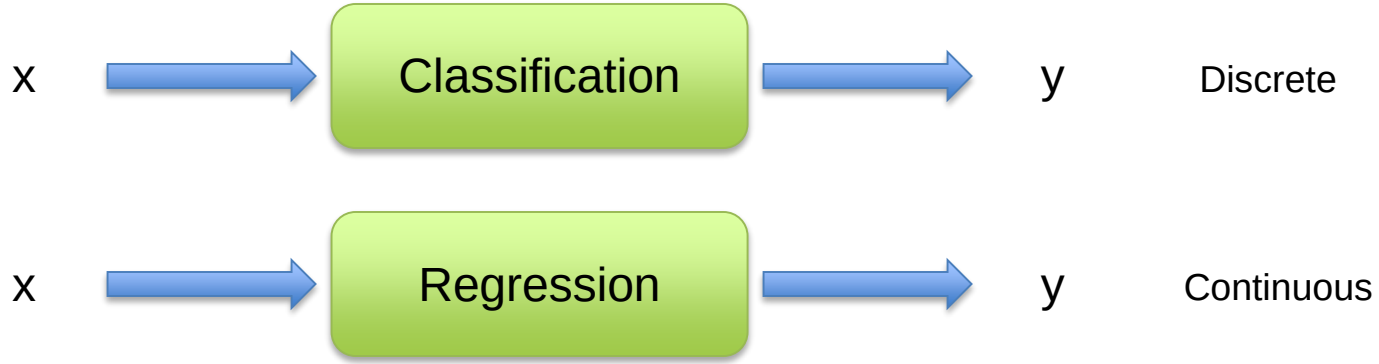
VBM683
Machine Learning

Pinar Duygulu

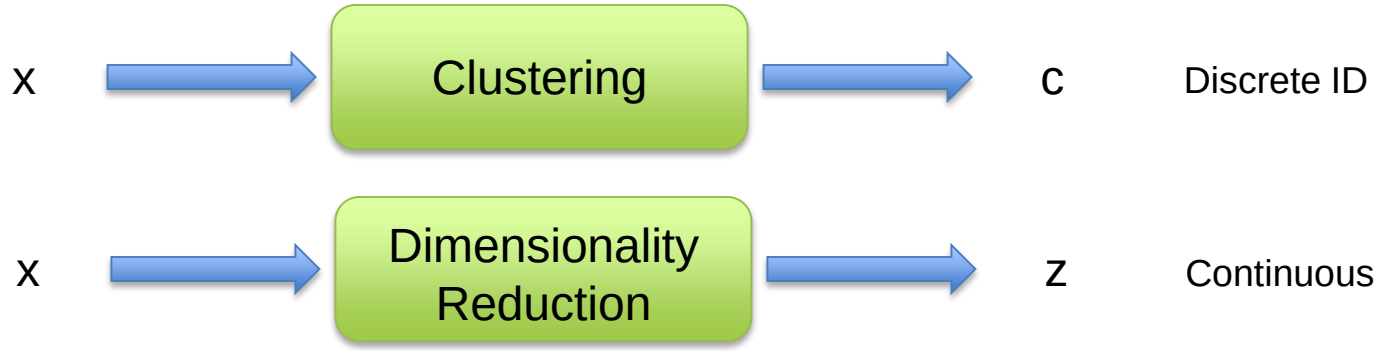
Slides are adapted from
Dhruv Batra, Aykut Erdem
Barnabas Poczos, and Aarti Singh

Tasks

Supervised Learning



Unsupervised Learning

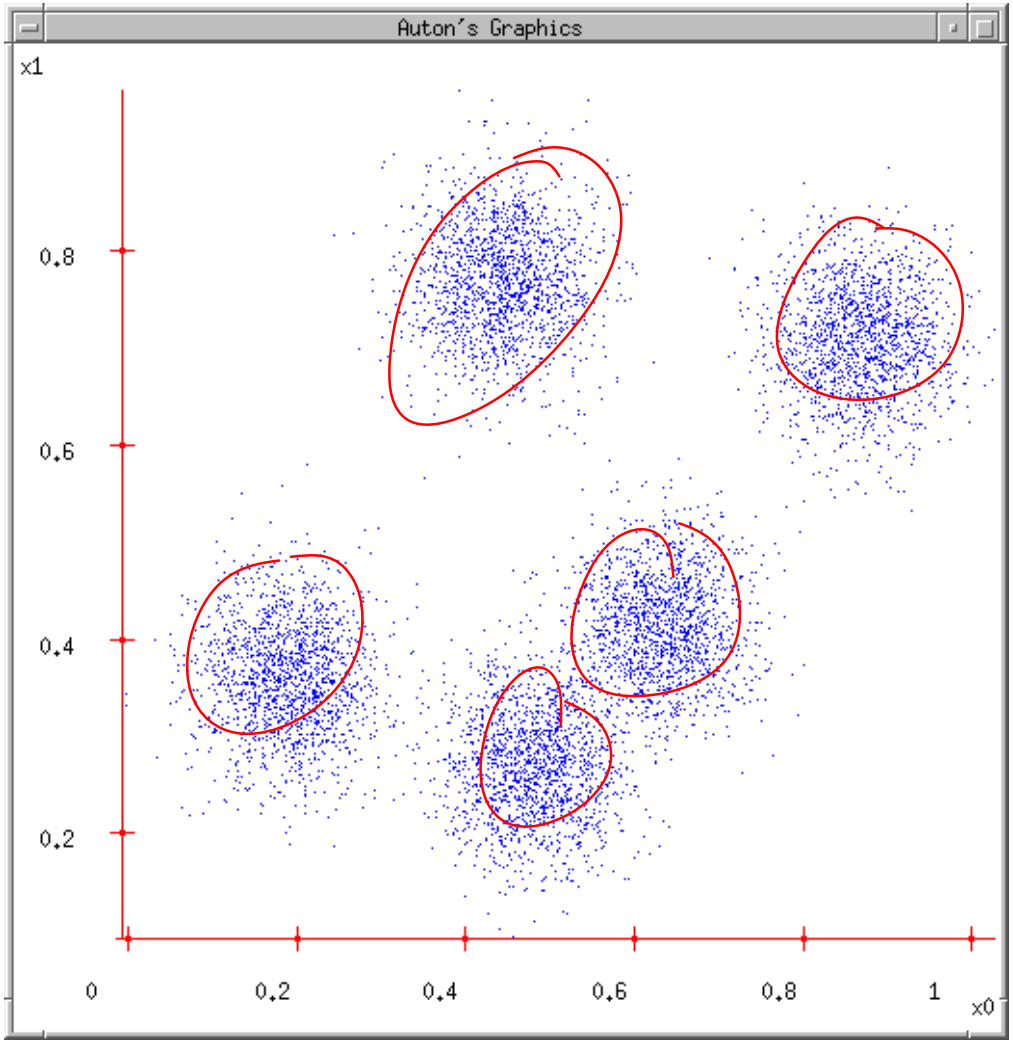


Unsupervised Learning

- Learning only with X
  - Y not present in training data

- Some example unsupervised learning problems:
  - Clustering / Factor Analysis
  - Dimensionality Reduction / Embeddings
  - Density Estimation with Mixture Models

New Topic: Clustering



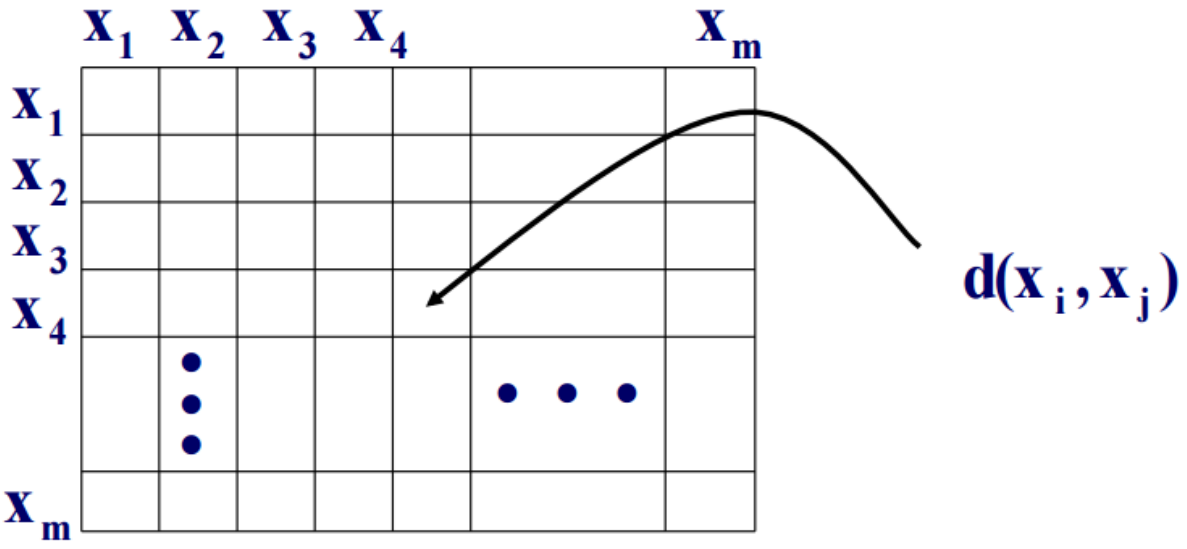
Slide Credit: Carlos Guestrin

Synonyms

- Clustering
- Vector Quantization
- Latent Variable Models
- Hidden Variable Models
- Mixture Models
- Algorithms:
  - K-means
  - Expectation Maximization (EM)

Distance measures

- In studying clustering techniques we will assume that we are given a matrix of distances between all pairs of data points:



slide by Julia Hockenmeier

What is Similarity/Dissimilarity?

Hard to define!
But we know it
when we see it

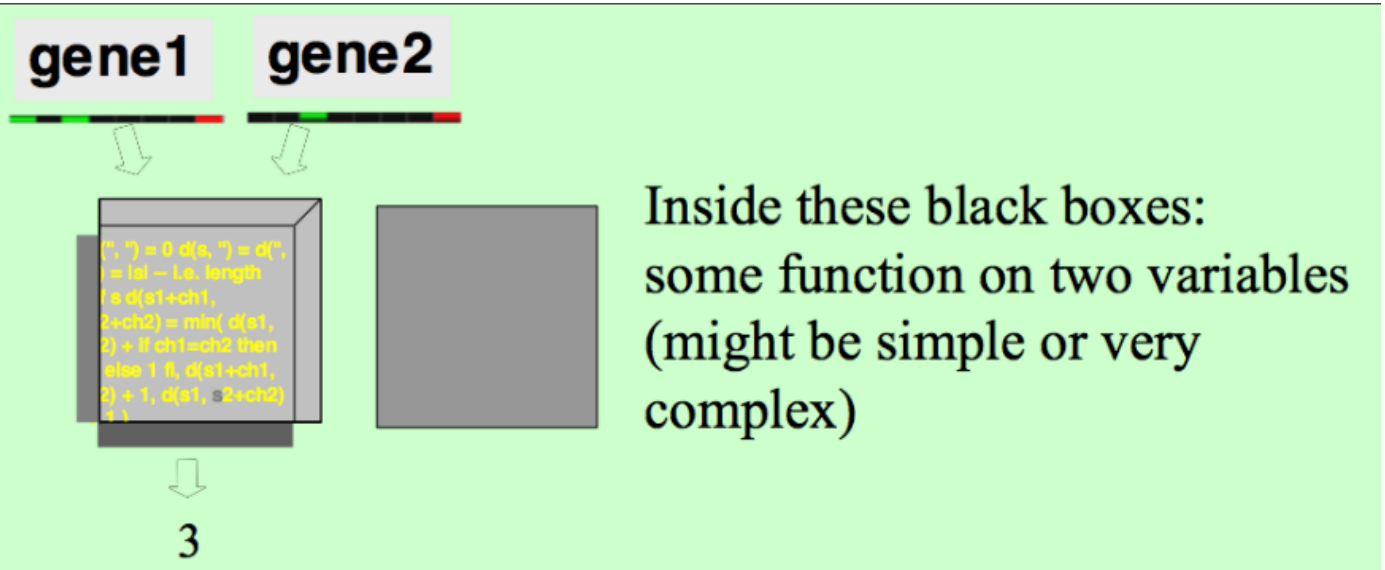
- The real meaning of similarity is a philosophical question. We will take a more pragmatic approach.
- Depends on representation and algorithm. For many rep.//alg., easier to think in terms of a distance (rather than similarity) between vectors.

slide by Eric Xing

Defining Distance Measures

- **Definition:** Let O_1 and O_2 be two objects from the universe of possible objects. The distance (dissimilarity) between O_1 and O_2 is a real number denoted by $D(O_1, O_2)$.

slide by Andrew Moore



A few examples:

- Euclidean distance

$$d(x,y) = \sqrt{\sum_i (x_i - y_i)^2}$$

- Correlation coefficient

$$s(x,y) = \frac{\sum_i (x_i - \mu_x)(y_i - \mu_y)}{\sigma_x \sigma_y}$$

- Similarity rather than distance
- Can determine similar trends

slide by Andrew Moore

What properties should a distance measure have?

- Symmetric
  - $D(A,B) = D(B,A)$
  - Otherwise, we can say A looks like B but B does not look like A
- Positivity, and self-similarity
  - $D(A,B) \geq 0$, and $D(A,B) = 0$ iff $A = B$
  - Otherwise there will different objects that we cannot tell apart
- Triangle inequality
  - $D(A,B) + D(B,C) \geq D(A,C)$
  - Otherwise one can say "A is like B, B is like C, but A is not like C at all"

slide by Alan Fern

Distance measures

- Euclidean (L_2)

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

- Manhattan (L_1)

$$d(\mathbf{x},\mathbf{y}) = |\mathbf{x} - \mathbf{y}| = \sum_{i=1}^{d}|x_i - y_i|$$

- Infinity (Sup) Distance L_∞

$$d(\mathbf{x},\mathbf{y}) = \max_{1\le i\le d}|x_i - y_i|$$

- Note that $L_\infty < L_1 < L_2$, but different distances do not induce the same ordering on points.

slide by Julia Hockenmaier

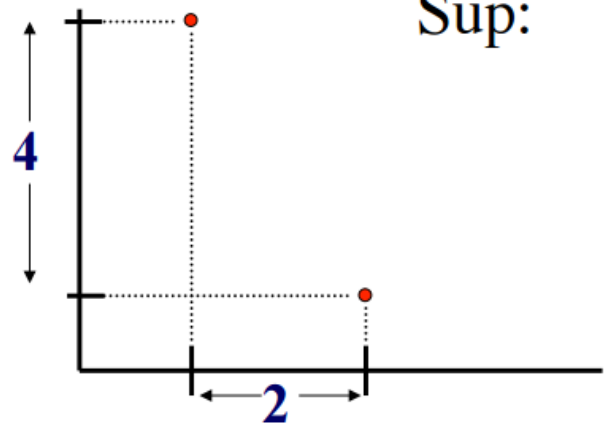
Distance measures

$\mathbf{x} = (x_1, x_2)$
$\mathbf{y} = (x_1 - 2, x_2 + 4)$

Euclidean: $(4^2 + 2^2)^{1/2} = 4.47$

Manhattan: $4 + 2 = 6$

Sup: $Max(4, 2) = 4$



slide by Julia Hockenmeier

Distance measures

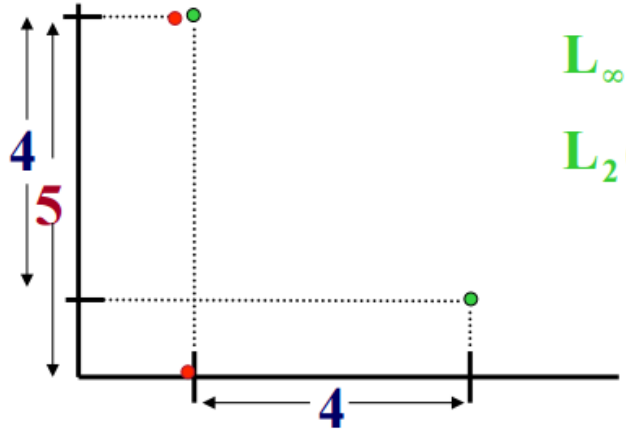
- Different distances do not induce the same ordering on points

$$L_\infty(a,b) = 5$$

$$L_2(a,b) = (5^2 + \varepsilon^2)^{1/2} = 5 + \varepsilon$$

$$L_\infty(c,d) = 4$$

$$L_2(c,d) = (4^2 + 4^2)^{1/2} = 4\sqrt{2} = 5.66$$



$$L_\infty(c,d) < L_\infty(a,b)$$

$$L_2(c,d) > L_2(a,b)$$

slide by Julia Hockenmeier

Distance measures

- Clustering is sensitive to the distance measure
- Sometimes it is beneficial to use a distance measure that is invariant to transformations that are natural to the problem:
  - Mahalanobis distance:
    - ✓ Shift and scale invariance

slide by Julia Hockenmeier

Mahalanobis Distance

$$d(\mathbf{x},\mathbf{y}) = \sqrt{(\mathbf{x}-\mathbf{y})^T \Sigma (\mathbf{x}-\mathbf{y})}$$

Σ is a (symmetric) Covariance Matrix:

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x_i, \text{ (average of the data)}$$

$$\Sigma = \frac{1}{m}\sum_{i=1}^{m} (\mathbf{x}-\mu)(\mathbf{x}-\mu)^T, \text{a matrix of size } m \times m$$

Translates all the axes to a mean = 0 and variance = 1 (shift and scale invariance)

slide by Julia Hockenmaier

Distance measures

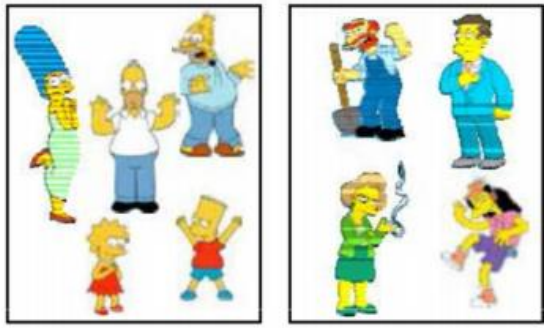
- Some algorithms require distances between a point x and a set of points A d(x, A)
  This might be defined e.g. as min/max/avg distance between x and any point in A.

- Others require distances between two sets of points A, B, d(A, B).
  This might be defined e.g as min/max/avg distance between any point in A and any point in B.

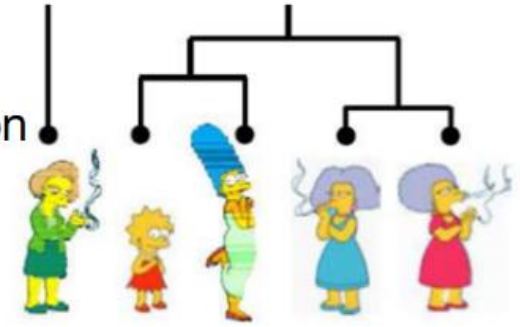
slide by Julia Hockenmeier

Clustering algorithms

- **Partitioning algorithms**
  - Construct various partitions and then evaluate them by some criterion
    - K-means
    - Mixture of Gaussians
    - Spectral Clustering



- **Hierarchical algorithms**
  - Create a hierarchical decomposition of the set of objects using some criterion
  - Bottom-up – agglomerative
  - Top-down – divisive



slide by Eric Xing

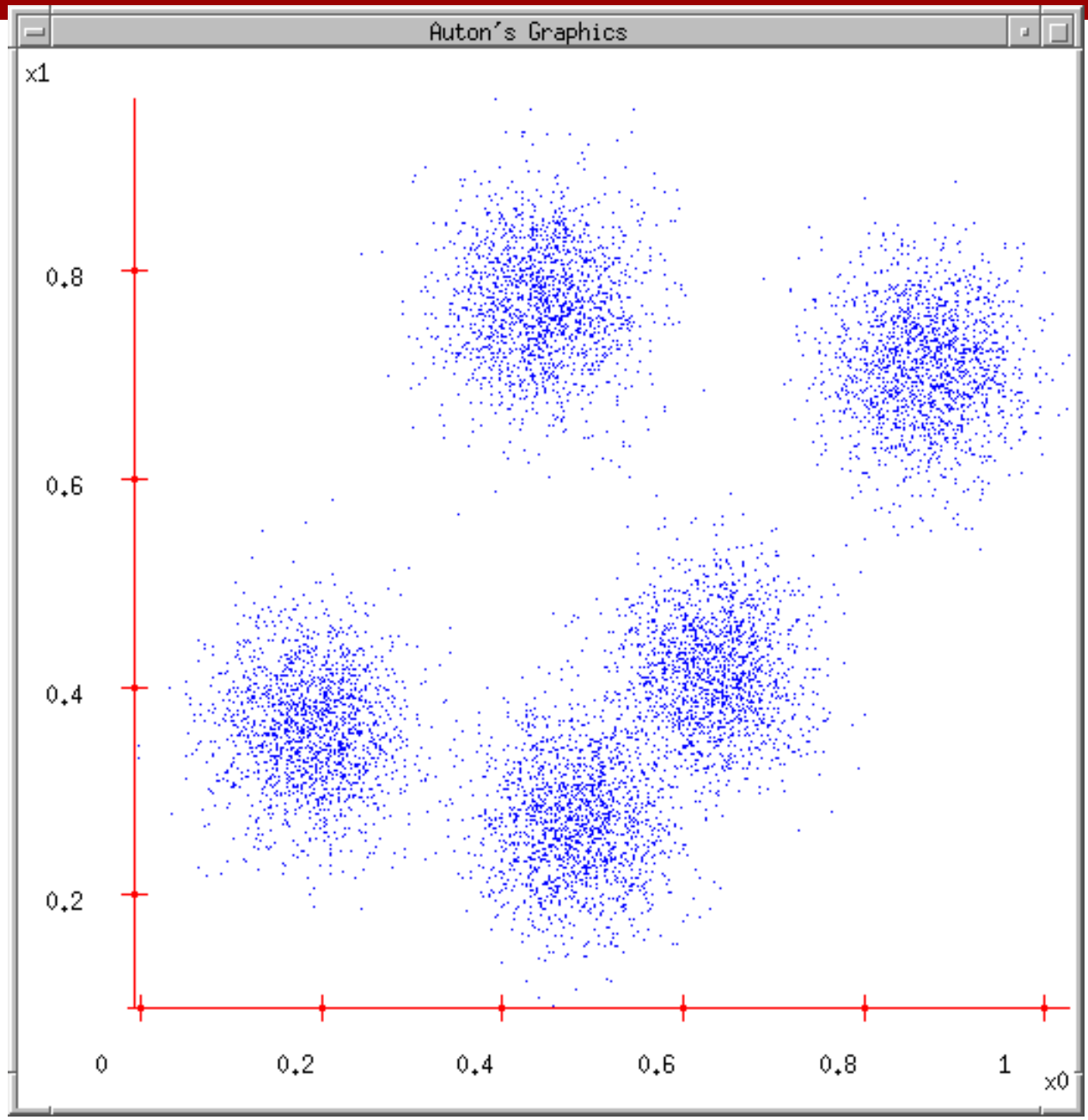
Desirable Properties of a Clustering Algorithm

- Scalability (in terms of both time and space
- Ability to deal with different data types
- Minimal requirements for domain knowledge to determine input parameters
- Ability to deal with noisy data
- Interpretability and usability

- Optional
  - Incorporation of user-specified constraints

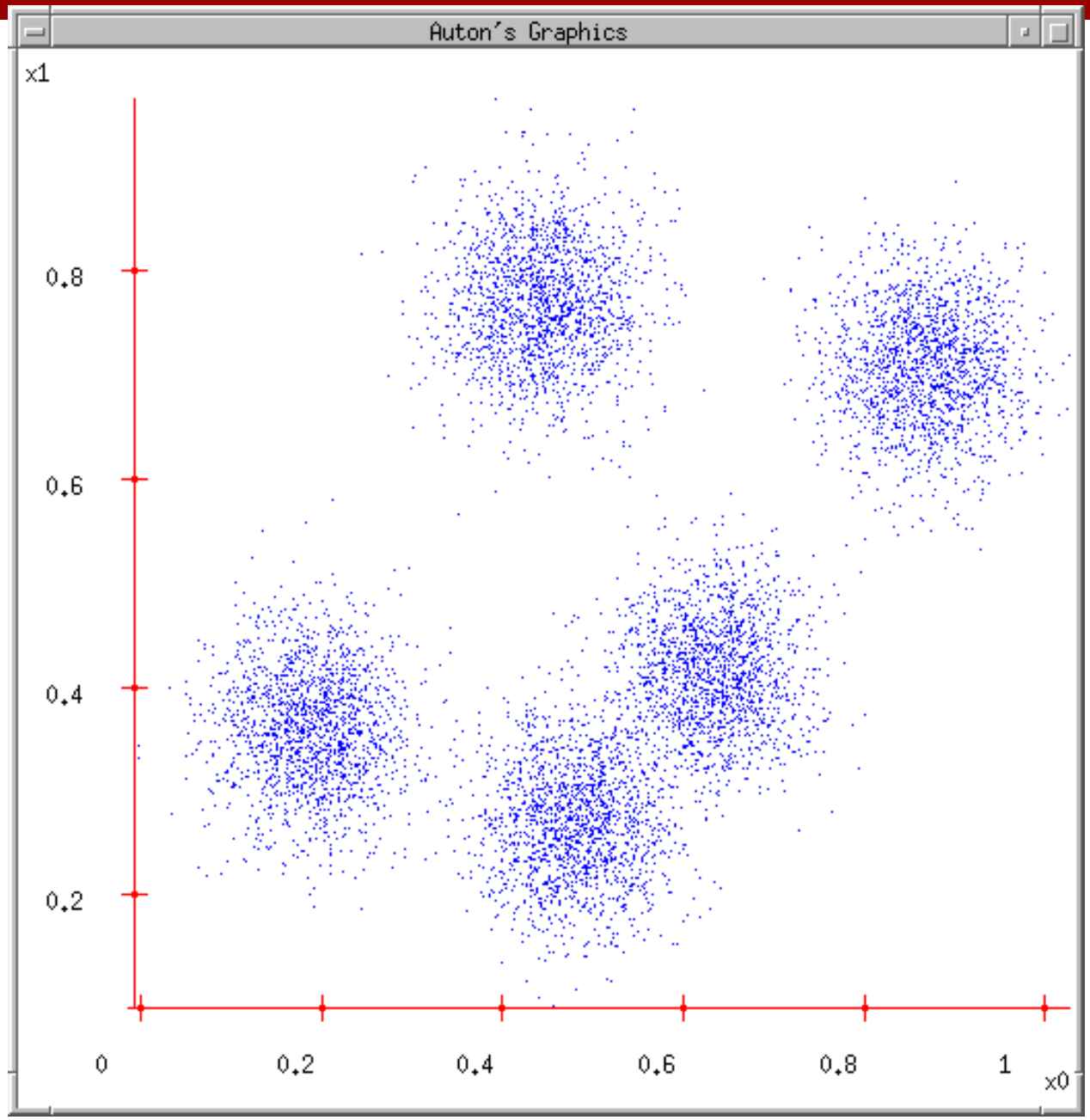
slide by Andrew Moore

Some Data



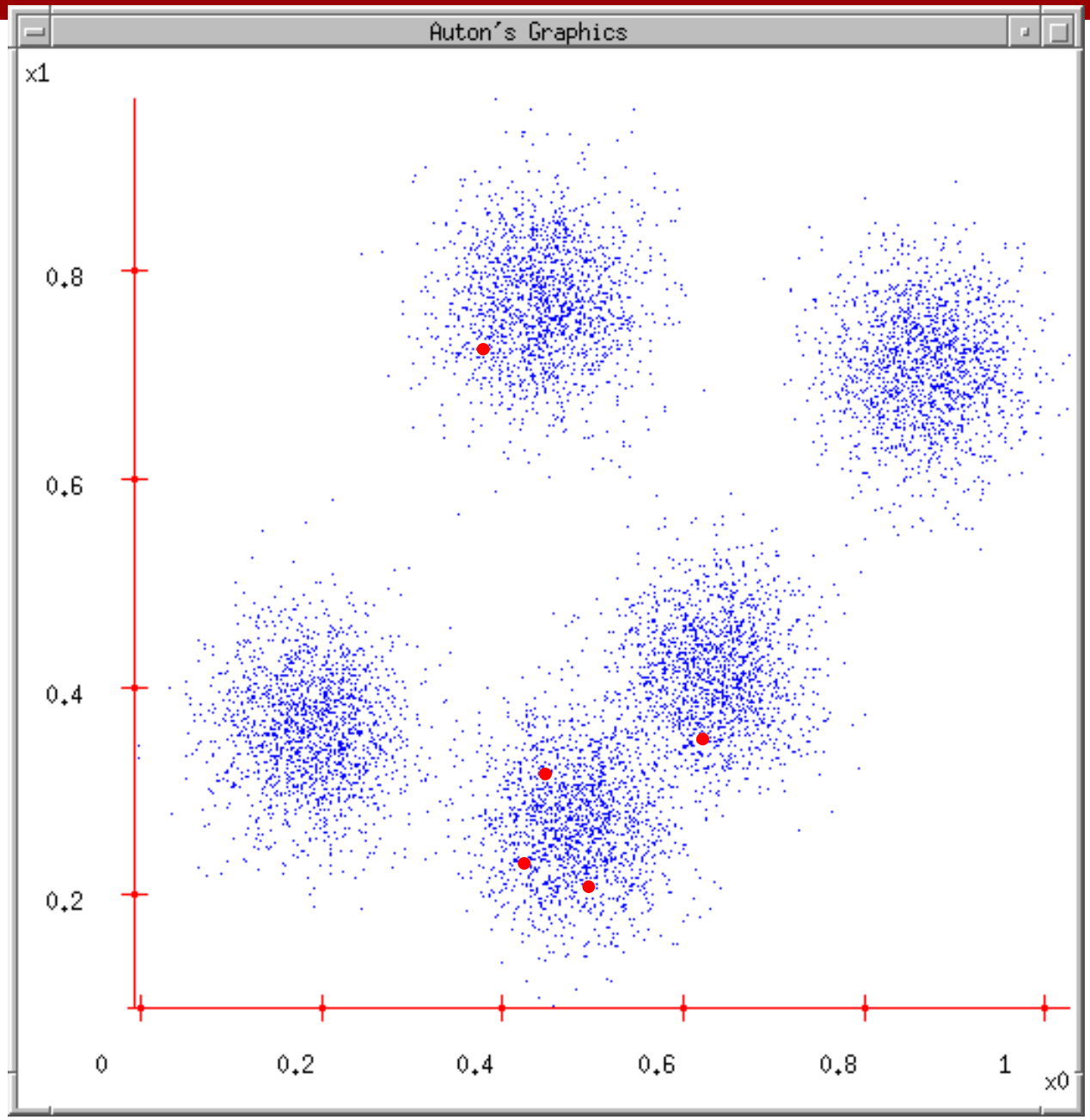
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*



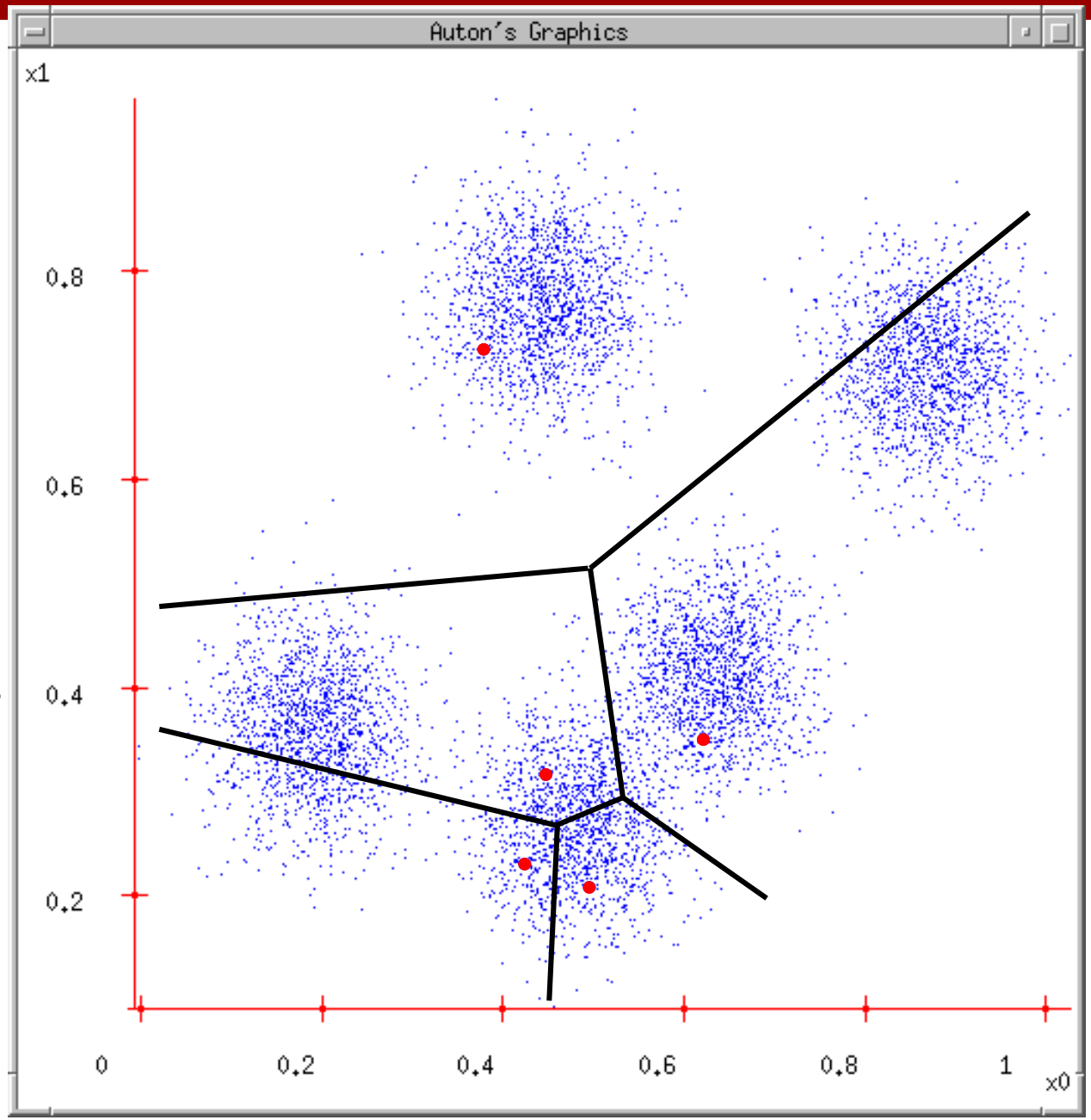
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations



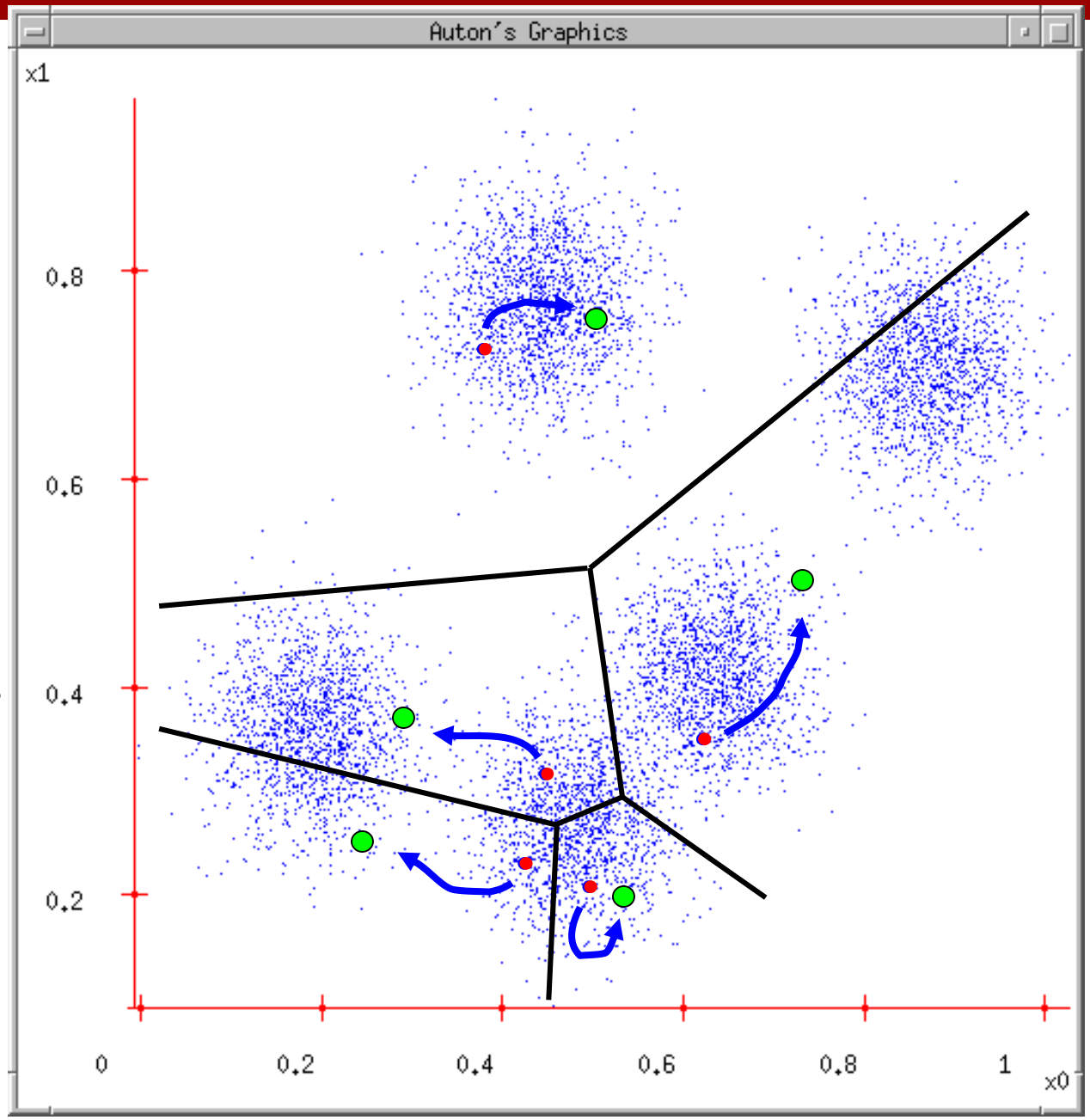
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to. (Thus each Center "owns" a set of datapoints)



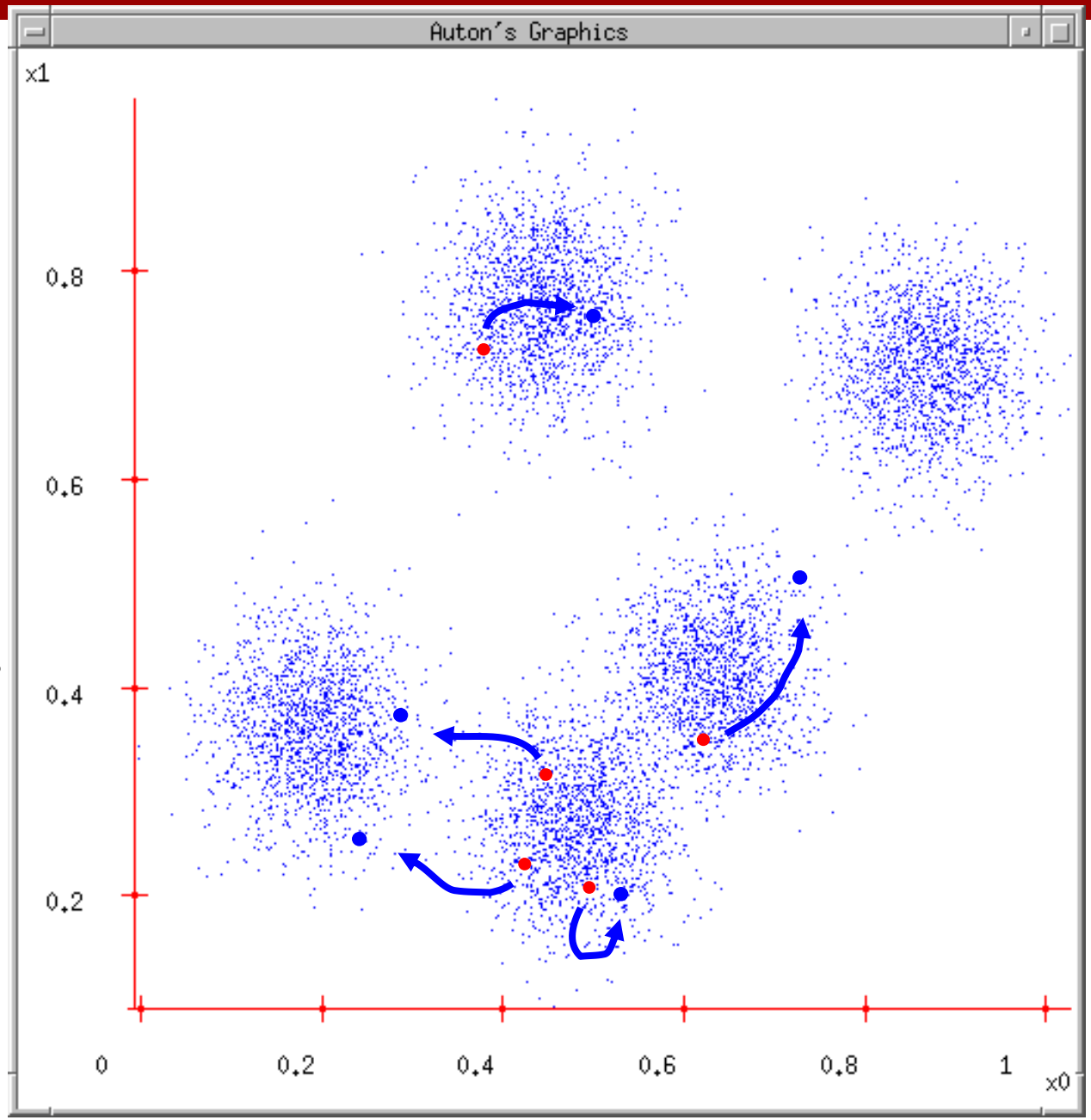
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center finds the centroid of the points it owns

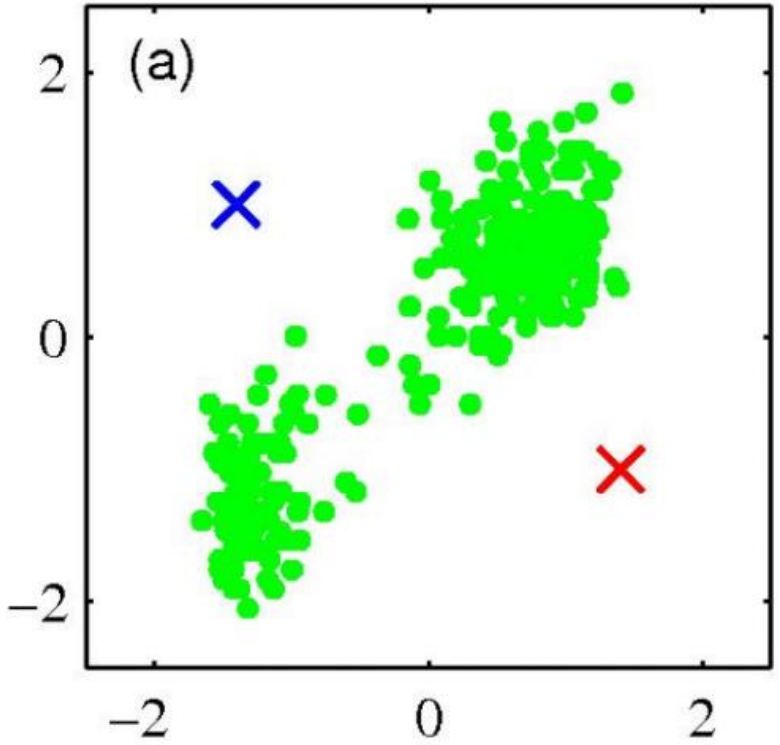


K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center finds the centroid of the points it owns
5. …Repeat until terminated!



K-Means Clustering: Example

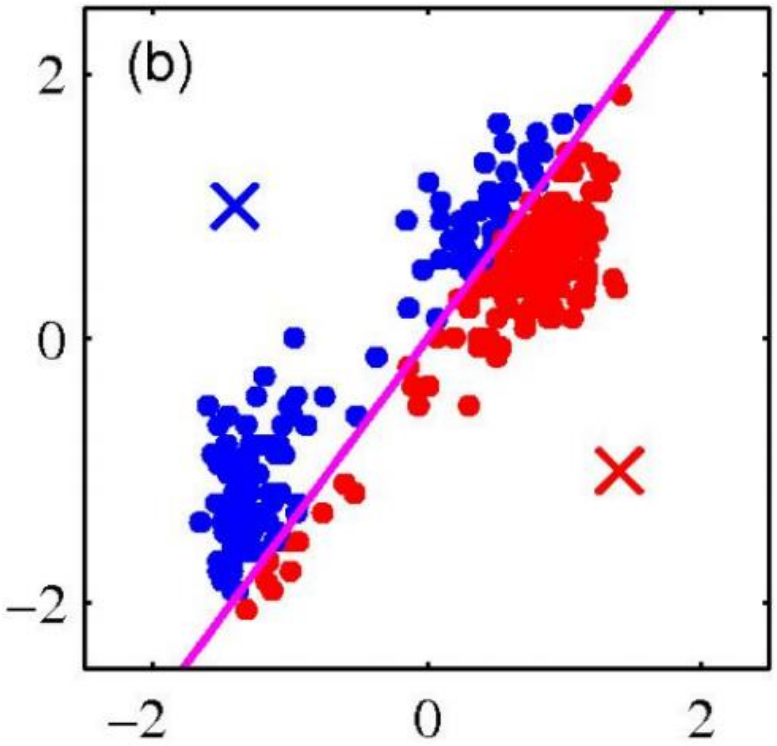


- Pick *K* random points as cluster centers (means)

Shown here for K=2

slide by David Sontag

K-Means Clustering: Example

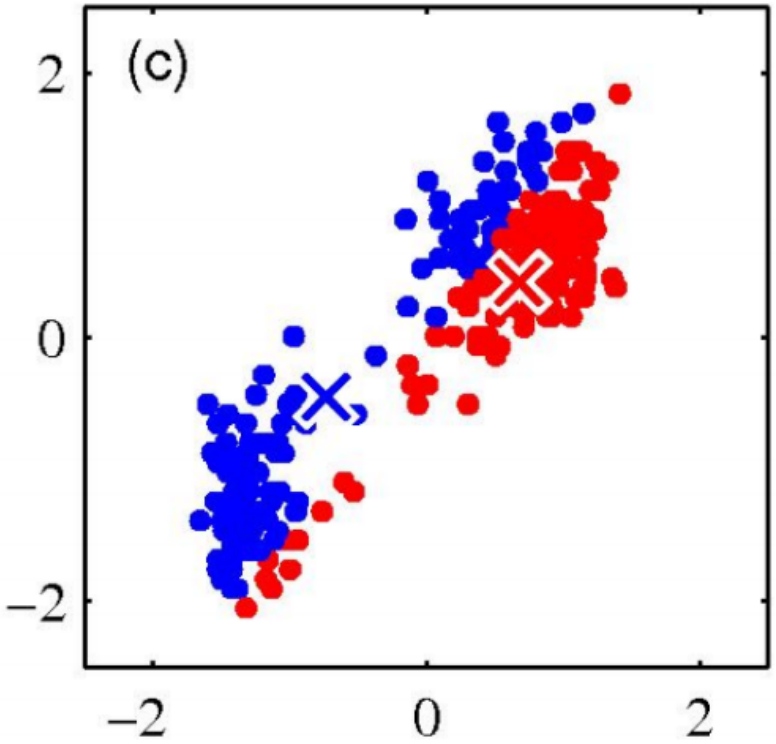


Iterative Step 1

- Assign data points to closest cluster centers

slide by David Sontag

K-Means Clustering: Example

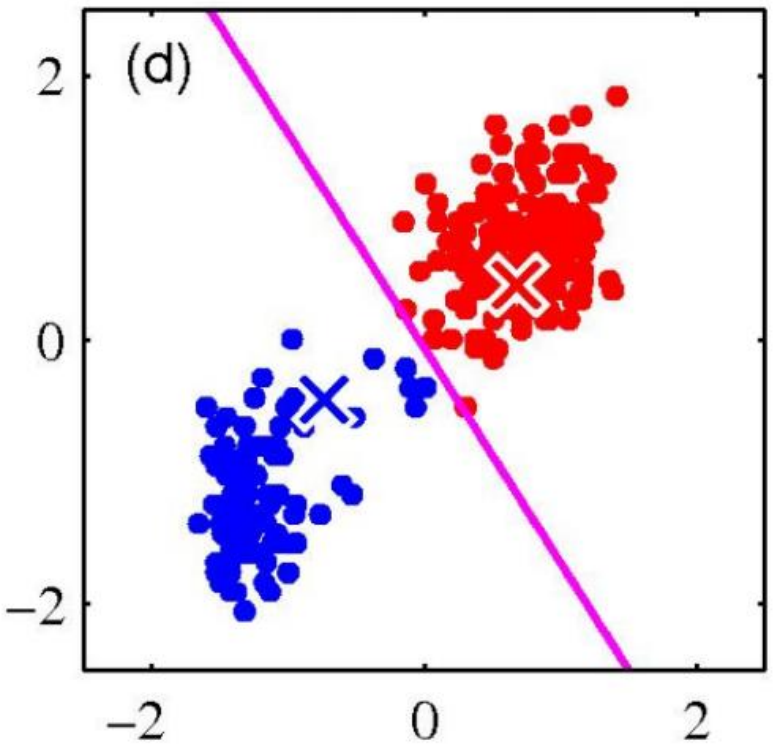


Iterative Step 2

- Change the cluster center to the average of the assigned points

slide by David Sontag

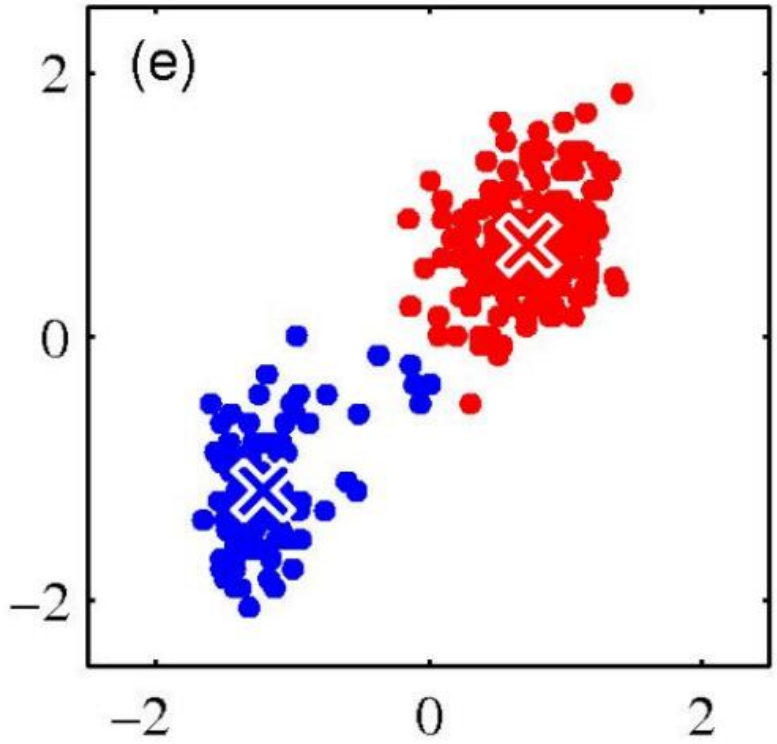
K-Means Clustering: Example



- Repeat until convergence

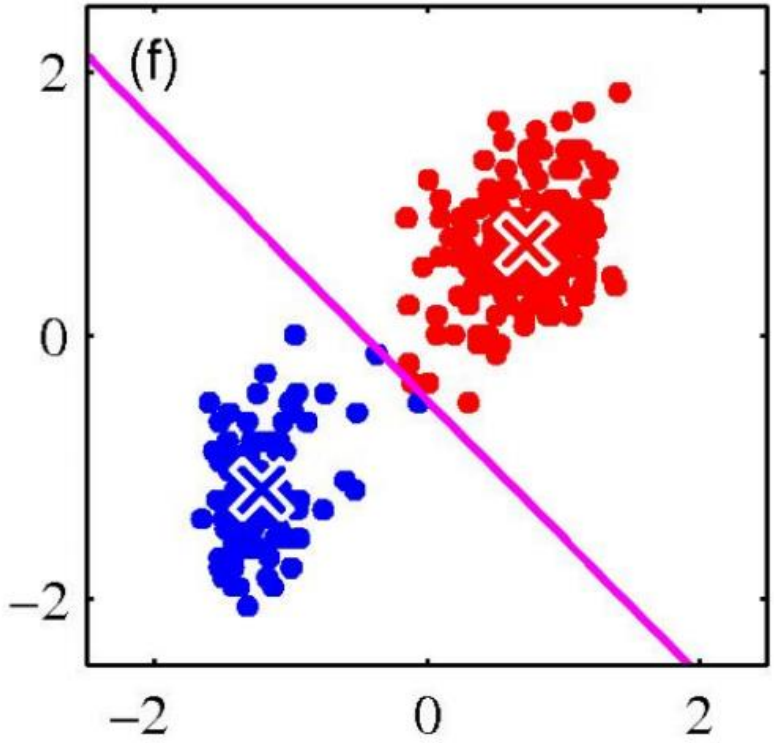
slide by David Sontag

K-Means Clustering: Example



slide by David Sontag

K-Means Clustering: Example



slide by David Sontag

K-means

- Randomly initialize *k* centers
  - $\mu^{(0)} = \mu_1^{(0)}, \ldots, \mu_k^{(0)}$

- **Assign**:
  - Assign each point i∈{1,…n} to nearest center:
  - $C(i) \longleftarrow \underset{j}{\operatorname{argmin}} ||\mathbf{x}_i - \boldsymbol{\mu}_j||^2$

- **Recenter**:
  - μ_j becomes centroid of its points

K-means

- Demo
  - http://mlehman.github.io/kmeans-javascript/

What is K-means optimizing?

- Objective F($\boldsymbol{\mu}$,C): function of centers $\boldsymbol{\mu}$ and point allocations C:
  - $F(\boldsymbol{\mu}, C) = \sum_{i=1}^{N} ||\mathbf{x}_i - \boldsymbol{\mu}_{C(i)}||^2$
  - 1-of-k encoding $F(\boldsymbol{\mu}, \boldsymbol{a}) = \sum_{i=1}^{N} \sum_{j=1}^{k} a_{ij} ||\mathbf{x}_i - \boldsymbol{\mu}_j||^2$
- Optimal K-means:
  - $\min_{\mu} \min_{a} F(\mu, a)$

Coordinate descent algorithms

- Want: $\min_a \min_b F(a,b)$
- Coordinate descent:
  - fix a, minimize b
  - fix b, minimize a
  - repeat
- Converges!!!
  - if F is bounded
  - to a (often good) local optimum
    - as we saw in applet (play with it!)
- K-means is a coordinate descent algorithm!

K-means as Co-ordinate Descent

- Optimize objective function:

$$\min_{\boldsymbol{\mu}_1,\ldots,\boldsymbol{\mu}_k} \min_{\boldsymbol{a}_1,\ldots,\boldsymbol{a}_N} F(\boldsymbol{\mu}, \boldsymbol{a}) = \min_{\boldsymbol{\mu}_1,\ldots,\boldsymbol{\mu}_k} \min_{\boldsymbol{a}_1,\ldots,\boldsymbol{a}_N} \sum_{i=1}^{N}\sum_{j=1}^{k} a_{ij}||\mathbf{x}_i - \boldsymbol{\mu}_j||^2$$

- Fix $\boldsymbol{\mu}$, optimize a (or C)

K-means as Co-ordinate Descent

- Optimize objective function:

$$\min_{\boldsymbol{\mu}_1,\ldots,\boldsymbol{\mu}_k} \min_{\boldsymbol{a}_1,\ldots,\boldsymbol{a}_N} F(\boldsymbol{\mu}, \boldsymbol{a}) = \min_{\boldsymbol{\mu}_1,\ldots,\boldsymbol{\mu}_k} \min_{\boldsymbol{a}_1,\ldots,\boldsymbol{a}_N} \sum_{i=1}^{N}\sum_{j=1}^{k} a_{ij}||\mathbf{x}_i - \boldsymbol{\mu}_j||^2$$

- Fix a (or C), optimize $\boldsymbol{\mu}$

Properties of K-Means Algorithms

- Guaranteed to converge in a finite number of iterations

- Running time per iteration:
  1. Assign data points to closest cluster center
     $O(KN)$ time
  2. Change the cluster center to the average of its assigned points
     $O(N)$ time

slide by David Sontag

K-means Clustering Problem

Given a set of observations $(x_1, x_2, \ldots, x_n)$, where $x_i \in \mathbb{R}^d$

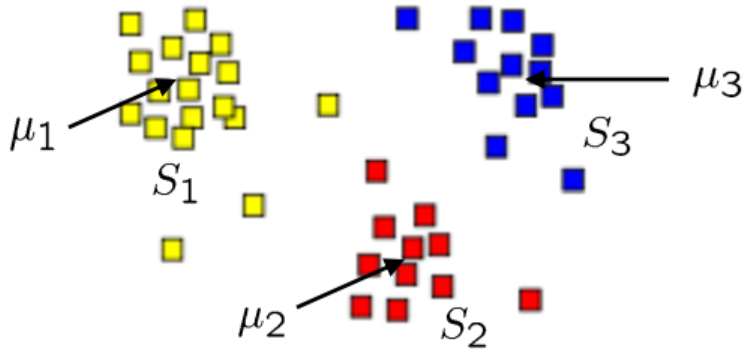
***K*-means clustering problem:**

Partition the *n* observations into *K* sets ($K \leq n$) $\mathbf{S} = \{S_1, S_2, \ldots, S_K\}$ such that the sets minimize the within-cluster sum of squares:

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{K} \sum_{\mathbf{x}_j \in S_i} \left\| \mathbf{x}_j - \boldsymbol{\mu}_i \right\|^2$$

where μ_i is the mean of points in set S_i.

K=3



K-Means Convergence

Objective $\min_{\mu}\min_{C}\sum_{i=1}^{k}\sum_{x\in C_i}|x-\mu_i|^2$

1. Fix μ, optimize C:

$$\min_{C}\sum_{i=1}^{k}\sum_{x\in C_i}|x-\mu_i|^2=\min_{c}\sum_{i}^{n}\left|x_i-\mu_{x_i}\right|^2$$

Step 1 of kmeans

2. Fix C, optimize μ:

$$\min_{\mu}\sum_{i=1}^{k}\sum_{x\in C_i}|x-\mu_i|^2$$

– Take partial derivative of μ_i and set to zero, we have

$$\mu_i=\frac{1}{|C_i|}\sum_{x\in C_i}x$$

Step 2 of kmeans

K-Means takes an alternating optimization approach, each step is guaranteed to decrease the objective – thus guaranteed to converge

slide by Alan Fern

Example: K-Means for Segmentation

K=2

Goal of Segmentation is to partition an image into regions each of which has reasonably homogenous visual appearance.

Original

slide by David Sontag

Example: K-Means for Segmentation

slide by David Sontag

Example: Vector quantization

FIGURE 14.9. *Sir Ronald A. Fisher* $(1890 - 1962)$ *was one of the founders of modern day statistics, to whom we owe maximum-likelihood, sufficiency, and many other fundamental concepts. The image on the left is a* 1024×1024 *grayscale image at* 8 *bits per pixel. The center image is the result of* 2×2 *block VQ, using* 200 *code vectors, with a compression rate of* 1.9 *bits/pixel. The right image uses only four code vectors, with a compression rate of* 0.50 *bits/pixel*

[Figure from Hastie *et al.* book]

slide by David Sontag

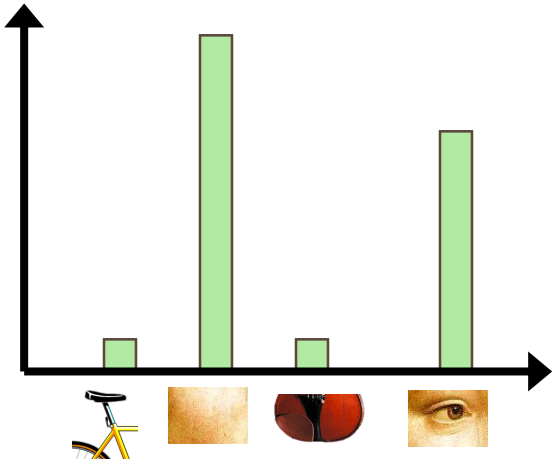
One important use of K-means

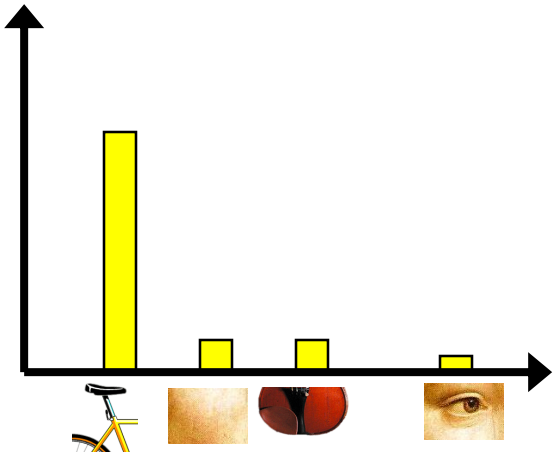
- Bag-of-word models in computer vision

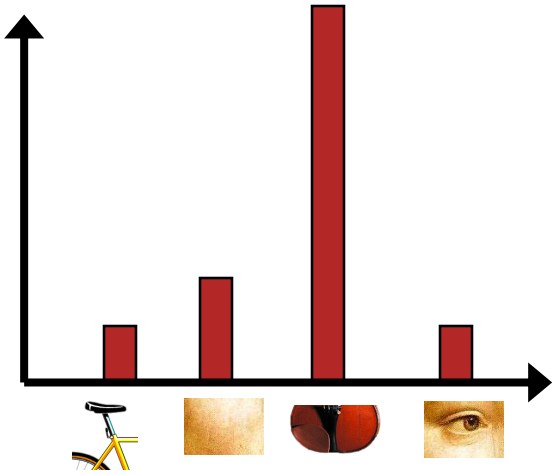
Bag of Words model

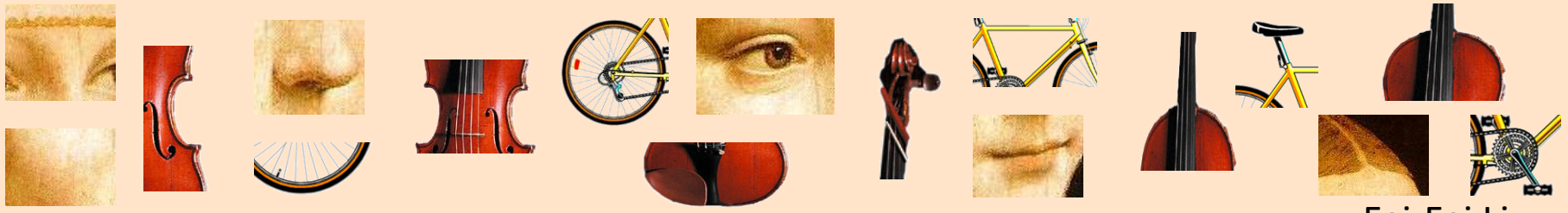
aardvark	0
about	2
all	2
Africa	1
apple	0
anxious	0
...	
gas	1
...	
oil	1
…	
Zaire	0

Object → Bag of 'words'

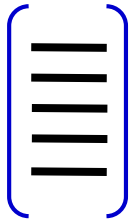






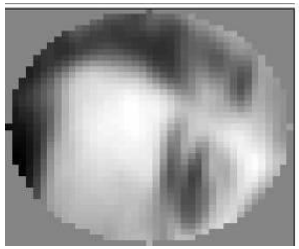


Interest Point Features

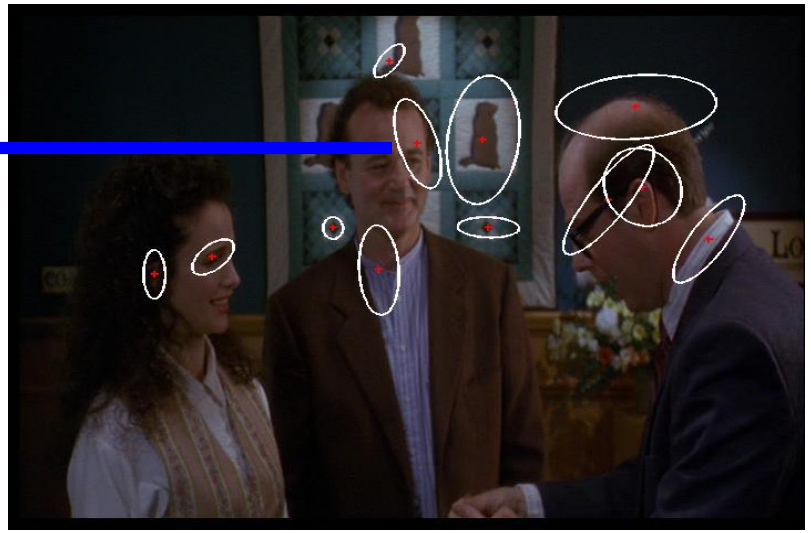


Compute SIFT descriptor

[Lowe'99]



Normalize patch



Detect patches

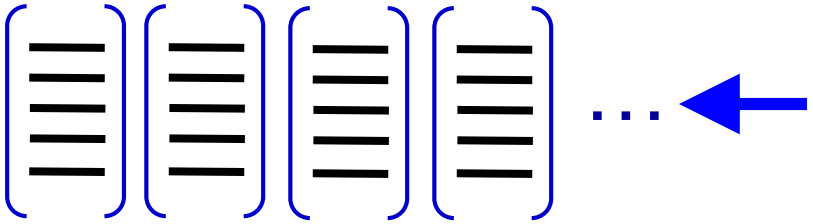
[Mikojaczyk and Schmid '02]

[Matas et al. '02]

[Sivic et al. '03]

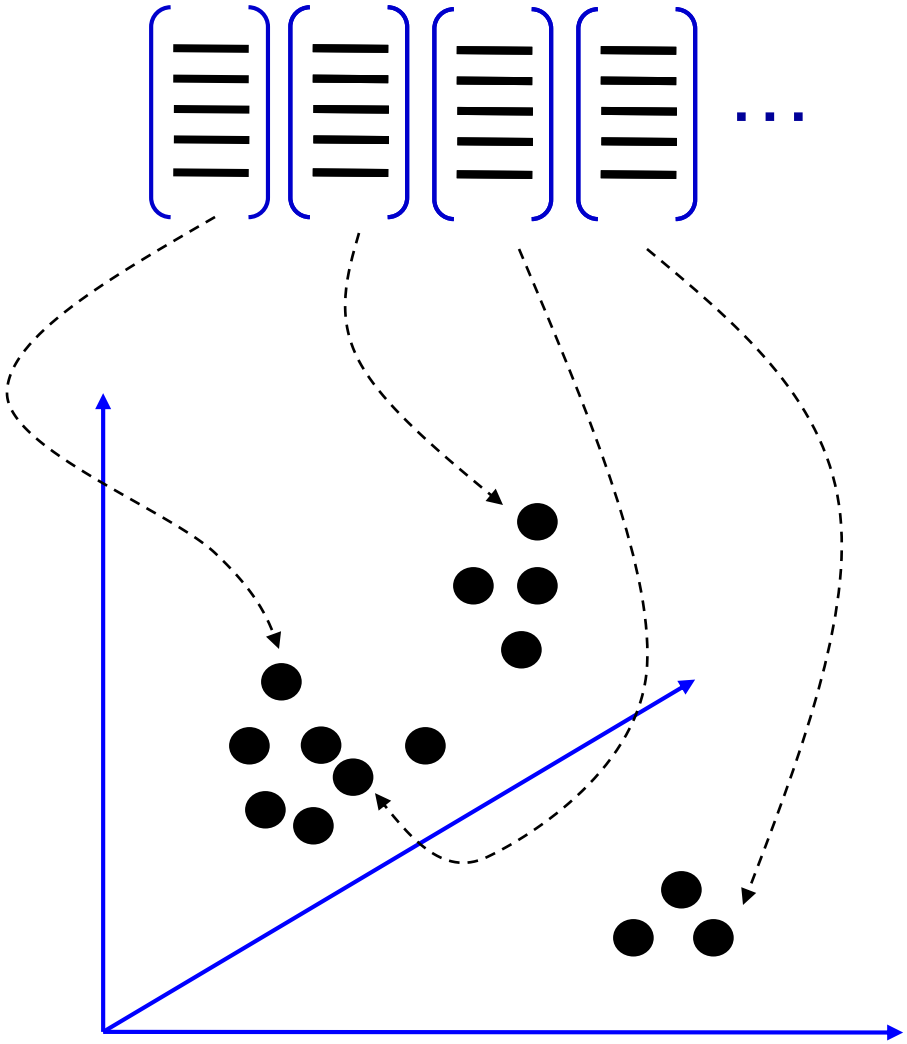
Slide credit: Josef Sivic

Patch Features



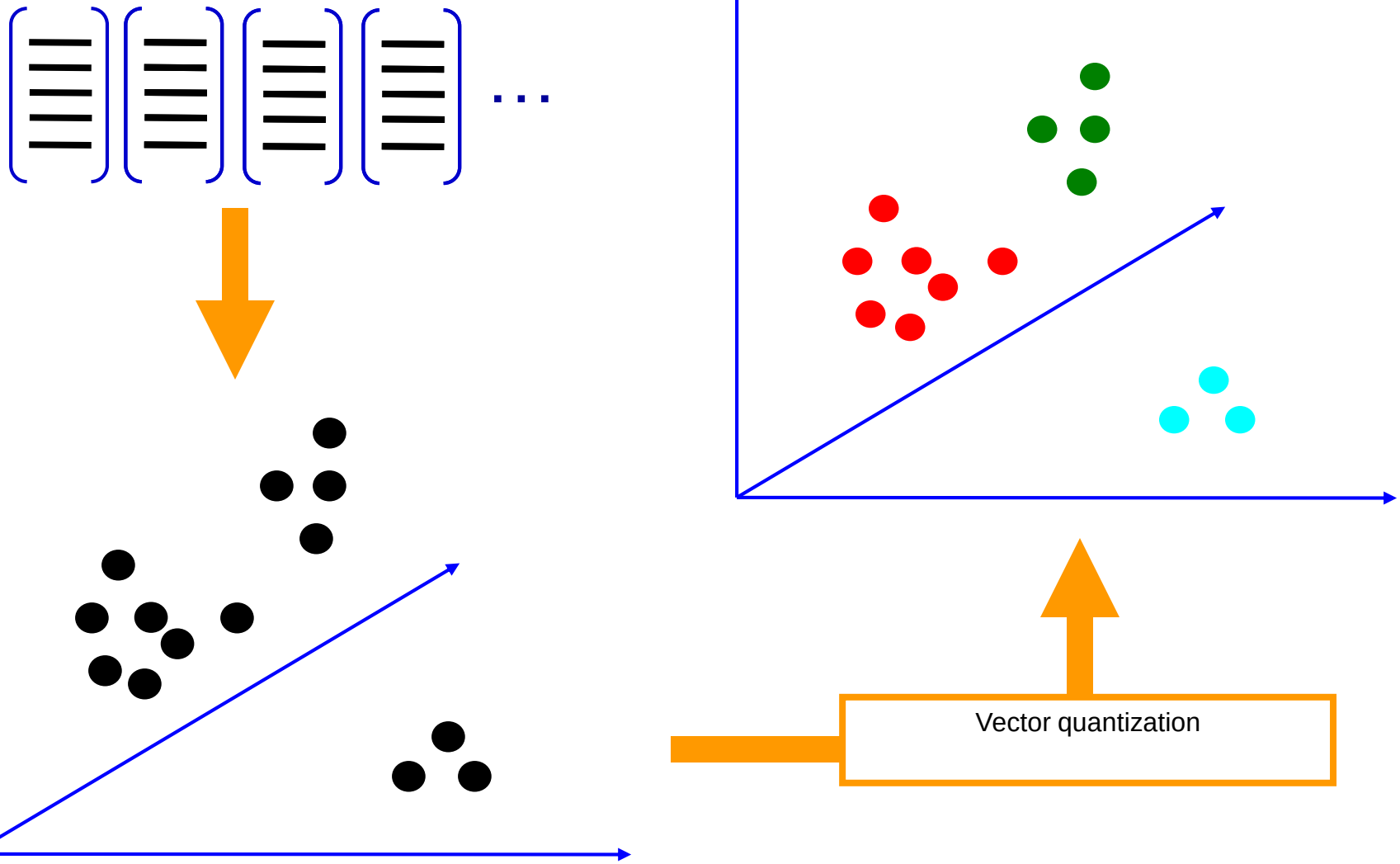
Slide credit: Josef Sivic

dictionary formation



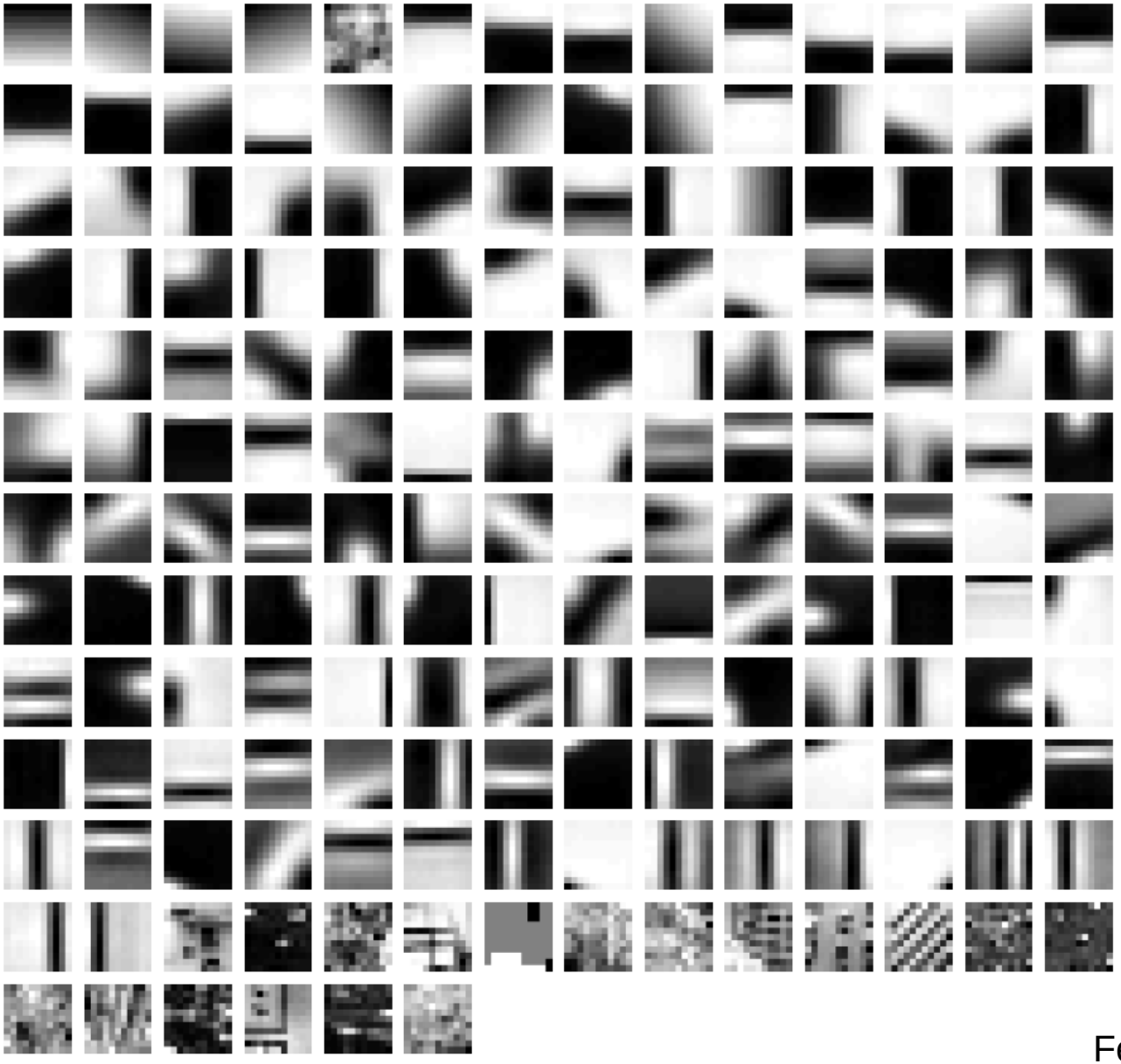
Slide credit: Josef Sivic

Clustering (usually k-means)



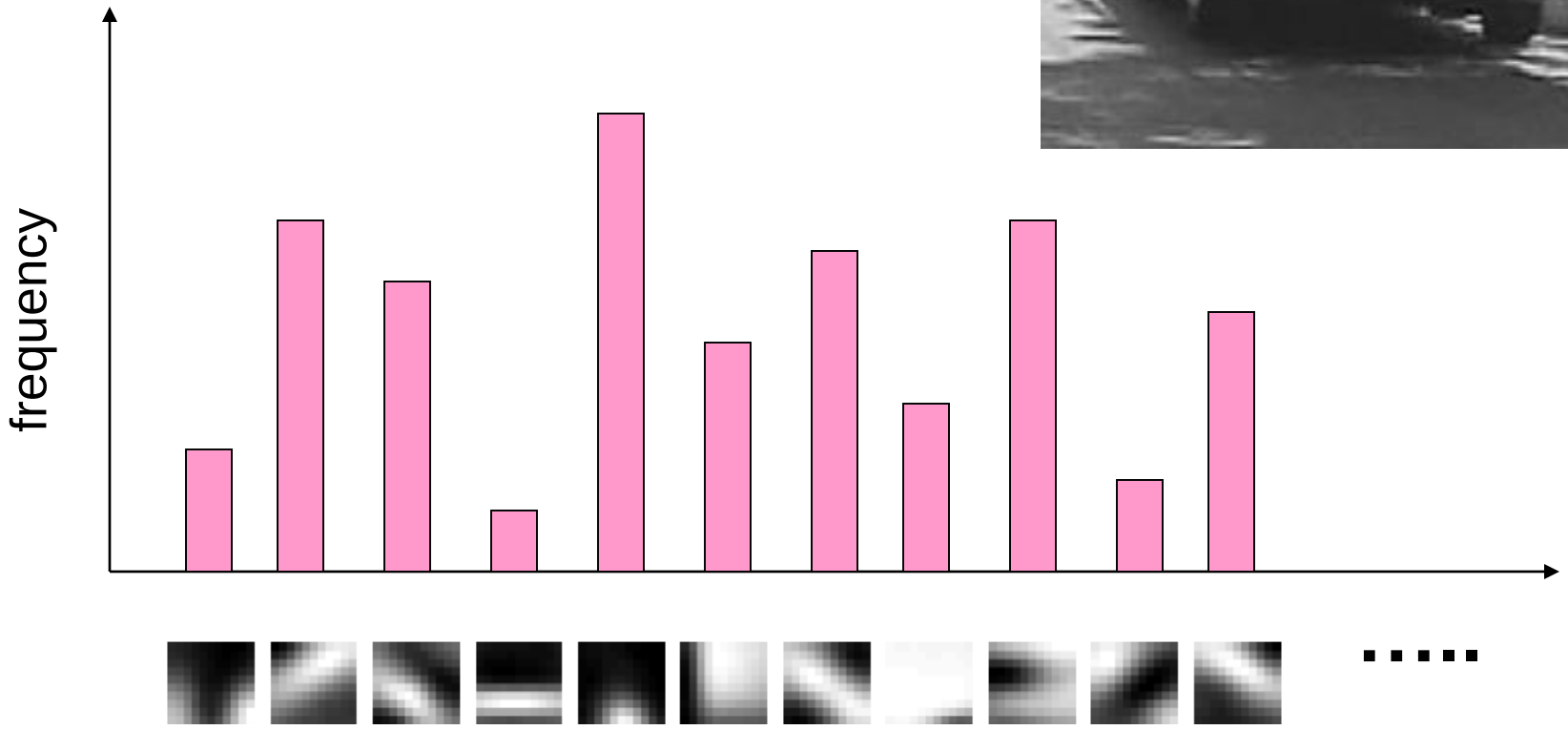
Slide credit: Josef Sivic

Clustered Image Patches



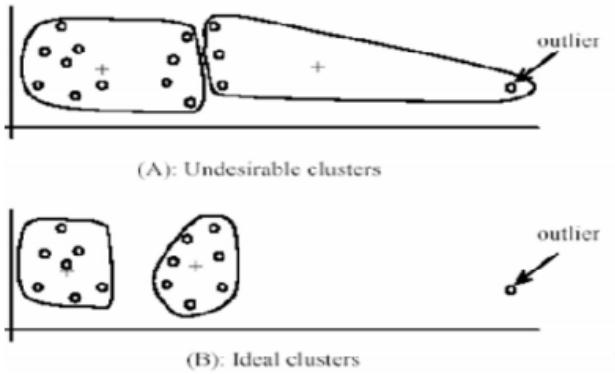
Fei-Fei et al. 2005

Image representation

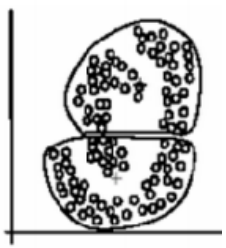


K-Means Clustering: Some Issues

- How to set k?
- Sensitive to initial centers
- Sensitive to outliers
- Detects spherical clusters
- Assuming means can be computed



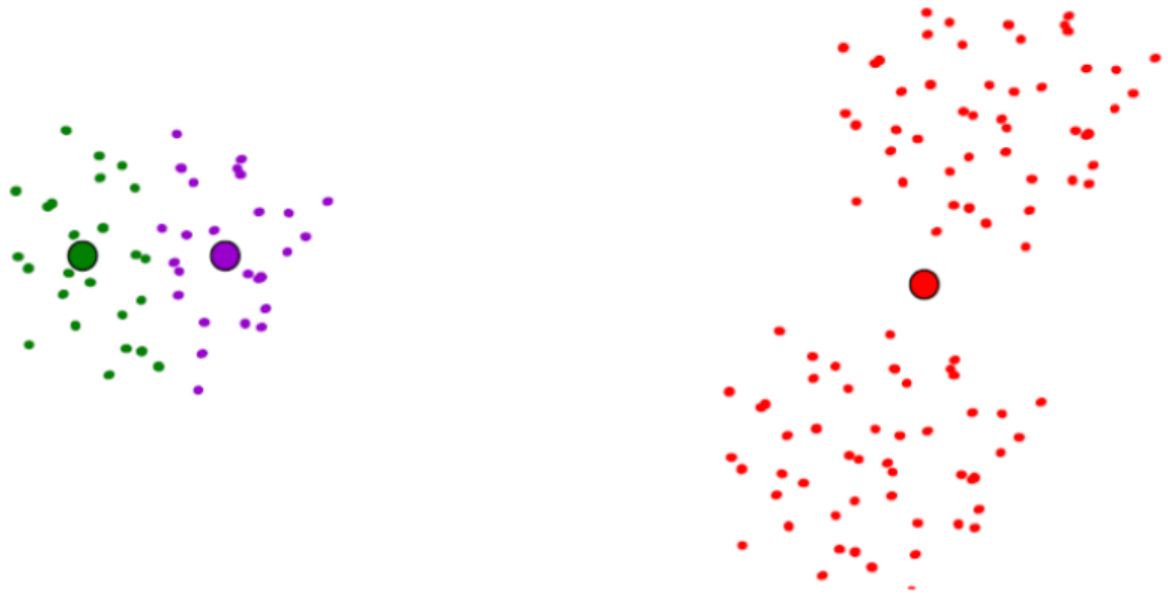

(A): Two natural clusters


(B): *k*-means clusters

slide by Kristen Grauman

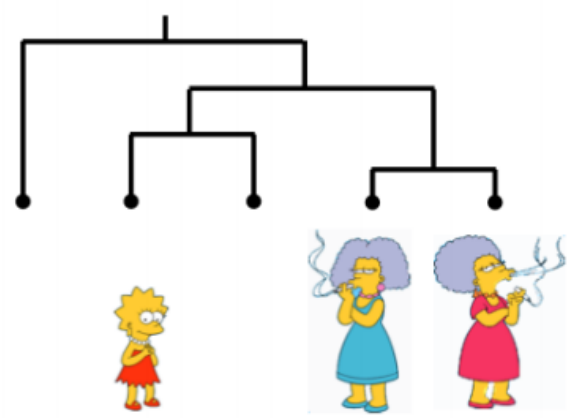
Seed Choice

Seed Choice



Hierarchical Clustering

- Bottom-Up (agglomerative): Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.

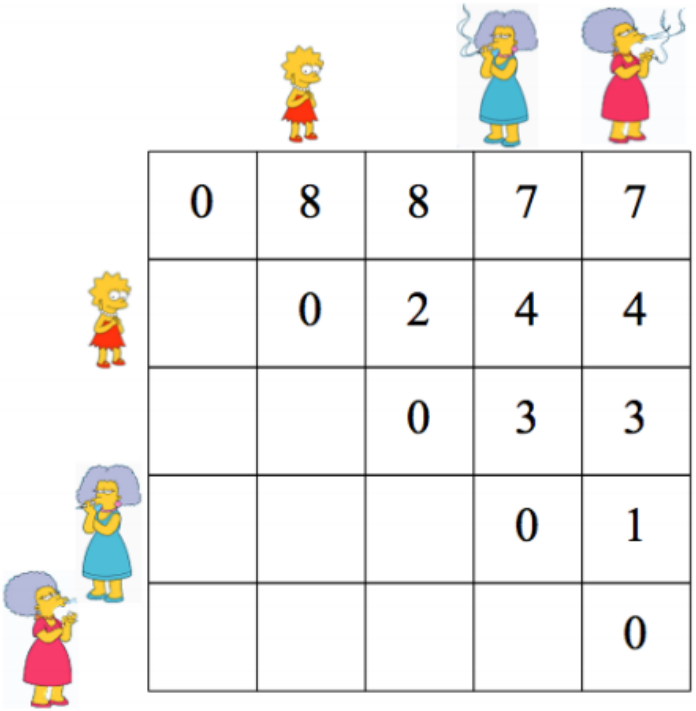


- The number of dendrograms with
  n leafs = (2n -3)!/[(2(n -2)) (n -2)!]

Number of leafs	Number of possible Dendrongrams
2	1
3	3
4	15
5	105
…	…
10	34,459,425

slide by Andrew Moore

We begin with a distance matrix which contains the distances between every pair of objects in our dataset



0	8	8	7	7
	0	2	4	4
		0	3	3
			0	1
				0

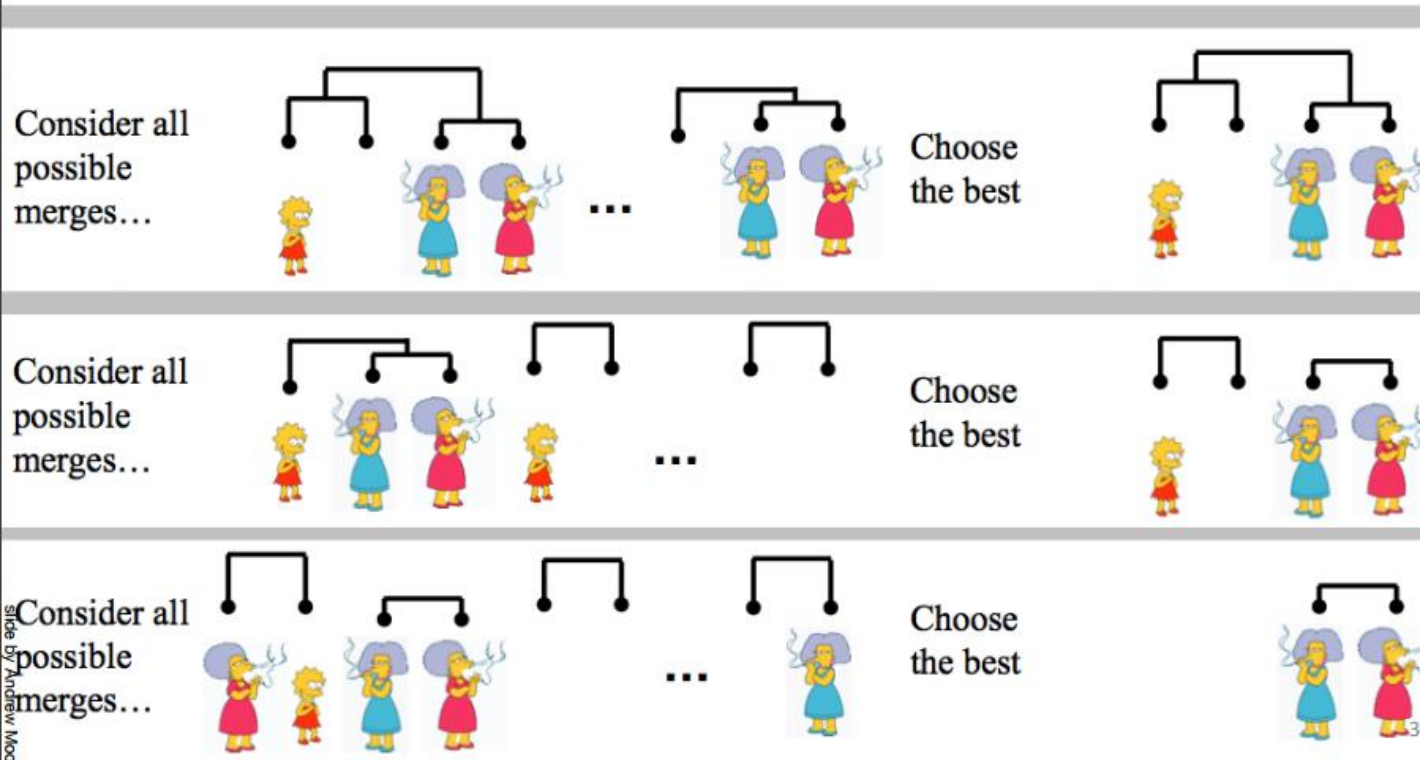
$D(\ ,) = 8$

$D(\ ,) = 1$

slide by Andrew Moore

Bottom-Up (agglomerative):

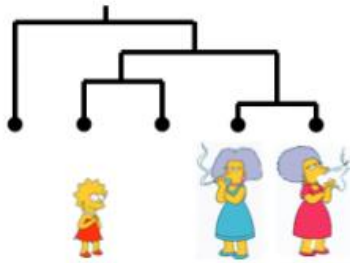
Start with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.



slide by Andrew Moore

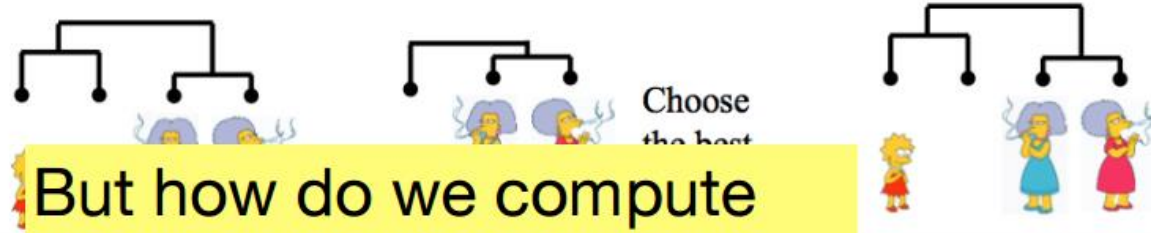
Bottom-Up (agglomerative):

Start with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.



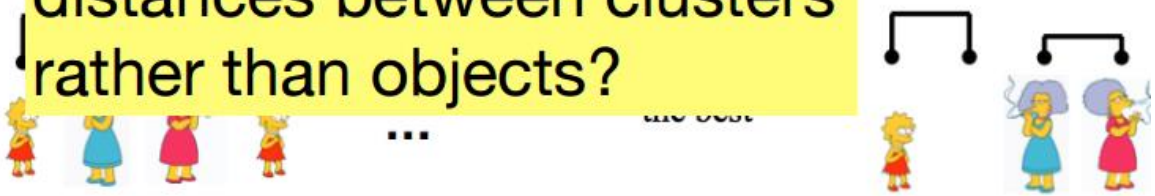
Consider all possible merges…

Choose the best



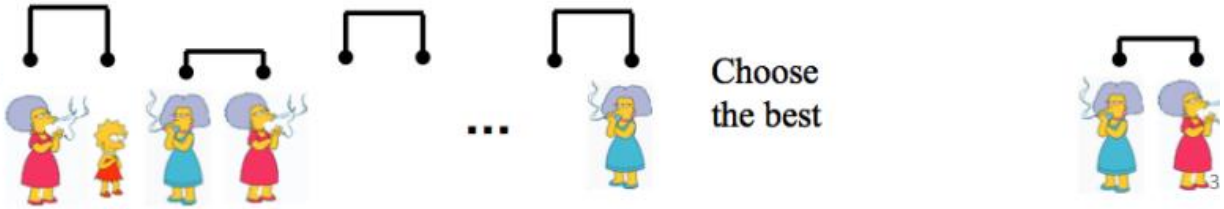
But how do we compute distances between clusters rather than objects?

Consider all possible merges…



Consider all possible merges…

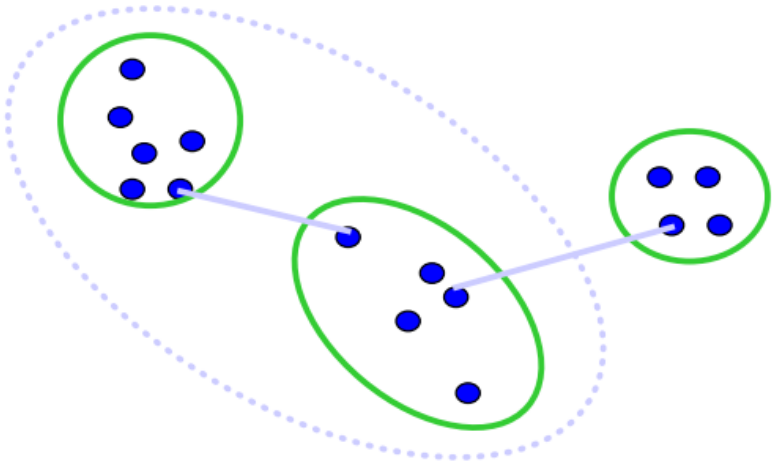
Choose the best



slide by Andrew Moore

Computing distance between clusters: **Single Link**

- Cluster distance = distance of two closest members in each class



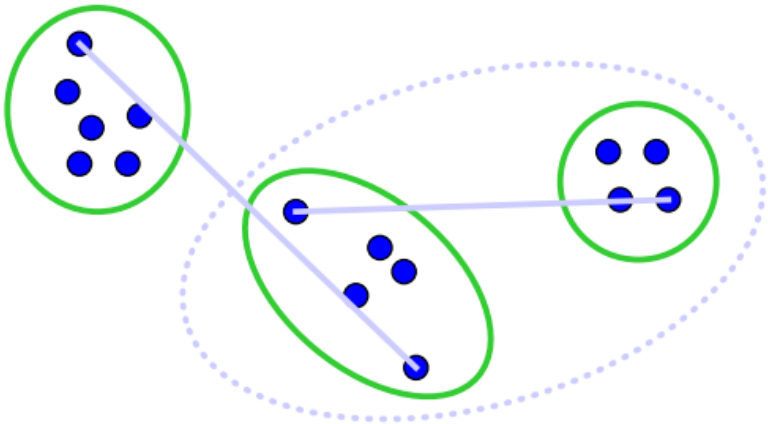
- Potentially long and skinny clusters

slide by Andrew Moore

Computing distance between clusters: **Complete Link**

- Cluster distance = distance of two farthest members in each class

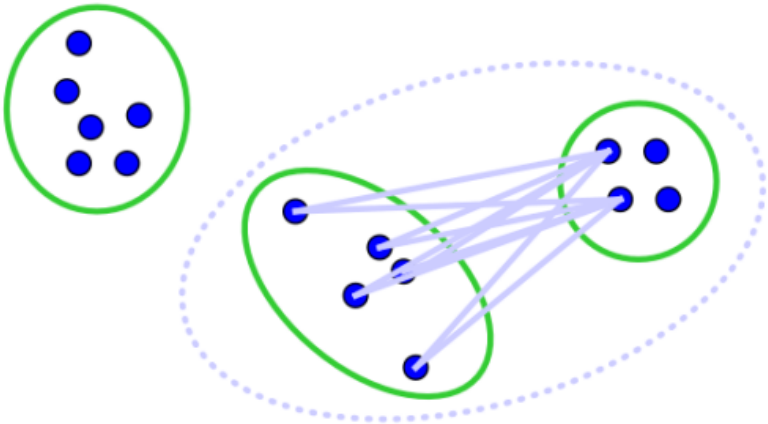
- Tight clusters



slide by Andrew Moore

Computing distance between clusters: **Average Link**

- Cluster distance = average distance of all pairs



- The most widely used measure
- Robust against noise

slide by Andrew Moore

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

slide by Derek Hoiem

What is a good clustering?

- Internal criterion: A good clustering will produce high quality clusters in which:
  - the intra-class (that is, intra-cluster) similarity is high
  - the inter-class similarity is low
  - The measured quality of a clustering depends on both the obj. representation and the similarity measure used
- External criteria for clustering quality
  - Quality measured by its ability to discover some or all of the hidden patterns or latent classes in gold standard data
  - Assesses a clustering with respect to ground truth
  - Example:
    - Purity
    - Entropy of classes in clusters (or Mutual Information between classes and clusters)

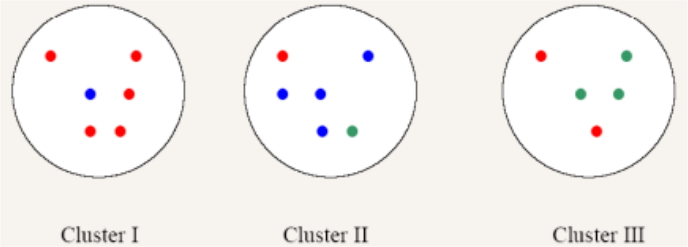
slide by Eric P. Xing

External Evaluation of Cluster Quality

- Simple measure: purity, the ratio between the dominant class in the cluster and the size of cluster
  - Assume documents with C gold standard classes, while our clustering algorithms produce K clusters, $\omega_1, \omega_2, \ldots, \omega_K$ with n_i members.

$$Purity(w_i) = \frac{1}{n_i} \max_j (n_{ij}) \quad j \in C$$

  - Example:



Cluster I: Purity = 1/6 (max(5, 1, 0)) = 5/6
Cluster II: Purity = 1/6 (max(1, 4, 1)) = 4/6
Cluster III: Purity = 1/5 (max(2, 0, 3)) = 3/5

slide by Eric P. Xing

External Evaluation of Cluster Quality

- Let:

$$TC = TC_1 \cup TC_2 \cup ... \cup TC_n$$

$$CC = CC_1 \cup CC_2 \cup ... \cup CC_m$$

be the target and computed clusterings, respectively.

- $TC = CC$ = original set of data
- Define the following:
  - a: number of pairs of items that belong to the same cluster in both CC and TC
  - b: number of pairs of items that belong to different clusters in both CC and TC
  - c: number of pairs of items that belong to the same cluster in CC but different clusters in TC
  - d: number of pairs of items that belong to the same cluster in TC but different clusters in CC

slide by Christophe Giraud-Carrier

External Evaluation of Cluster Quality

$$P = \frac{a}{a+c}$$

$$R = \frac{a}{a+d}$$

$$F = \frac{2 \times P \times R}{P+R}$$

F-measure

$$\frac{a+b}{a+b+c+d}$$

Rand Index

Measure of clustering agreement: how similar are these two ways of partitioning the data?

slide by Christophe Giraud-Carrier

External Evaluation of Cluster Quality

$$\frac{a+b}{a+b+c+d}$$

Rand Index

$$\frac{2(ab-cd)}{(a+c)(c+b)+(a+d)(d+b)}$$

Adjusted Rand Index

Extension of the Rand index that attempts to account for items that may have been clustered by chance

slide by Christophe Giraud-Carrier

External Evaluation of Cluster Quality

$$Entropy(CC_i) = \sum_{TC_j \in TC} -p(TC_j \mid CC_i) \log p(TC_j \mid CC_i)$$

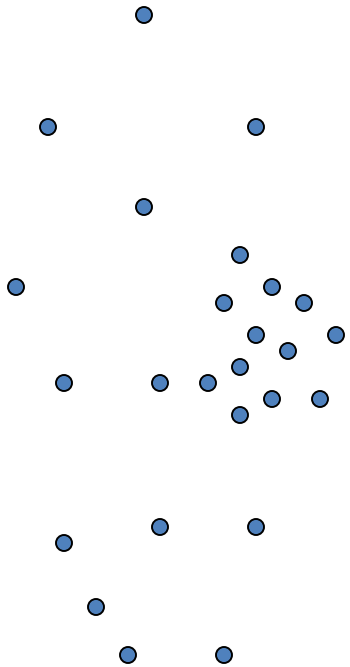
$$AvgEntropy(CC) = \sum_{i=1}^{m} \frac{|CC_i|}{|CC|} Entropy(CC_i)$$

Average Entropy

Measure of purity with respect to the target clustering

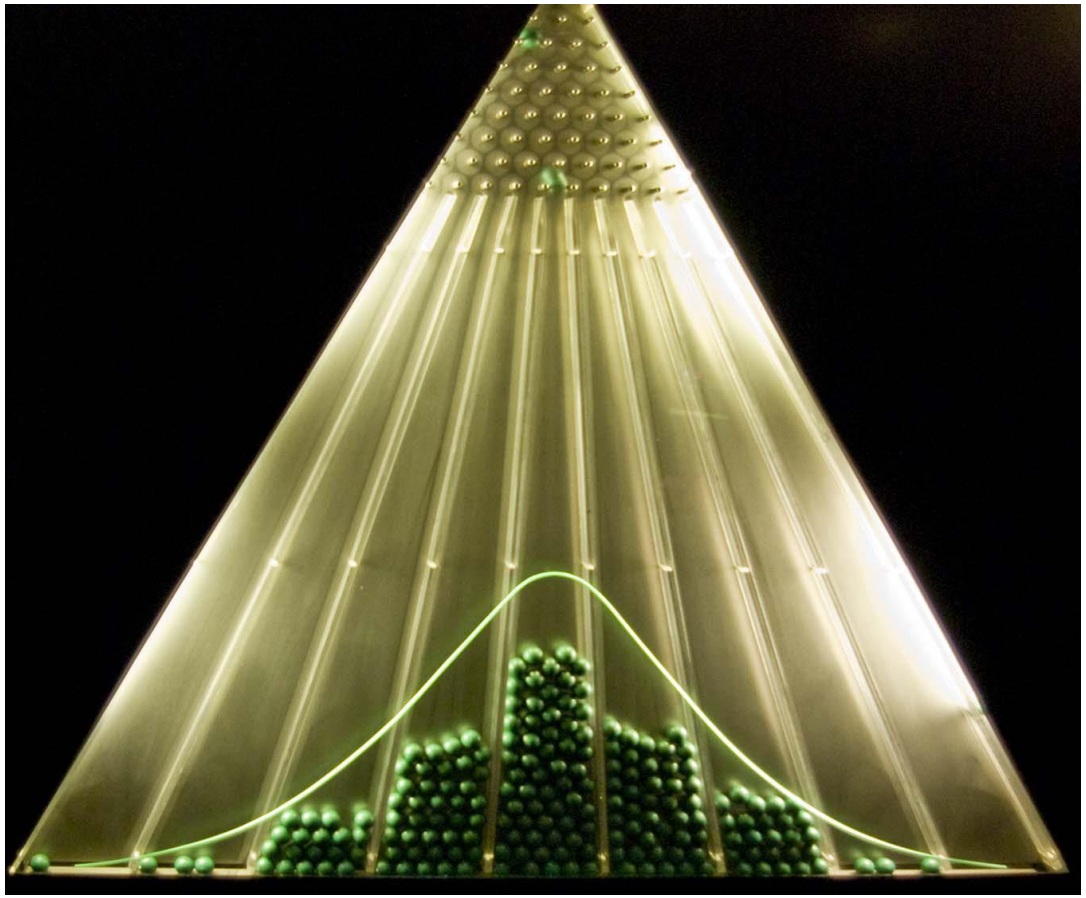
slide by Christophe Giraud-Carrier

(One) bad case for k-means



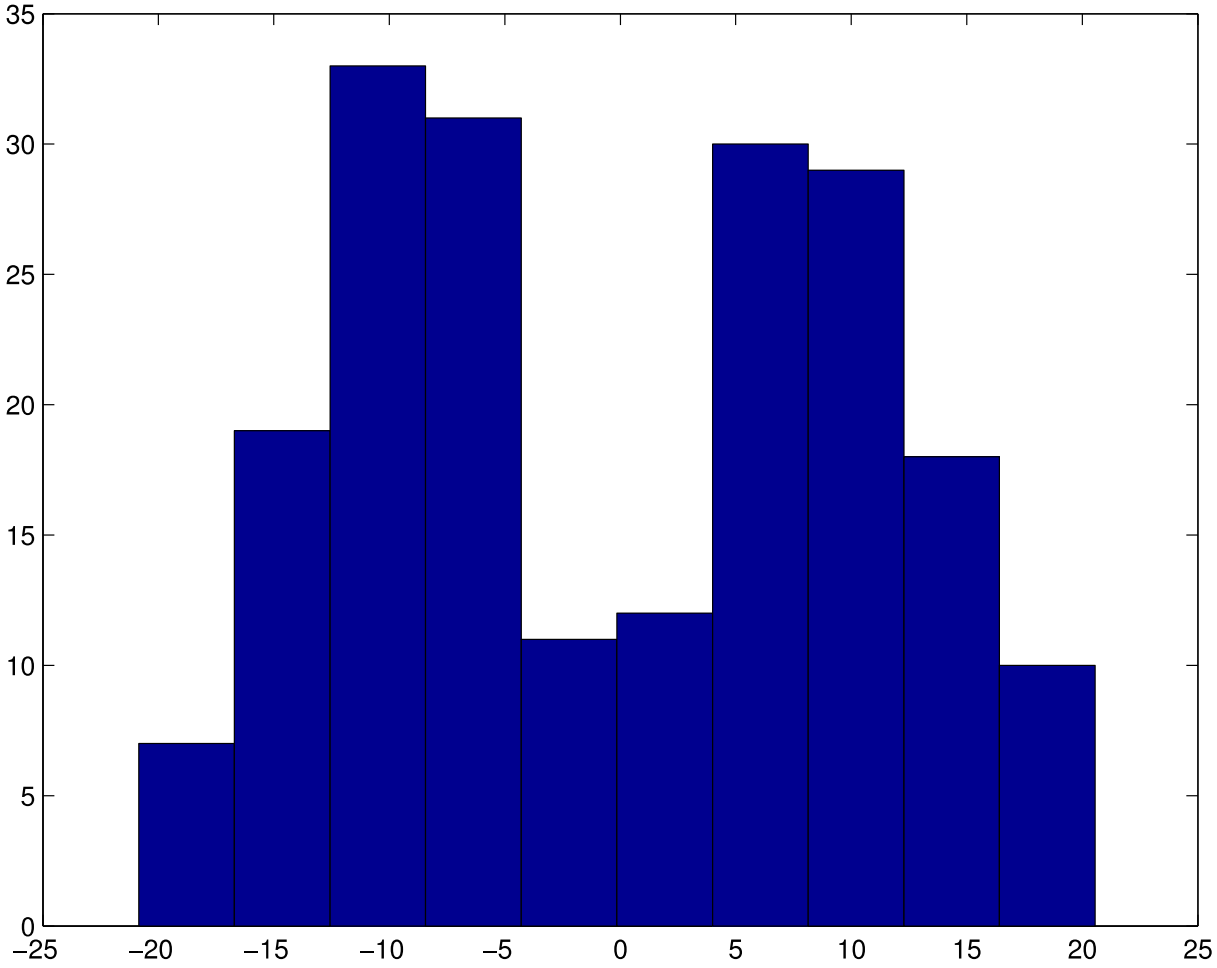
- Clusters may overlap
- Some clusters may be “wider” than others

- GMM to the rescue!

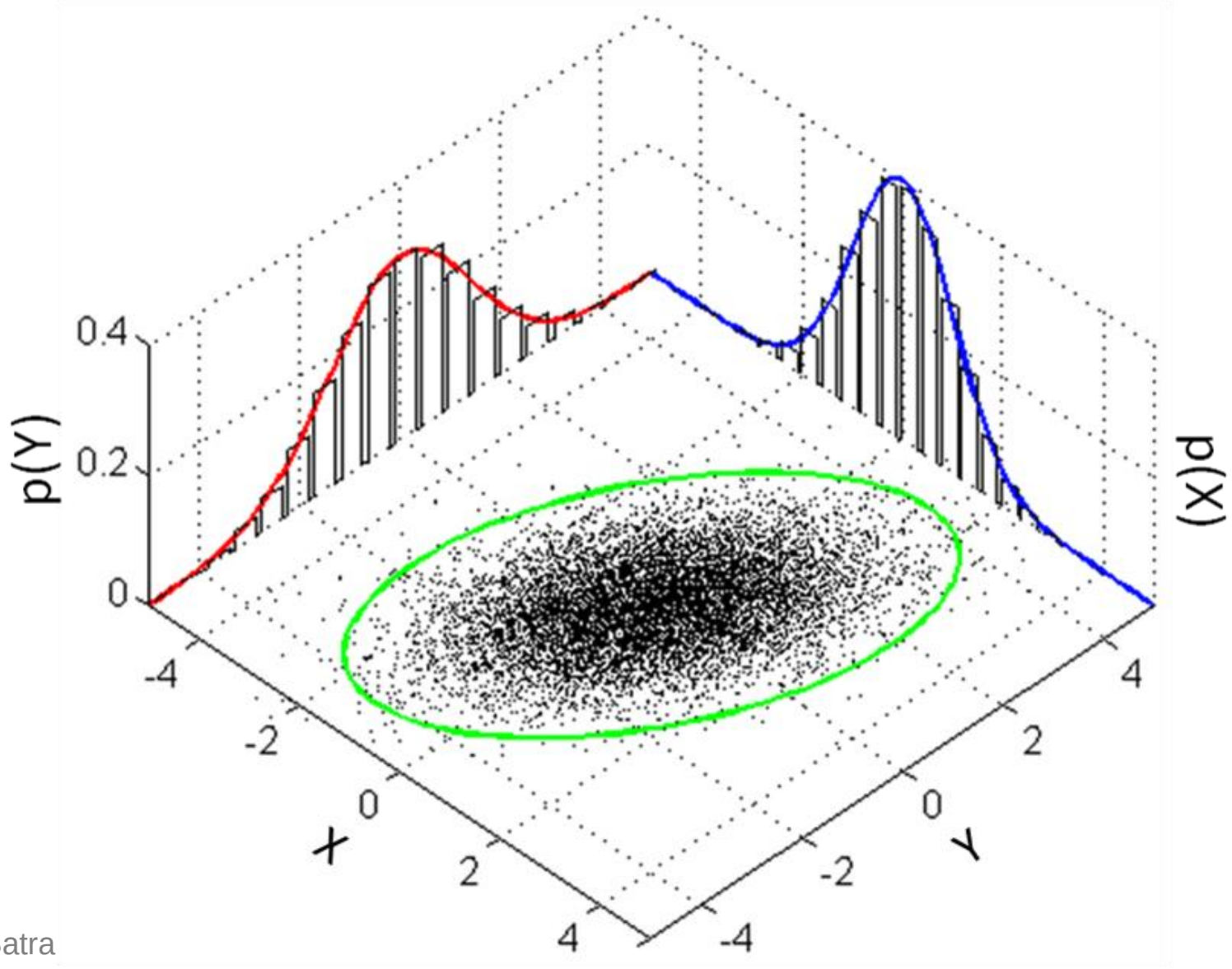


$$P(x \mid \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

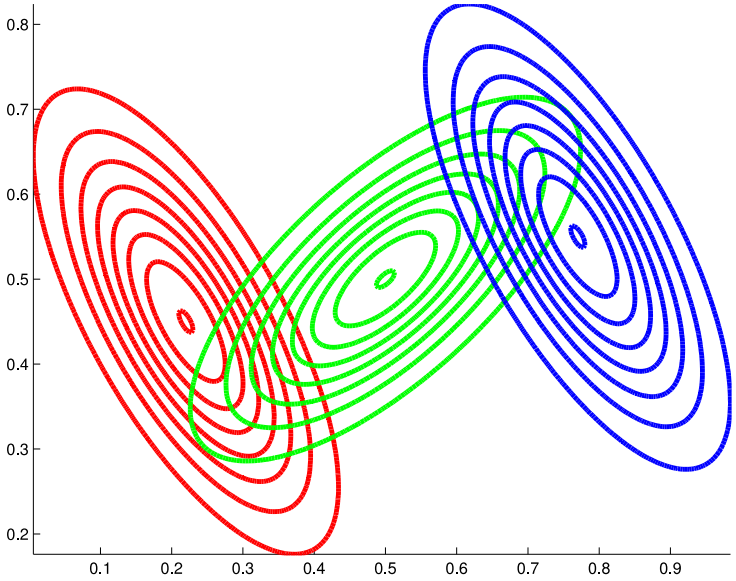
GMM



Recall Multi-variate Gaussians



GMM



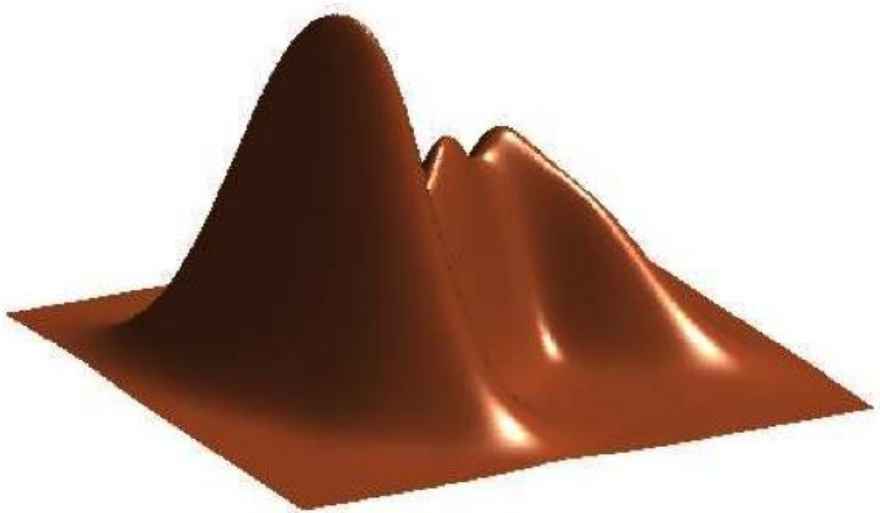


Figure Credit: Kevin Murphy

Special case: spherical Gaussians and hard assignments

- If P(X|Z=k) is spherical, with same σ for all classes:

$$P(\mathbf{x}_i \mid z=j) \propto \exp\left[-\frac{1}{2\sigma^2}\left\|\mathbf{x}_i - \mu_j\right\|^2\right]$$

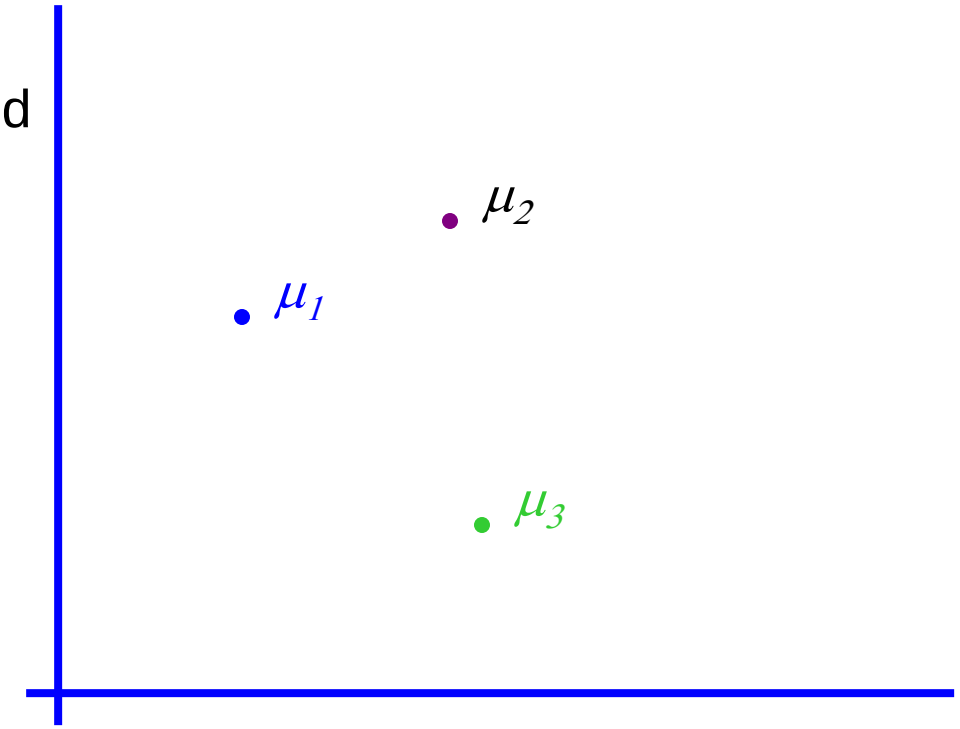
- If each x_i belongs to one class C(i) (hard assignment), marginal likelihood:

$$\prod_{i=1}^{N}\sum_{j=1}^{k} P(\mathbf{x}_i, y=j) \propto \prod_{i=1}^{N} \exp\left[-\frac{1}{2\sigma^2}\left\|\mathbf{x}_i - \mu_{C(i)}\right\|^2\right]$$

- M(M)LE same as K-means!!!

The K-means GMM assumption

- There are k components
- Component *i* has an associated mean vector μ_i

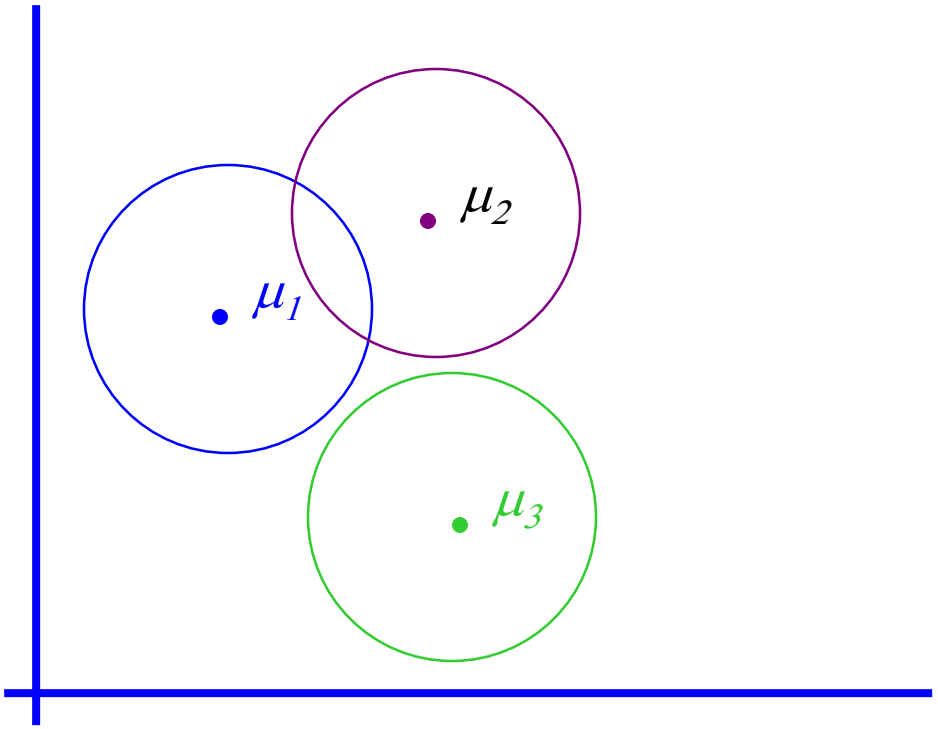


The K-means GMM assumption

- There are k components
- Component *i* has an associated mean vector μ_i

Each component generates data from a Gaussian with mean m_i and covariance matrix $\sigma^2 I$

Each data point is generated according to the following recipe:

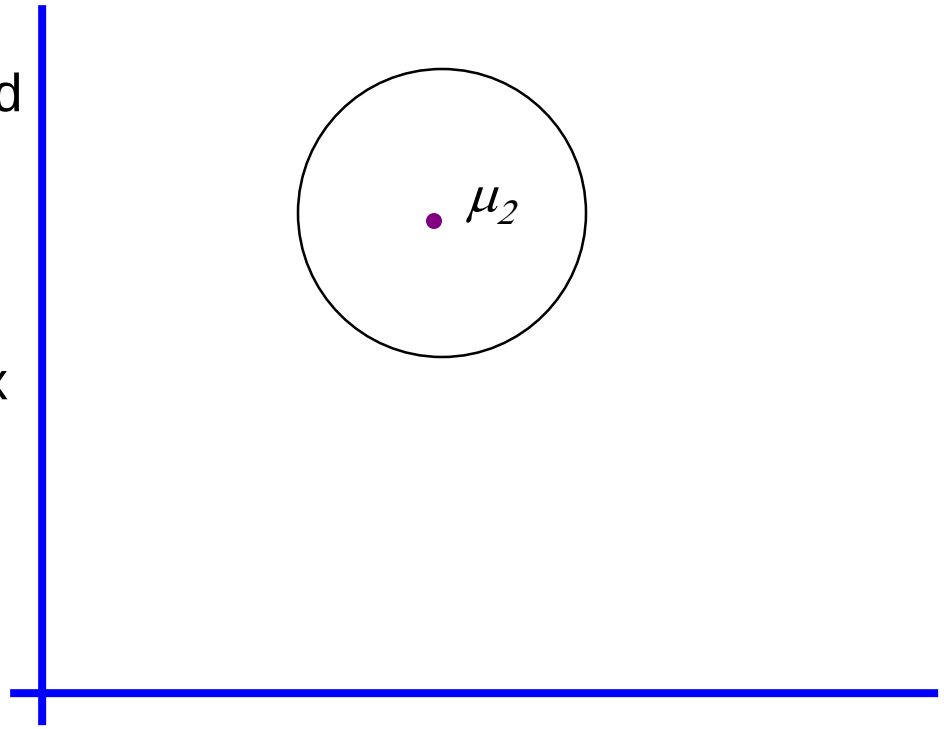


The K-means GMM assumption

- There are k components
- Component *i* has an associated mean vector μ_i
- Each component generates data from a Gaussian with mean m_i and covariance matrix $\sigma^2 I$

Each data point is generated according to the following recipe:

1. Pick a component at random: Choose component i with probability *P(y=i)*

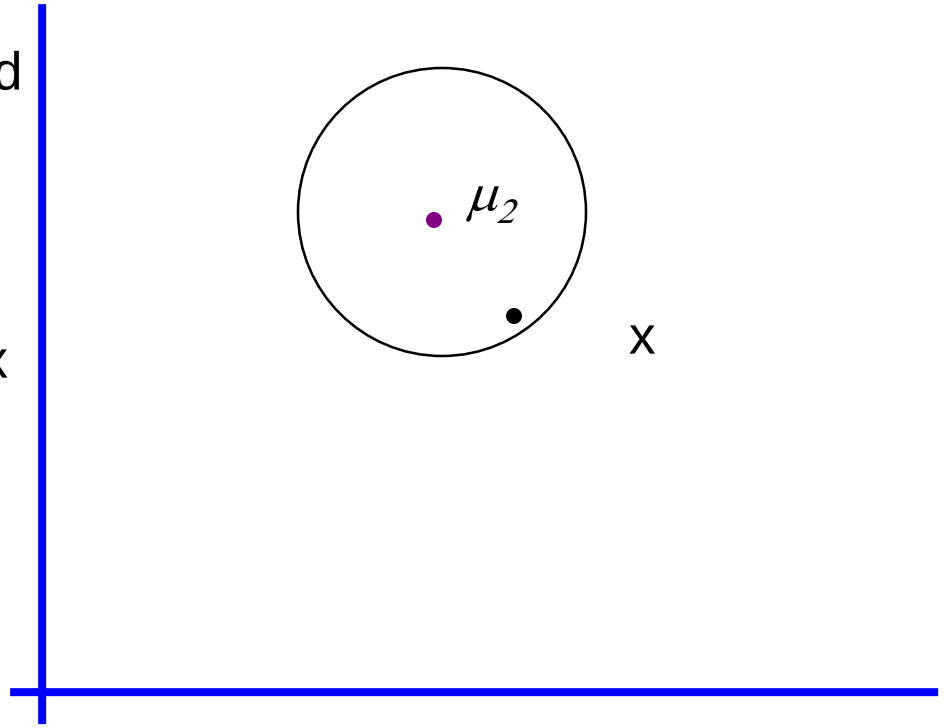


The K-means GMM assumption

- There are k components
- Component *i* has an associated mean vector μ_i
- Each component generates data from a Gaussian with mean m_i and covariance matrix $\sigma^2 \boldsymbol{I}$

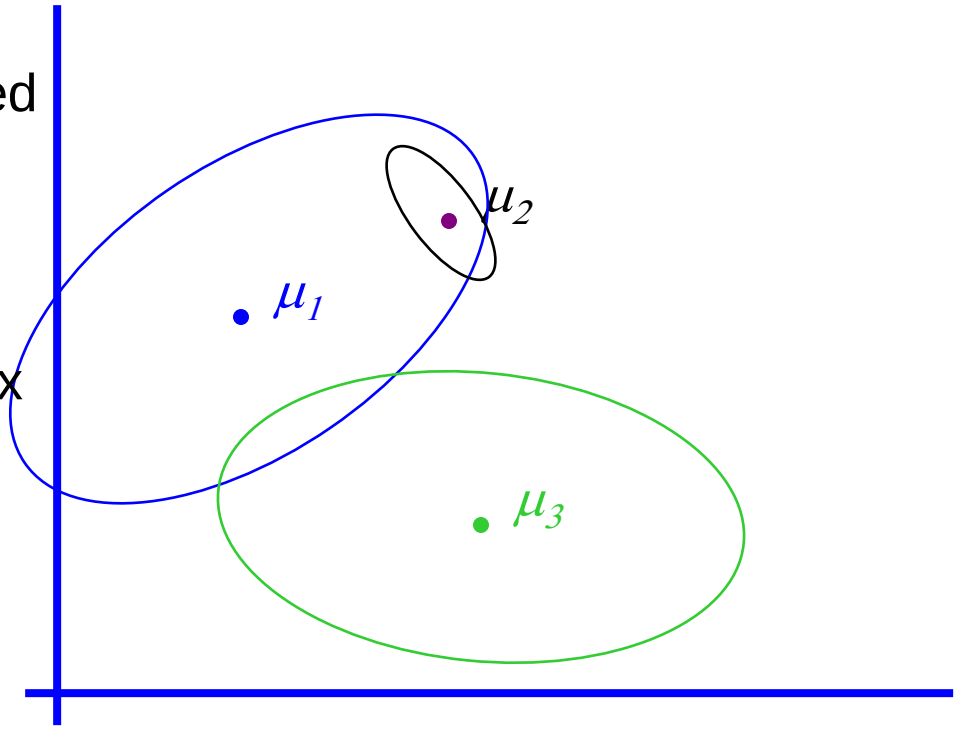
Each data point is generated according to the following recipe:

1. Pick a component at random: Choose component i with probability *P(y=i)*
2. Datapoint ~ $\mathrm{N}(\mu_i, \sigma^2 \boldsymbol{I})$



The General GMM assumption

- There are k components
- Component *i* has an associated mean vector m_i
- Each component generates data from a Gaussian with mean m_i and covariance matrix Σ_i



Each data point is generated according to the following recipe:

1. Pick a component at random: Choose component i with probability *P(y=i)*
2. Datapoint ~ N(m_i, Σ_i)

K-means vs GMM

- K-Means
  - http://home.deib.polimi.it/matteucc/Clustering/tutorial_html/AppletKM.html
- GMM
  - http://www.socr.ucla.edu/applets.dir/mixtureem.html

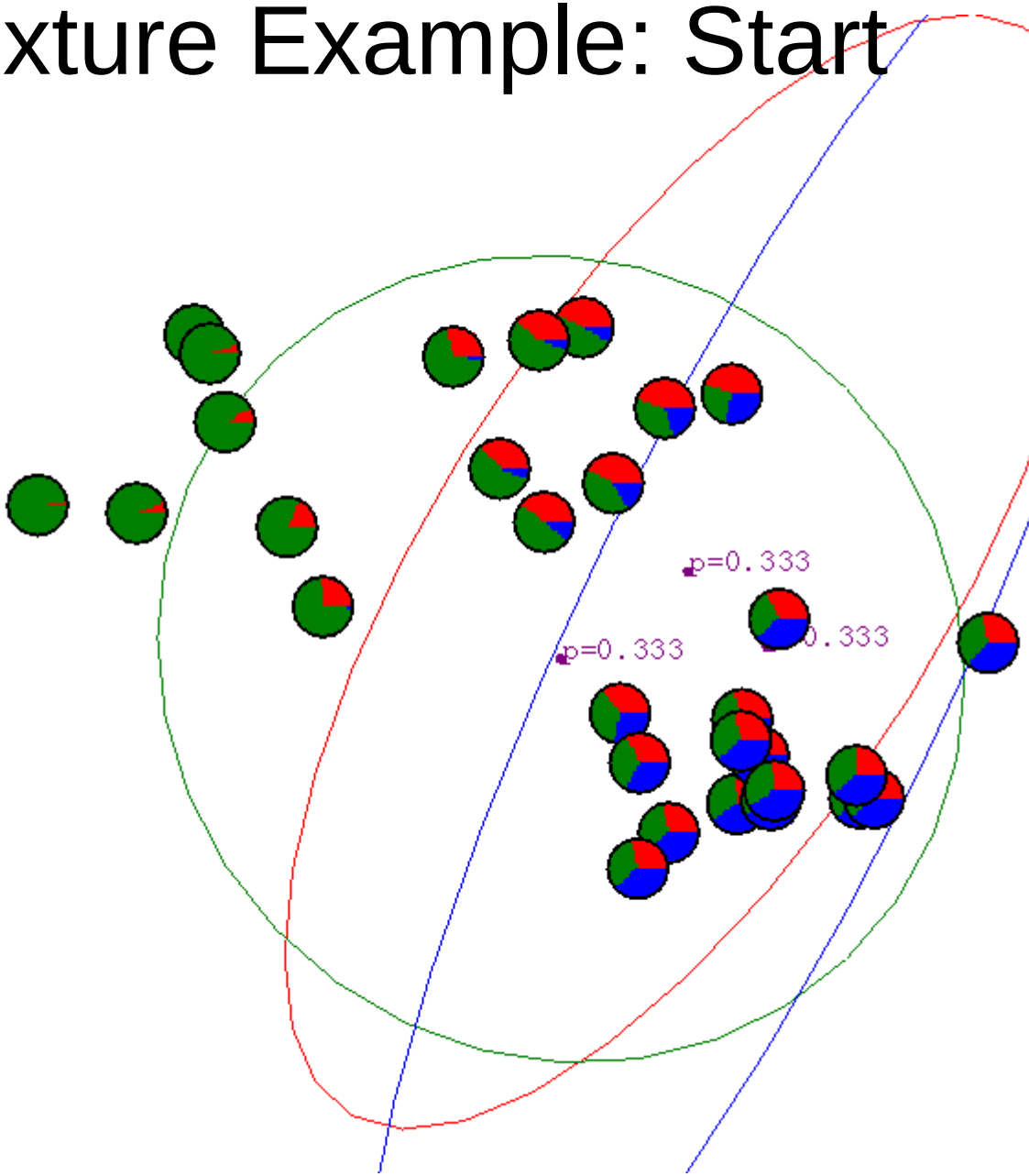
EM

- Expectation Maximization [Dempster ‘77]
- Often looks like “soft” K-means
- Extremely general
- Extremely useful algorithm
  - Essentially THE goto algorithm for unsupervised learning
- Plan
  - EM for learning GMM parameters
  - EM for general unsupervised learning problems

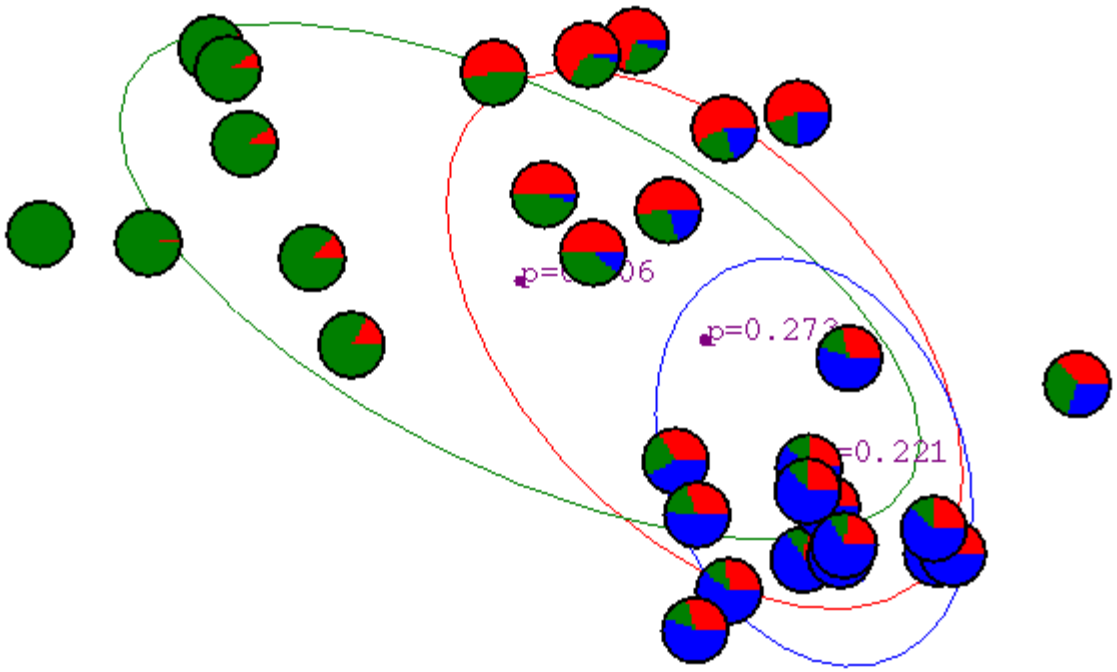
EM for Learning GMMs

- Simple Update Rules
  - E-Step: estimate $P(z_i = j \mid x_i)$
  - M-Step: maximize full likelihood weighted by posterior

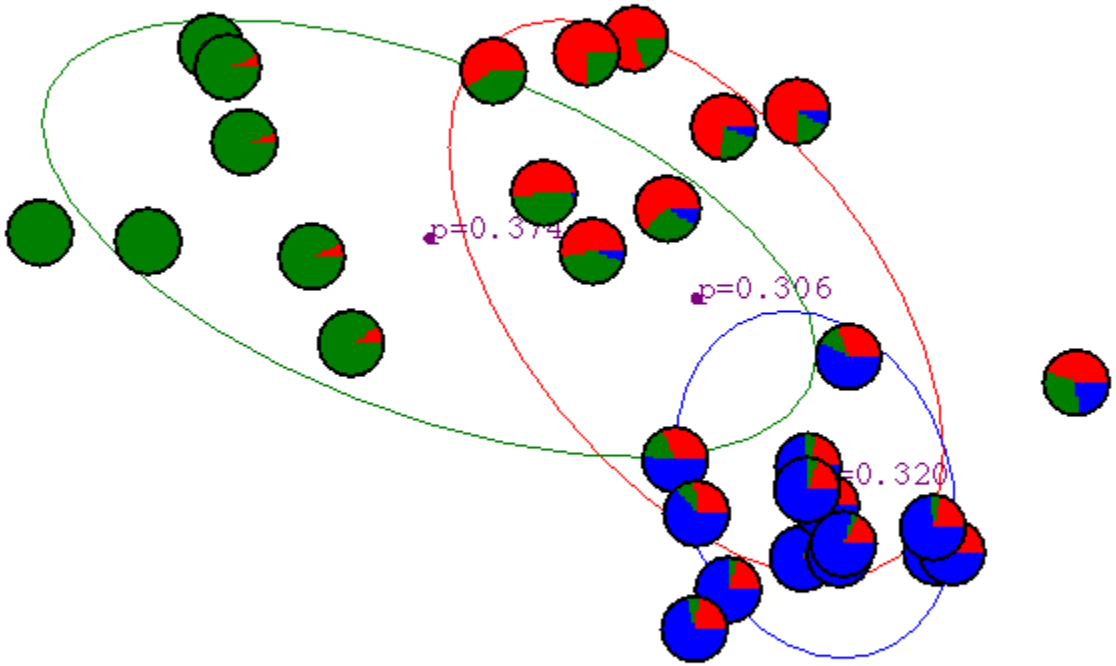
Gaussian Mixture Example: Start



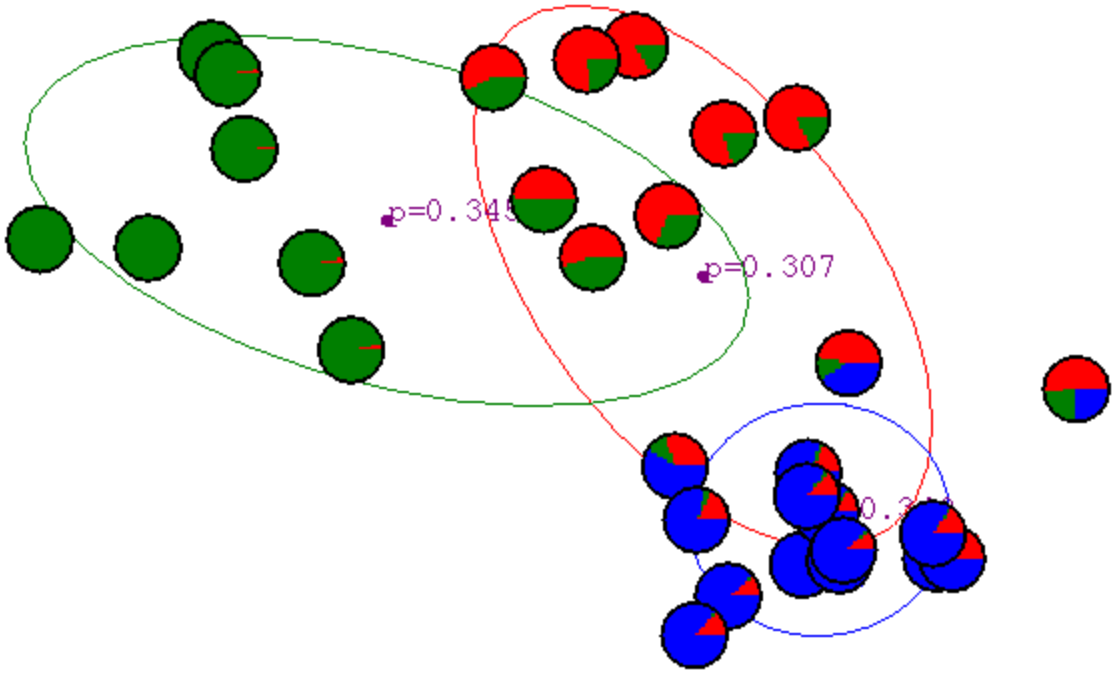
After 1st iteration



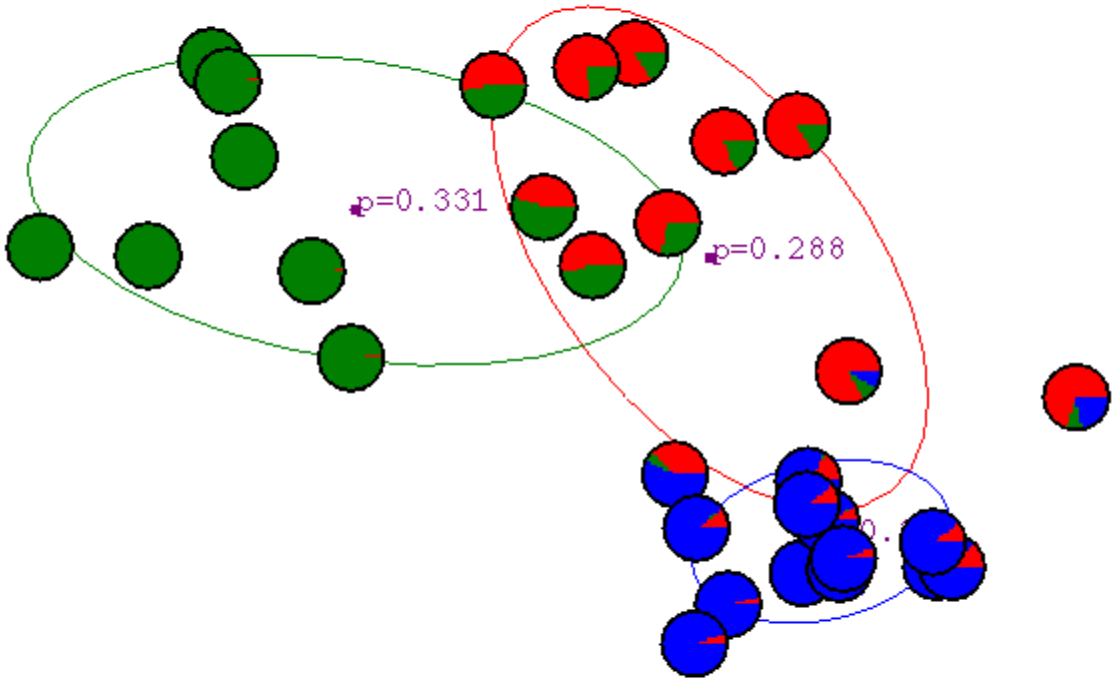
After 2nd iteration



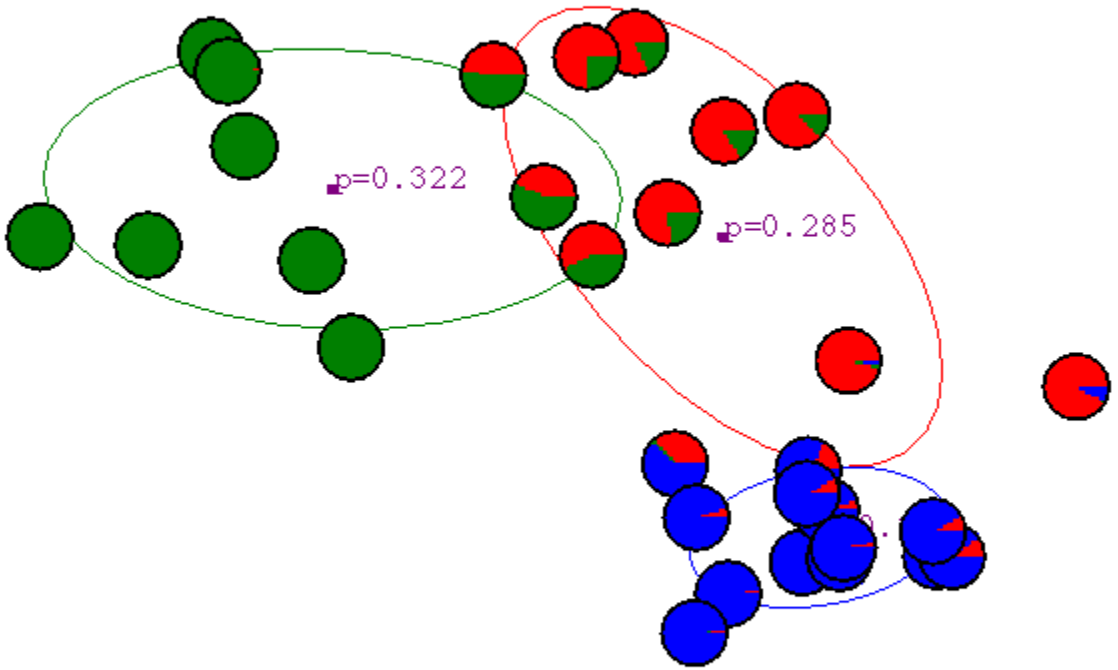
After 3rd iteration



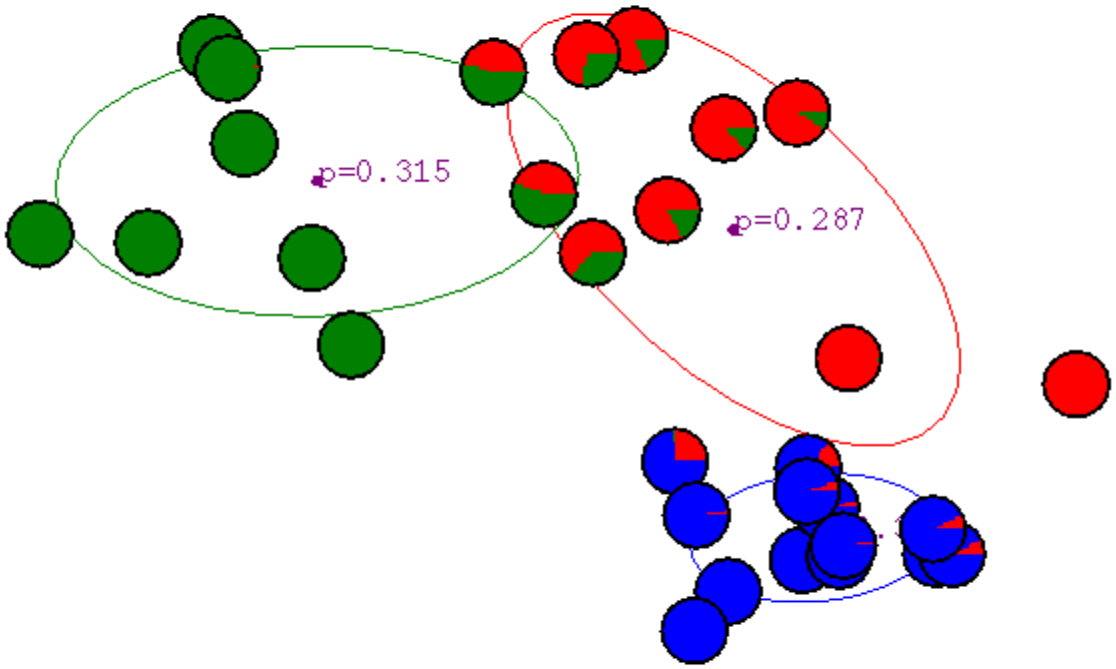
After 4th iteration



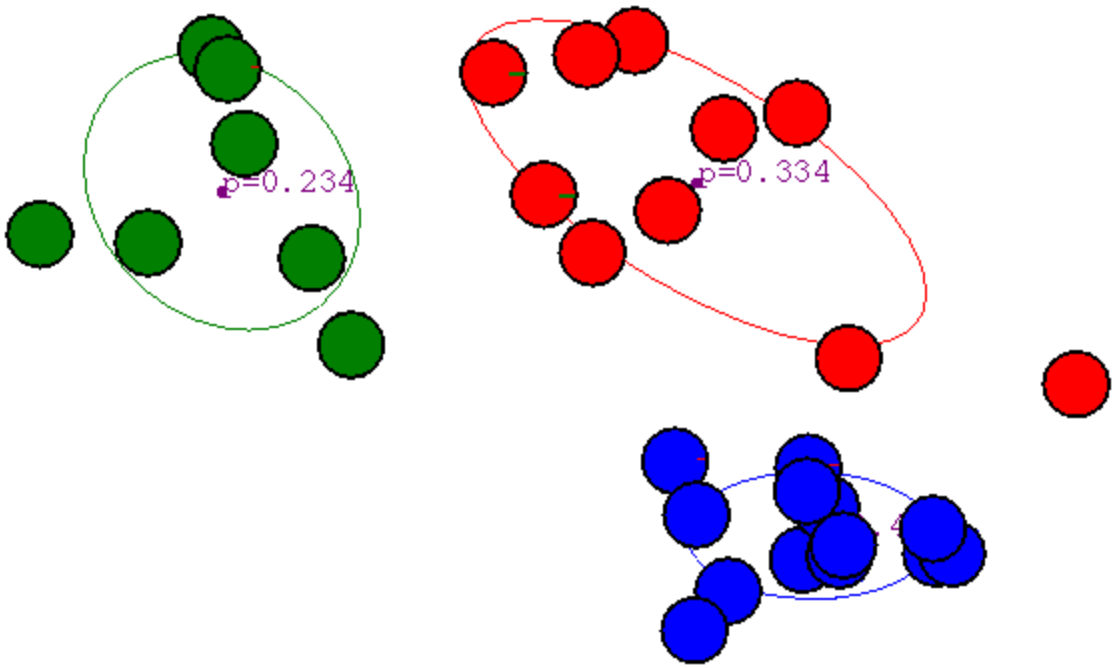
After 5th iteration



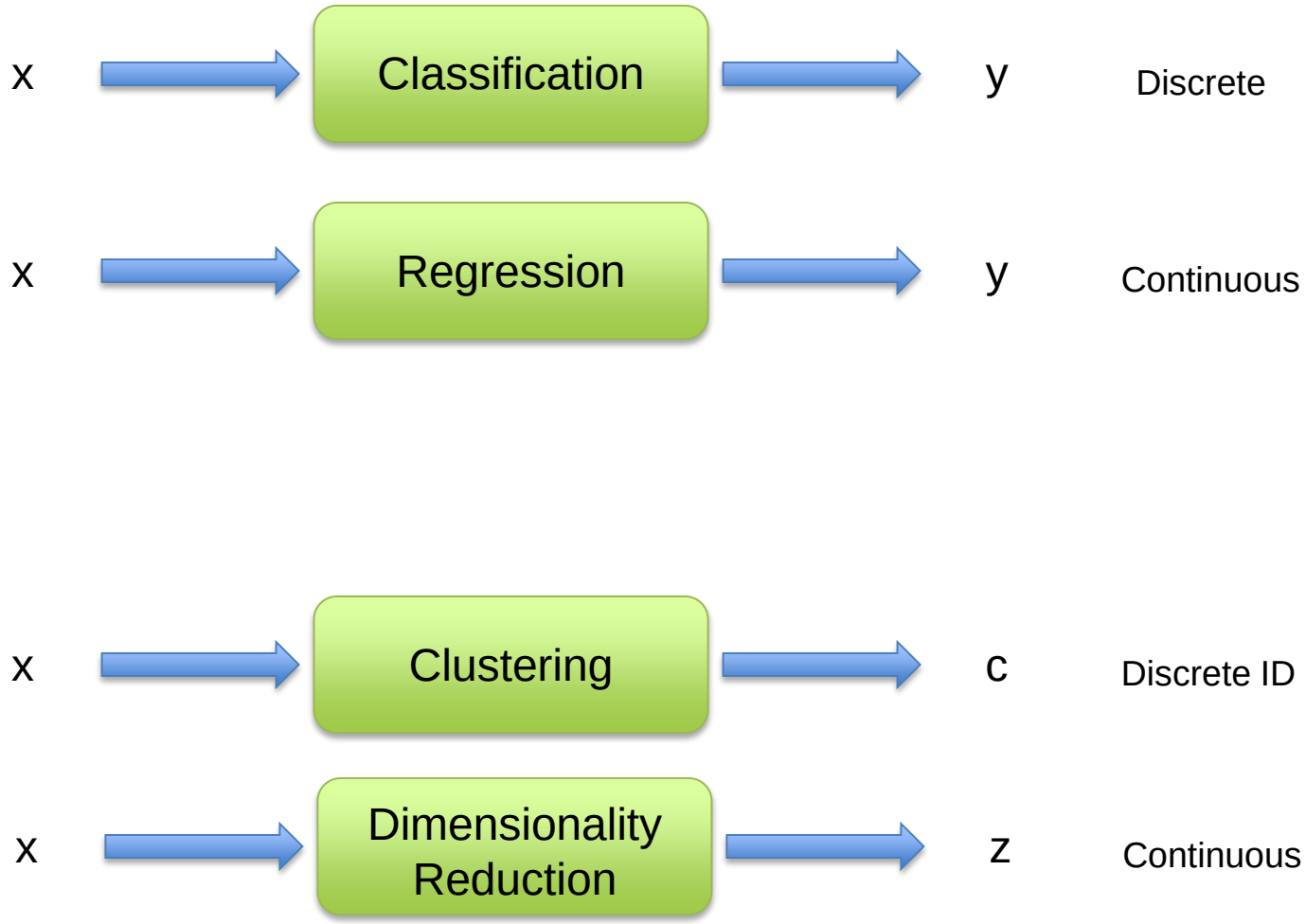
After 6th iteration



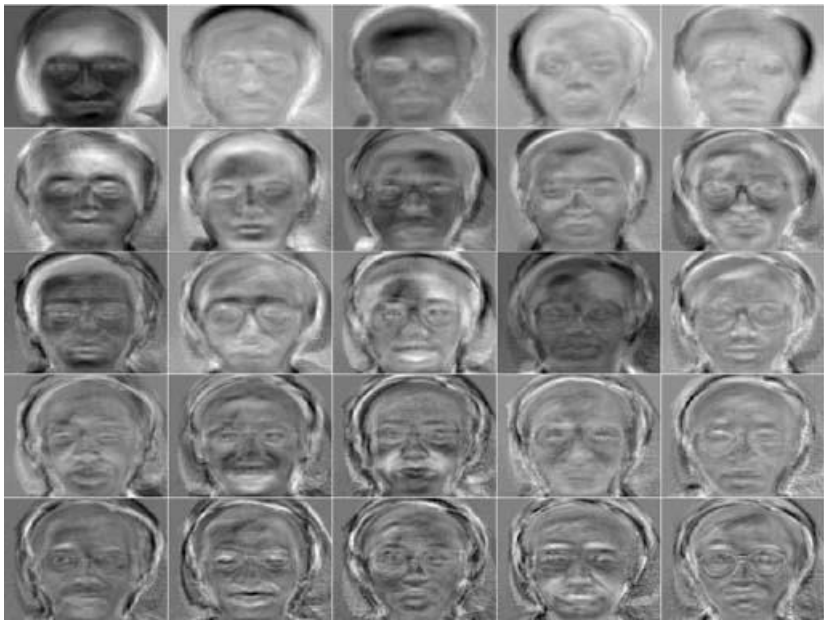
After 20th iteration



Tasks


x
Classification
y
Discrete
x
Regression
y
Continuous
x
Clustering
c
Discrete ID
x
Dimensionality Reduction
z
Continuous

New Topic: PCA



Synonyms

- Principal Component Analysis
- Karhunen–Loève transform

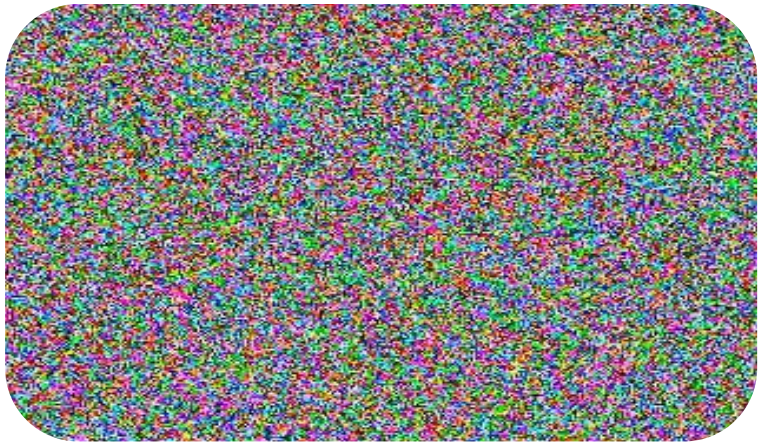
- Eigen-Faces
- Eigen-<Insert-your-problem-domain>

- PCA is a Dimensionality Reduction Algorithm

- Other Dimensionality Reduction algorithms
  - Linear Discriminant Analysis (LDA)
  - Independent Component Analysis (ICA)
  - Local Linear Embedding (LLE)
  - …

Dimensionality reduction

- Input data may have thousands or millions of dimensions!
  - e.g., images have 5M pixels



Dimensionality reduction

- Input data may have thousands or millions of dimensions!
  - e.g., images have 5M pixels

- **Dimensionality reduction**: represent data with fewer dimensions
  - easier learning – fewer parameters
  - visualization – hard to visualize more than 3D or 4D
  - discover “intrinsic dimensionality” of data
    - high dimensional data that is truly lower dimensional

PCA / KL-Transform

- De-correlation view
  - Make features uncorrelated
  - No projection yet
- Max-variance view:
  - Project data to lower dimensions
  - Maximize variance in lower dimensions
- Synthesis / Min-error view:
  - Project data to lower dimensions
  - Minimize reconstruction error
- All views lead to same solution

Basic PCA algorithm

- **Center data (**subtract mean)
- **Estimate covariance**
- **Find eigenvectors and values of covariance**
- **Principle components:** choose k eigenvectors with highest corresponding values

Video

- https://youtu.be/pSRA8GpWIrA?t=162

Video

- What if the dimension is high?
  - Covariance matrix is d x d
  - For high d, Eigen decomposition is very slow… $O(d^3)$
- Use Singular Value Decomposition (SVD)
  - finds k-eigenvectors
  - great implementations $O(N^2d)$

PCA Applications

- Data Visualization
- Data Compression
- Noise Reduction
- Learning
- Anomaly detection

slide by Barnabás Póczos and Aarti Singh

Data Visualization

Example:

- Given 53 blood and urine samples (features) from 65 people.
- How can we visualize the measurements?

Data Visualization

- Matrix format (65x53)

	H-WBC	H-RBC	H-Hgb	H-Hct	H-MCV	H-MCH	H-MCHC
A1	8.0000	4.8200	14.1000	41.0000	85.0000	29.0000	34.0000
A2	7.3000	5.0200	14.7000	43.0000	86.0000	29.0000	34.0000
A3	4.3000	4.4800	14.1000	41.0000	91.0000	32.0000	35.0000
A4	7.5000	4.4700	14.9000	45.0000	101.0000	33.0000	33.0000
A5	7.3000	5.5200	15.4000	46.0000	84.0000	28.0000	33.0000
A6	6.9000	4.8600	16.0000	47.0000	97.0000	33.0000	34.0000
A7	7.8000	4.6800	14.7000	43.0000	92.0000	31.0000	34.0000
A8	8.6000	4.8200	15.8000	42.0000	88.0000	33.0000	37.0000
A9	5.1000	4.7100	14.0000	43.0000	92.0000	30.0000	32.0000

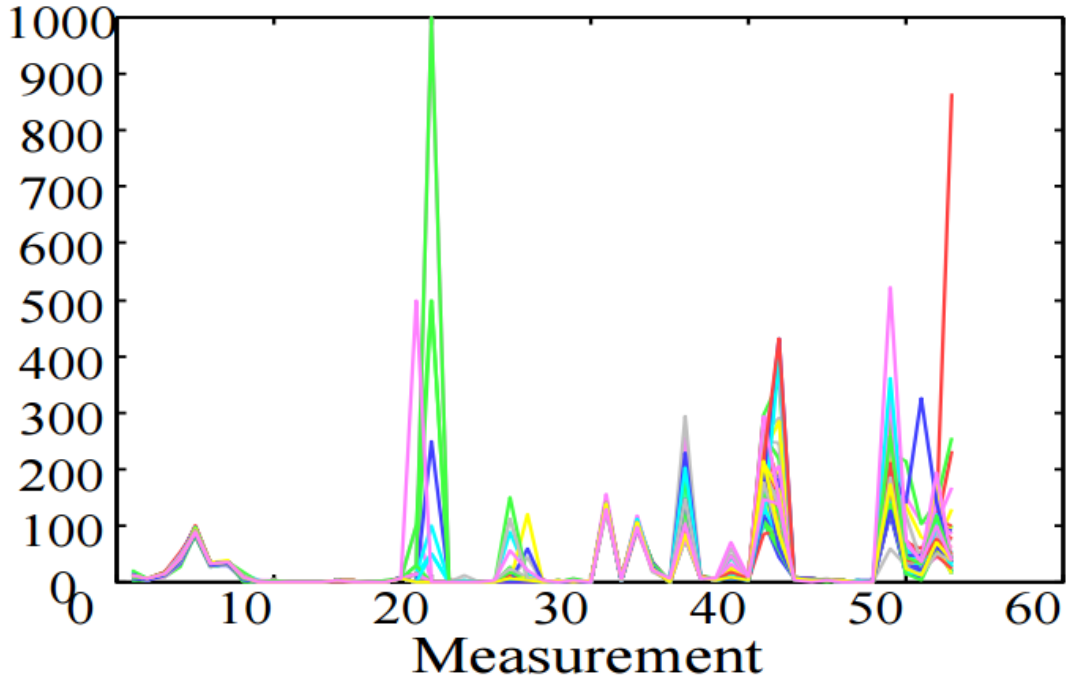
Instances

Features

Difficult to see the correlations between the features...

Data Visualization

- Spectral format (65 curves, one for each person)

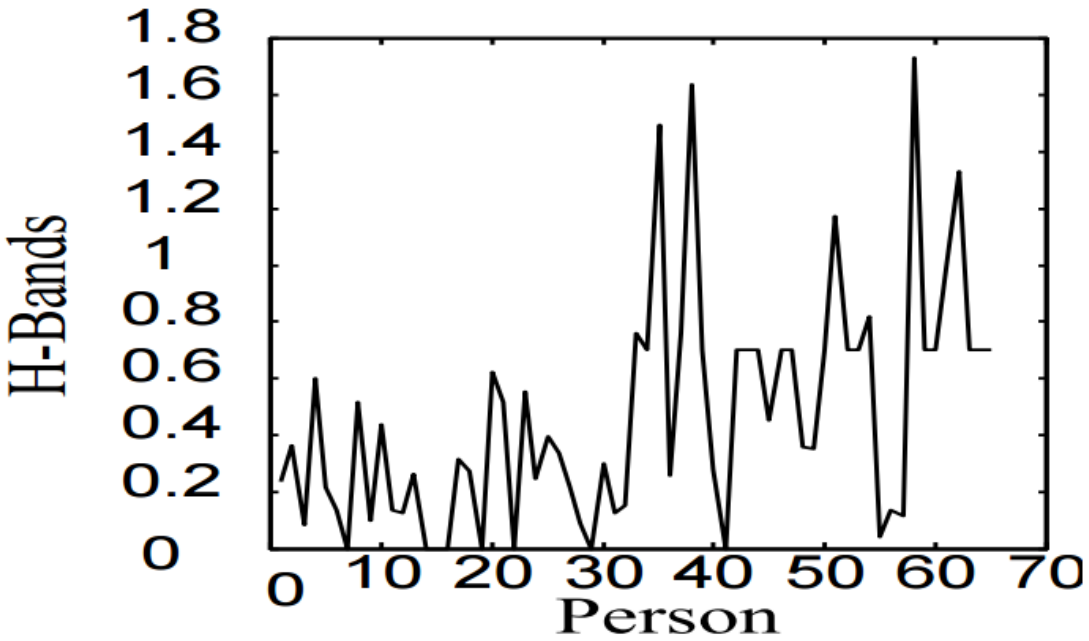


Difficult to compare the different patients...

slide by Barnabás Póczos and Aarti Singh

Data Visualization

- Spectral format (53 pictures, one for each feature)

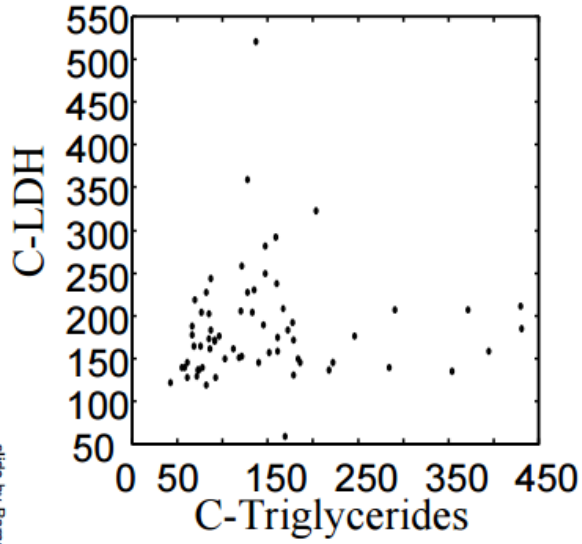


Difficult to see the correlations between the features...

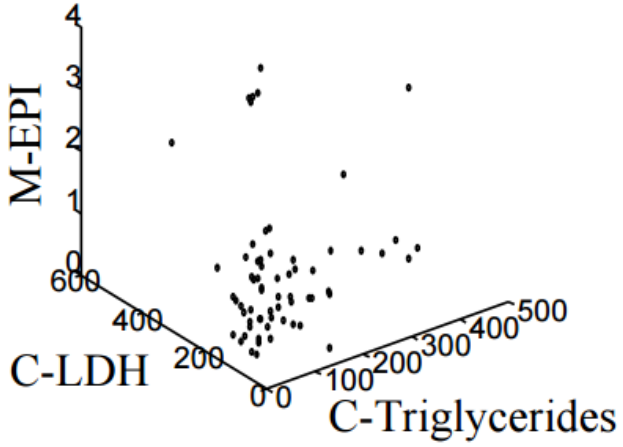
slide by Barnabás Póczos and Aarti Singh

Data Visualization

Bi-variate



Tri-variate



How can we visualize the other variables???

… difficult to see in 4 or higher dimensional spaces...

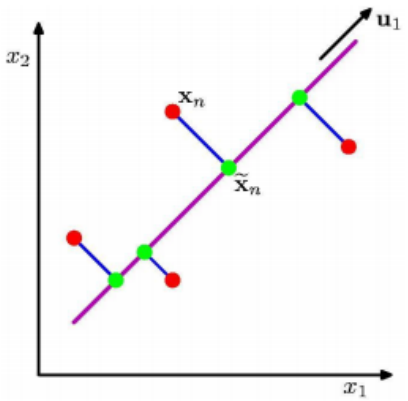
slide by Barnabás Póczos and Aarti Singh

Data Visualization

- Is there a representation better than the coordinate axes?
- Is it really necessary to show all the 53 dimensions?
  - ... what if there are strong correlations between the features?
- How could we find the smallest subspace of the 53-D space that keeps the most information about the original data?
- A solution: **Principal Component Analysis**

slide by Barnabás Póczos and Aarti Singh

Principal Component Analysis



PCA:

Orthogonal projection of the data onto a lower-dimension linear space that...

- maximizes variance of projected data (purple line)

- minimizes mean squared distance between
  - data point and
  - projections (sum of blue lines)

slide by Barnabás Póczos and Aarti Singh

Principal Component Analysis

Idea:

- Given data points in a d-dimensional space, project them into a lower dimensional space while preserving as much information as possible.
  - Find best planar approximation to 3D data
  - Find best 12-D approximation to 10^4-D data

- In particular, choose projection that minimizes **squared error** in reconstructing the original data.

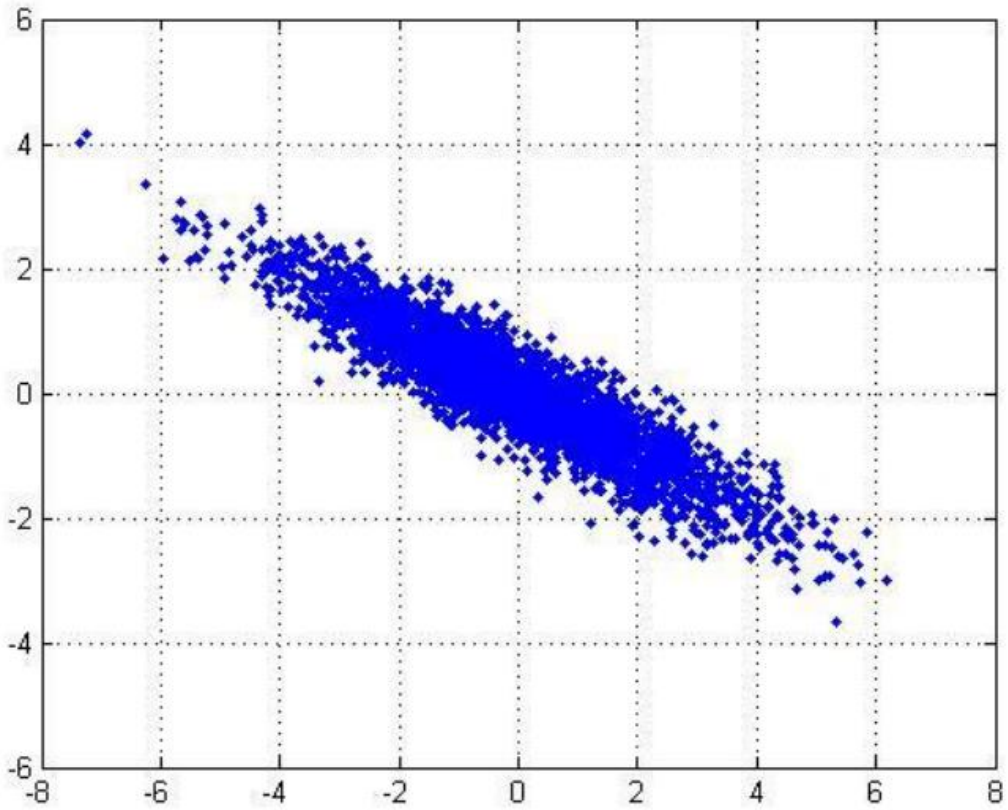
slide by Barnabás Póczos and Aarti Singh

Principal Component Analysis

- **PCA Vectors** originate from the center of mass.
- Principal component #1: points in the direction of the **largest variance**.
- Each subsequent principal component
  - is **orthogonal** to the previous ones, and
  - points in the directions of the **largest variance of the residual subspace**

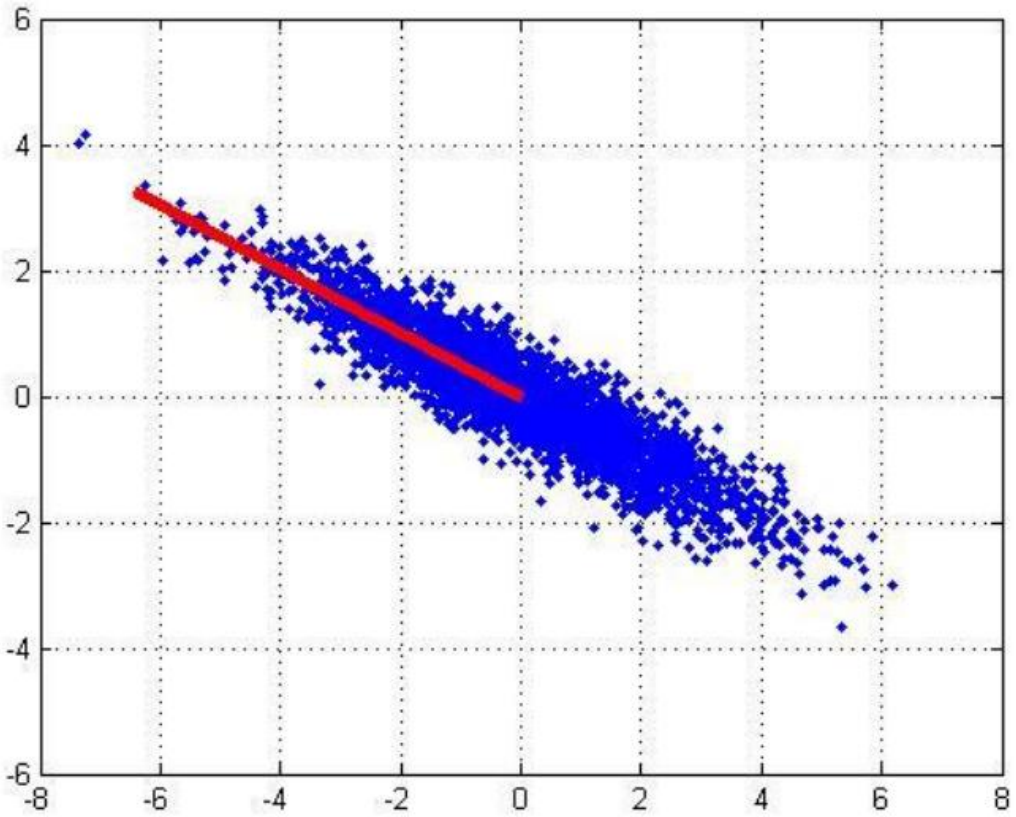
slide by Barnabás Póczos and Aarti Singh

2D Gaussian dataset



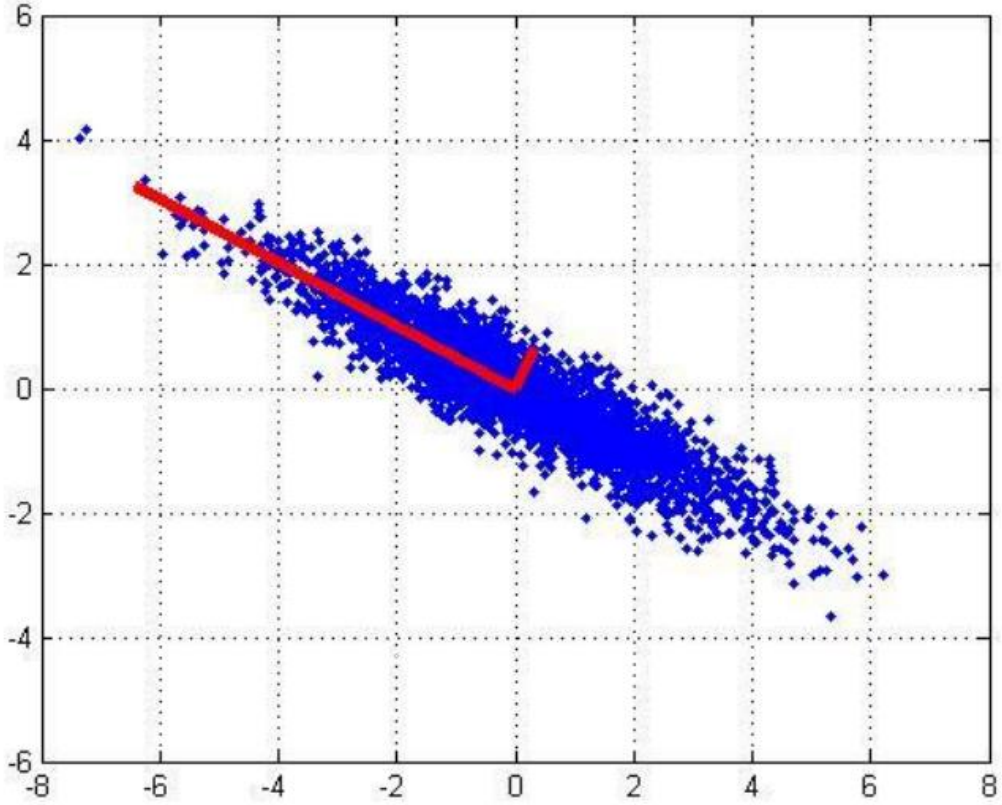
slide by Barnabás Póczos and Aarti Singh

1st PCA axis



slide by Barnabás Póczos and Aarti Singh

2nd PCA axis



slide by Barnabás Póczos and Aarti Singh

Face Recognition

- Want to identify specific person, based on facial image
- Robust to glasses, lighting,…

  ⇒ Can't just use the given 256 x 256 pixels

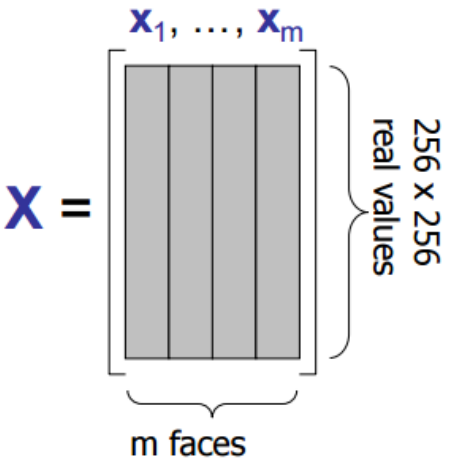
Applying PCA: Eigenfaces

Method A: Build a PCA subspace for *each person* and check which subspace can reconstruct the test image the best

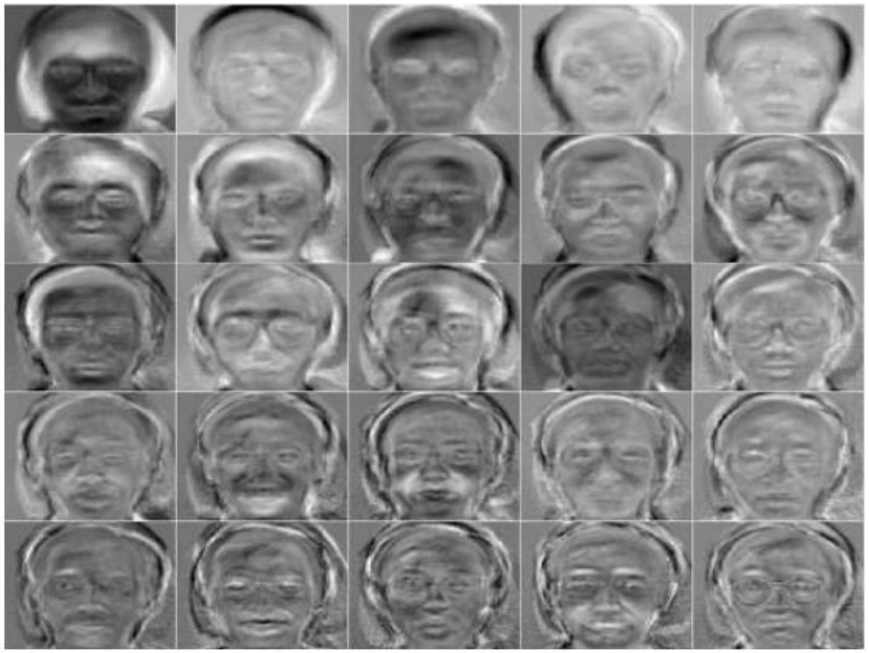
Method B: Build one PCA database for the *whole dataset* and then classify based on the weights.

Applying PCA: Eigenfaces

- Example data set: Images of faces
  - Eigenface approach
    [Turk & Pentland], [Sirovich & Kirby]
- Each face **x** is …
  - 256 × 256 values (luminance at location)
  - **x** in $\Re^{256\times256}$ (view as 64K dim vector)
- Form $\mathbf{X} = [\, \mathbf{x}_1, \ldots, \mathbf{x}_m \,]$ **centered** data mtx
- Compute $\Sigma = \mathbf{X}\mathbf{X}^T$
- Problem: Σ is 64K × 64K … HUGE!!!

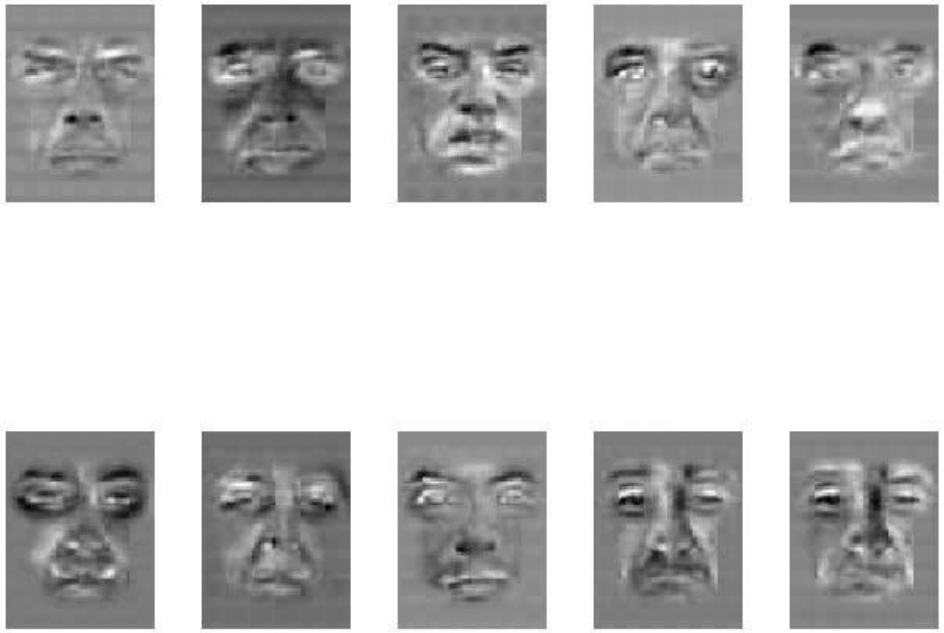


Principle Components (Method B)

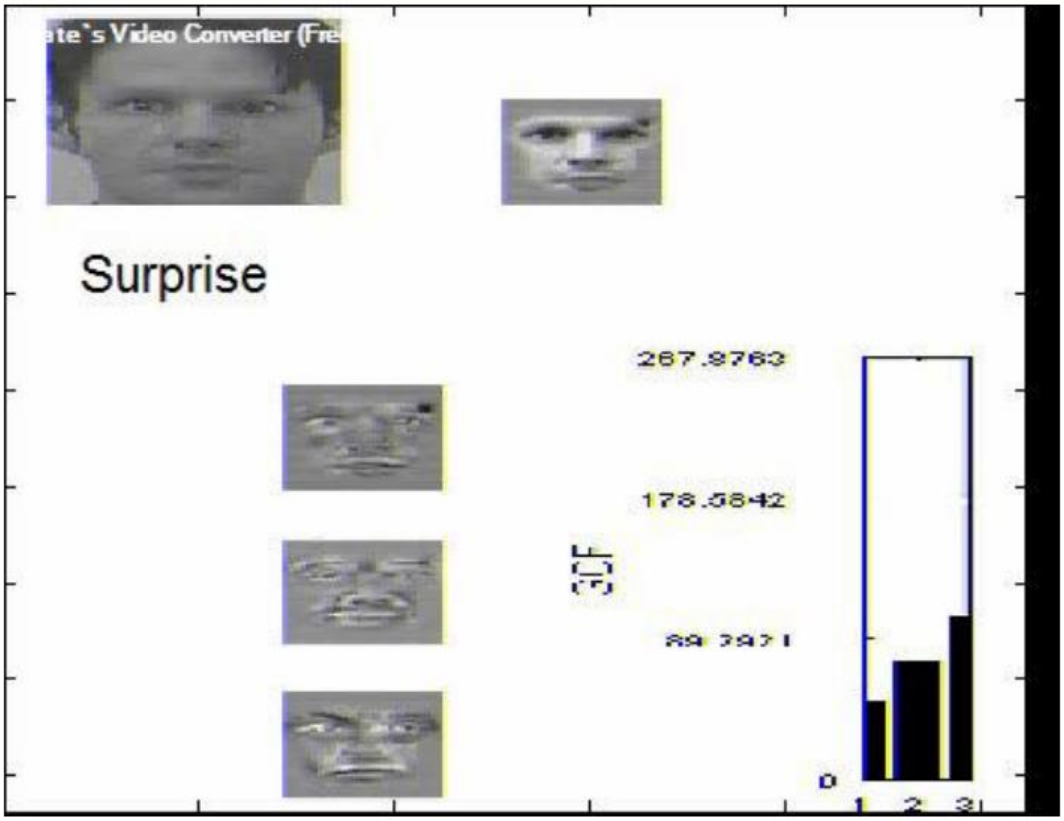


Happiness subspace (method A)

Disgust subspace (method A)



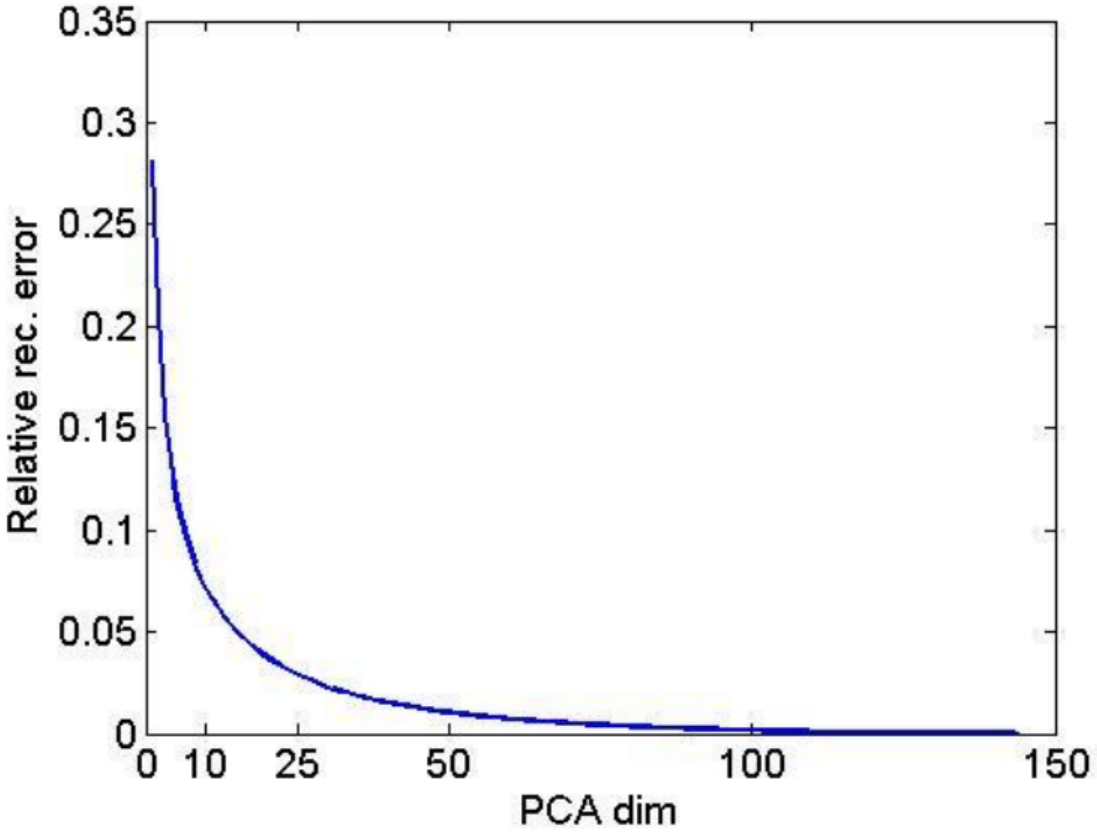
Facial Expression Recognition Movies



Original Image

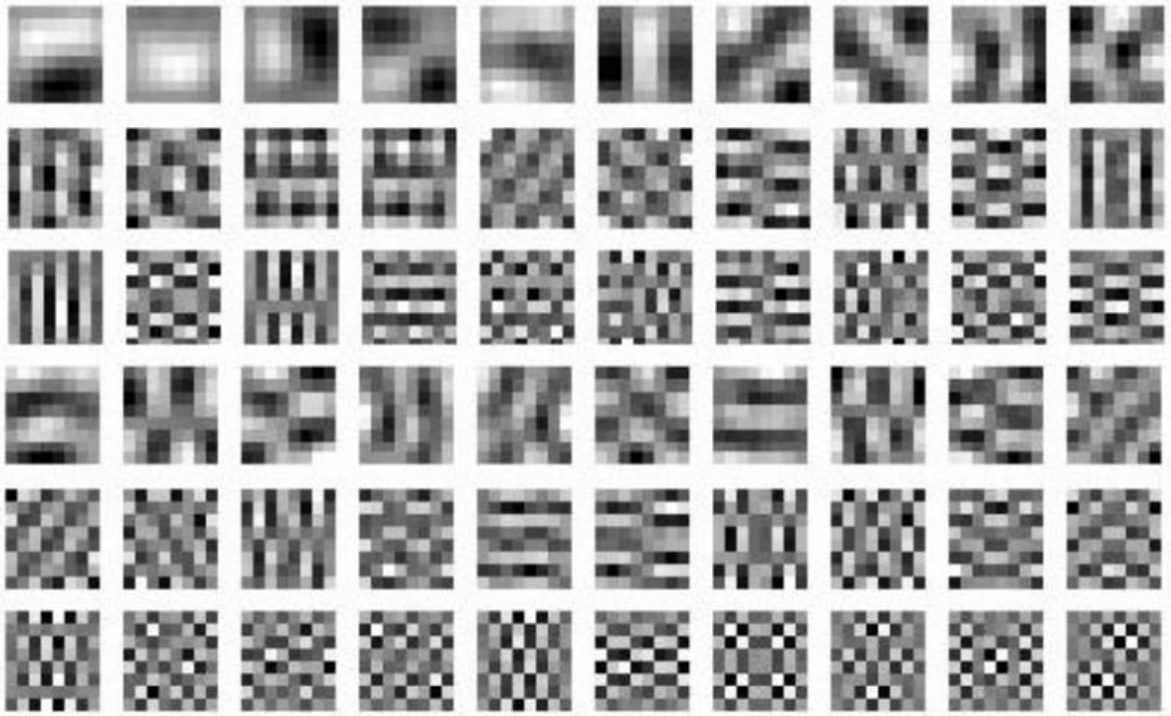
- Divide the original 372x492 image into patches:
  - Each patch is an instance that contains 12x12 pixels on a grid
- Consider each as a 144-D vector

L_2 error and PCA dim



PCA compression: 144D $\Rightarrow$ 60D

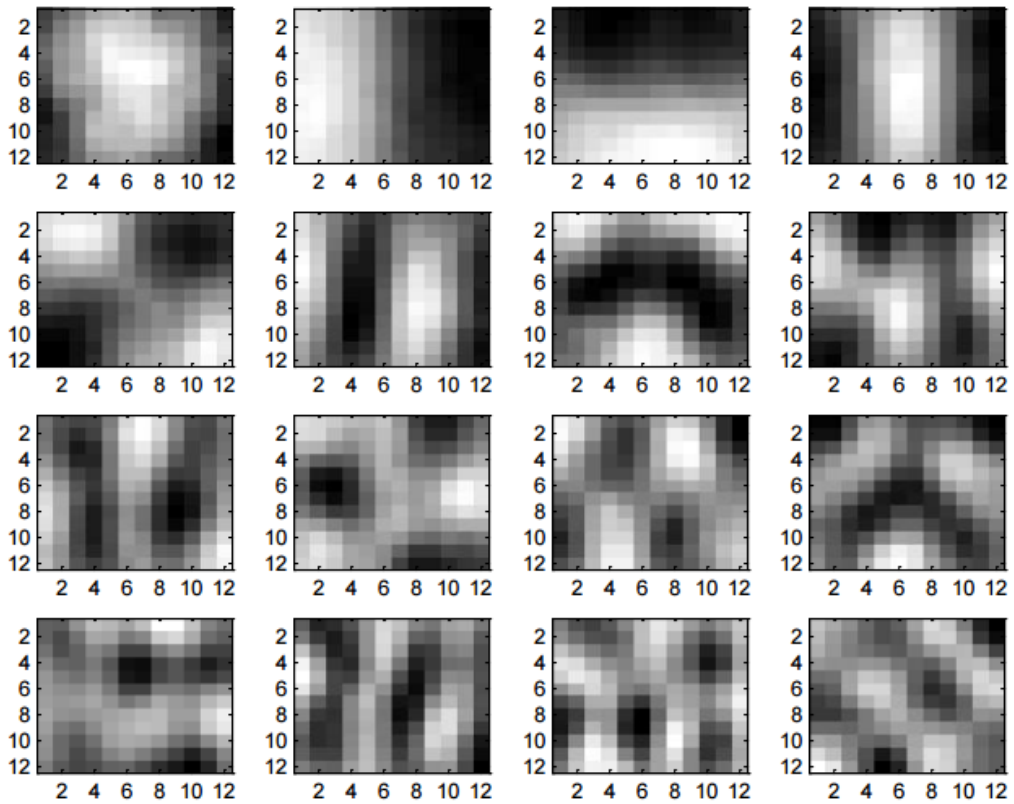
60 most important eigenvectors



Looks like the discrete cosine bases of JPG!...

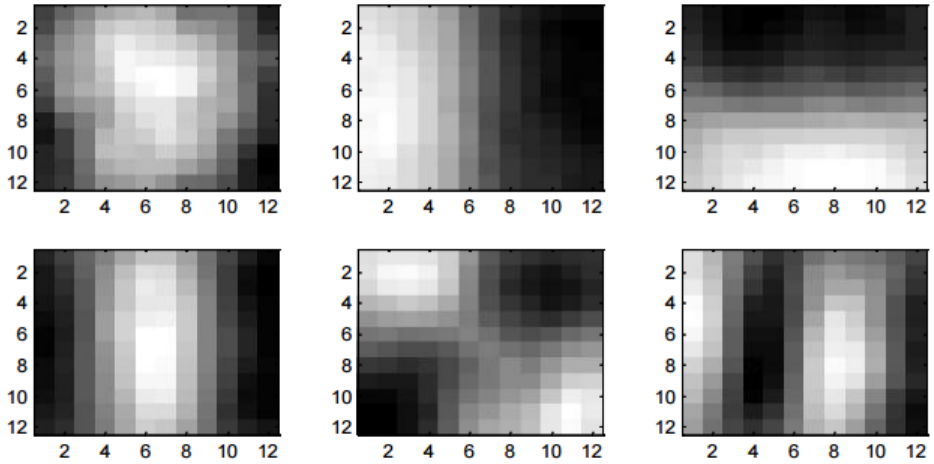
PCA compression: 144D $\Rightarrow$ 16D

16 most important eigenvectors



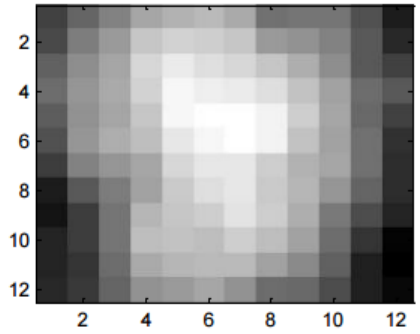
PCA compression: 144D $\Rightarrow$ 6D

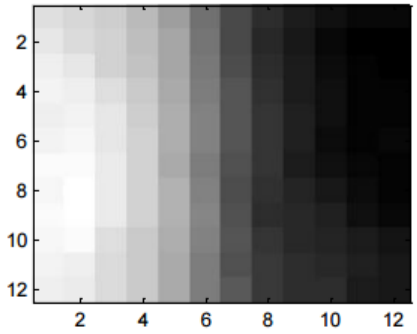
6 most important eigenvectors

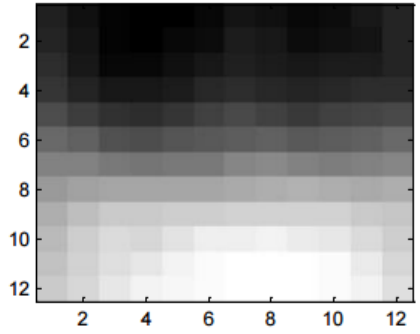


PCA compression: 144D $\Rightarrow$ 3D

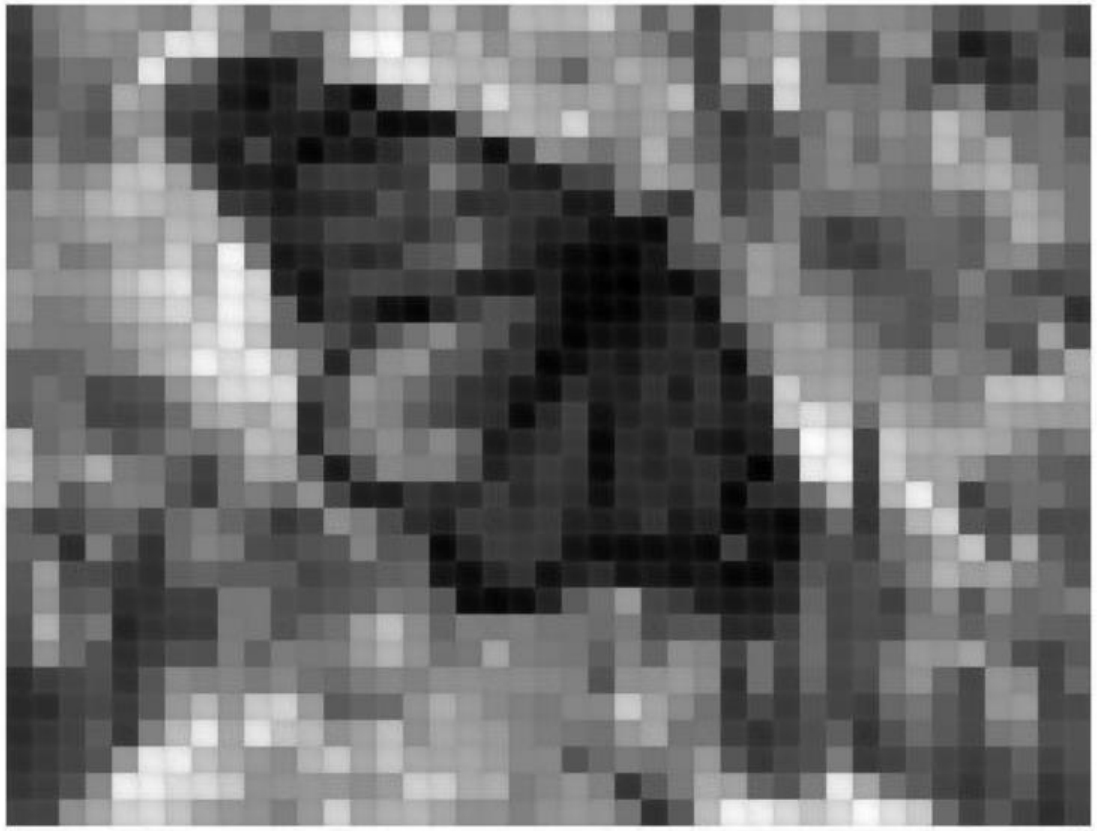
3 most important eigenvectors



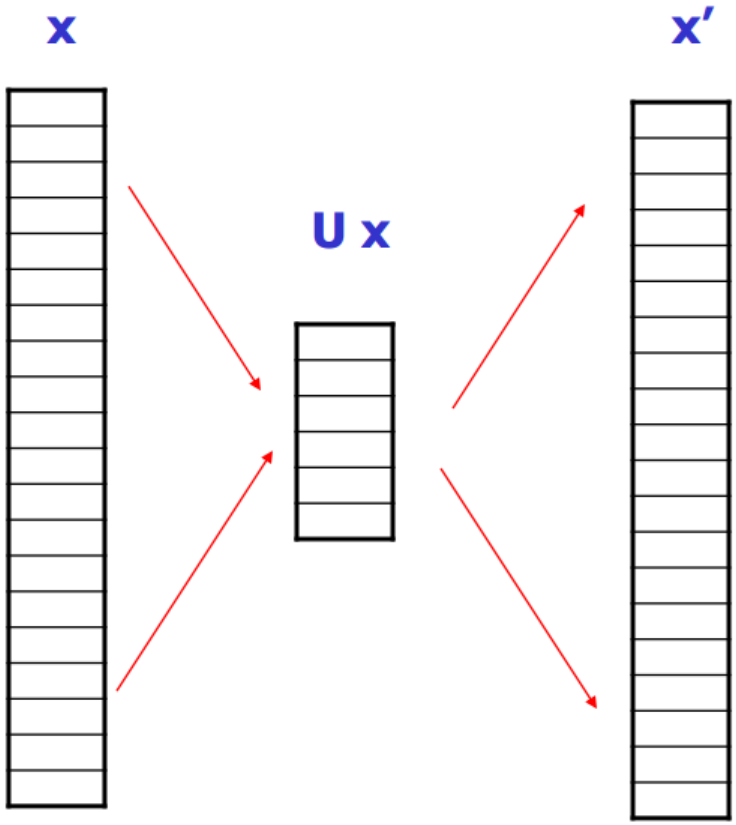




PCA compression: 144D $\Rightarrow$ 1D



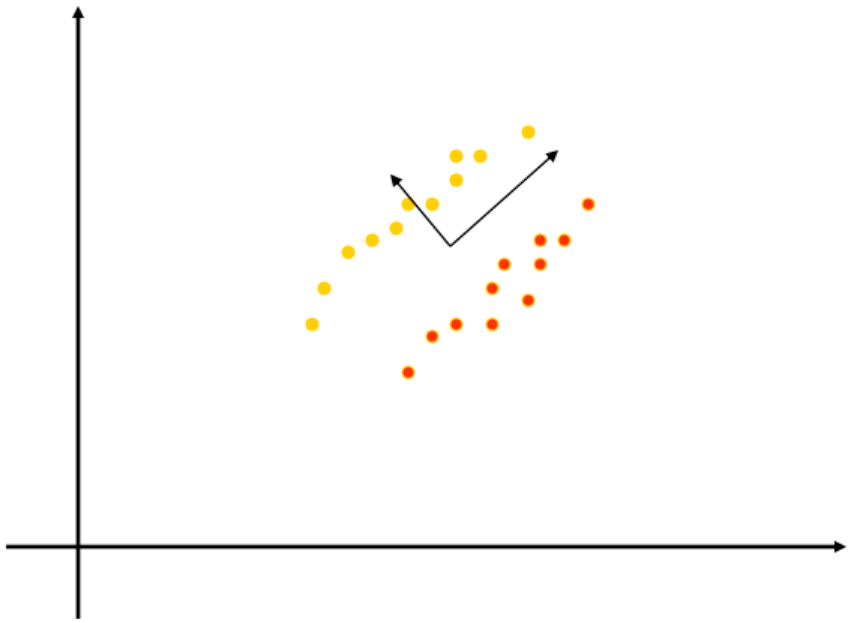
Noise Filtering



Noisy image

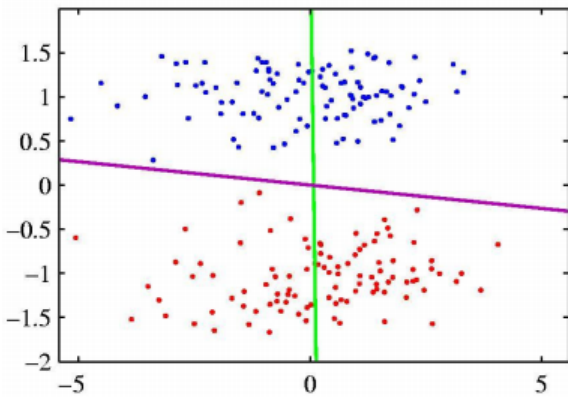
Denoised image using 15 PCA components

Problematic Data Set for PCA



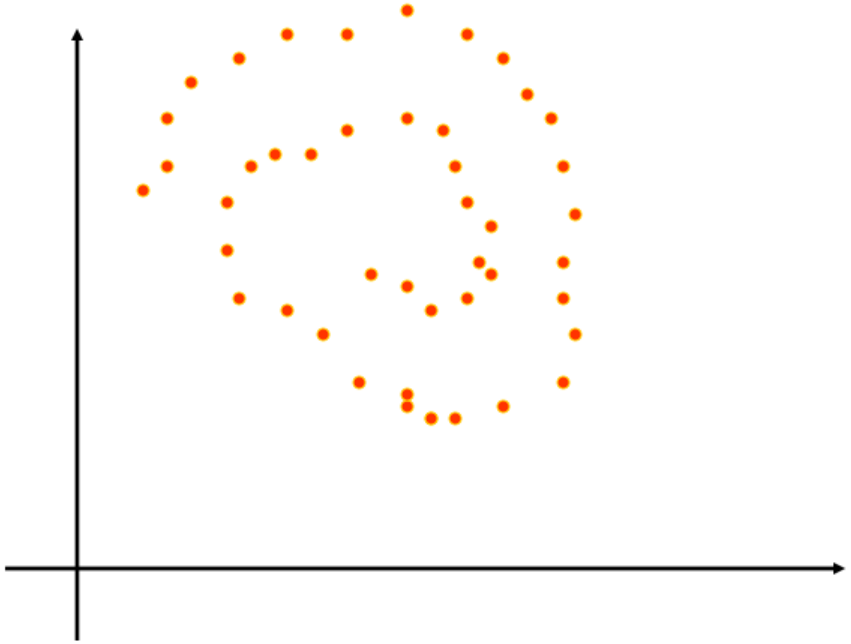
PCA doesn't know labels!

PCA vs Fisher Linear Discriminant



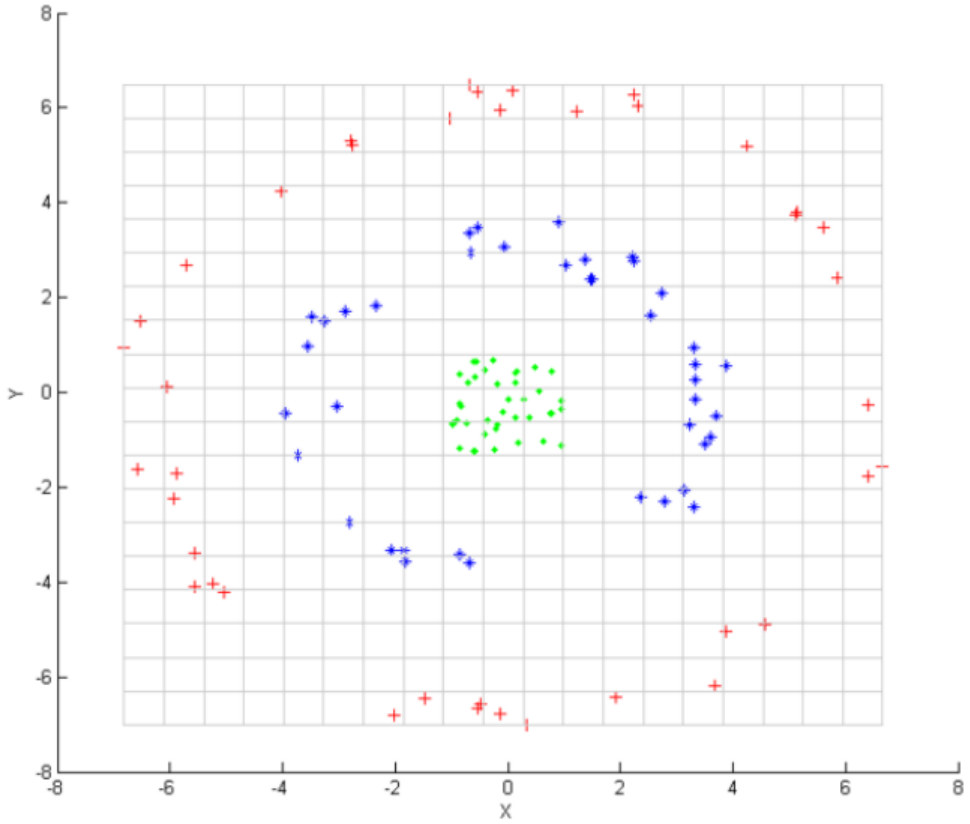
- PCA maximizes variance, *independent of class*
  ⇒ magenta
- FLD attempts to separate classes
  ⇒ green line

Problematic Data Set for PCA



PCA cannot capture NON-LINEAR structure!

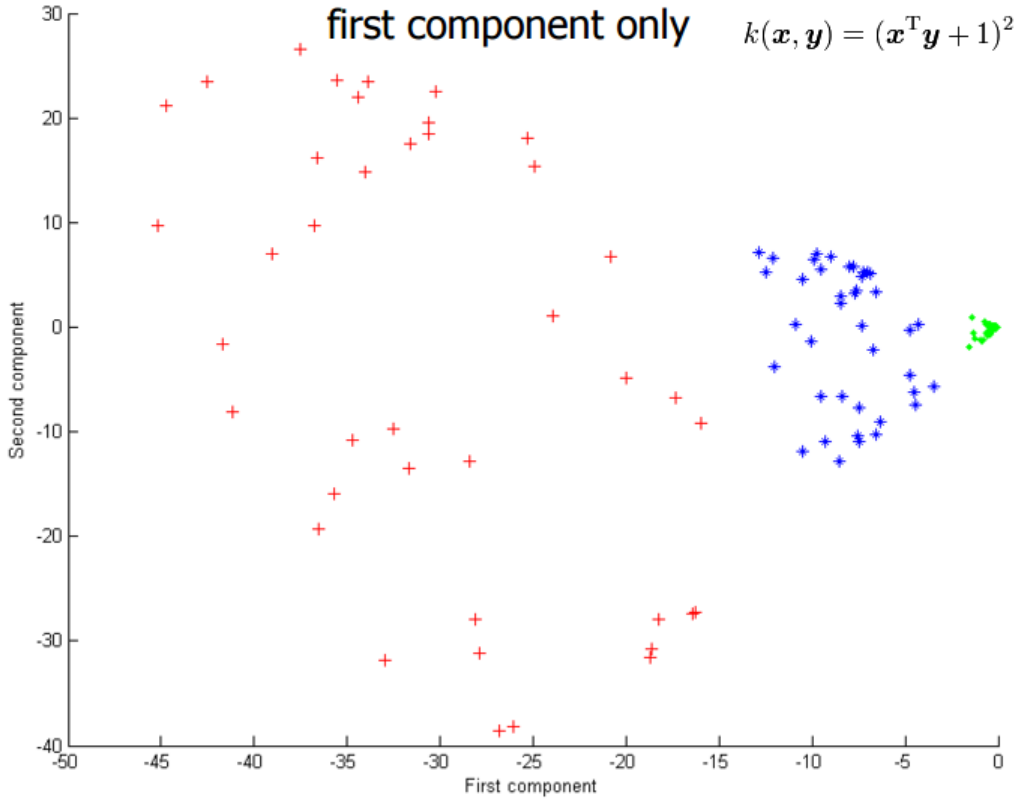
Input points before kernel PCA



http://en.wikipedia.org/wiki/Kernel_principal_component_analysis

Output after kernel PCA

The three groups are distinguishable using the first component only $k(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{x}^{\mathrm{T}}\boldsymbol{y} + 1)^2$



What you need to know

- Dimensionality Reduction
  - why and when its important
    - visualization
    - compression
    - faster learning
- Principle Component Analysis
  - KL Transform view
    - Notes have reconstruction error and max variance views too
  - Relationship to covariance matrix and eigenvectors
  - using SVD for PCA