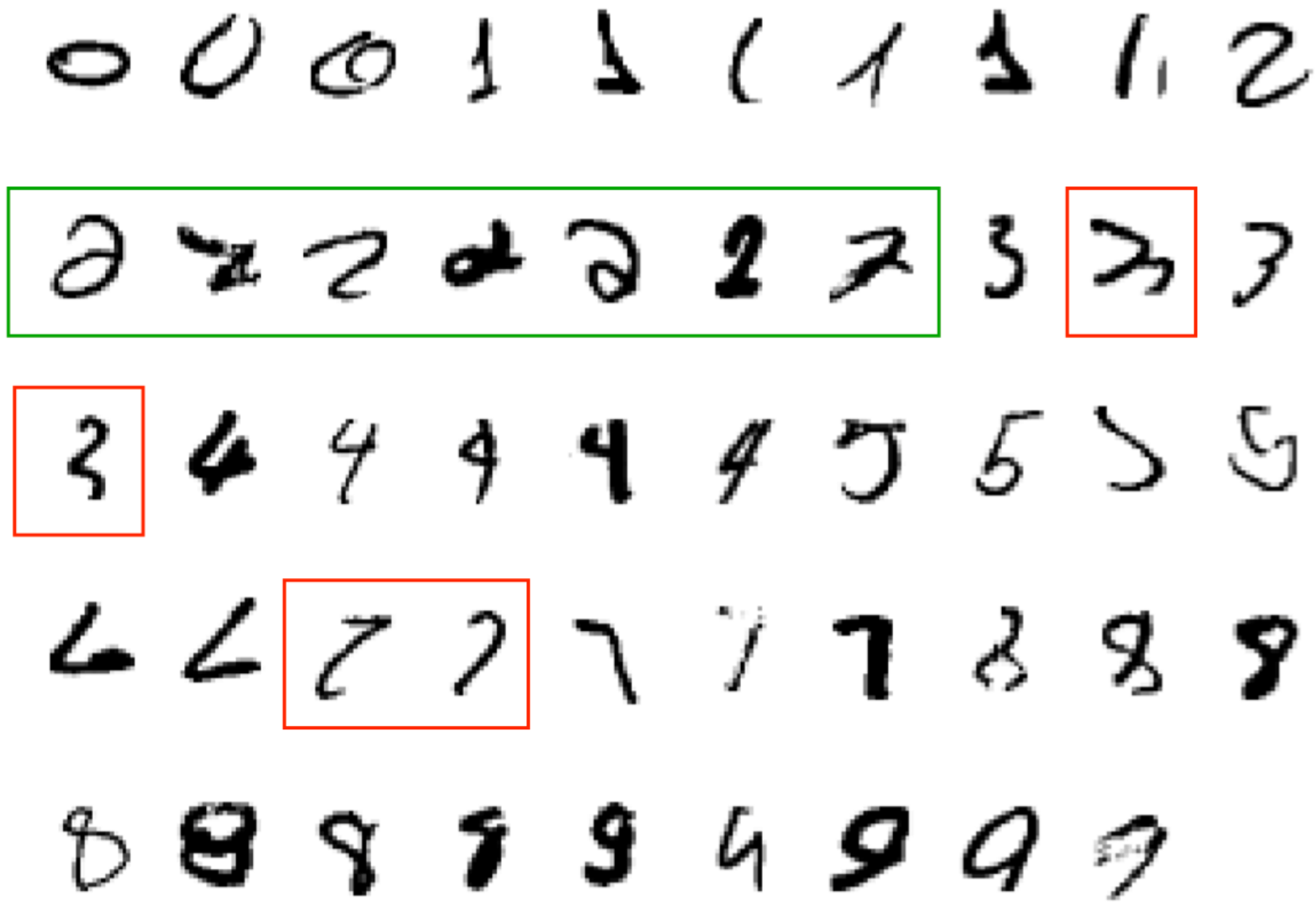
Machine Learning Introduction

Faisal Qureshi
faisal.qureshi@uoit.ca

Machine Learning

- Making predictions or decisions from data
- Machine learning is easily the greatest export from computing to other scientific disciplines

What makes a “2”?



[Source: Urtasun]

Machine Learning

- It is very hard to write programs that solve problems like recognizing a handwritten digit
  - What distinguishes a 2 from a 7?
  - How does our brain do it?
- Instead of writing a program by hand, we collect examples that specify the correct output for a given input
- A machine learning algorithm then takes these examples and produces a program that does the job
  - The program produced by the learning algorithm may look very different from a typical hand-written program. It may contain millions of numbers.
  - If we do it right, the program works for new cases as well as the ones we trained it on.

Machine Learning and Statistics

- Machine learning deals with inference in the presence of uncertainty
    - Statistical theory to build models
- Machine learning can be seen as applying computational techniques to statistical problems

Applications of Machine Learning

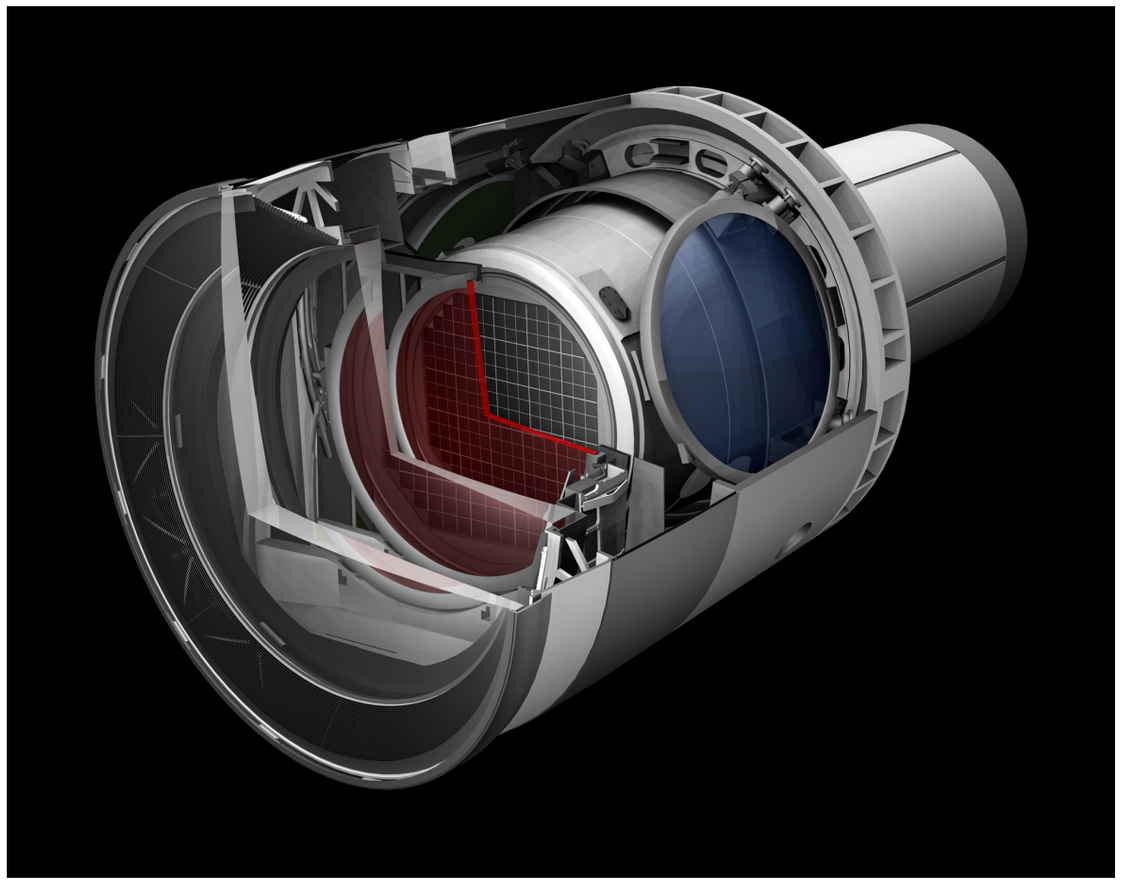
- Recognizing patterns: face recognition, spoken language understanding
- Digital images and videos: autonomous vehicles
- Recognizing anomalies: unusual sequence of credit card transactions
- Spam filtering & fraud detection
- Recommendation system: ads by Google, Amazon
- Information retrieval: find similar images
- And many many more …

Big Data and Machine Learning

- Machine learning assumes access to large corpus of data for *training*

- "Large" text dataset
  - 1,000,000 words in 1967
  - 1,000,000,000,000 words in 2006

Machine Learning



SLAC / LSST Camera

15 Terabytes every night

"When the Sloan Digital Sky Survey started work in 2000, its telescope in New Mexico collected more data in its first few weeks than had been amassed in the entire history of astronomy.

Now, a decade later, its archive contains a whopping 140 terabytes of information.

A successor, the Large Synoptic Survey Telescope, due to come on stream in Chile in 2016, will acquire that quantity of data every five days."

The Economist, February 2010

Types of Learning

- Supervised: correct output known for each training example
  - Classification
  - Regression
- Unsupervised: create internal representation of the input, capturing regularities/structure in data
  - Clustering
  - Dimensionality reduction
- Reinforcement learning: pick actions/policy to maximize payoffs

Types of Learning

	Supervised learning	**Unsupervised learning**
Discrete	Classification or Categorization	Clustering
Continuous	Regression	Dimensionality reduction

Types of Learning

	Supervised learning	**Unsupervised learning**
Discrete	Classification or Categorization	Clustering
Continuous	Regression	Dimensionality reduction

Image Categorization

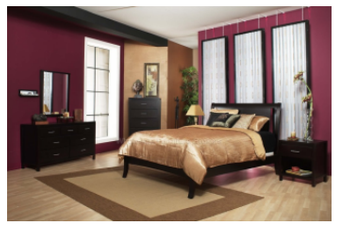
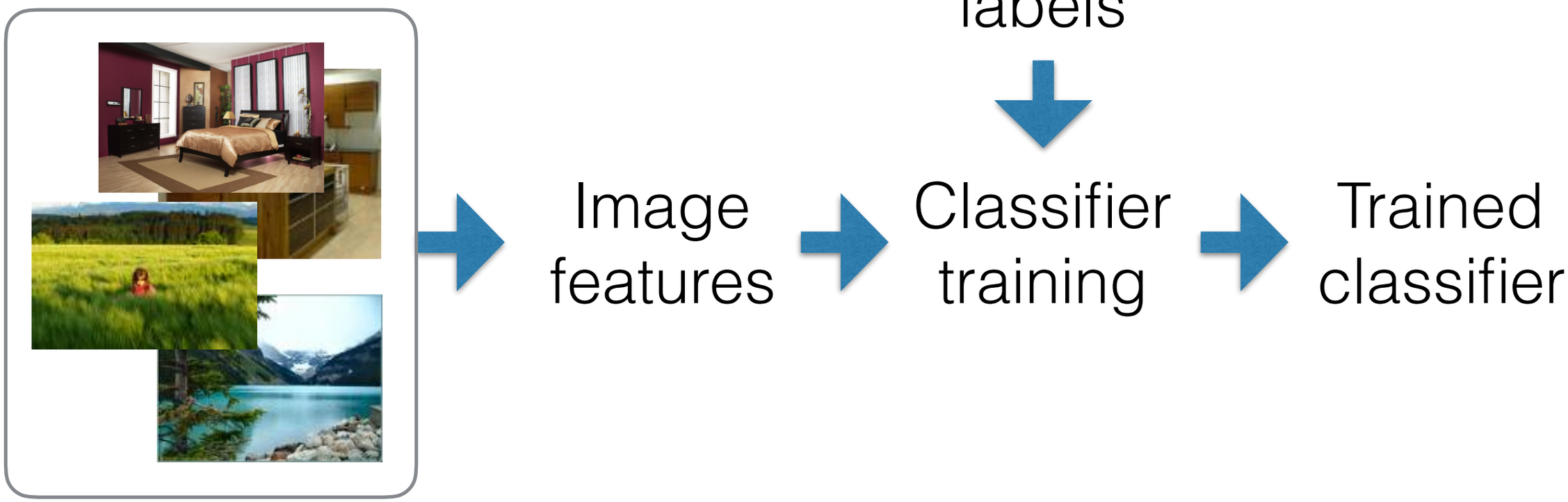
Images	Labels	
	*Bedroom*	*Indoor*
	*Kitchen*	*Indoor*
	*Grassy Plain*	*Outdoor*
	*Lake*	*Outdoor*

Image Categorization



Training images

Image Categorization

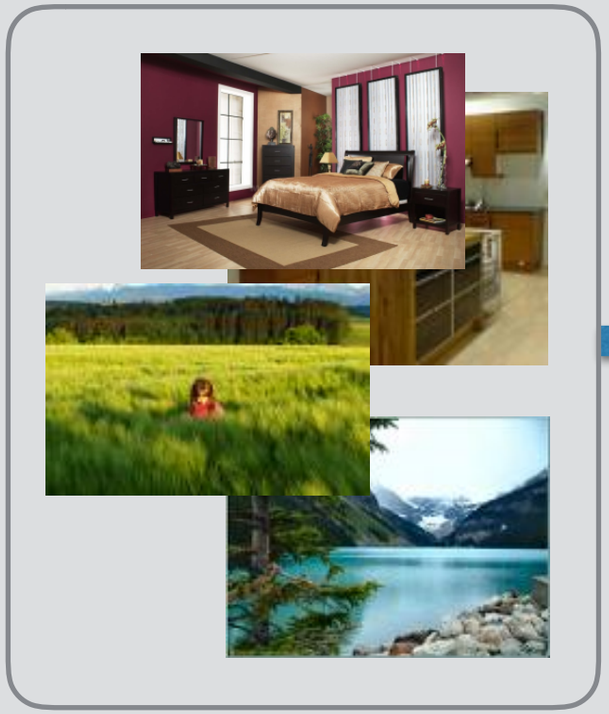
Test Image

Image features

Trained classifier

Label

Outdoor, grass



Training labels

Image features

Classifier training

Trained classifier

Training images

Image features

Trained classifier

Label

Outdoor, grass

Test Image

Image Features

- Coverage: captures relevant info
- Concision: parsimonious
- Independence: features are independently useful for prediction

Is intensity a good feature for face recognition?

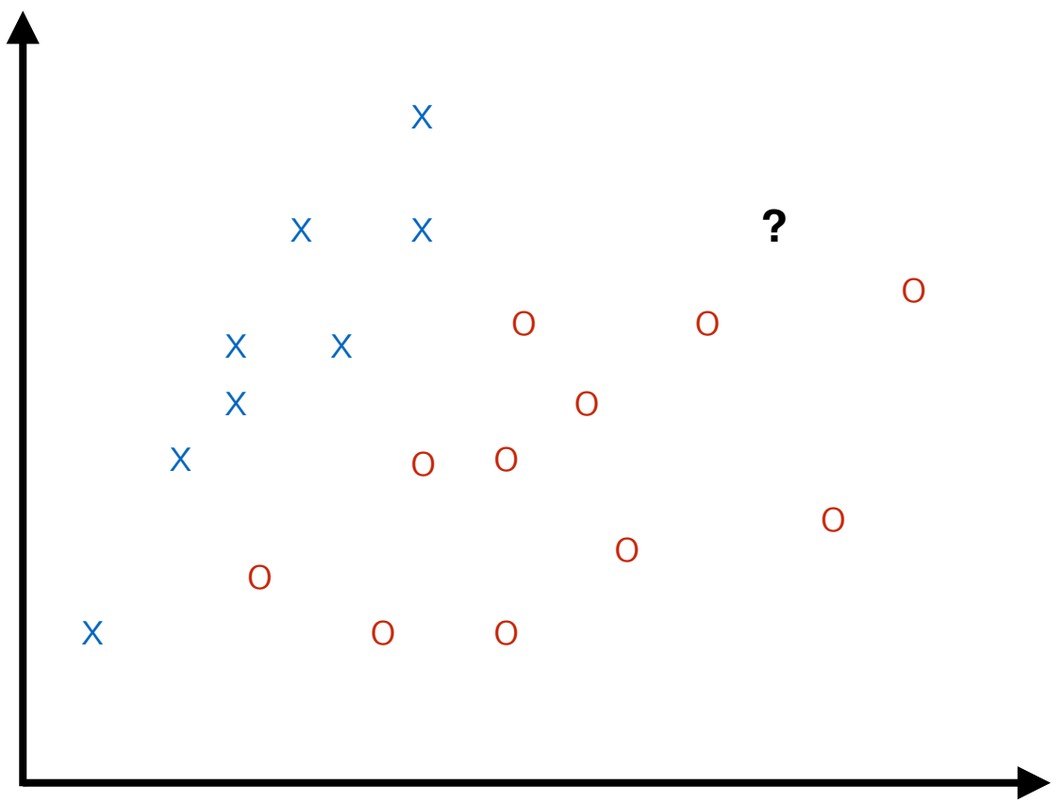
Image features (representations)

- Intensity, gradients, colors, textures, Fourier and Discrete Cosine transform coefficients, etc.

- Histograms of color, gradients, textures, SIFT, FAST, etc.

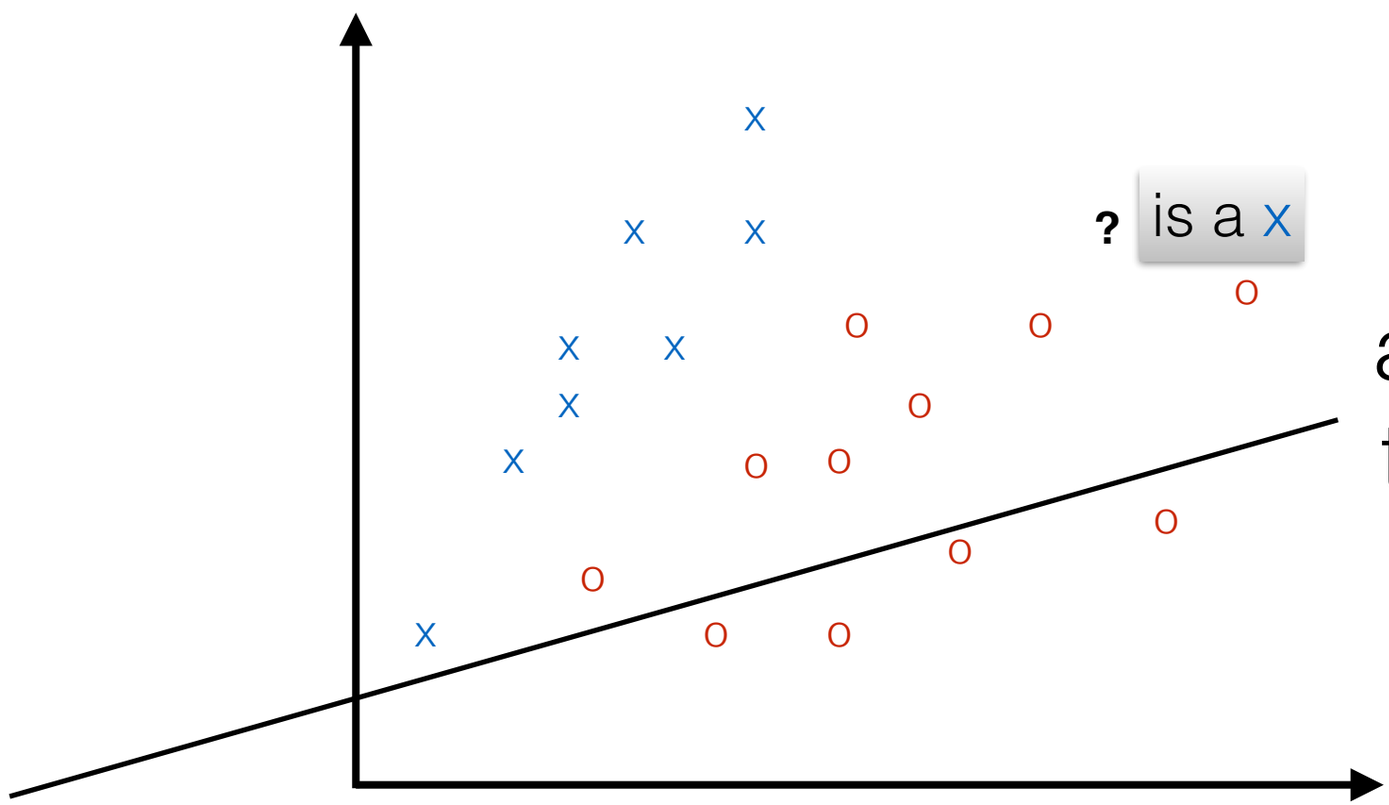
Learning a Classifier

- Given some *features* with corresponding *labels* learn a function to predict the *label* of a previous unseen *feature*



Learning a Classifier

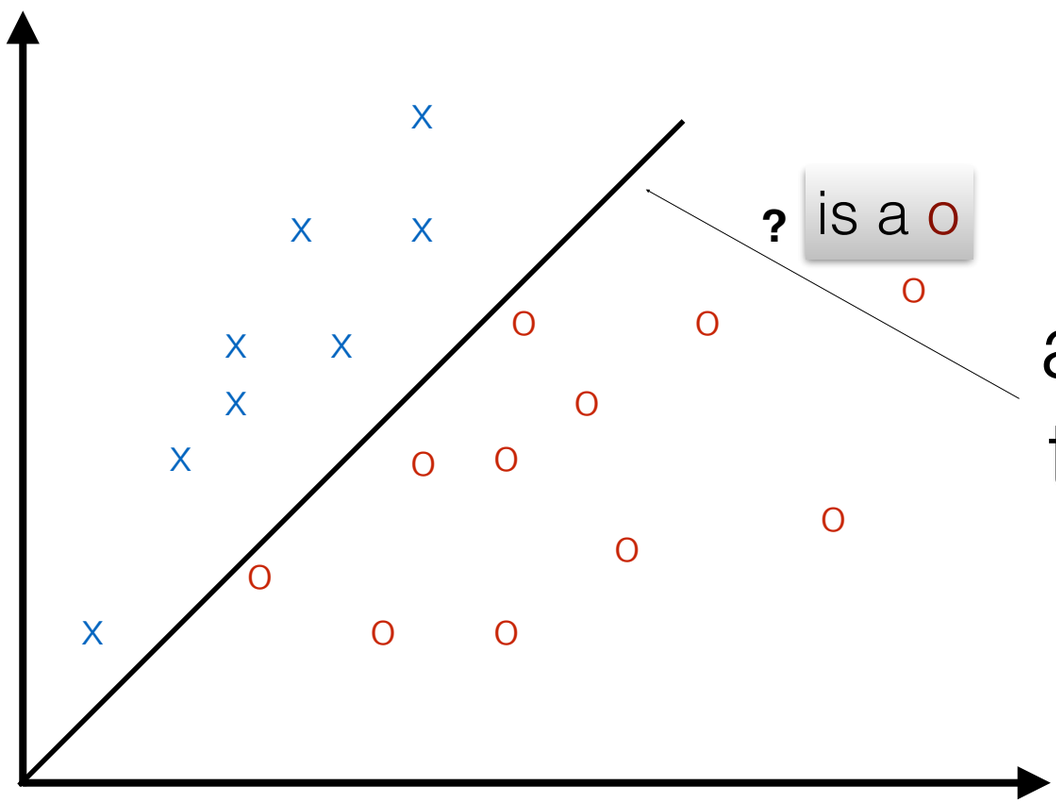
- Given some *features* with corresponding *labels* learn a function to predict the *label* of a previous unseen *feature*



any thing above this line is an ‘x’

Learning a Classifier

- Given some *features* with corresponding *labels* learn a function to predict the *label* of a previous unseen *feature*



Learning a Classifier

- Training labels dictate if two examples are similar or different

- Features are used to compute the degree of dissimilarity (or distance) between two examples

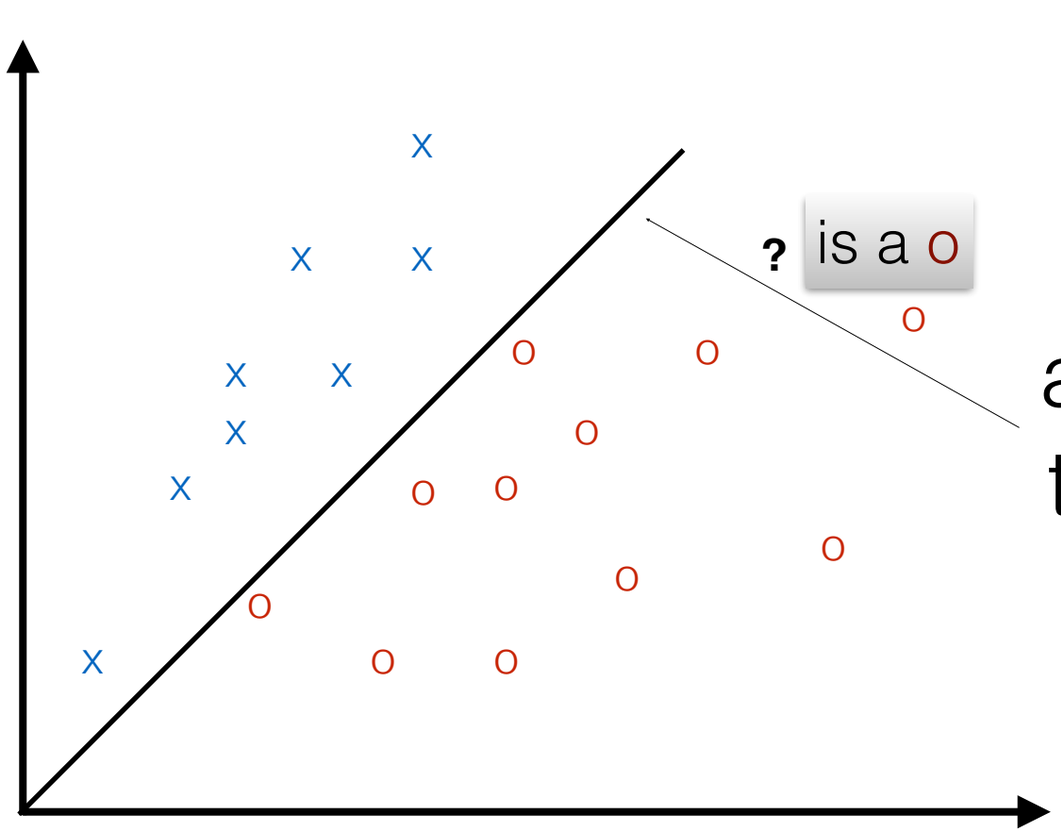
  - These can be used to define visual similarity between two examples

Classifiers

- K-nearest Neighbours
- Naive Bayes
- SVM
- Logistic Regression
- RBMs
- Decision Trees
- *and many others*

Learning a Classifier

- Given some *features* with corresponding *labels* learn a function to predict the *label* of a previous unseen *feature*



This is a prediction function

any thing above this line is an ‘x’

Learning a Classifier

Given a training set (*labeled examples*)

$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots, (\mathbf{x}_N, y_N)$$

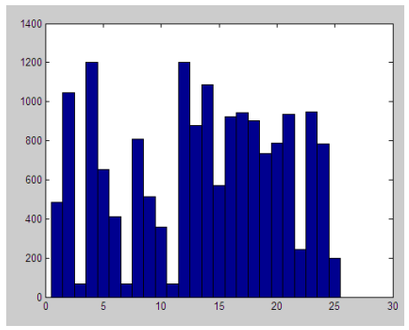
Learn a *prediction function* $y = f(\mathbf{x})$ that can be used to assign a label y given a previously unseen example $\mathbf{x}$

Training

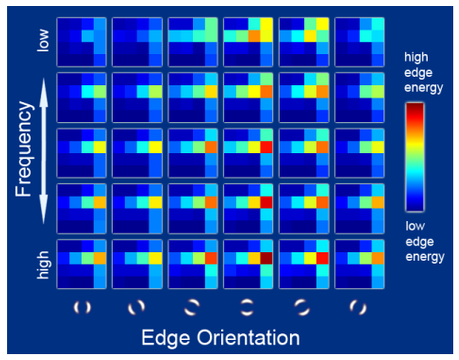
Features $\mathbf{x}$

Raw pixels

Histograms



Gist descriptors

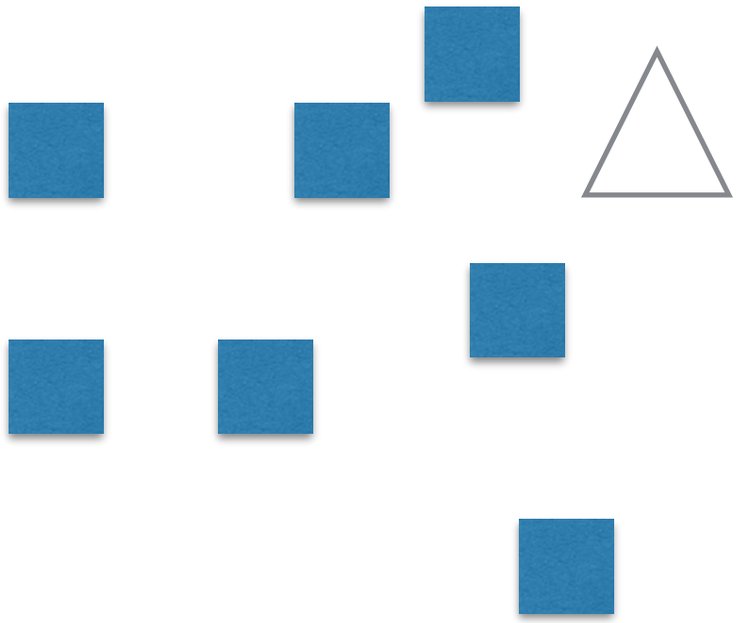


a point in a high-dimensional space

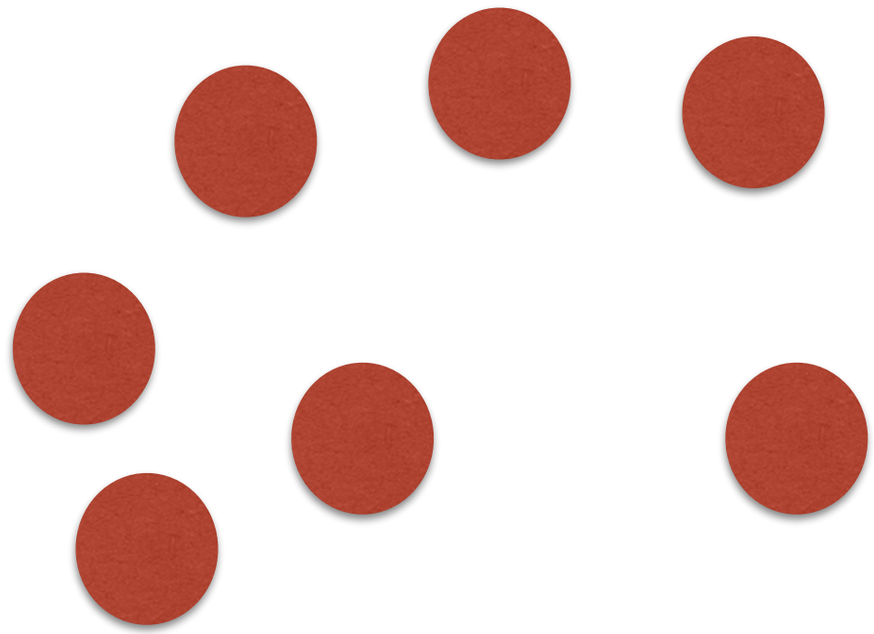
K-nearest Neighbour

Prediction function:

$$f(\mathbf{x}) = \text{label of the nearest training example}$$



Training examples from class 1

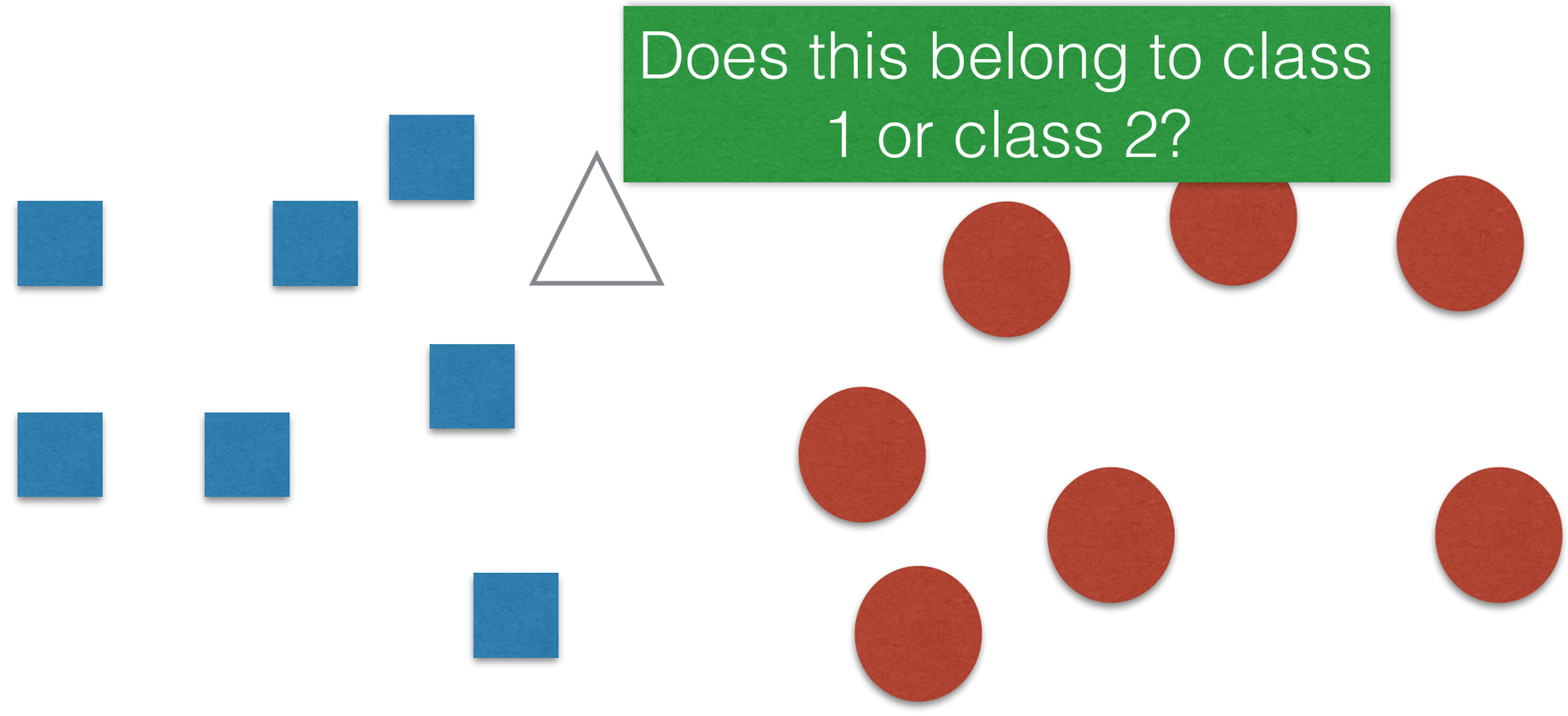


Training examples from class 2

K-nearest Neighbour

Prediction function:

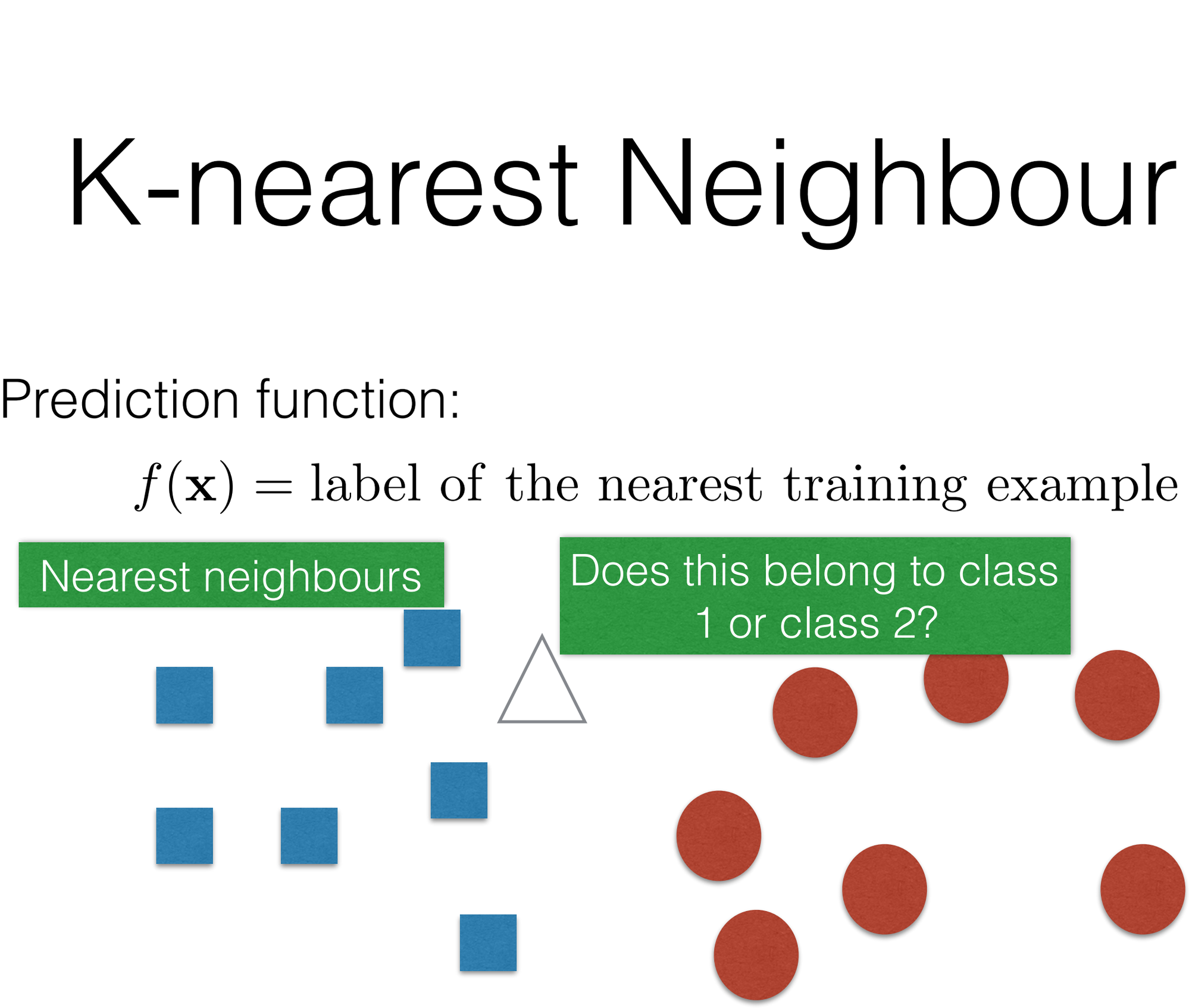
$$f(\mathbf{x}) = \text{label of the nearest training example}$$



K-nearest Neighbour

Prediction function:

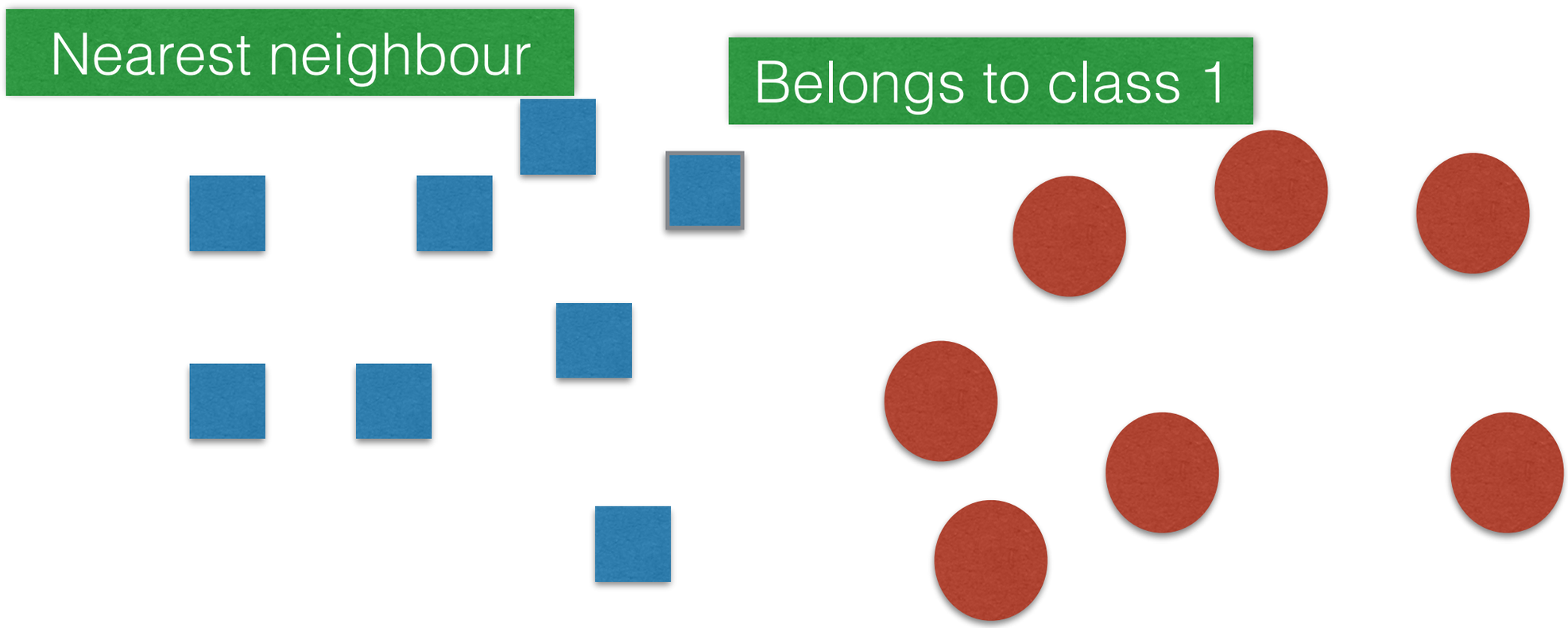
$$f(\mathbf{x}) = \text{label of the nearest training example}$$



K-nearest Neighbour

Prediction function:

$$f(\mathbf{x}) = \text{label of the nearest training example}$$



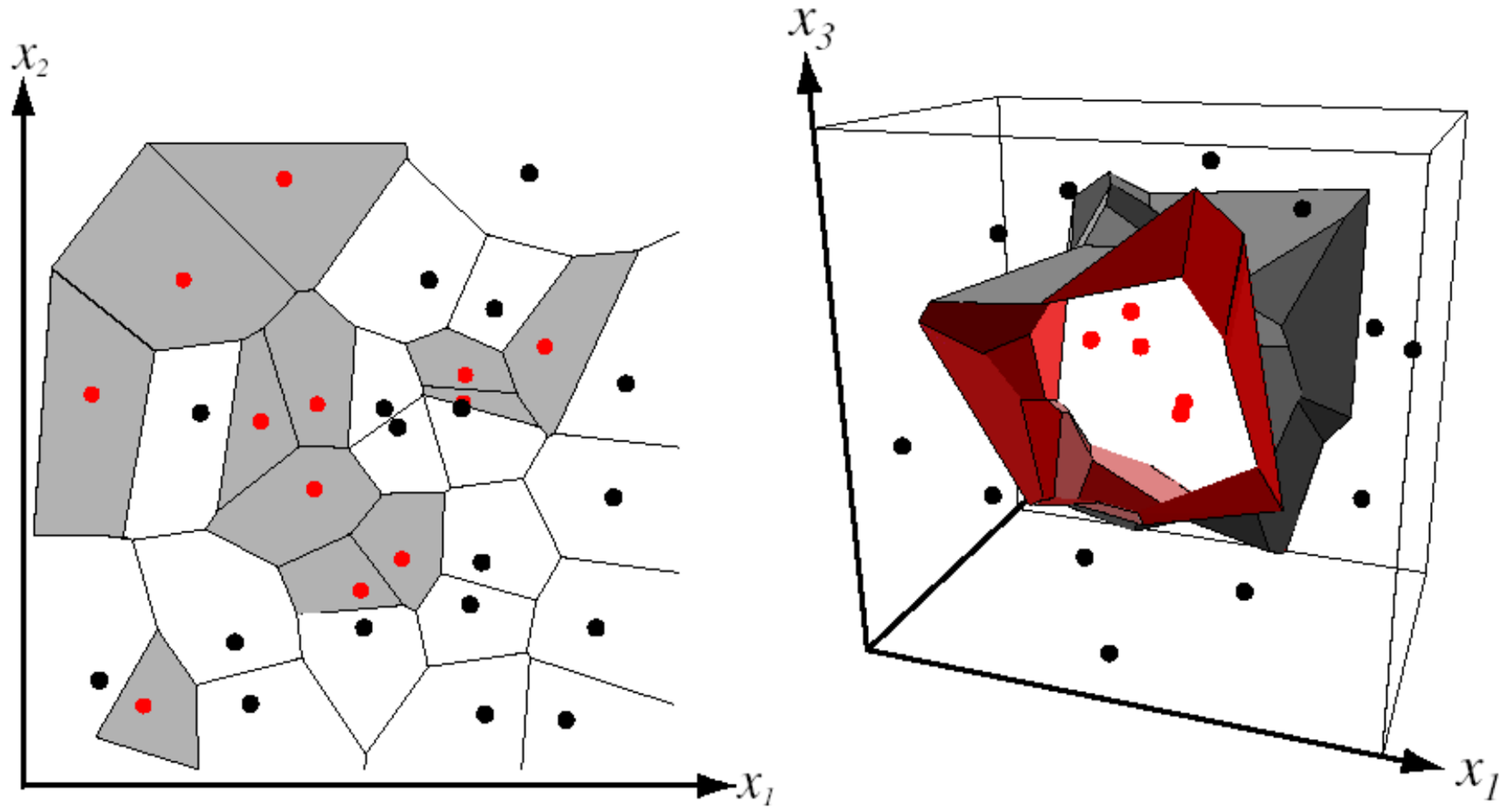
K-nearest Neighbour

Prediction function:

$$f(\mathbf{x}) = \text{label of the nearest training example}$$

- We need a *distance* function
- No training needed
- We need an efficient scheme for finding distance between the query and training set

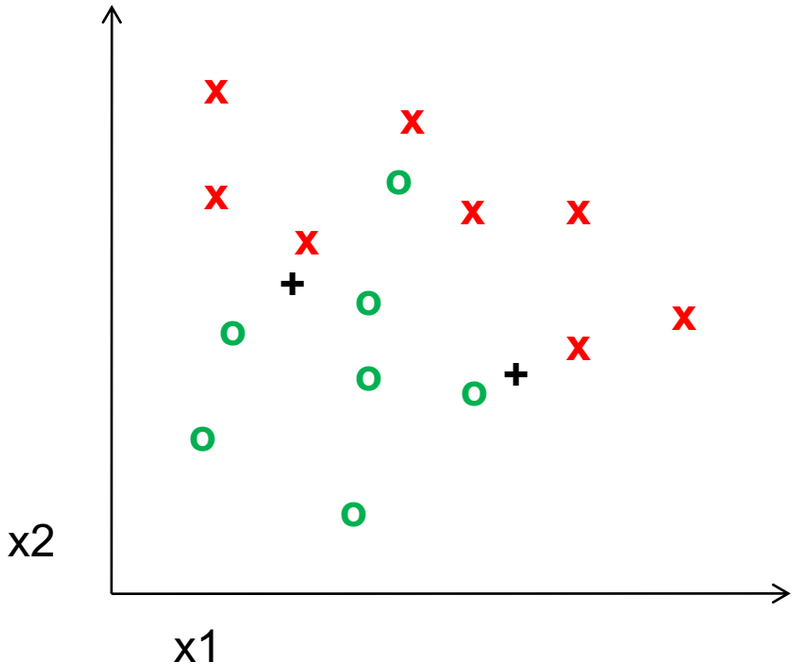
K-nearest Neighbours

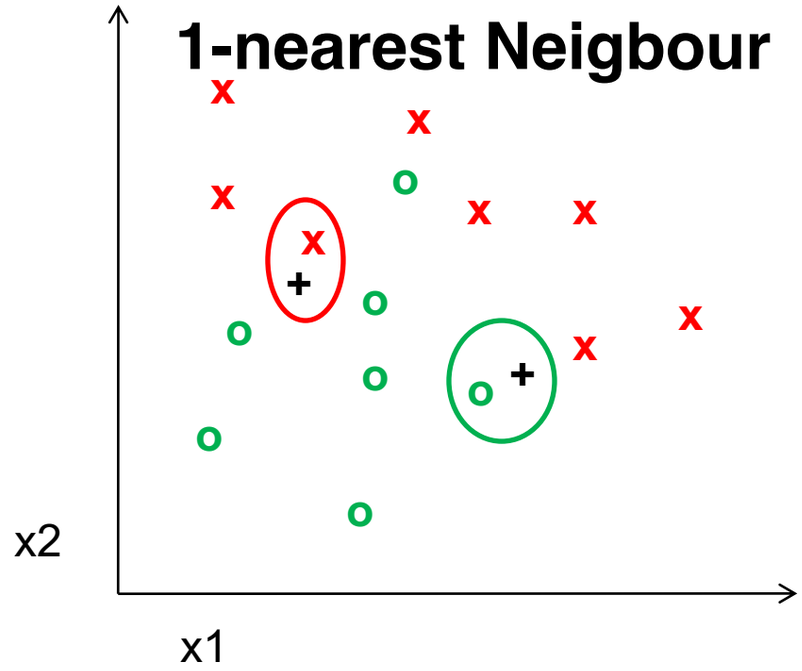


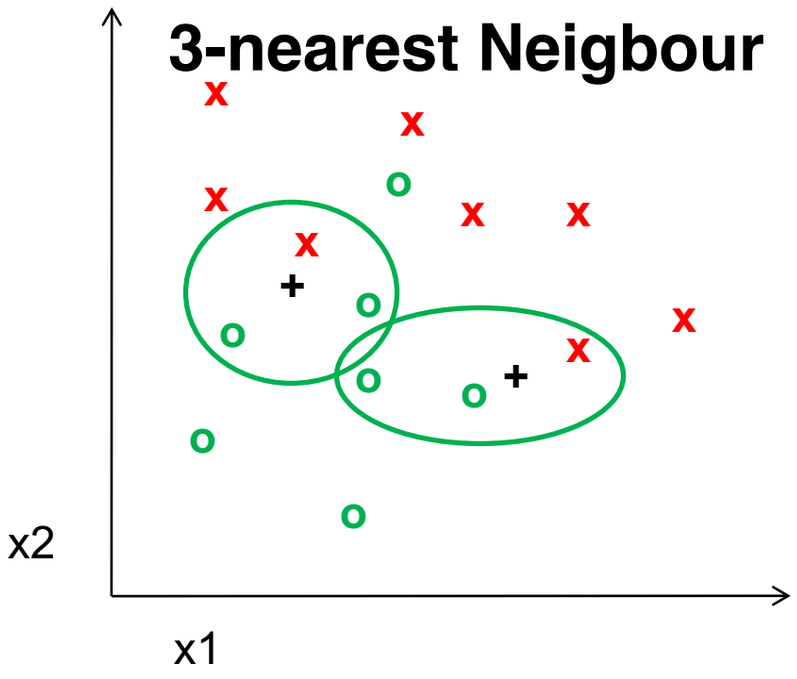
From Duda *et al.*

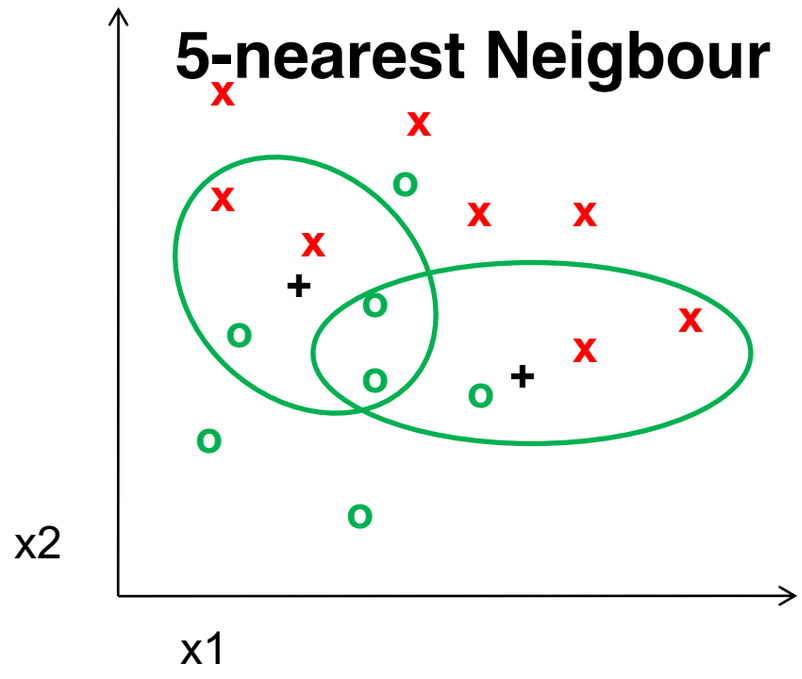
Voronoi partitioning of feature space for two-category 2D and 3D data

K-nearest Neighbours









K-nearest Neighbour

- With infinite examples, 1-nearest neighbour provably has error that is at most twice as Bayes optimal error

Naive Bayes

$$p(A|B) = \frac{p(B|A)p(A)}{p(B)}$$

Conditional Probability

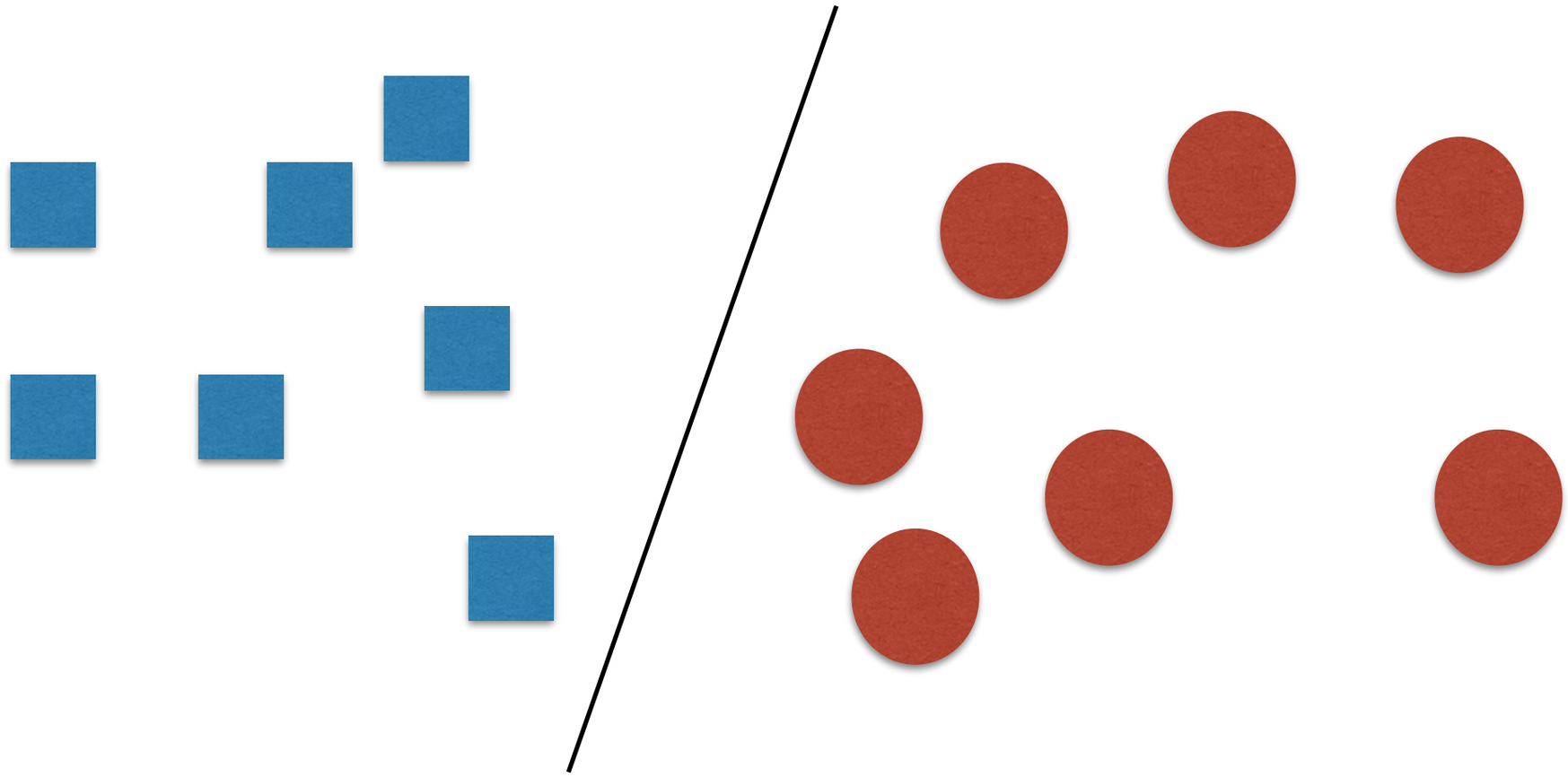
probability of A given B

$$p(c|f_1, f_2, f_3, \cdots, f_n) = \frac{p(f_1, f_2, f_3, \cdots, f_n|c)p(c)}{p(f_1, f_2, f_3, \cdots, f_n)}$$

Linear Classifier

Prediction function:

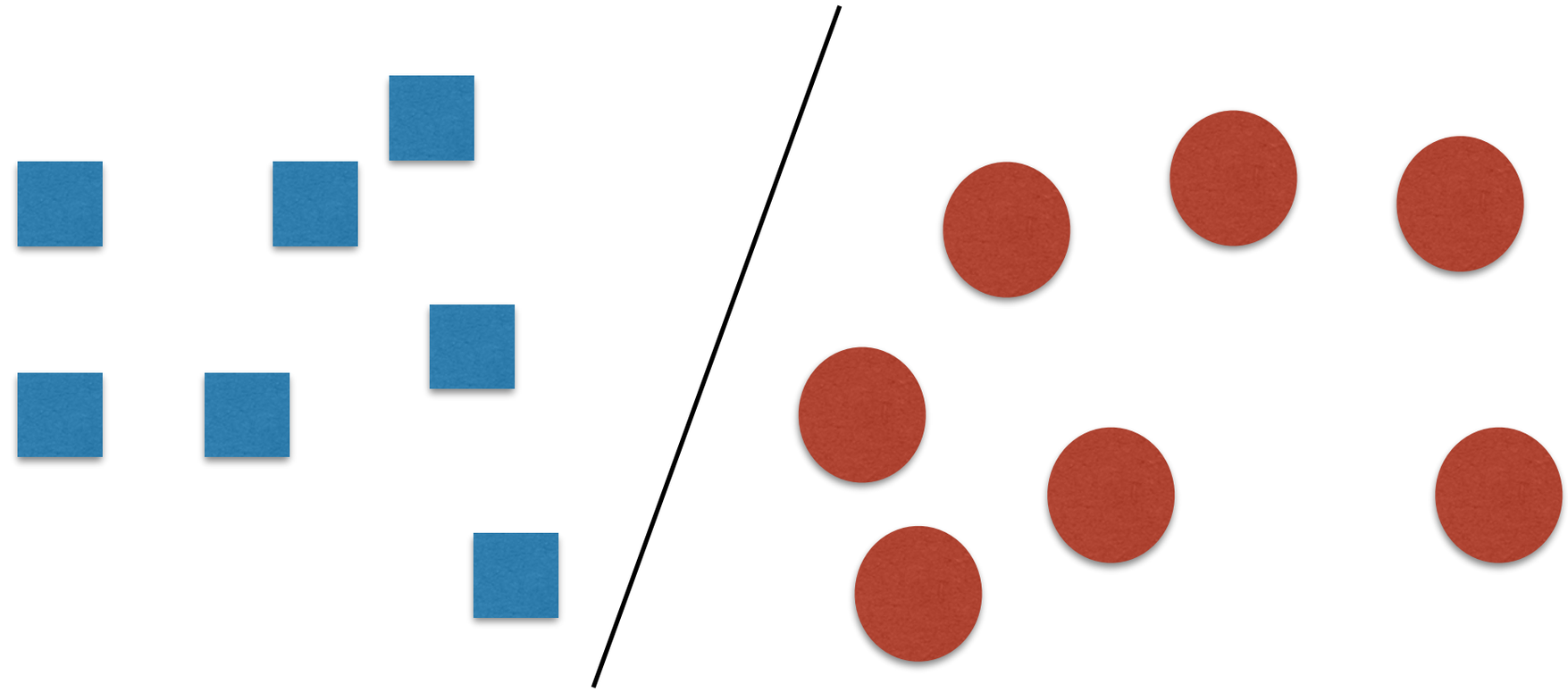
$$f(\mathbf{x}) = \mathrm{sgn}(\mathbf{w}^T\mathbf{x} + b)$$



Linear Classifier

Prediction function:

$$f(\mathbf{x}) = \text{sgn}(\mathbf{w}^T\mathbf{x} + b)$$

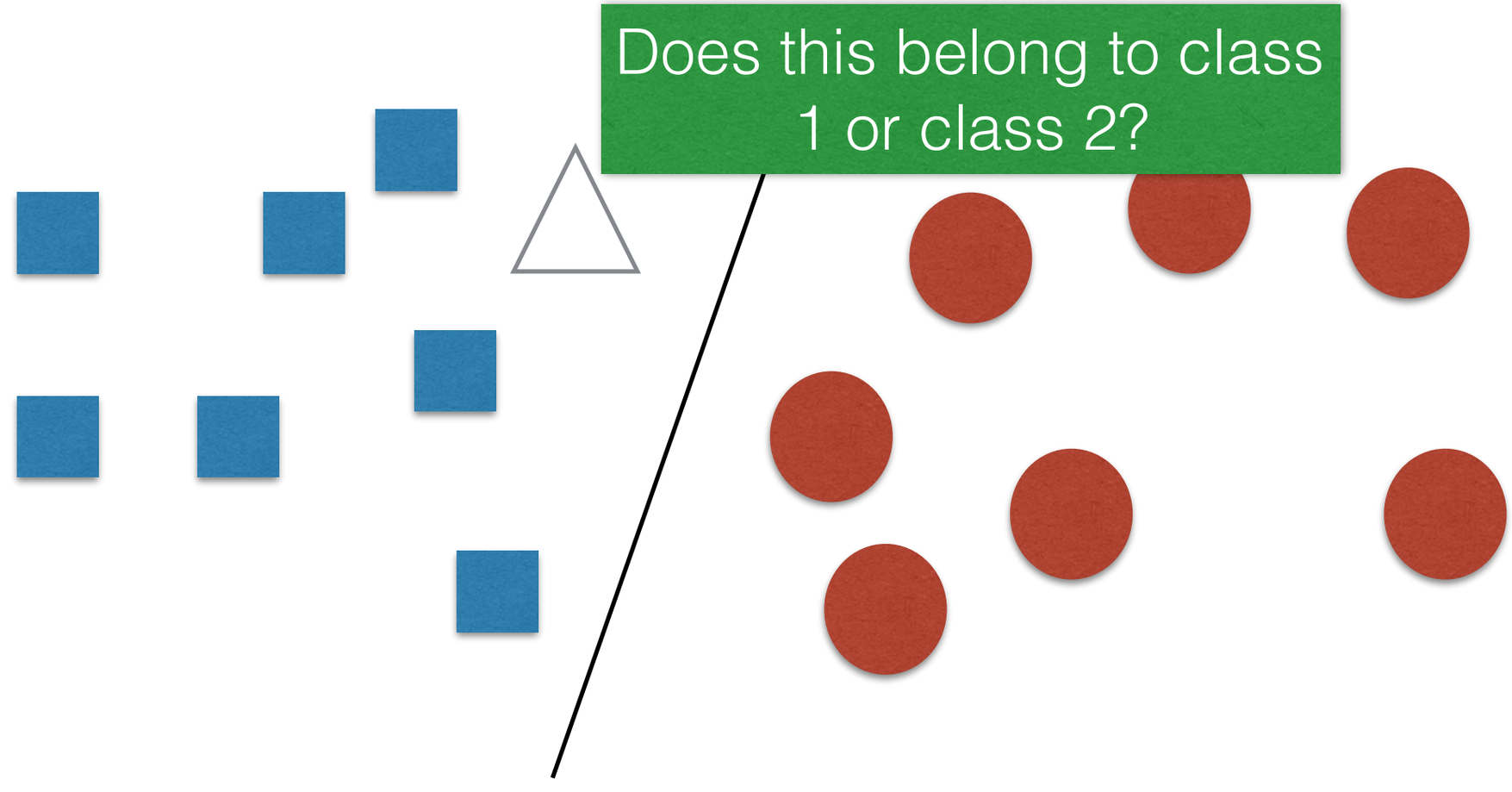


Learn a linear function that separates the two classes

Linear Classifier

Prediction function:

$$f(\mathbf{x}) = \text{sgn}(\mathbf{w}^T\mathbf{x} + b)$$



Linear Classifier

Prediction function:

$$f(\mathbf{x}) = \mathrm{sgn}(\mathbf{w}^T\mathbf{x} + b)$$

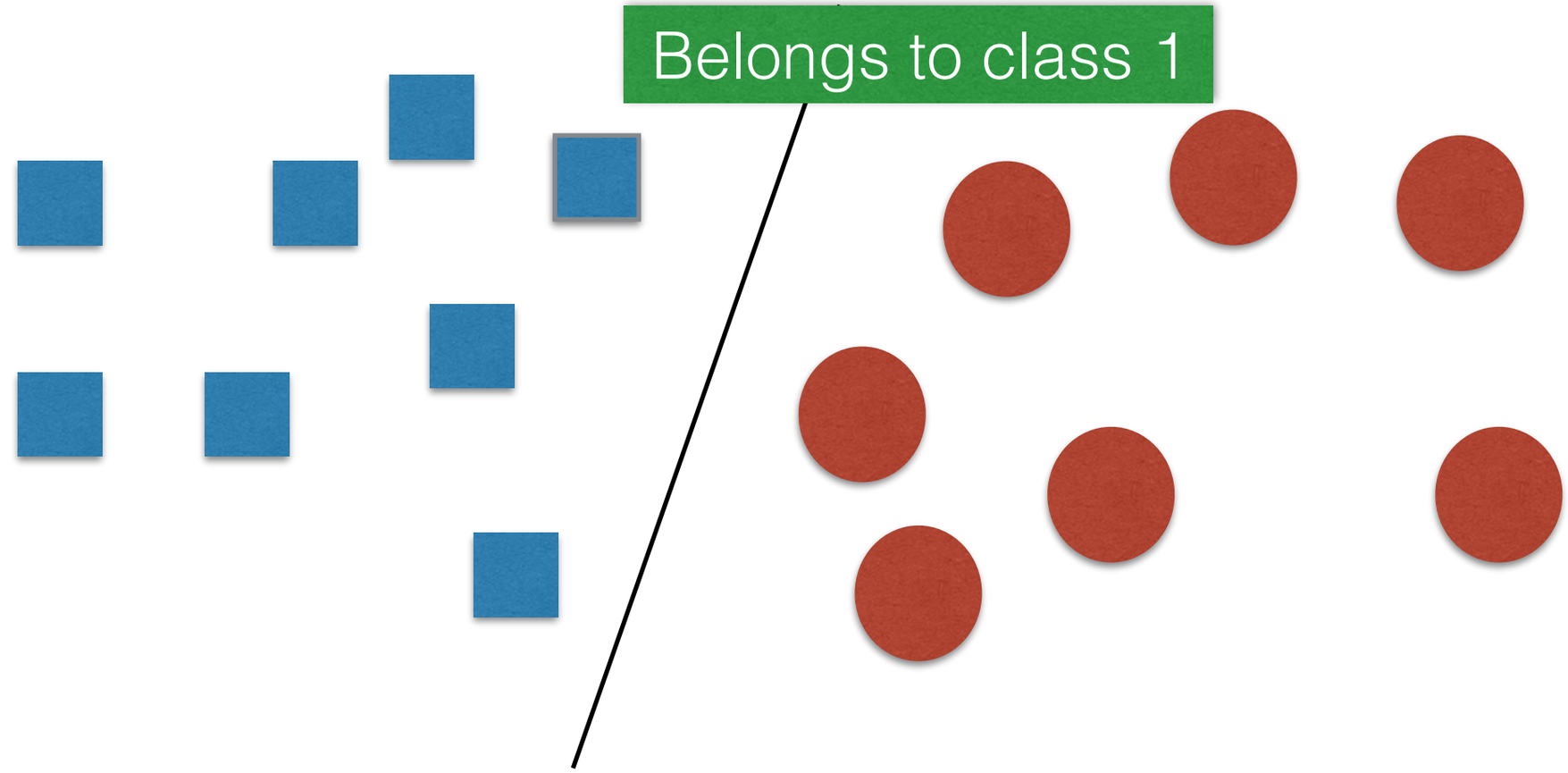
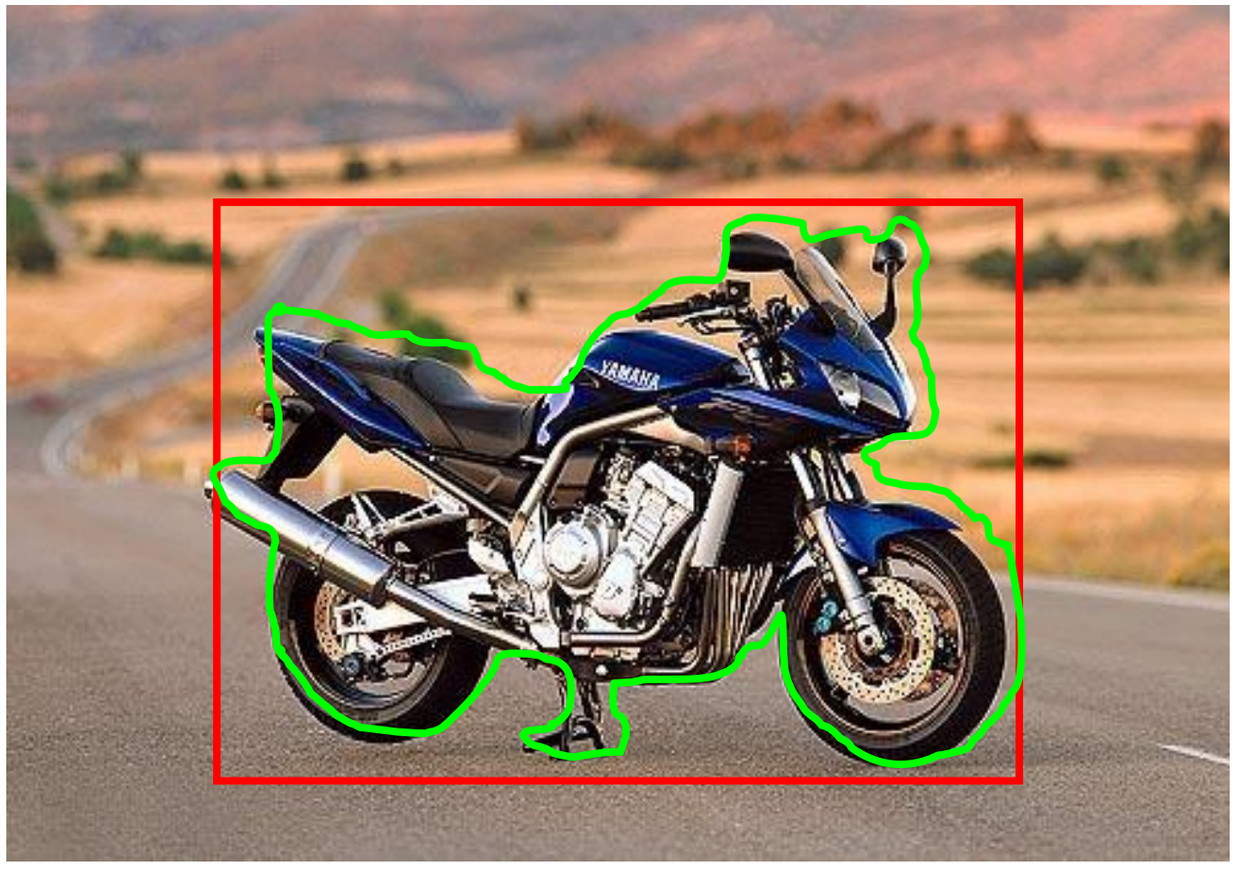


Image Recognition as *Classification*

- The *training set* contains annotated images



Contains motorbike

Generalization

- How well does a learned model generalize from the data it was trained on to a new test set?

Generalization

- Bias: how much learned model differs from the true model
  - Errors due to inaccurate assumptions and simplifications
- Variance: how much models learned over different training set differ from each other?
  - Inability to accurately estimate parameters from limited data

Variance

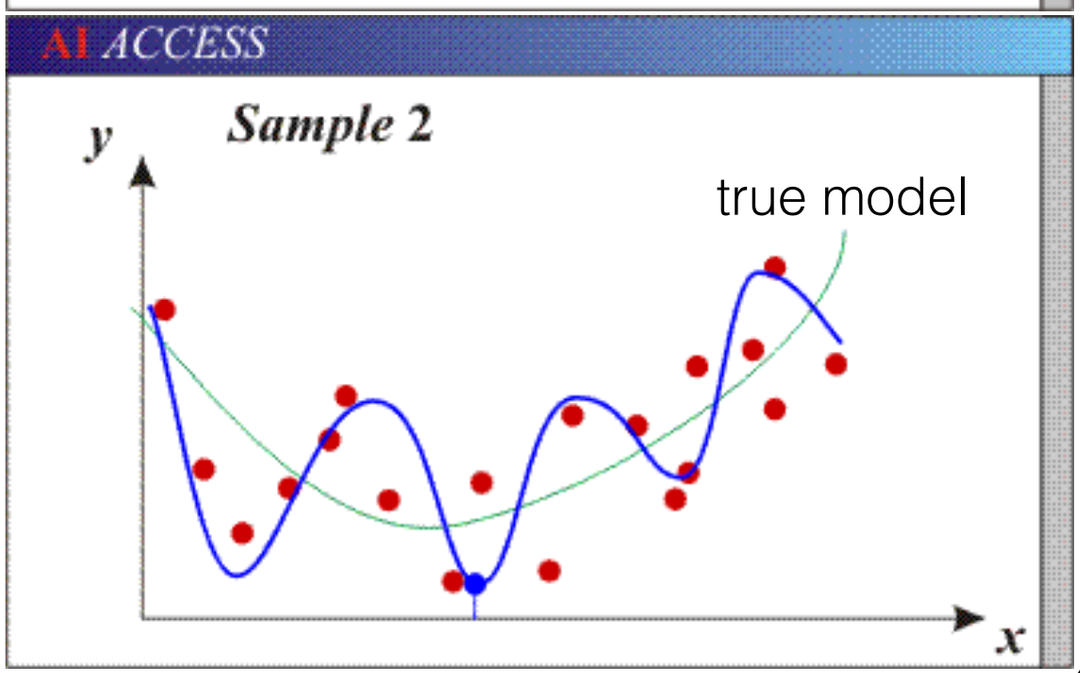
- Choose a simple classifier
- Regularize the parameters
- Get more training data

Generalization

- Undercutting: model is too simple and doesn't capture the relevant features of the training data
  - High bias, low variance
  - High training and test errors
- Overfitting: model is too complex and fits irrelevant features (noise) present in the training data
  - Low bias, high variance
  - Low training and high test errors

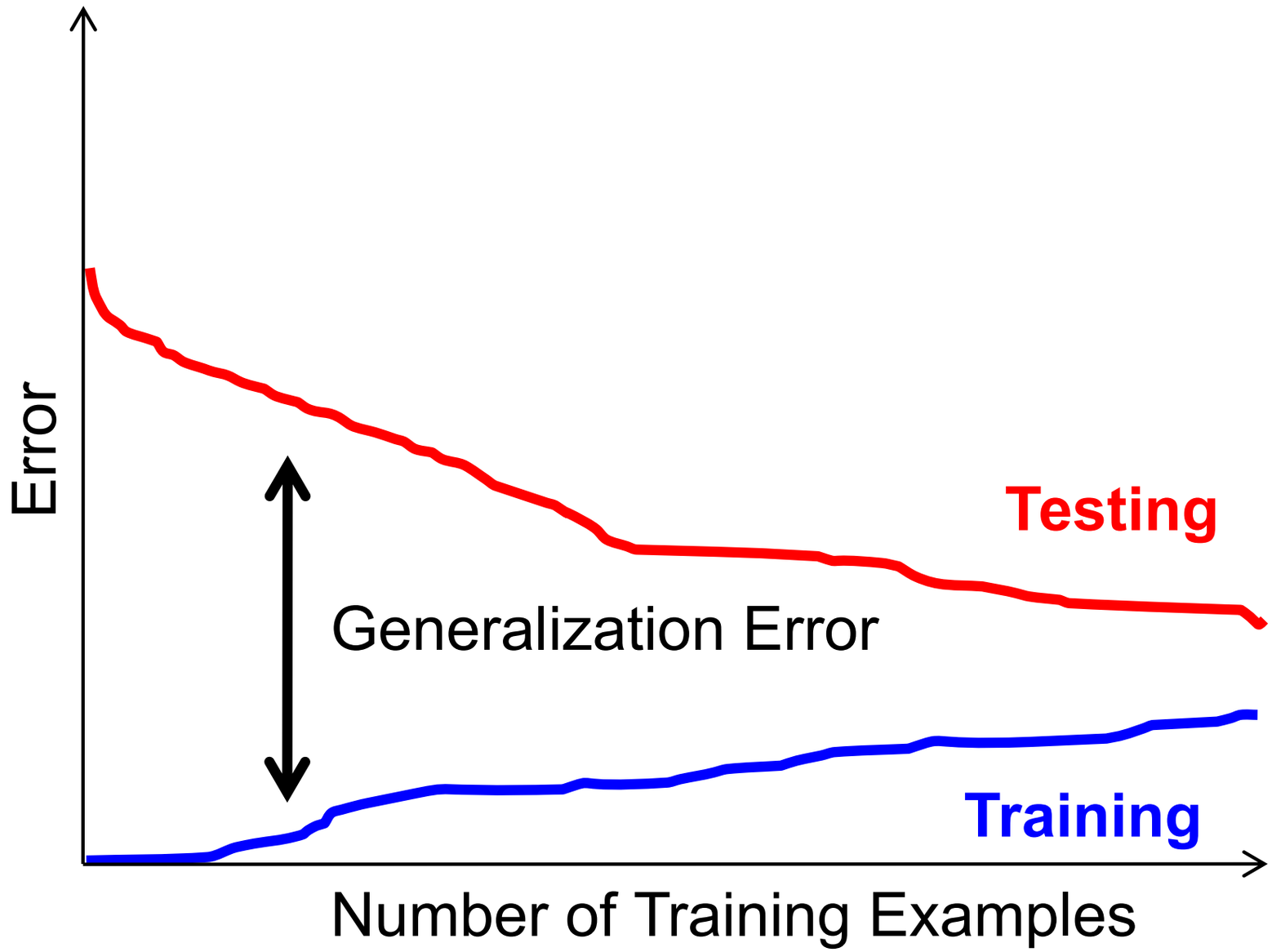
Generalization

Model with too few parameters

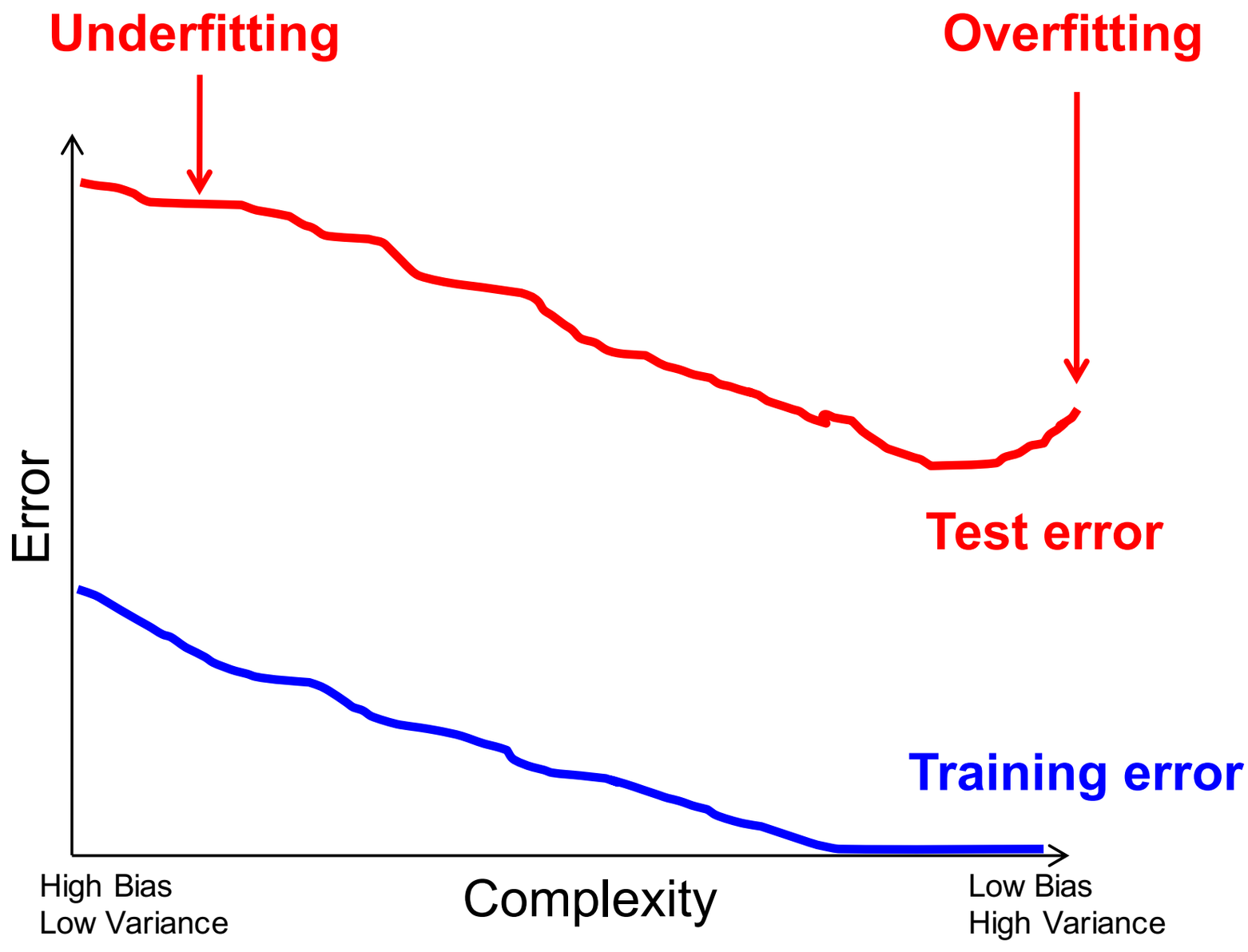


Model with too many parameters

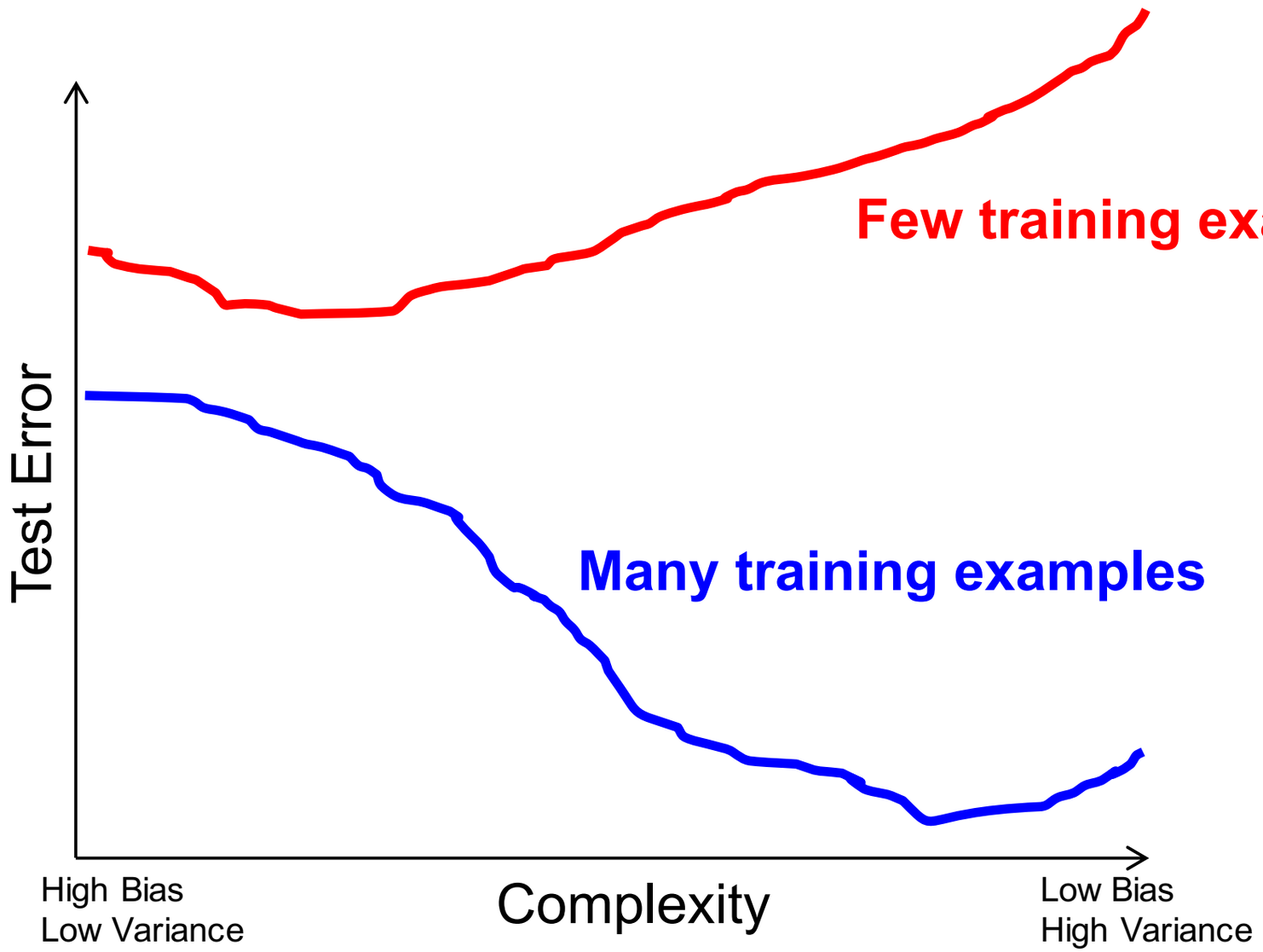
Training data size



Bias vs. Variance



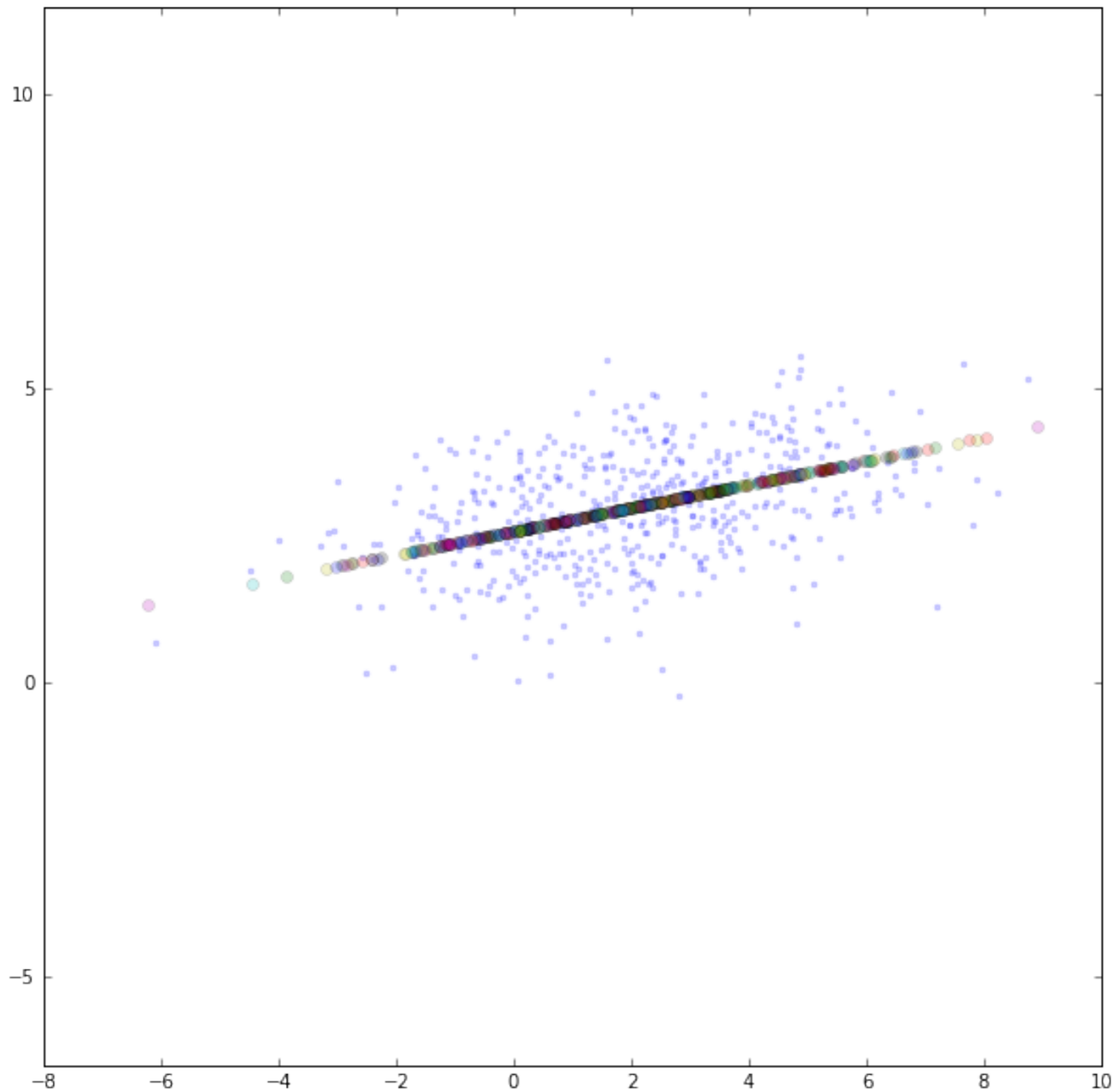
Bias vs. Variance



Types of Learning

	Supervised learning	**Unsupervised learning**
Discrete	Classification or Categorization	Clustering
Continuous	Regression	Dimensionality reduction

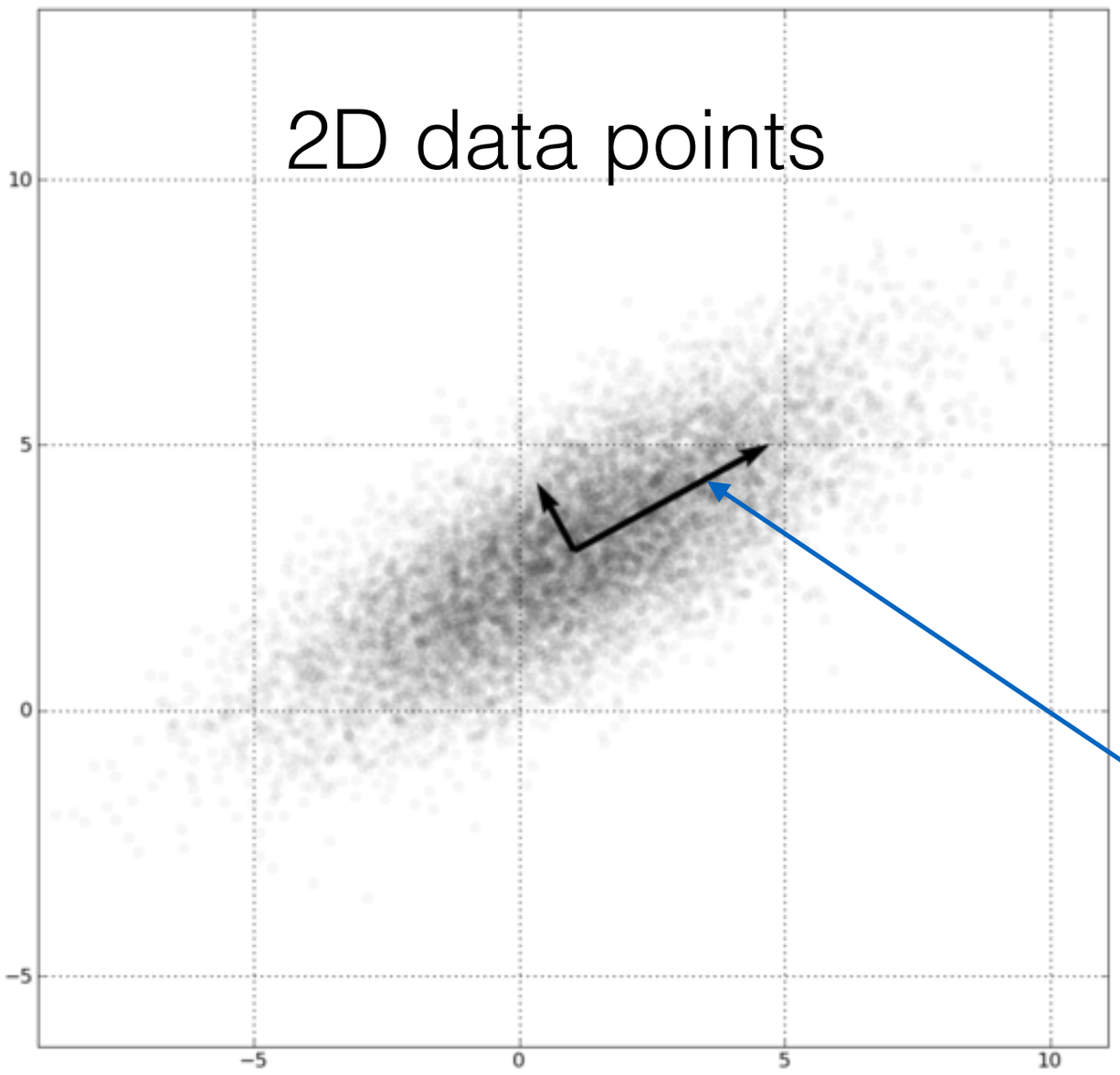
Principle Component Analysis (PCA)



Principle Component Analysis (PCA)

- PCA takes advantage of correlations in data dimensions to produce the best possible lower dimensional representations (according to the reconstruction error)
- PCA should **not** be used for discovering patterns
- PCA should **not** be used for making predictions
- PCA should be **used for dimensionality reduction**

PCA



All data points can be *projected* onto this axis, which will reduce data dimension to 1

PCA

Given n d-dimensional data points

$$\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \cdots, \mathbf{x}_n \quad \text{where} \quad \mathbf{x}_i \in R^d$$

Compute

$$\mathbf{z}_i = \mathbf{x}_i - \mu$$

Generate matrix $\mathbf{Z} \in R^{d \times n}$

$$\begin{bmatrix} \vdots & \vdots & \vdots & & \vdots \\ \mathbf{z}_1 & \mathbf{z}_2 & \mathbf{z}_3 & \cdots & \mathbf{z}_n \\ \vdots & \vdots & \vdots & & \vdots \end{bmatrix}$$

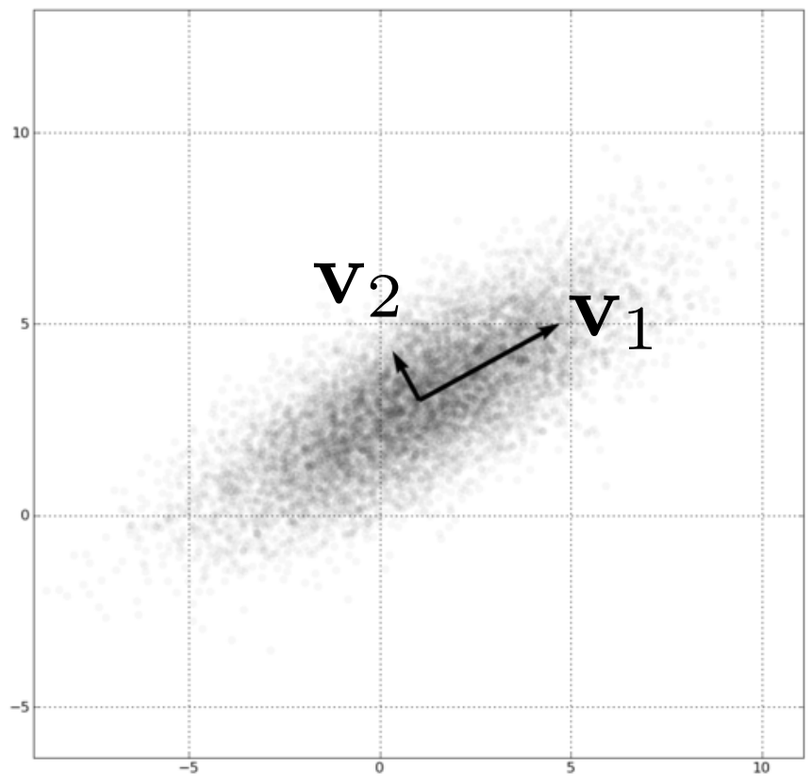
PCA

Compute $\mathbf{Z}\mathbf{Z}^T \in R^{d \times d}$

Compute **eigenvectors** and **eigenvalues** of $\mathbf{Z}\mathbf{Z}^T$

$\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_d$ $\lambda_1, \lambda_2, \cdots, \lambda_d$

$\mathbf{v}_i \in R^d$ $\lambda_1 \geq \lambda_2 \geq \lambda_3 \cdots \lambda_d$



PCA

Construct a projection matrix consisting of the eigenvectors corresponding to top-k (k << n) eigenvalues

$$\mathbf{P} = \begin{bmatrix} \vdots & \vdots & \vdots & & \vdots \\ \vdots & \vdots & \vdots & & \vdots \\ \mathbf{v}_1 & \mathbf{v}_2 & \mathbf{v}_3 & \cdots & \mathbf{v}_k \\ \vdots & \vdots & \vdots & & \vdots \\ \vdots & \vdots & \vdots & & \vdots \end{bmatrix} \in R^{d \times k}$$

PCA

Project n, d dimensional points into k dimensional space (k << n) as follows

$$\mathbf{x}'_i = \mathbf{P}^T \mathbf{z}_i$$

Dimensionality reduction
data dimension has been reduced from n to k, where k << n

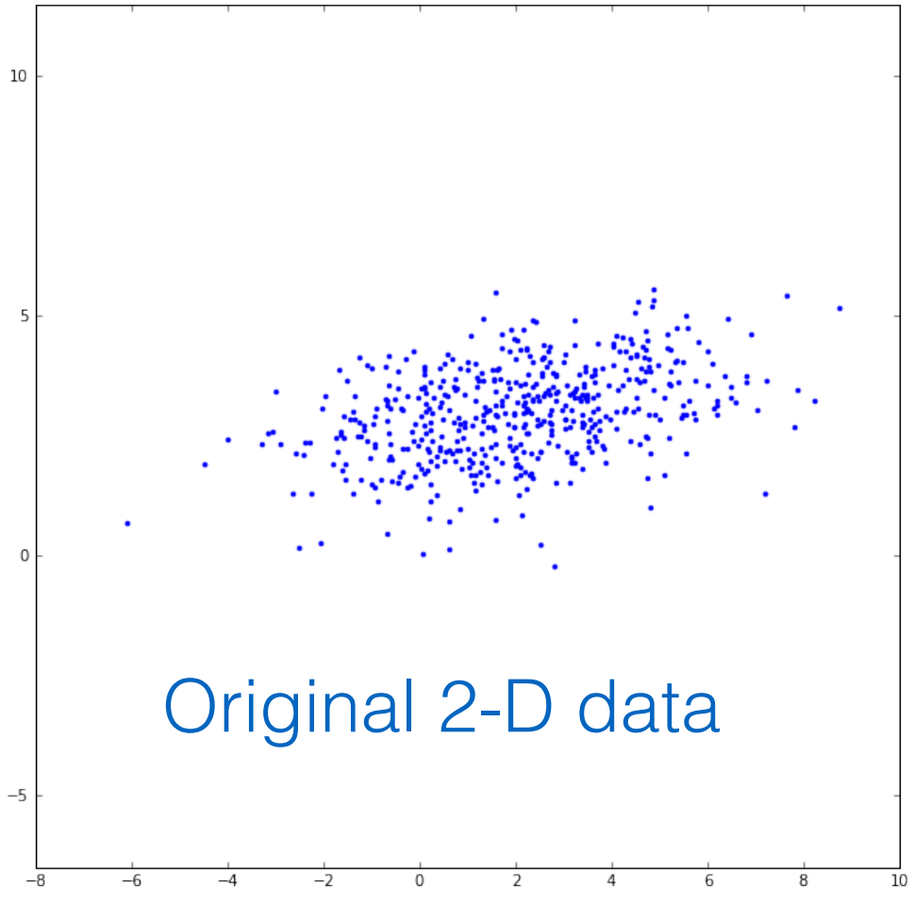
PCA

Reconstruct n-dimensional points from k-dimensional points

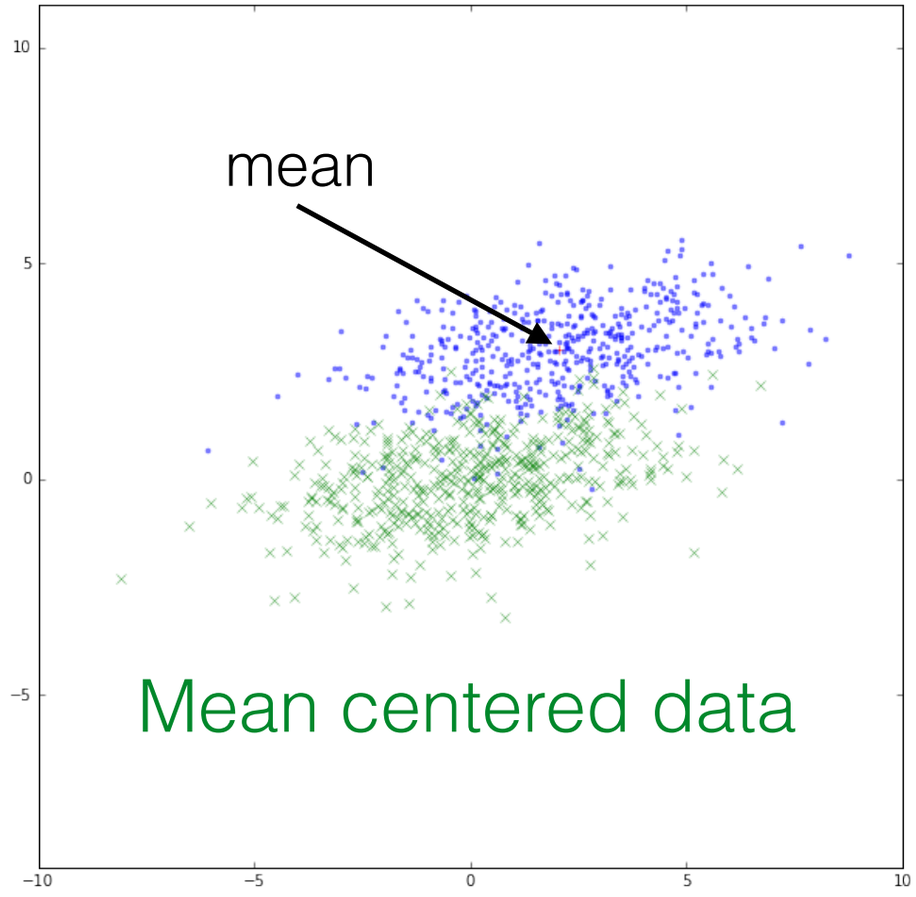
$$\mathbf{x}_i^{\text{reconstructed}} = \mathbf{P}\mathbf{x}'_i + \mu$$

Lossy Reconstruction

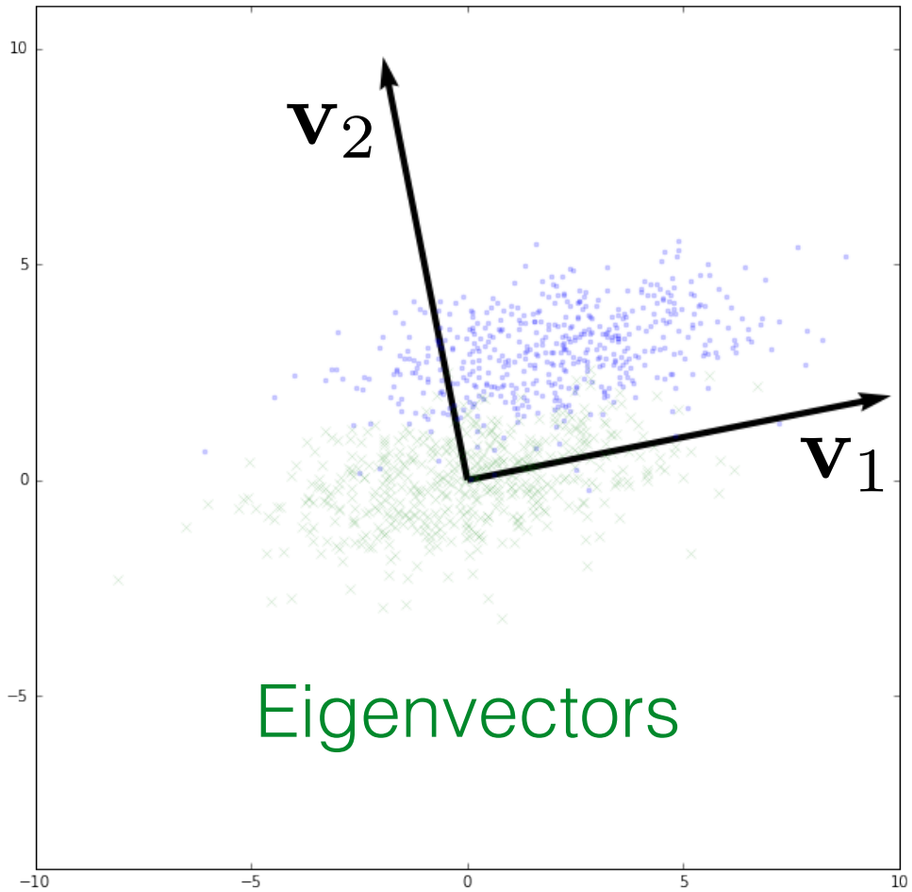
We can also reconstruct the original d-dimensional data from k-dimensional data



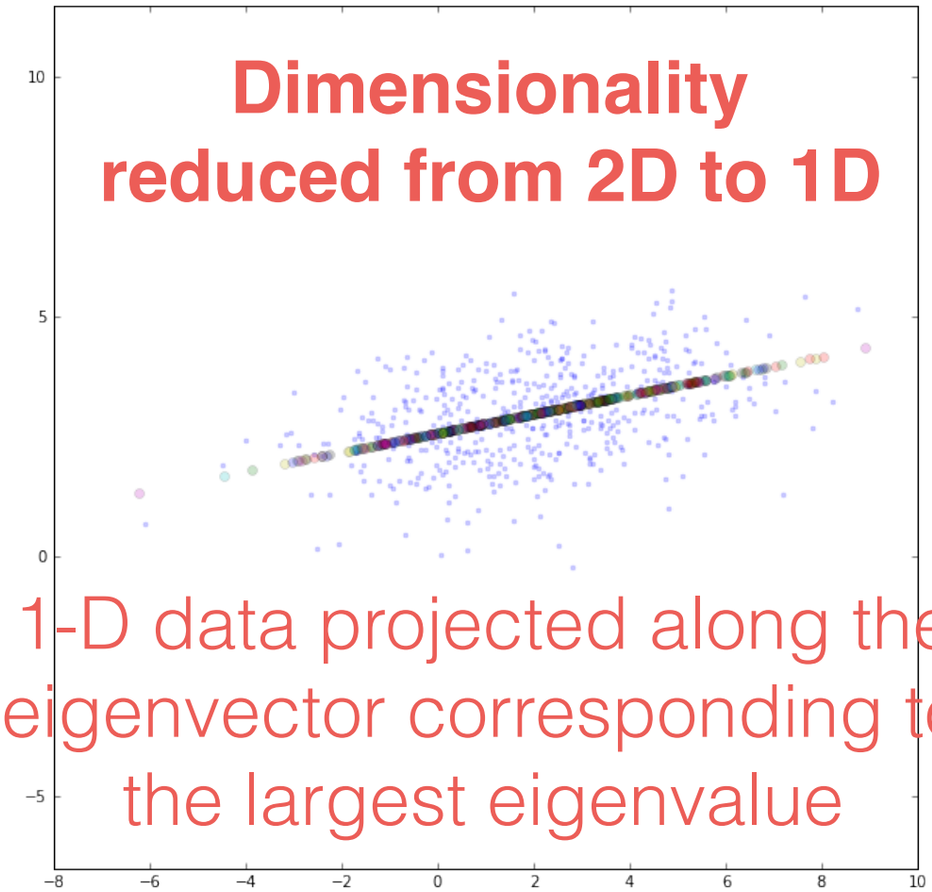
Original 2-D data



Mean centered data



Eigenvectors



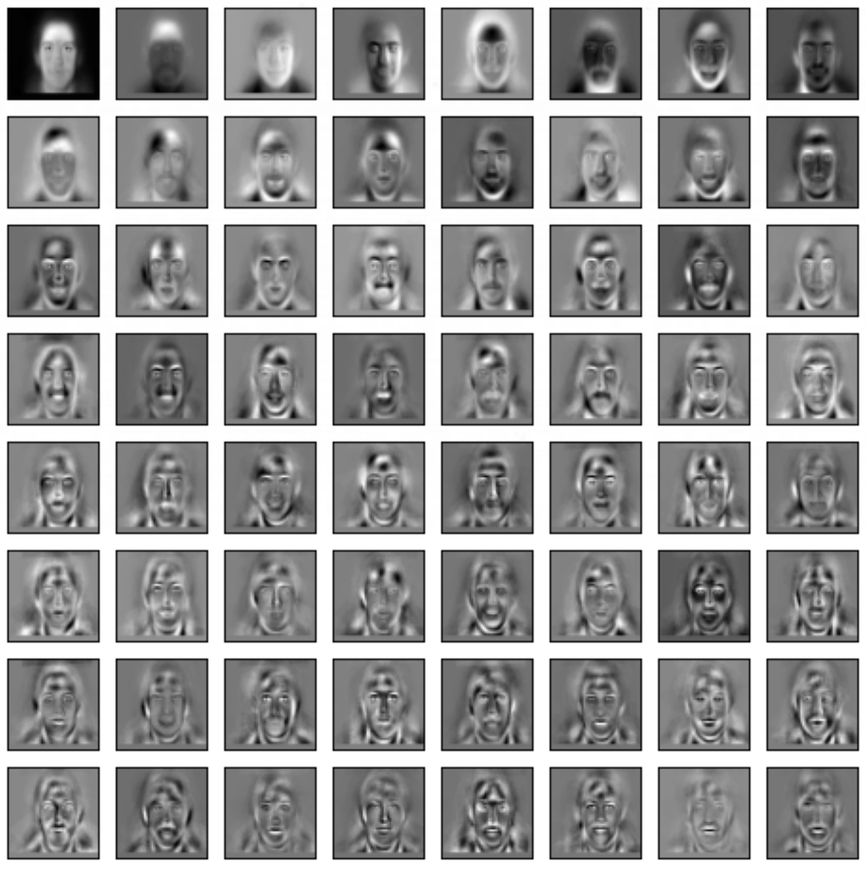
Dimensionality reduced from 2D to 1D

1-D data projected along the eigenvector corresponding to the largest eigenvalue

PCA Applications: Face Recognition

Eigenfaces for Recognition

Matthew Turk and Alex Pentland
Vision and Modeling Group
The Media Laboratory
Massachusetts Institute of Technology



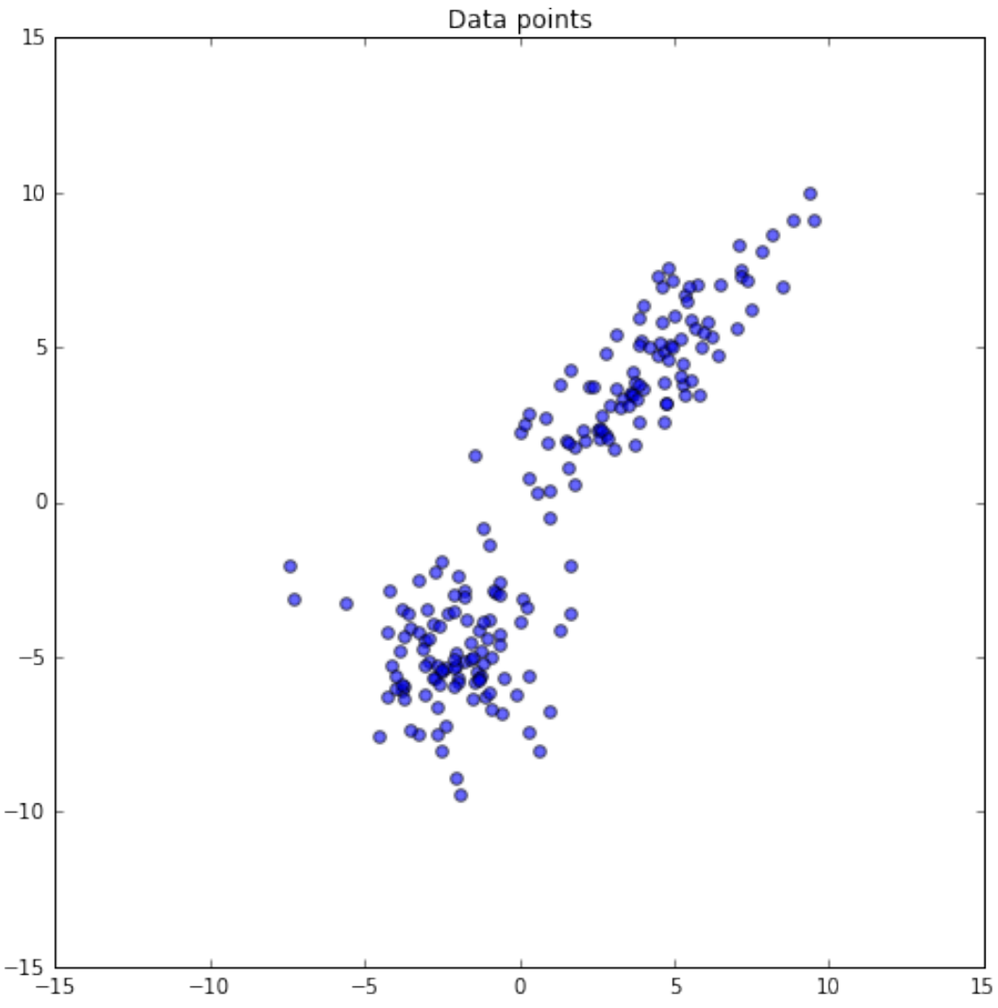
M. A. Turk and A. P. Pentland, "Face recognition using eigenfaces," Proceedings. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Maui, HI, 1991, pp. 586-591.

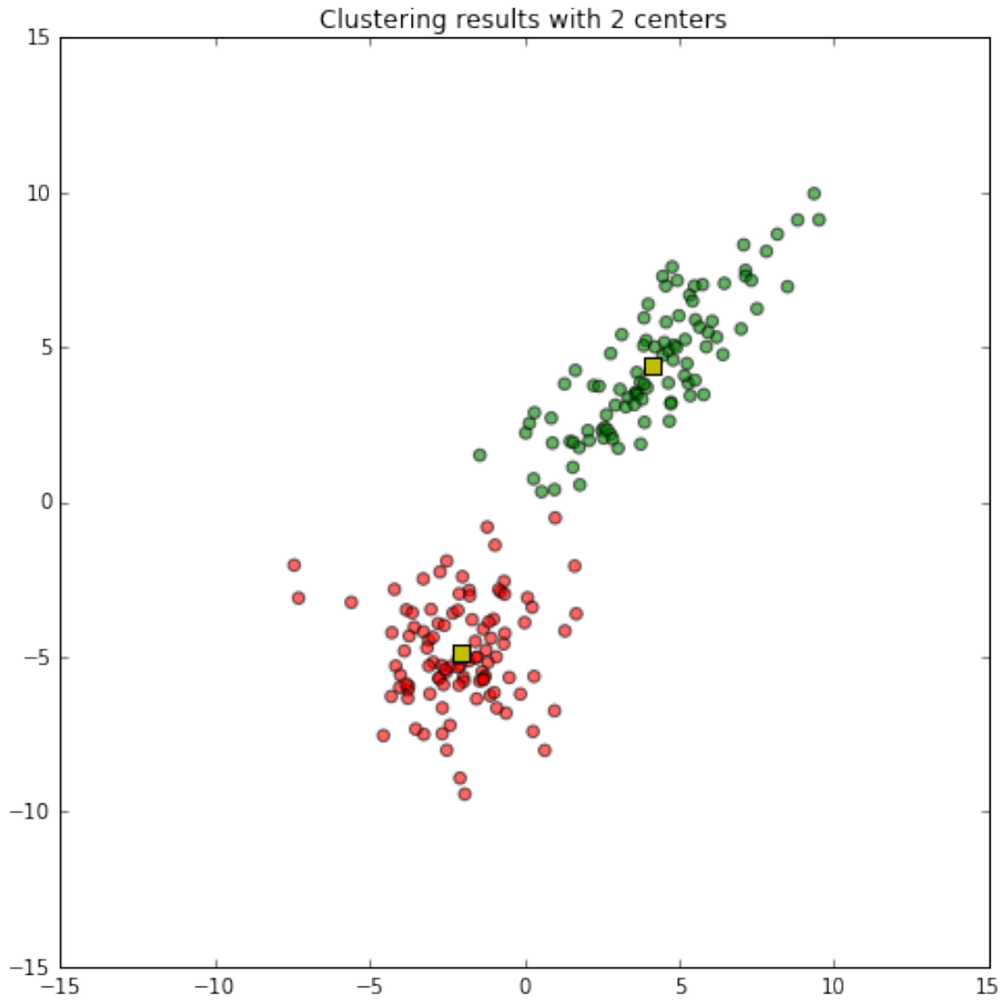
Types of Learning

	Supervised learning	**Unsupervised learning**
Discrete	Classification or Categorization	Clustering
Continuous	Regression	Dimensionality reduction

Clustering

- Group together similar points (items), and represent them with a single token





Clustering

- Group together similar points (items), and represent them with a single token

- Key challenges
  - What makes two points/images/patches/items similar?
  - How can we compute an overall grouping from pairwise similarities?

Slide credit: Derek Hoiem

Why perform clustering?

- Summarizing data
- Counting
- Segmentations
- Prediction

- Look at large amounts of data
- Represent high-dimensional vectors with a cluster number

- Histograms (texture, SIFT vectors, color, etc.)

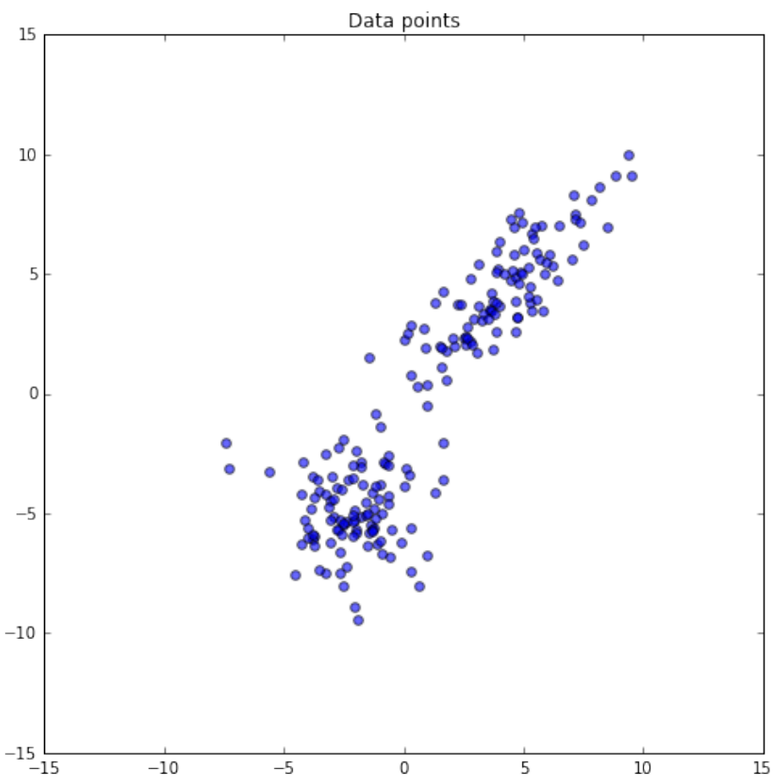
- Separate image into different regions

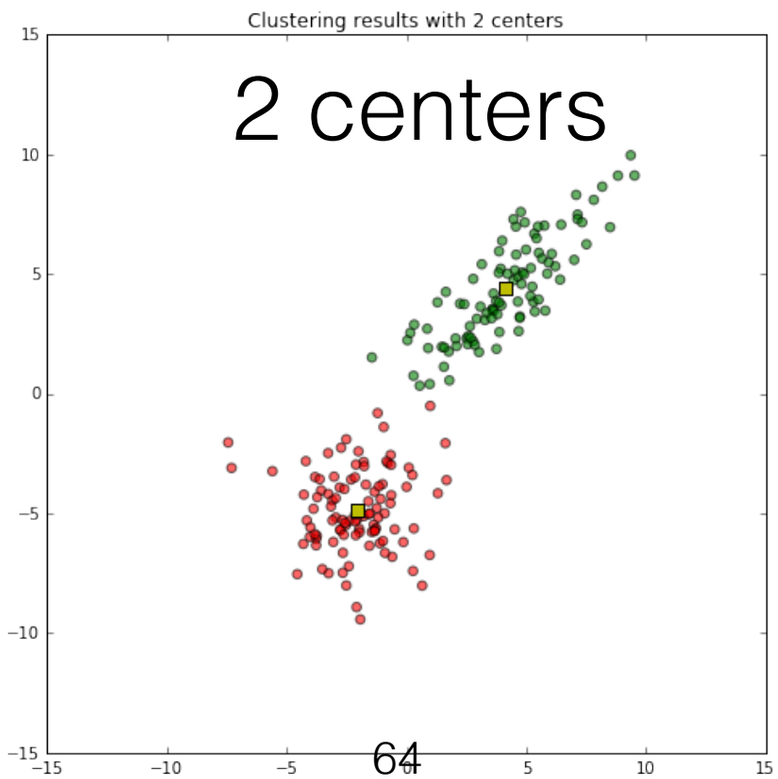
- Images recognition: image in the same cluster may have the same labels

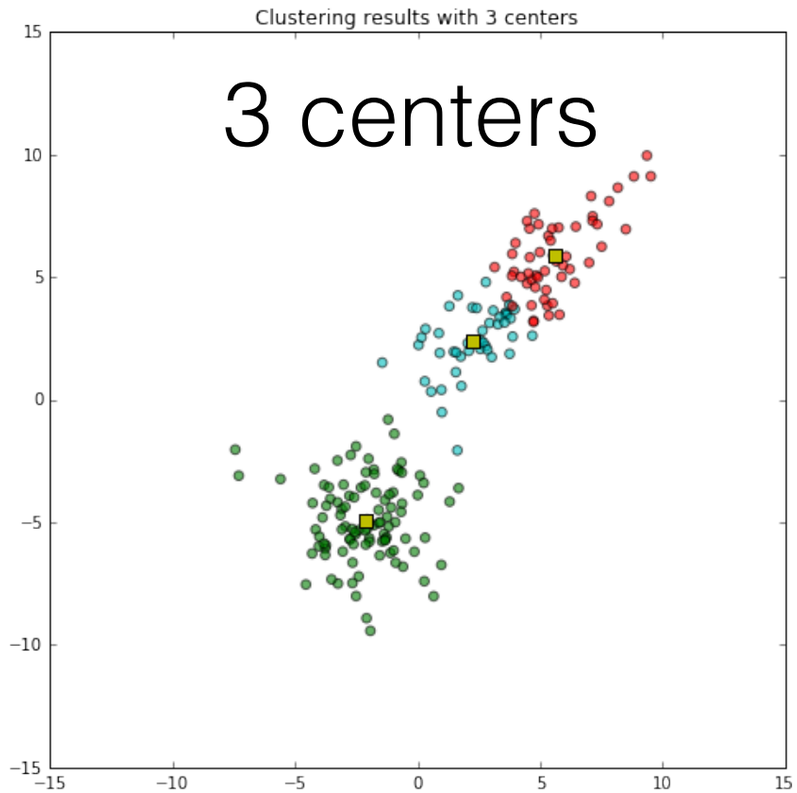
Clustering methods

- K-means
- Agglomerative clustering
- Mean-shift clustering
- Spectral clustering

K-means Clustering







K-means Clustering

- Objective: cluster to minimize variance in data given clusters
  - Preserve information

Given n data points: $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \cdots, \mathbf{x}_n$

Find k cluster centers

$$\mathbf{c}^*, \delta^* = \underset{\mathbf{c}, \delta}{\operatorname{argmin}} \frac{1}{n} \sum_{j}^{n} \sum_{i}^{k} \delta_{ij} \left(\mathbf{c}_i - \mathbf{x}_j \right)^2$$

cluster center i

data point j

whether or not data point j is assigned to cluster i

K-means Clustering

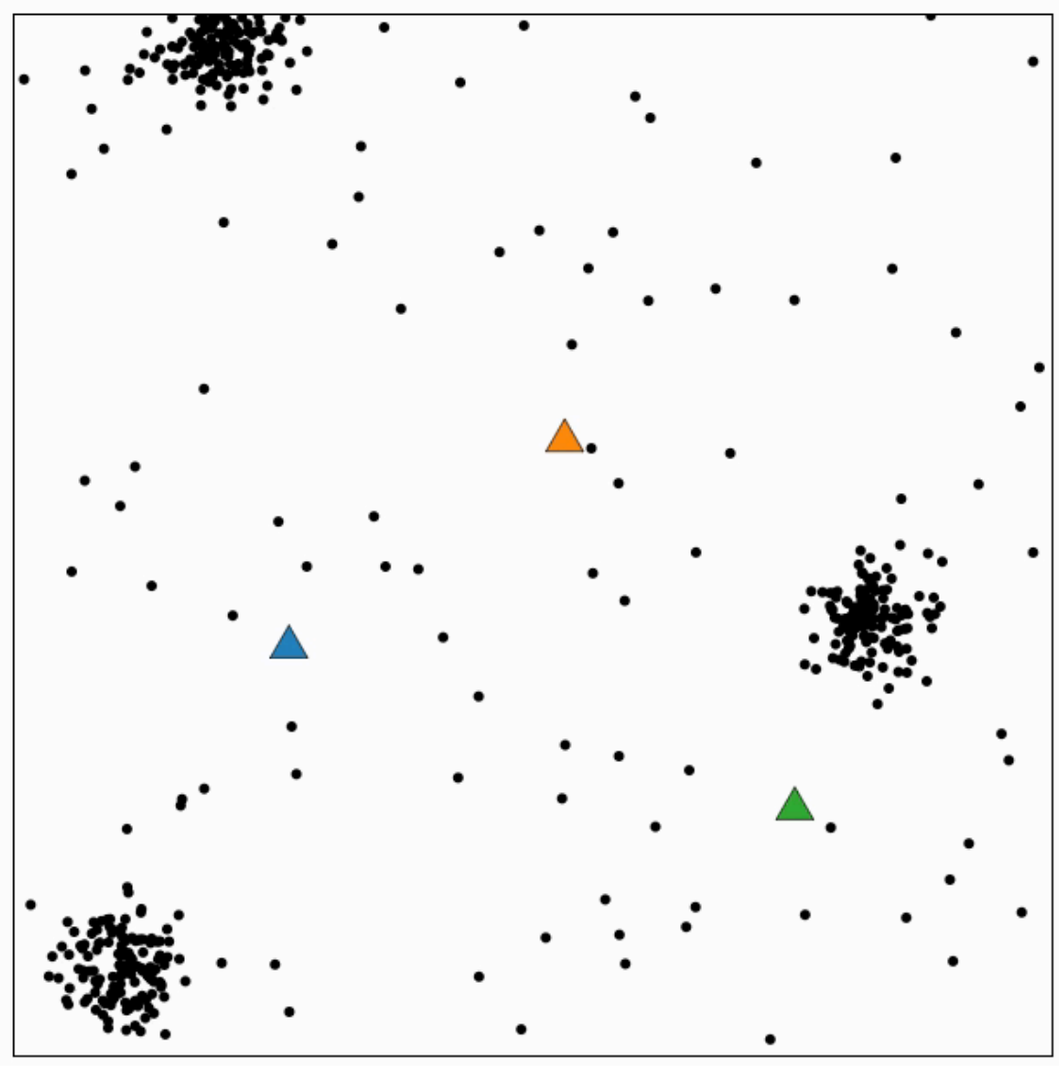
Algorithm

Initialize (randomly) centroids

Find closest centroid to each point. Group points that share the same centroid

Update each centroid to be the mean of the points in its group

Loop until convergence (number of iterations reached or centroids don't move)



http://stanford.edu/class/ee103/visualizations/kmeans/kmeans.html

K-means Clustering

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

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 the cluster enters as the mean of the points that belong to it

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

4. Repeat steps 2 and 3, until convergence is achieved

K-means Clustering

- Initialization
  - Randomly select k points as initial cluster centers
  - Greedily select k points to minimize residual
  - What if a cluster center sits on a data point?
- Distance/similarity measures
  - Euclidean, others …
- Optimization
  - Cannot guarantee that it will converge to *global minima*
  - Multiple restarts
- Choice of K?

Image Segmentation

K-means clustering using intensity or color

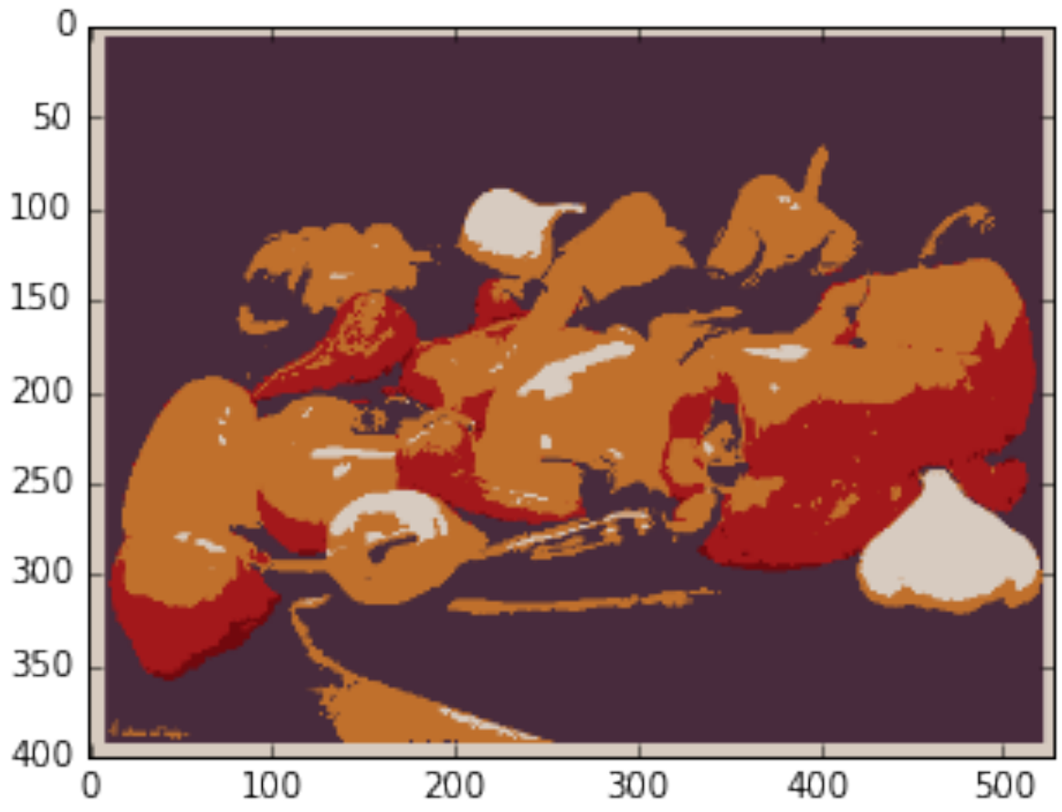
Image

Clusters on intensity

Clusters on color

Image Segmentation

Image



Each pixel is replaced by its cluster centre. The number of cluster is set to 5. Using RGB values.

K-means Clustering

- Pros
  - Find cluster centres that are good representation of data (reduces conditional variance)
  - Simple, fast* and easy to implement
- Cons
  - Need to select the number of clusters
  - Sensitive to outliers
  - Can get stuck in local minima
  - All clusters have the same parameters, i.e., distance/similarity measure is non-adaptive
  - *Each iteration is O(knd) for n, d-dimensional points, so it can be slow
- K-means is rarely used to image segmentation (pixel segmentation)

Commonly used distance/ similarity measures

- P-norms
  - City block (L1)
  - Euclidean (L2)
  - L-infinity
- Mahalanobis distance
  - Scaled Euclidean
- Cosine Distance

$$\|\mathbf{x}\|_p := \left(\sum_{i=1}^{n} |x_i|^p\right)^{1/p}.$$

$$\|\boldsymbol{x}\|_1 := \sum_{i=1}^{n} |x_i|.$$

$$\|\boldsymbol{x}\| := \sqrt{x_1^2 + \cdots + x_n^2}.$$

$$\|\mathbf{x}\|_\infty := \max\left(|x_1|, \ldots, |x_n|\right).$$

$$d(\vec{x}, \vec{y}) = \sqrt{\sum_{i=1}^{N} \frac{(x_i - y_i)^2}{\sigma_i^2}},$$

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|}$$

Here x_i is the distance between two points

Slide credit: James Hayes

How many cluster centers?

- Validation set
  - Try different numbers of clusters and look at performance

Evaluating Clusters

- Generative
  - How well are points reconstructed from the clusters?
- Discriminative
  - How well do the clusters correspond to labels? This is often termed as *purity*.
  - Unsupervised clustering doesn't aim to be discriminative

K-mediods Clustering

- Similar to K-means
  - Represent a cluster center with one of its members (data points), rather than the mean of its members
    - Choose the member (data point) that minimizes cluster similarity
- Applicable in situations where *mean* is not meaningful
  - Clustering hue values
  - Using L-infinity norm for similarity

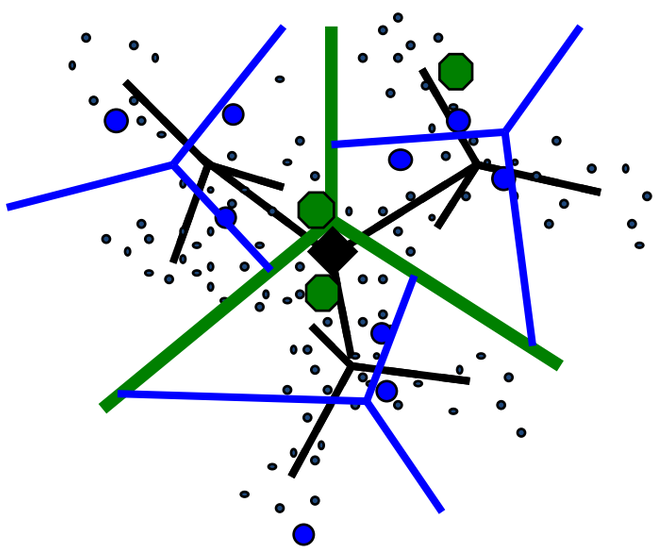
Slide credit: James Hayes

Building Visual Dictionaries

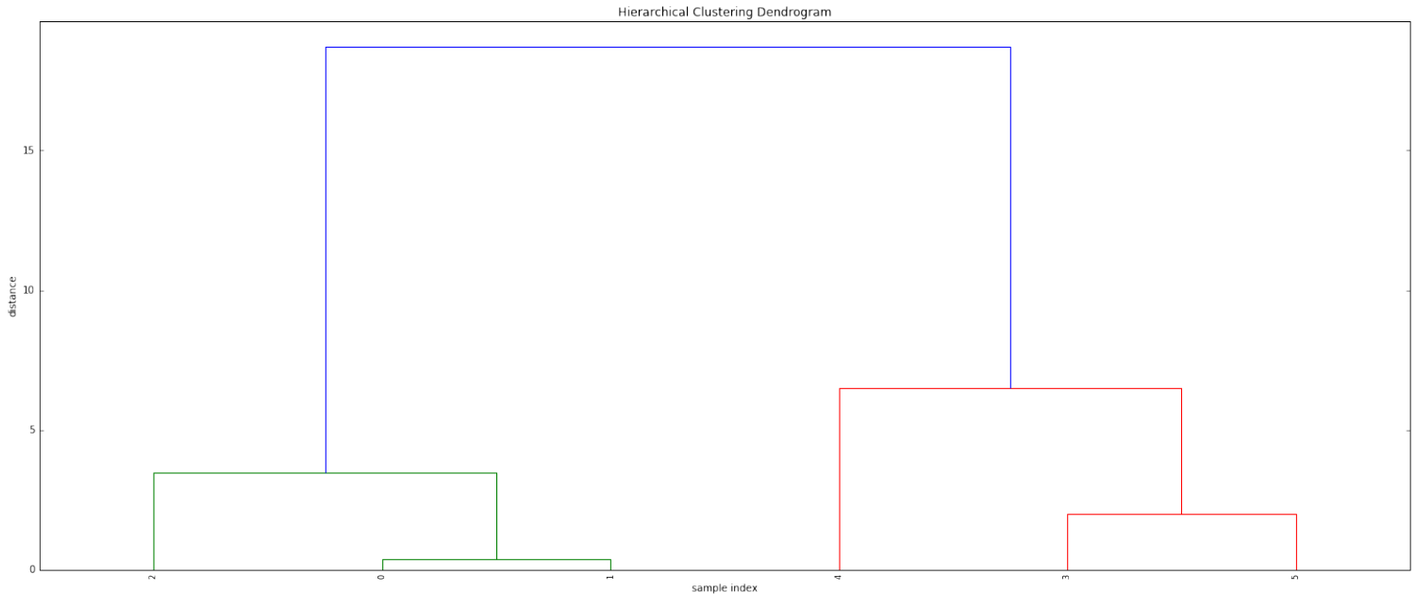
- Sample patches from a database
  - E.g., 128-dimensional SIFT features

- Cluster these patches
  - Clusters centers comprise (visual) dictionary

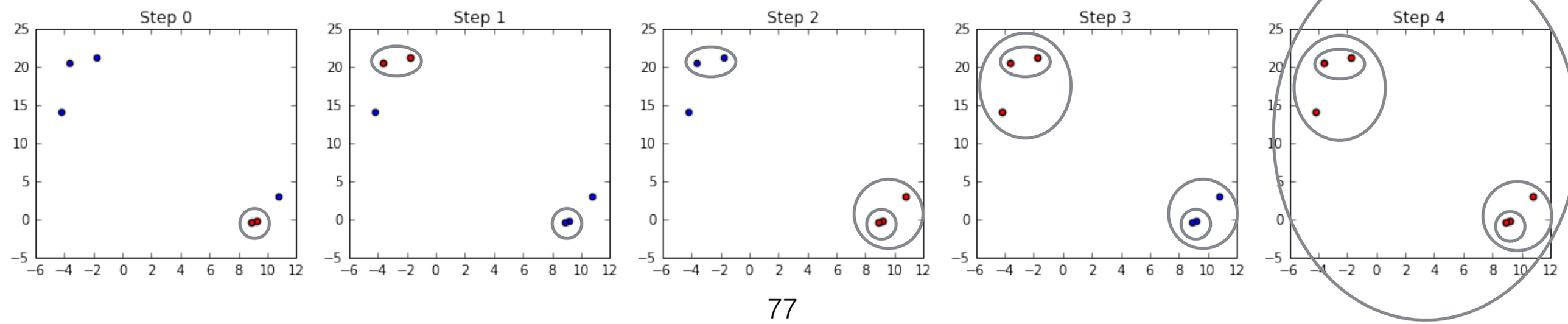
- Assign a codeword (number, cluster center) to each new patch (say 128-dimensional SIFT feature) according to the nearest cluster



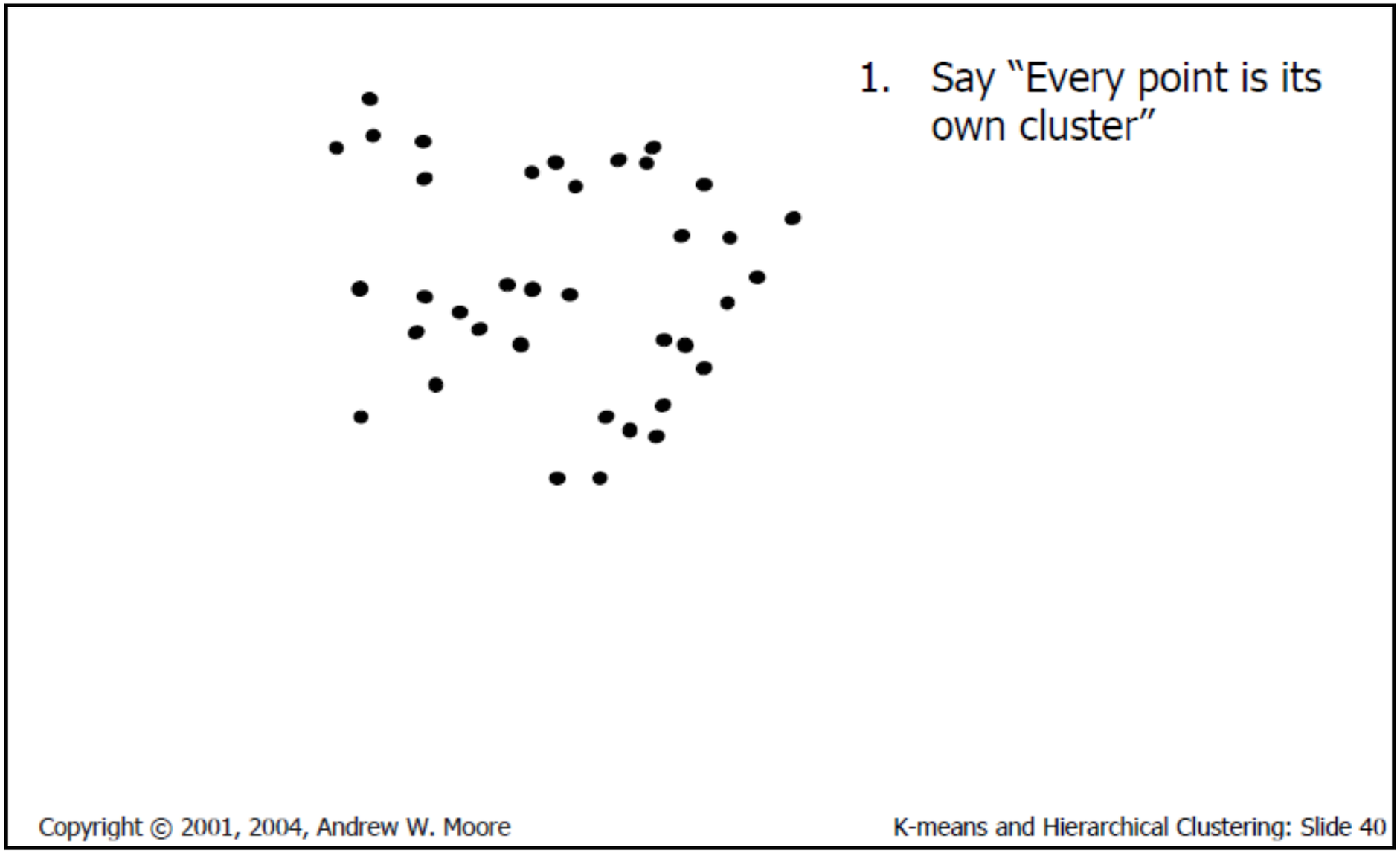
Slide credit: James Hayes



Agglomerative Clustering

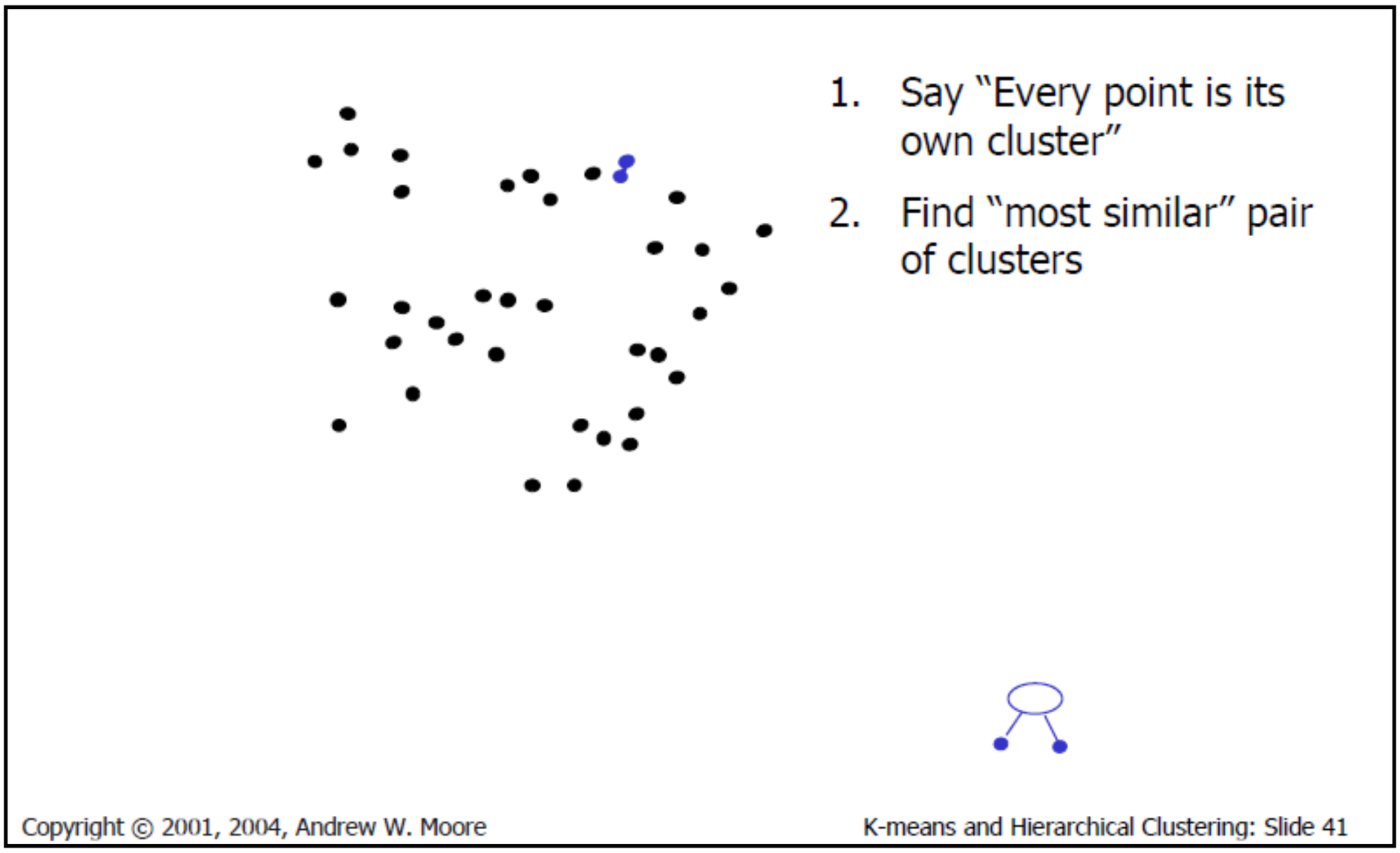


Agglomerative Clustering



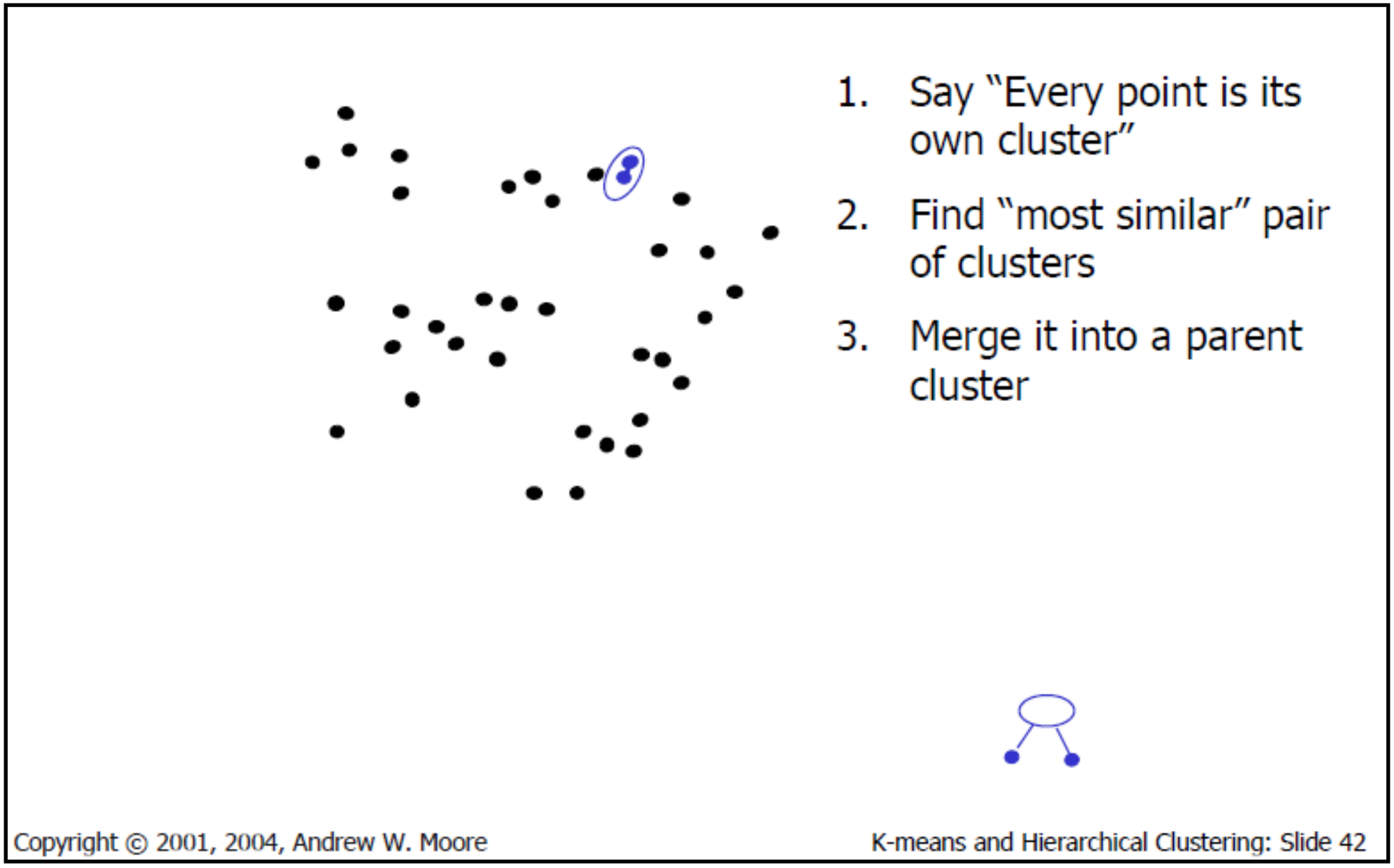
Slide credit: James Hayes

Agglomerative Clustering



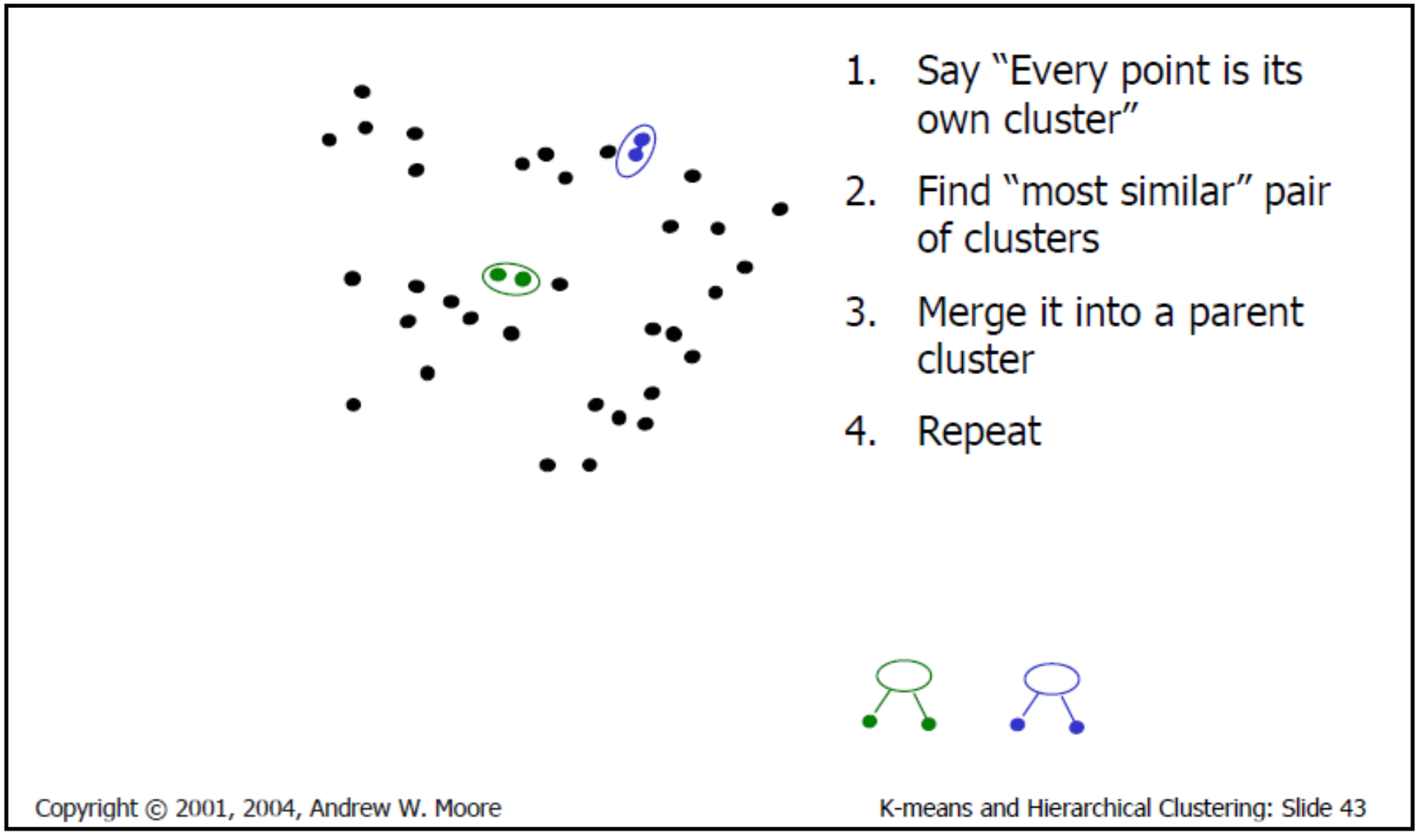
Slide credit: James Hayes

Agglomerative Clustering



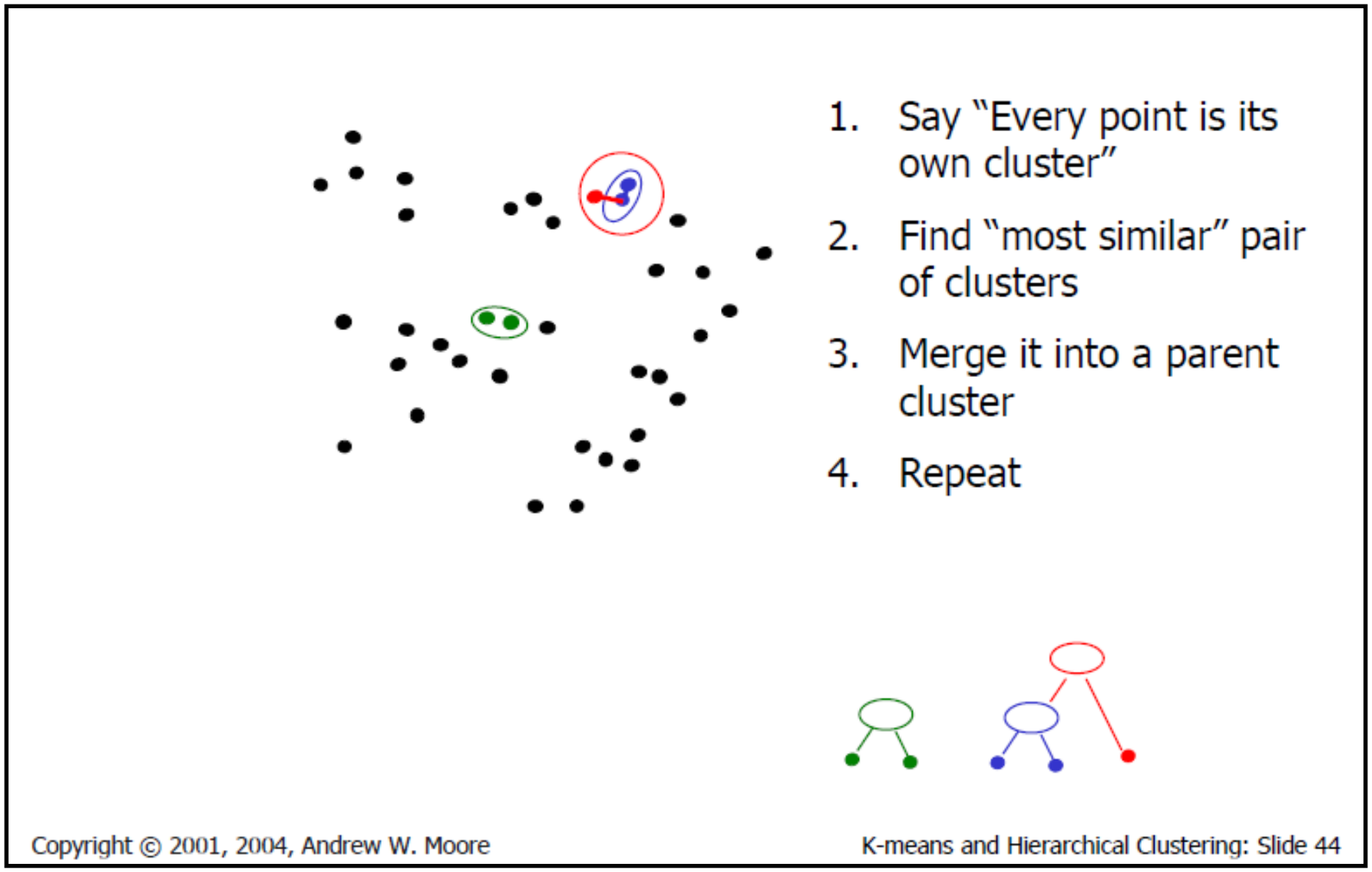
Slide credit: James Hays

Agglomerative Clustering



Slide credit: James Hays

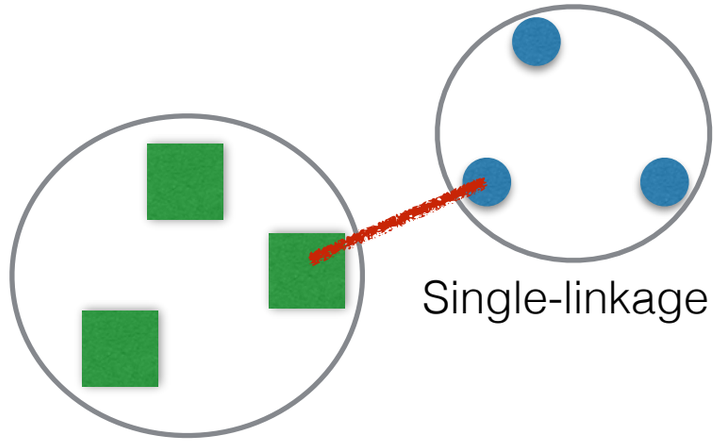
Agglomerative Clustering

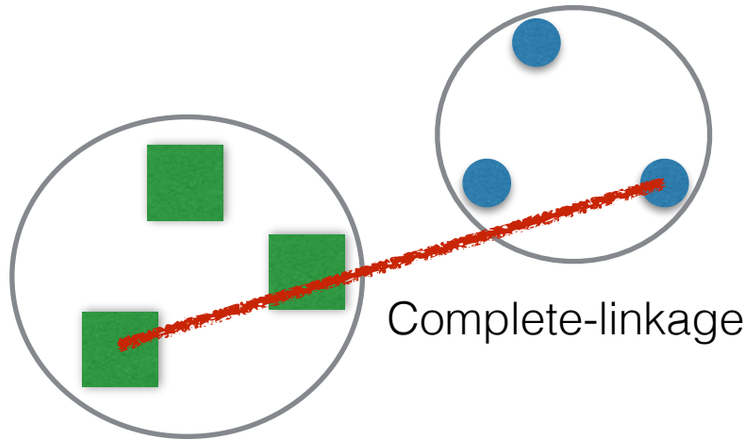


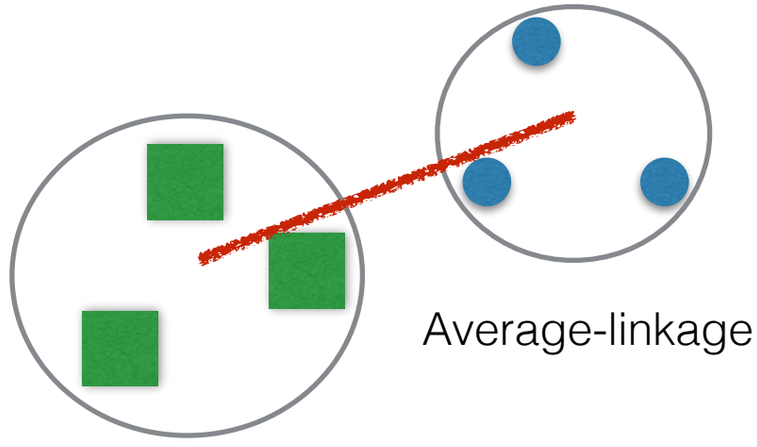
Slide credit: James Hayes

Agglomerative Clustering: Defining Cluster Similarity

- Single-linkage clustering (also called the connectedness or minimum method),
  - We consider the distance between one cluster and another cluster to be equal to the shortest distance from any member of one cluster to any member of the other cluster.
  - If the data consist of similarities, we consider the similarity between one cluster and another cluster to be equal to the greatest similarity from any member of one cluster to any member of the other cluster.
- Complete-linkage clustering (also called the diameter or maximum method)
  - We consider the distance between one cluster and another cluster to be equal to the greatest distance from any member of one cluster to any member of the other cluster.
- Average-linkage clustering
  - We consider the distance between one cluster and another cluster to be equal to the average distance from any member of one cluster to any member of the other cluster.
  - A variation on average-link clustering uses the median distance, which is much more outlier-proof than the average distance.

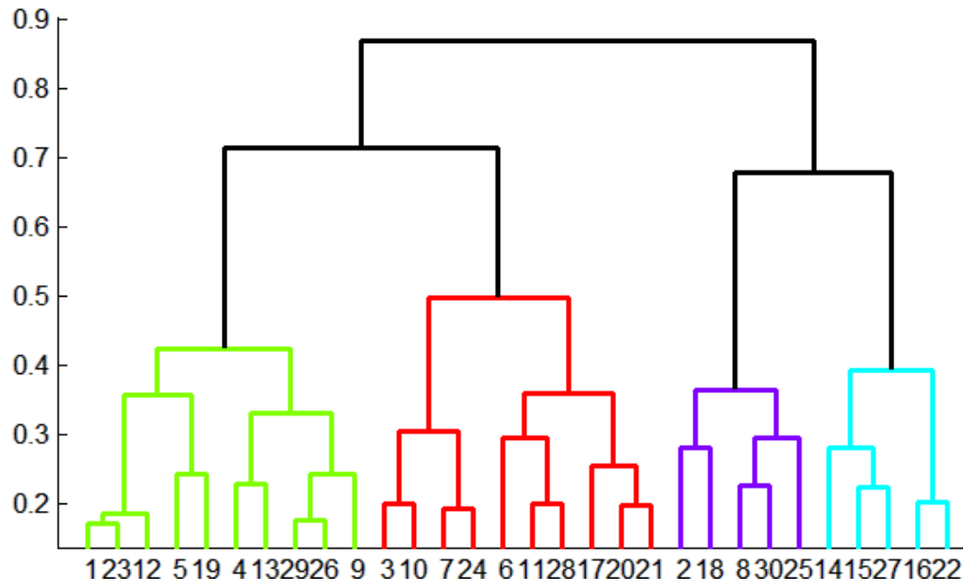






Agglomerative Clustering

- How many clusters?
  - Agglomerative clustering creates a tree (commonly referred to as a *dendrogram*)
  - Threshold based upon the maximum number of clusters
  - Threshold based upon distance of merges



dendrogram

Agglomerative Clustering

Single-linkage clustering (Johnson's algorithms)

1. *Begin with the disjoint clustering having level L(0) = 0 and sequence number m = 0.*
2. *Find the least dissimilar pair of clusters in the current clustering, say pair (r), (s), according to*

   $$d[(r),(s)] = \min d[(i),(j)]$$

   where the minimum is over all pairs of clusters in the current clustering.
3. *Increment the sequence number : m = m +1. Merge clusters (r) and (s) into a single cluster to form the next clustering m. Set the level of this clustering to*

   $$L(m) = d[(r),(s)]$$
4. *Update the proximity matrix, D, by deleting the rows and columns corresponding to clusters (r) and (s) and adding a row and column corresponding to the newly formed cluster. The proximity between the new cluster, denoted (r,s) and old cluster (k) is defined in this way:*

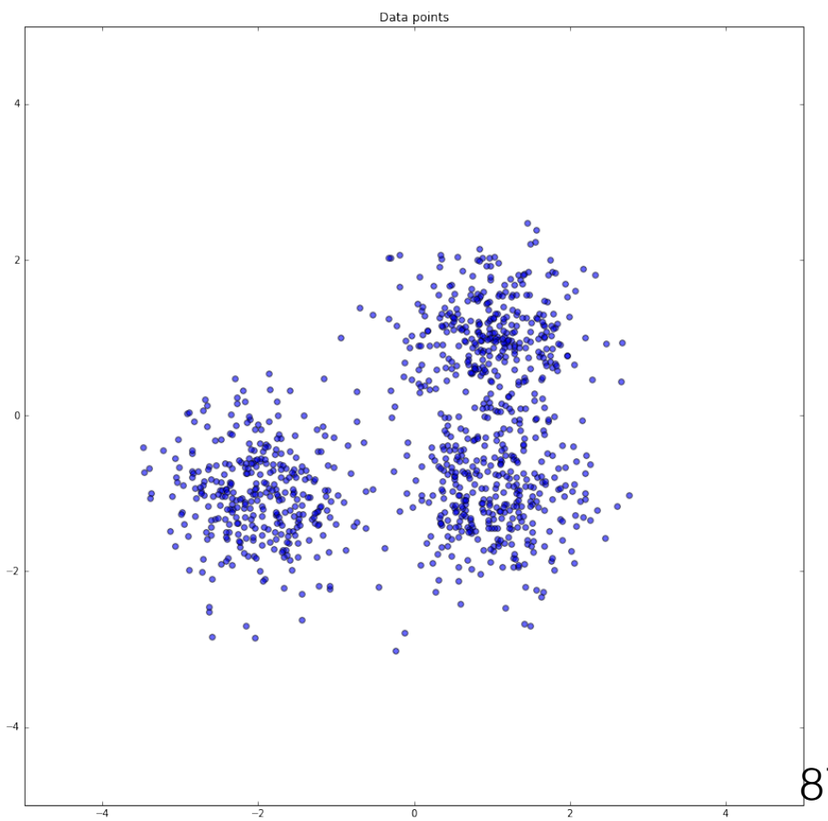
   $$d[(k), (r,s)] = \min d[(k),(r)], d[(k),(s)]$$
5. *If all objects are in one cluster, stop. Else, go to step 2.*

http://home.deib.polimi.it/matteucc/Clustering/tutorial_html/hierarchical.html

Agglomerative Clustering

- Pros
  - Simple to implement
  - Clusters have adaptive shapes
  - Provides a hierarchy of clusters
- Bad
  - These do not scale well. Time complexity is $O(n^2)$
  - May have imbalanced clusters
  - They cannot undo what was done previously
  - Need to choose the number of clusters
  - Needs to use an “ultrametric” to get meaningful hierarchy
    - Ultrametric space is special kind of metric space in which the triangle inequality is replaced with $d(x, z) \leq \max\left(d(x, y), d(y, z)\right)$

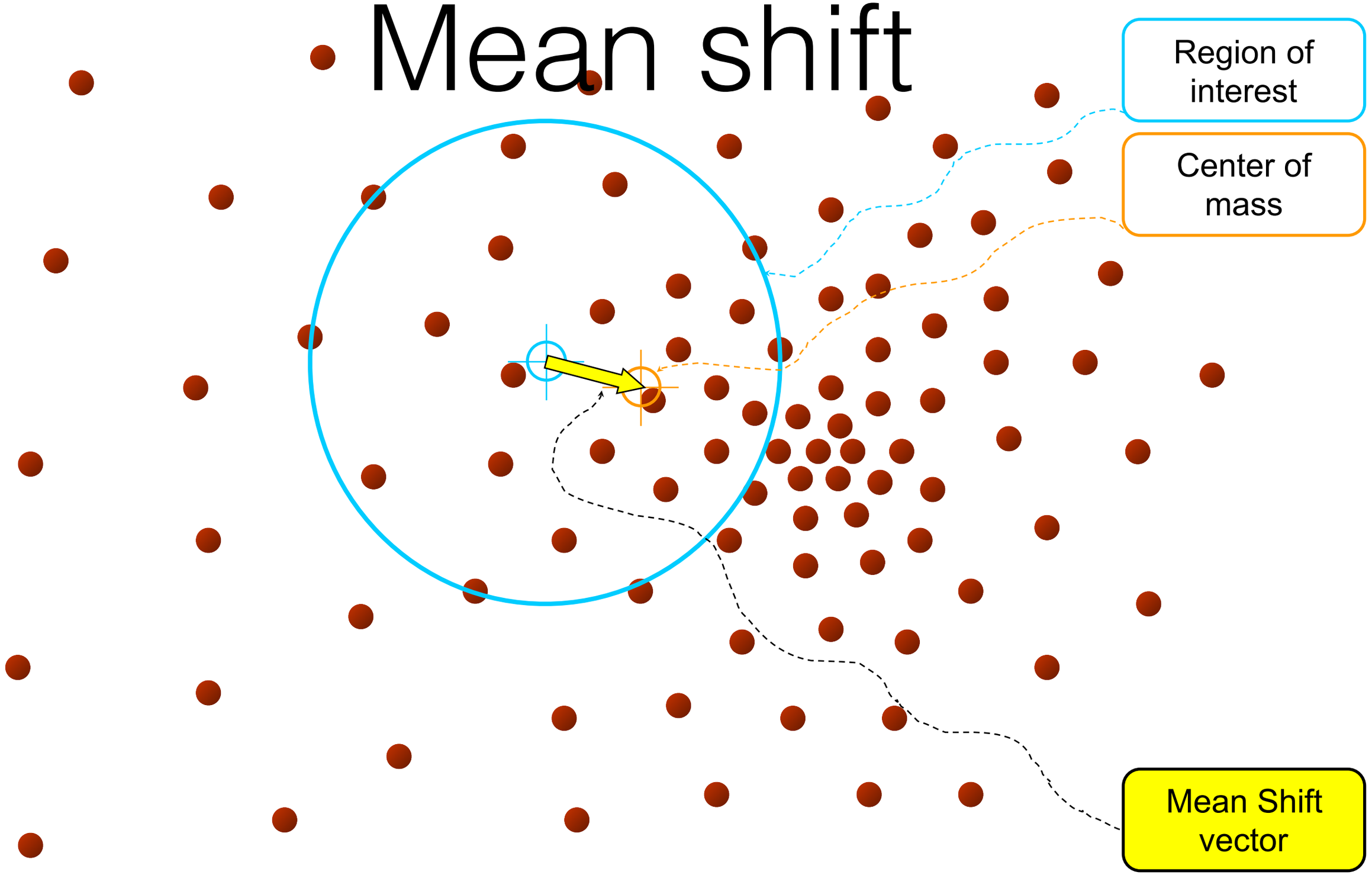
Mean Shift Clustering



Mean shift Clustering

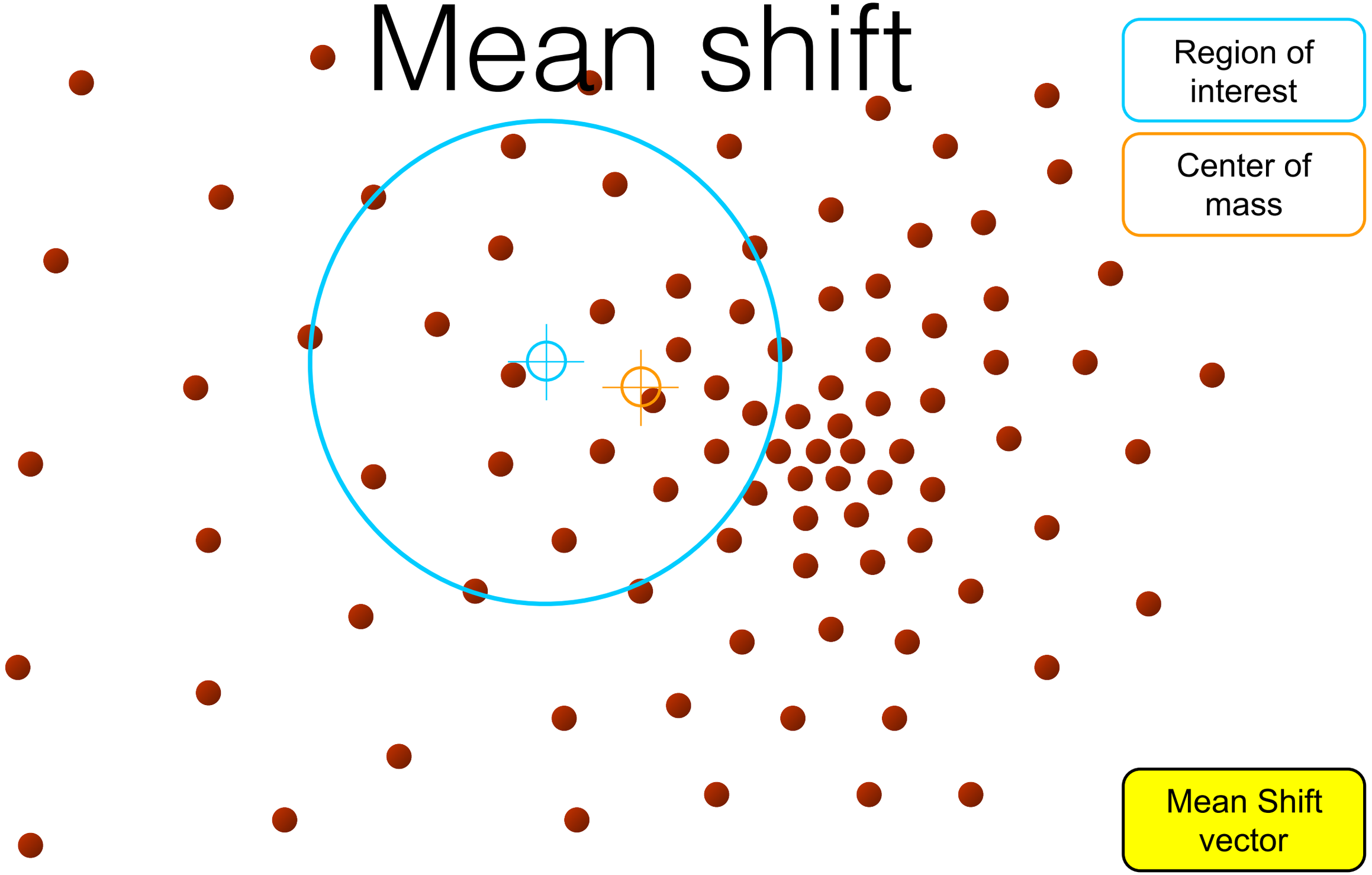
- The mean shift algorithm seeks *modes* of a given set of points
- Algorithm outline
    1. Choose kernel and bandwidth
    2. For each point
        a. Center a window on that point
        b. Compute the mean of the data in the search window
        c. Center the search window at the new mean location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

Mean shift

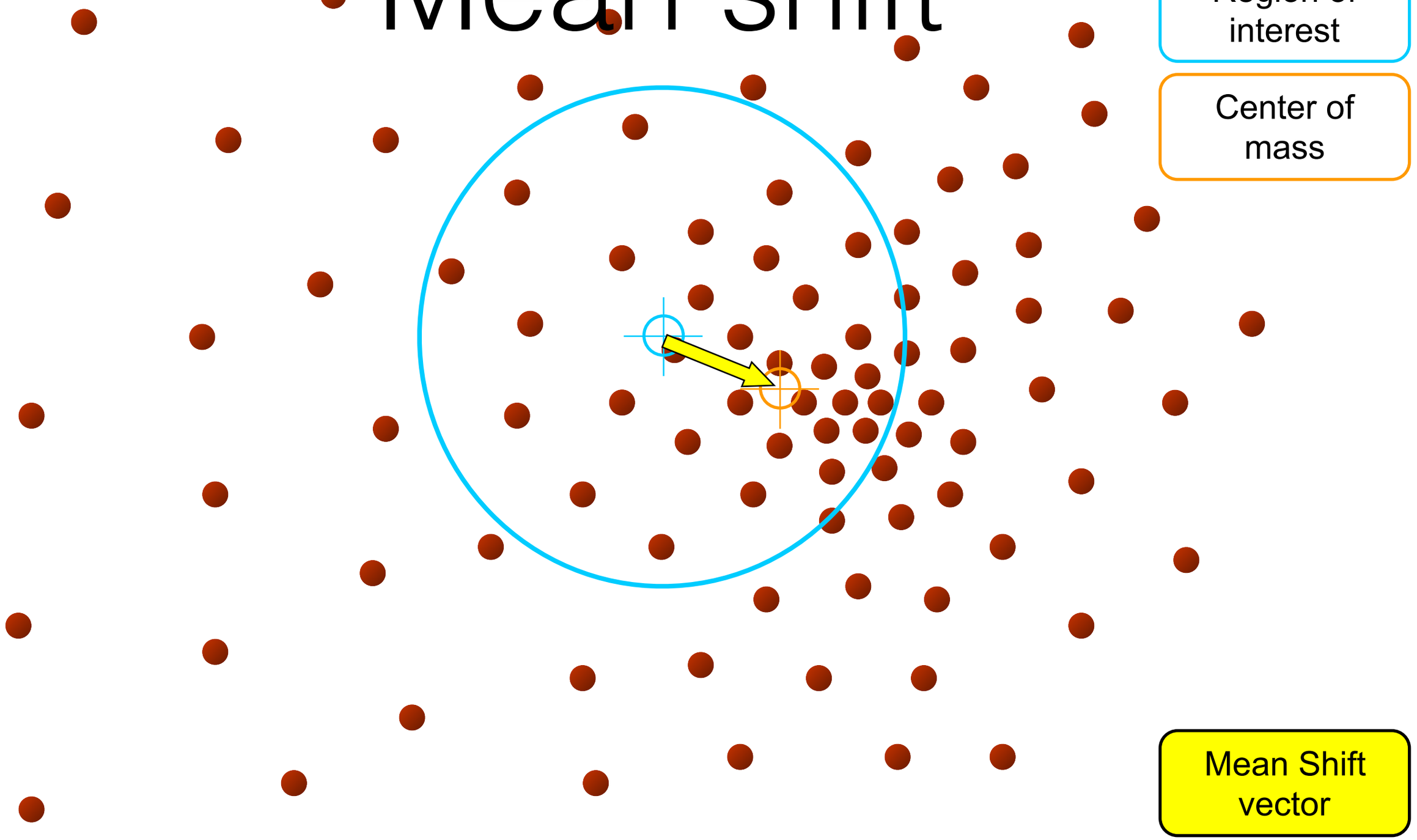


Slide by Y. Ukrainitz & B. Sarel

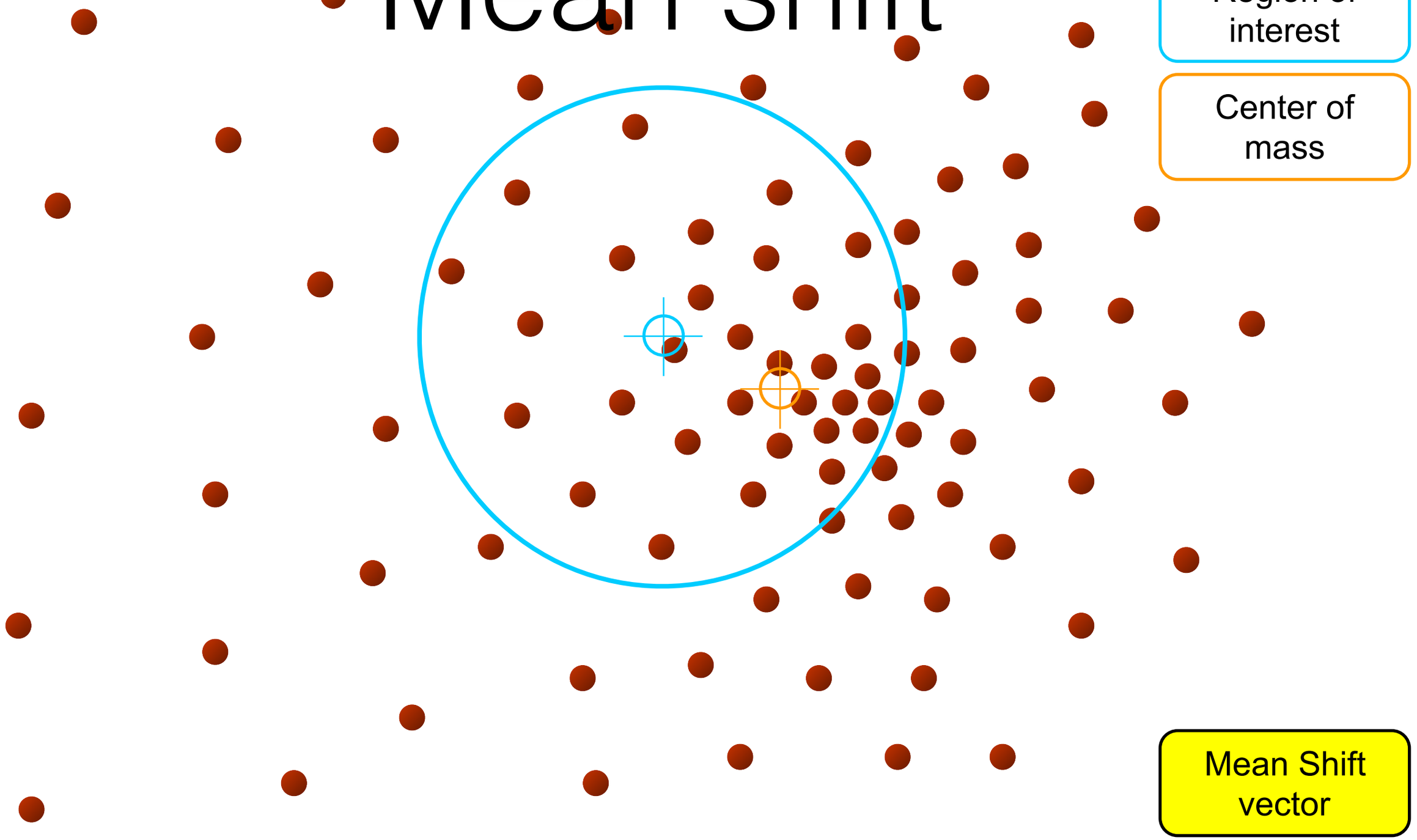
Mean shift



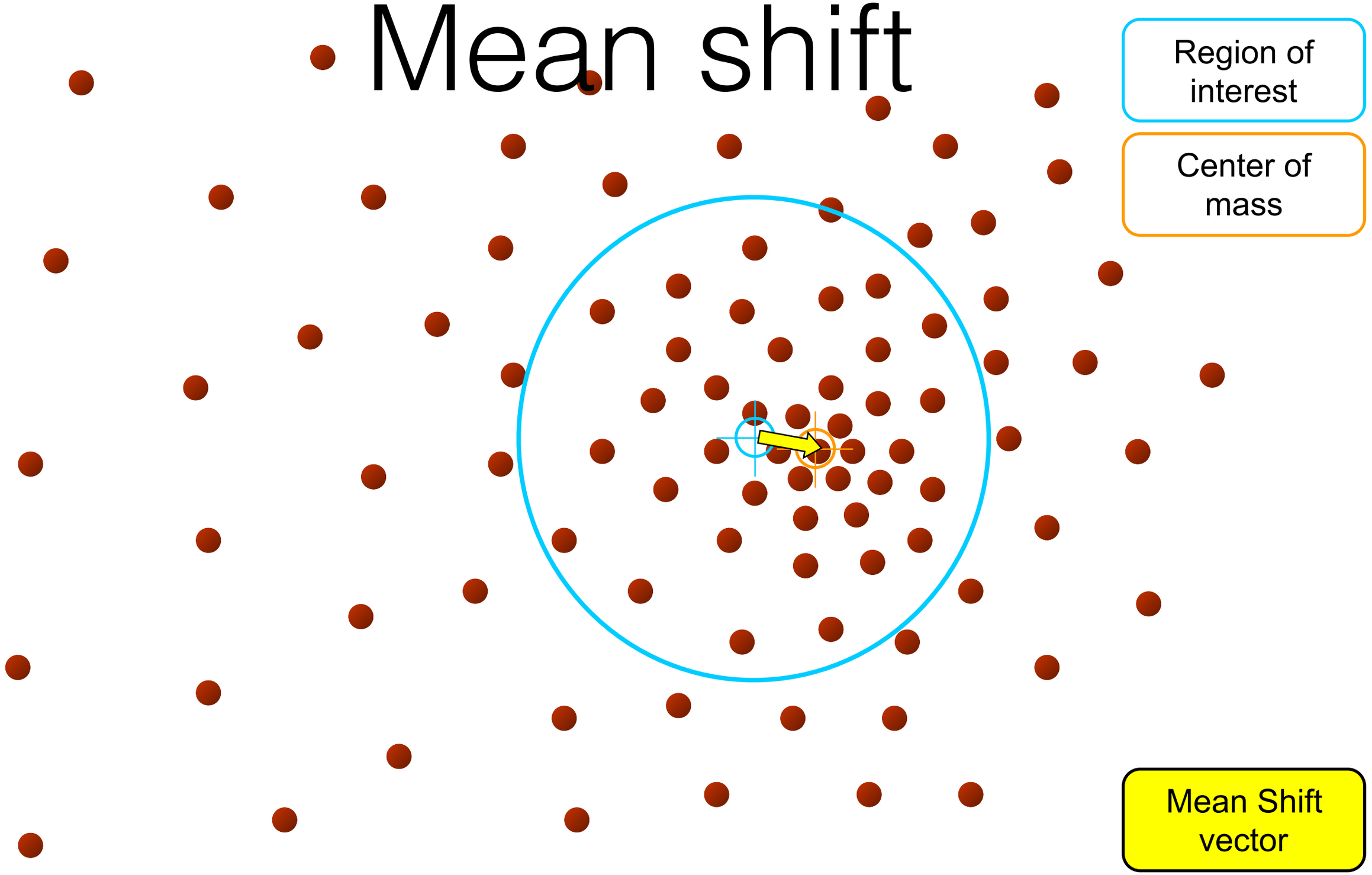
Mean shift



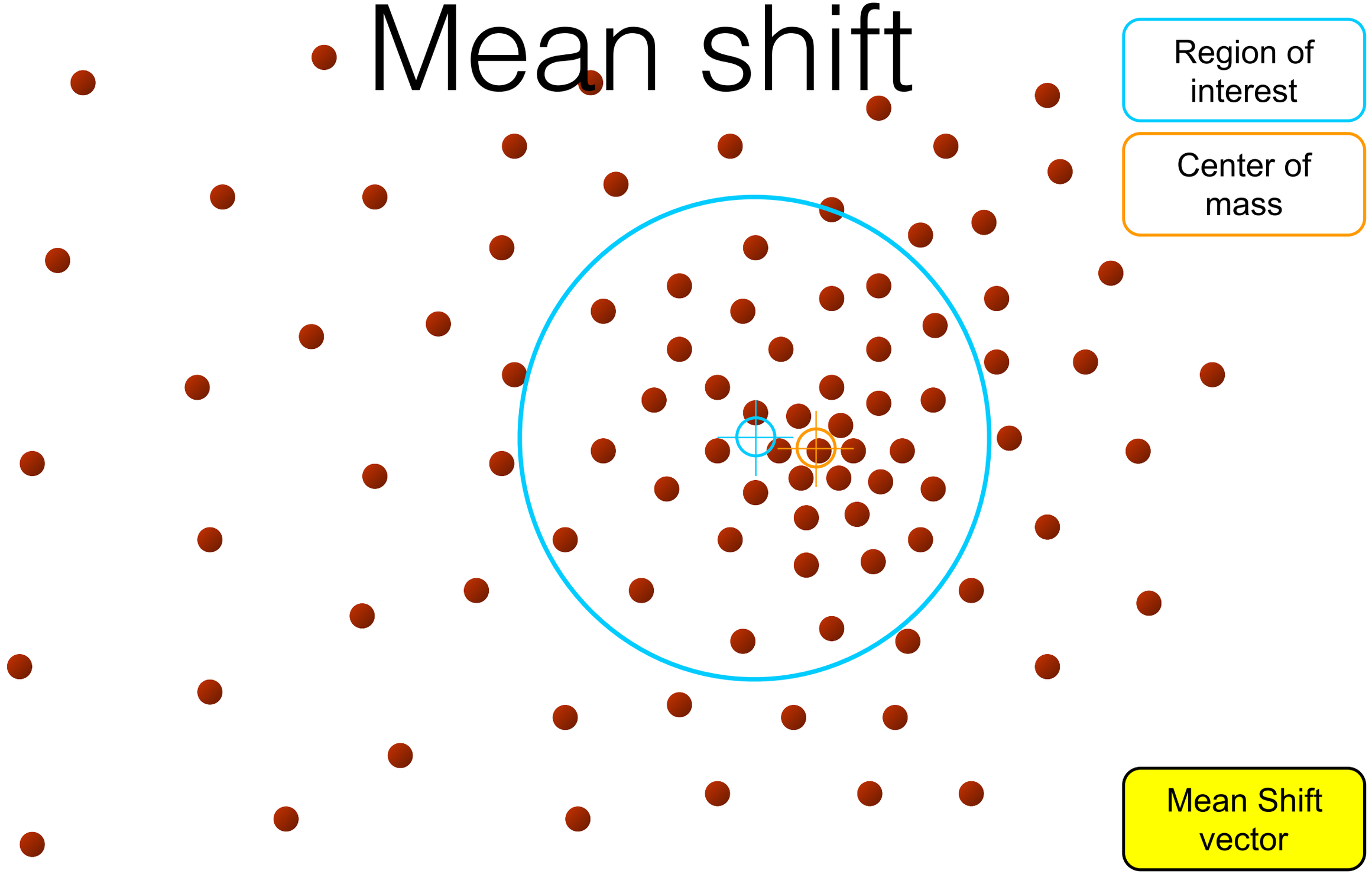
Mean shift



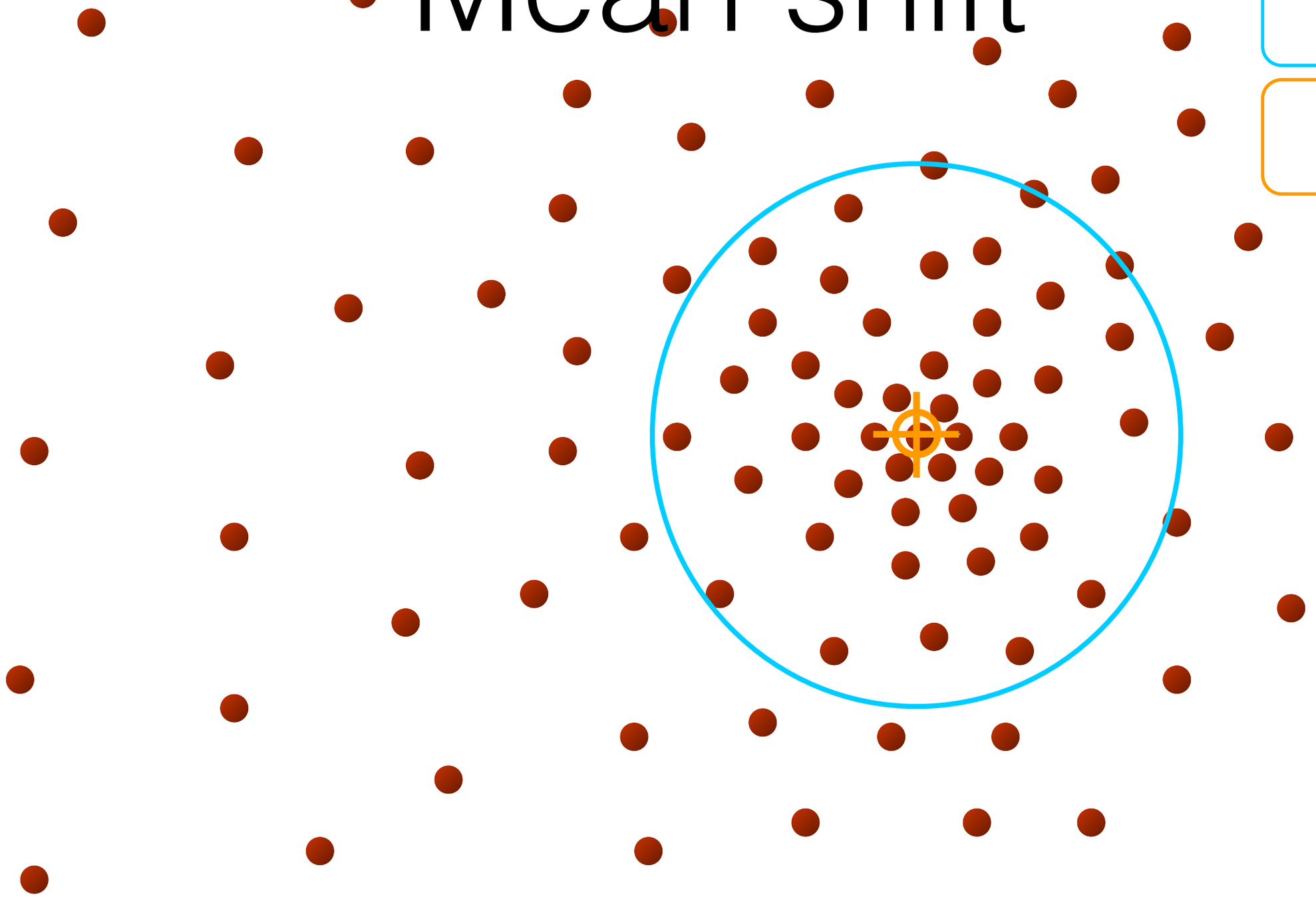
Mean shift



Mean shift



Mean shift



Kernel density estimation

- Kernel density estimation function

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

- Gaussian kernel

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

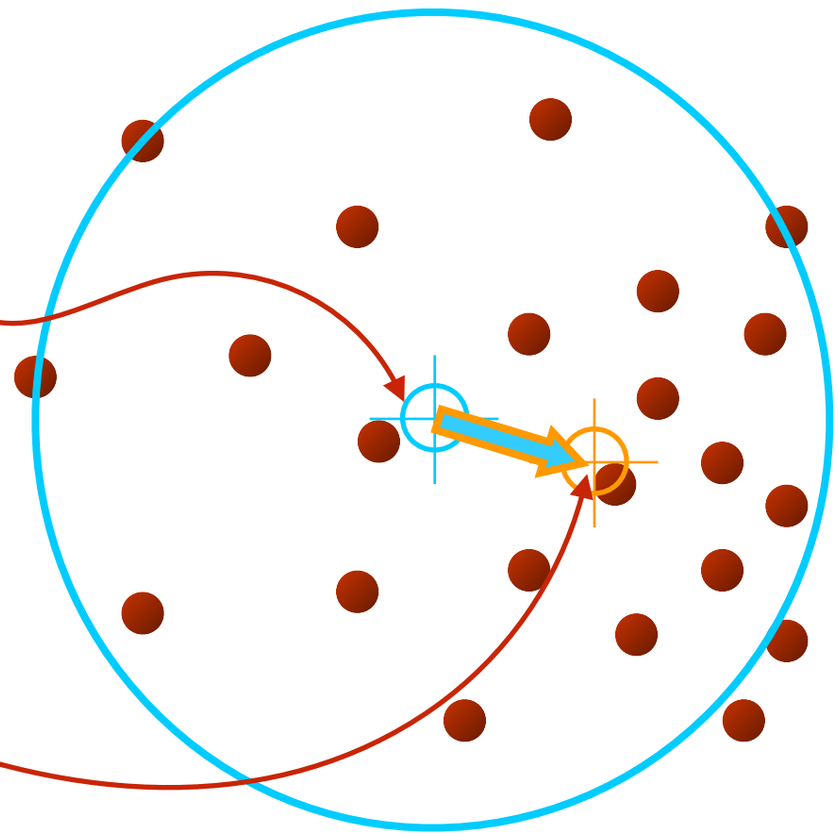
Slide credit: James Hays

Computing Mean Shift

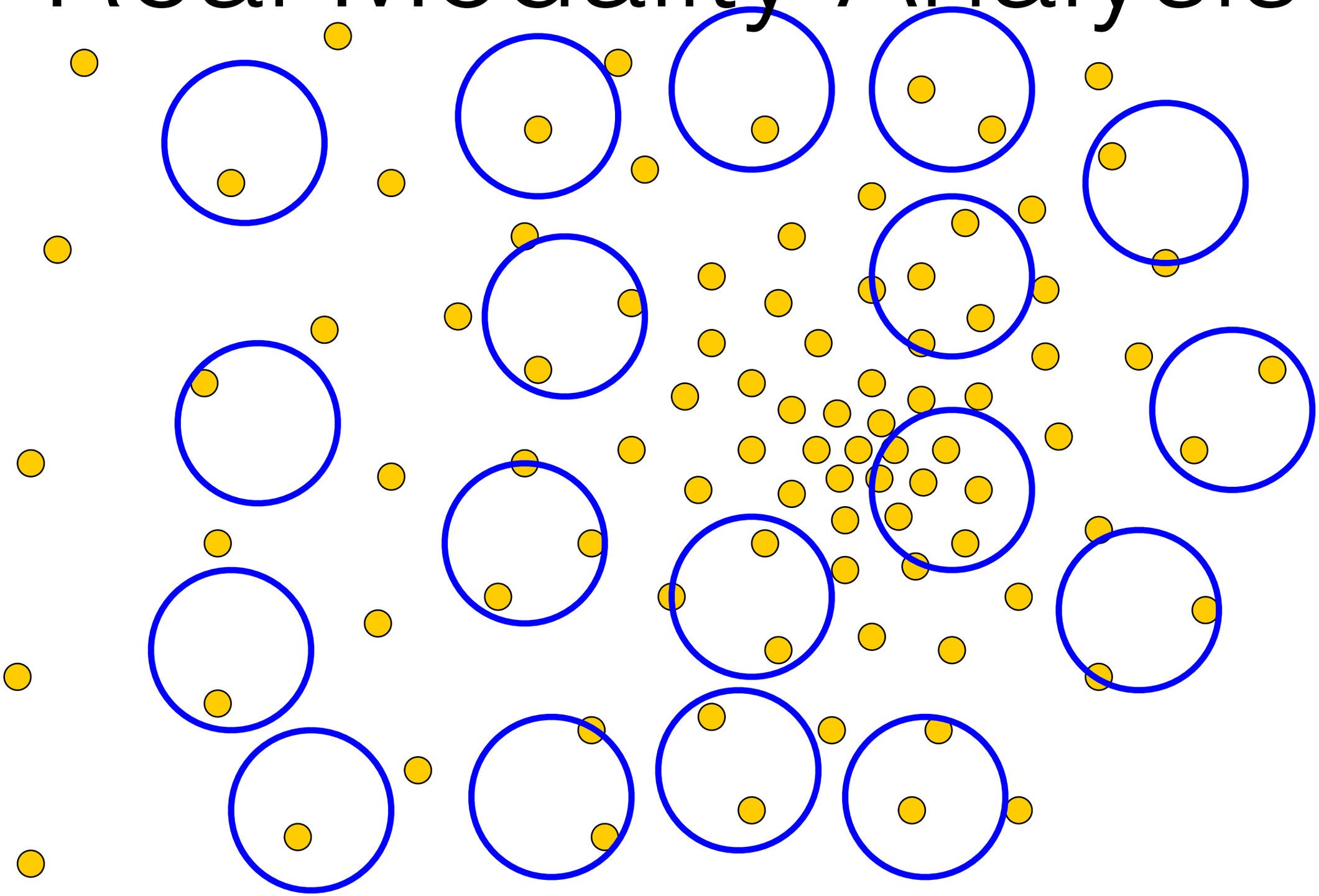
- Compute mean shift vector

- Shift the kernel window

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



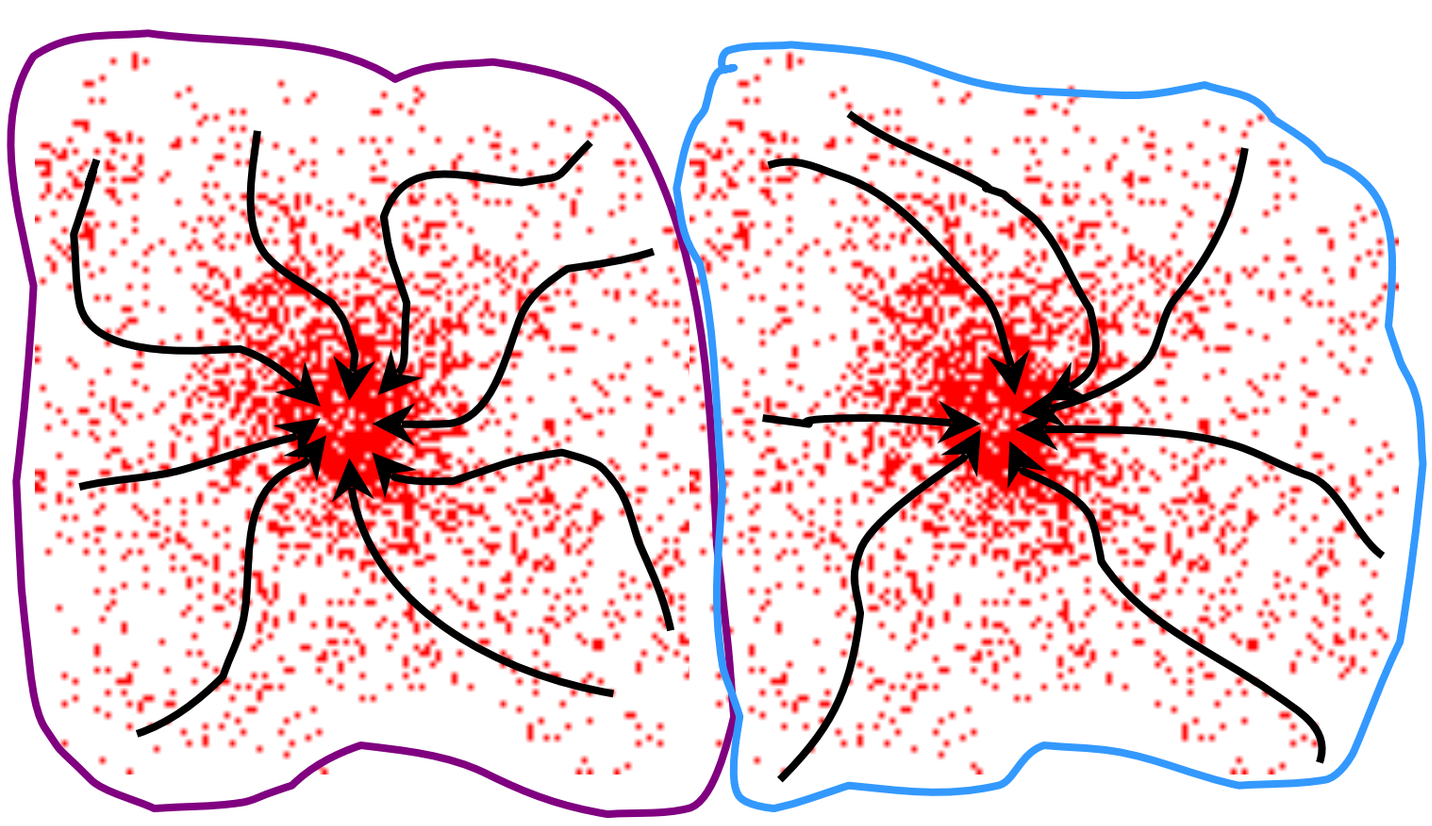
Real Modality Analysis



Slide credit: James Hayes

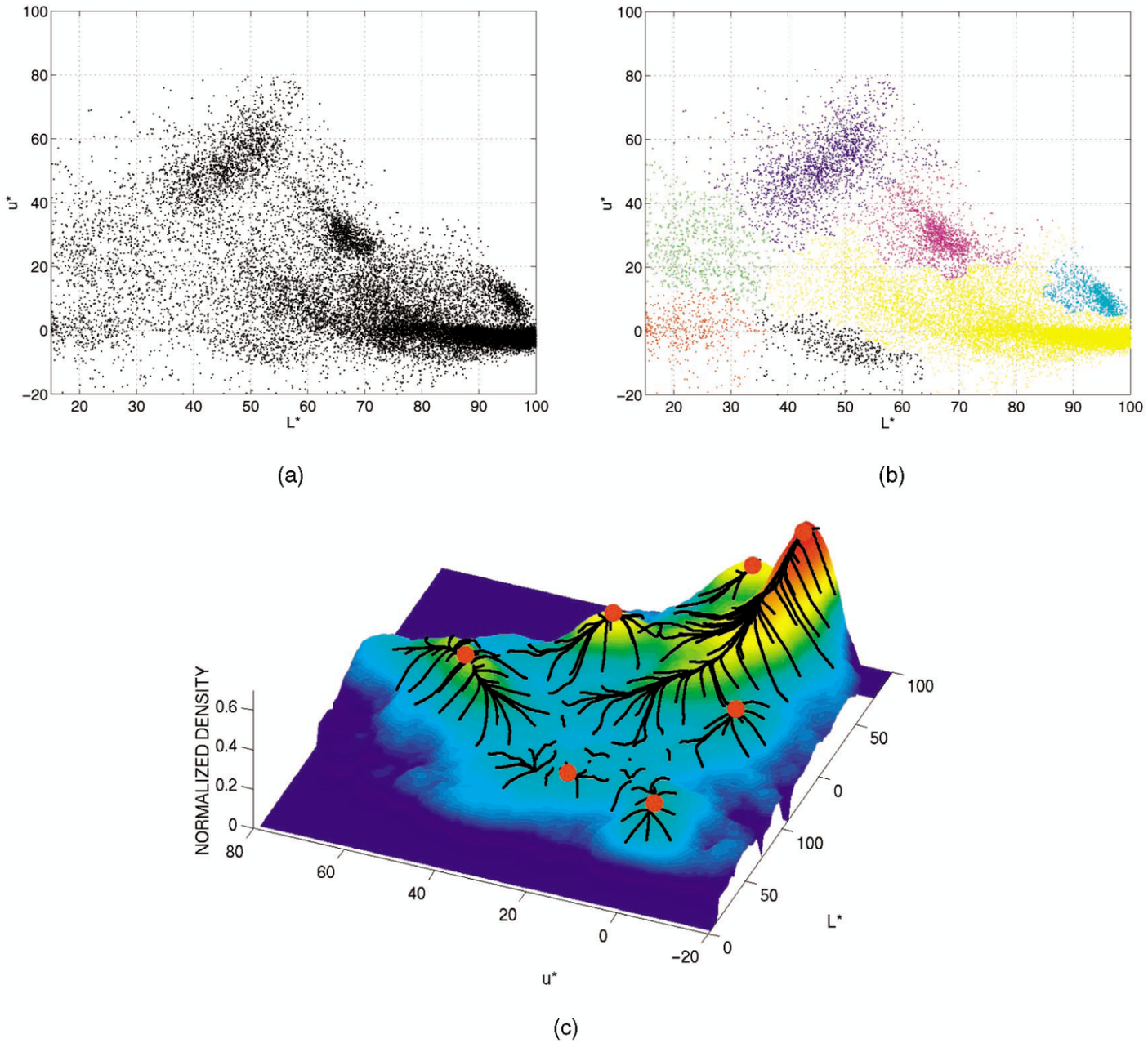
Attraction basin

- 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



Slide credit: James Hayes

Image segmentation using Mean Shift

- Compute features for each pixel (color, gradient, 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 is reached

- Merge windows that are within width of K_f and K_s

Mean shift

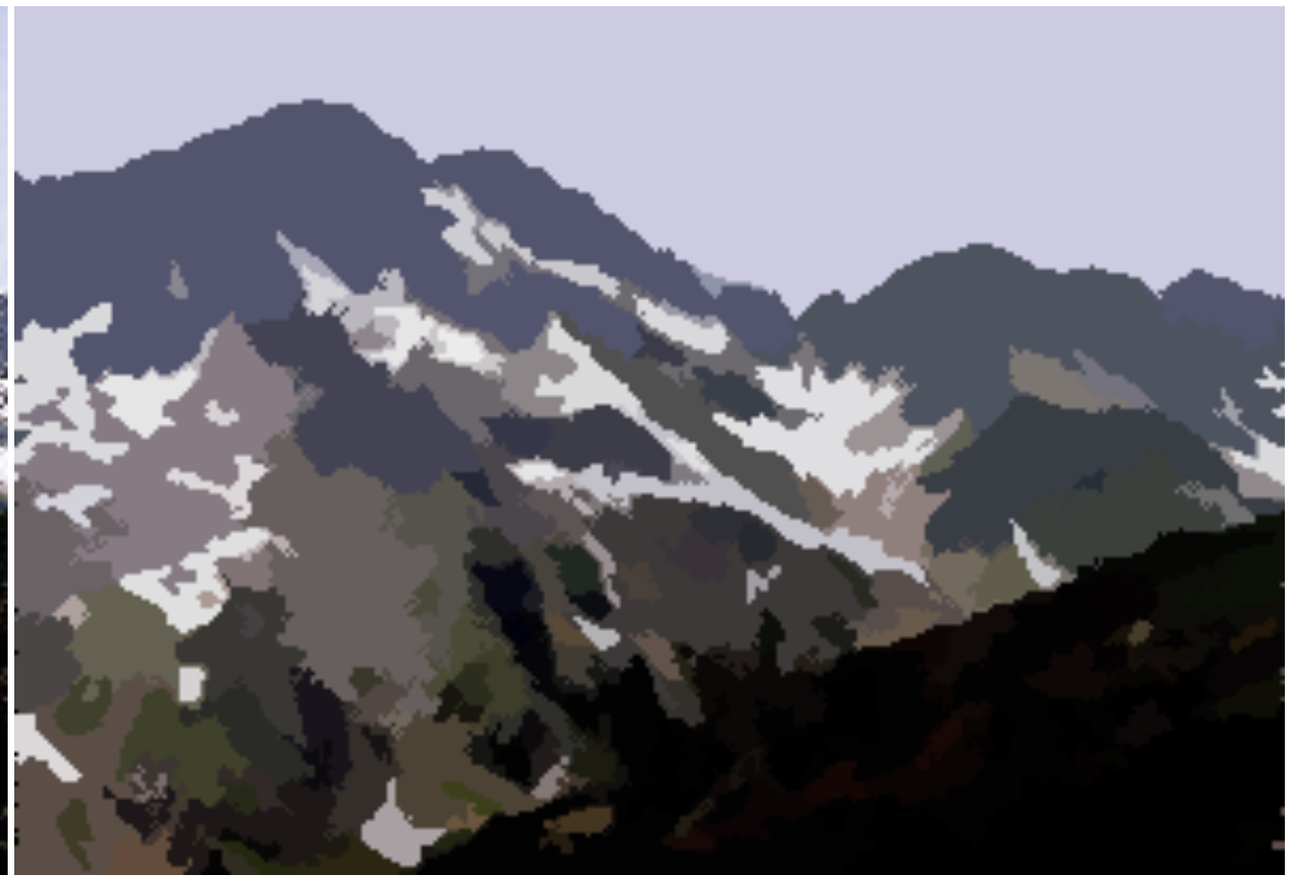
- Speed up
  - Binned estimation
  - Fast neighbour search
  - Update each window at each iteration

- Other tricks
  - Use kNN to determine window sizes adaptively

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

Mean shift

- Pros
  - Good general purpose segmentation
  - Flexible in number and shapes of regions
  - Robust to outliers
- Cons
  - Have to choose kernel size in advance
  - Not suitable for high-dimensional features (i.e., data points)
- When to use it?
  - Oversegmentation
  - Multiple segmentations
  - **Tracking**, clustering and filtering applications



http://www.caip.rutgers.edu/~comanici/MSPAMI/msPamiResults.html

http://www.caip.rutgers.edu/~comanici/MSPAMI/msPamiResults.html