Density Estimation and Clustering

Density Estimation

Properties of probability distributions

- Always greater than zero
- Integrates to 1

Common approaches to density estimation

- Parametric density estimation (distribution fitting)
- Histograms
- Kernel density estimation
- Gaussian mixture models

Parametric Density Estimation

If we have knowledge of a possible parametric form, we can estimate the parameters of the model

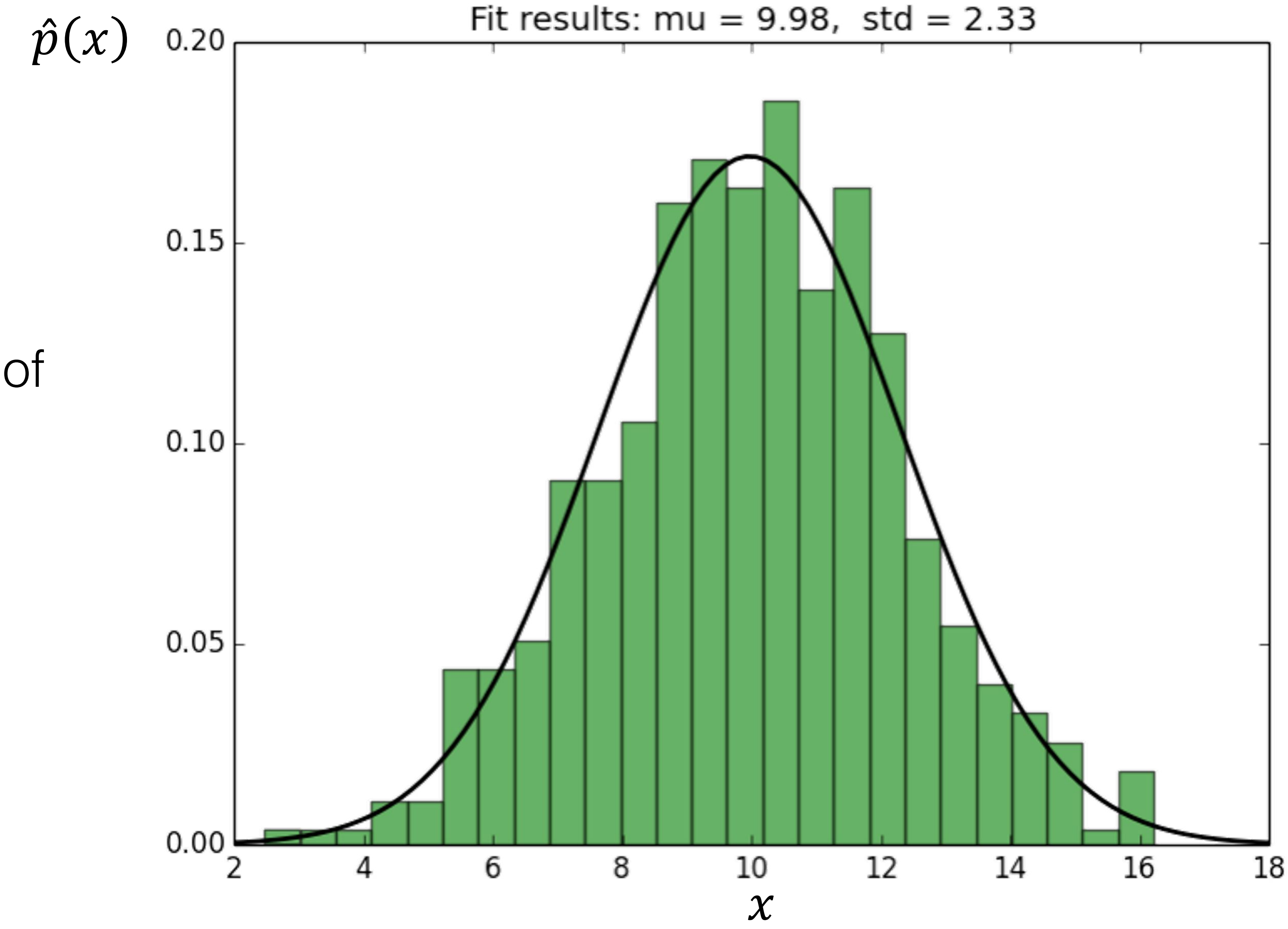
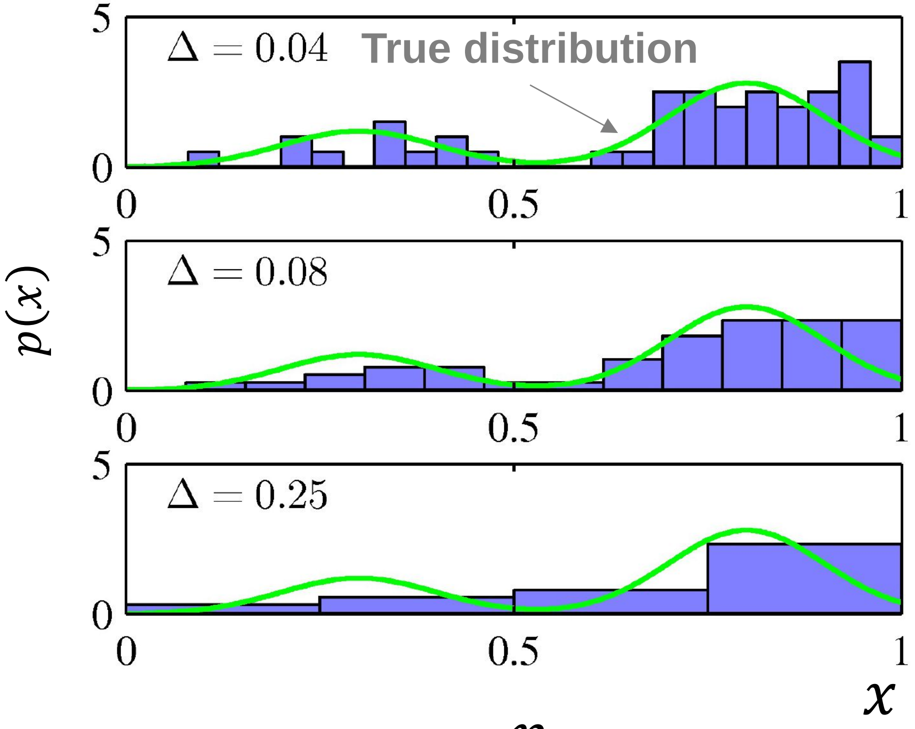


Image from: https://stackoverflow.com/questions/20011122/fitting-a-normal-distribution-to-1d-data

Histogram Density Estimation

Histogram



$$p(x) = \frac{n_i}{N\Delta_i}$$

n_i = # observations of x falling in bin i
N = total # observations
Δ_i = width of bin i

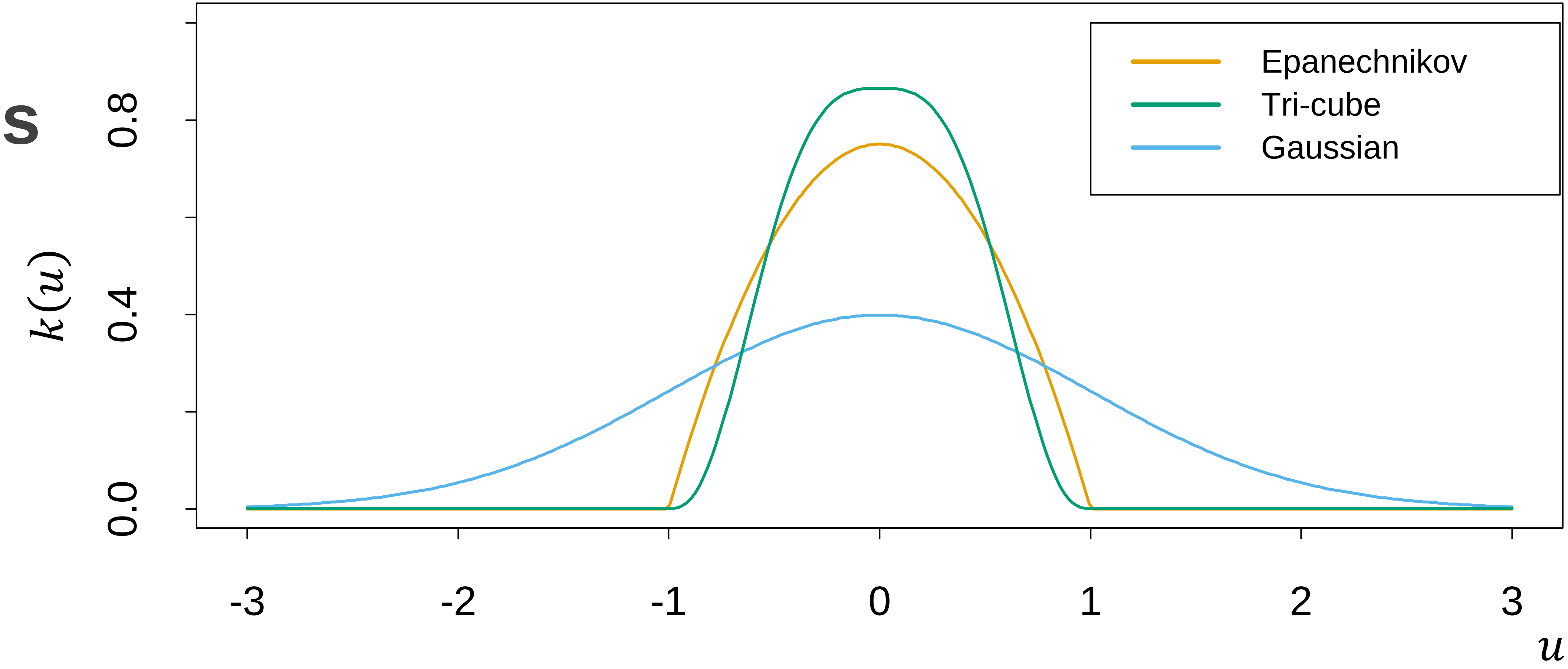
Highly dependent on the choice of bin width, Δ_i

Has discontinuities at the bin edges

Local neighborhoods do appear to be helpful

Bishop, Pattern Recognition and Machine Learning, 2006

Kernel Functions (window kernels)



Satisfy two properties:

$$k(u) \geq 0$$

$$\int k(u) du = 1$$

Epanechnikov

$$k(u) = \frac{3}{4}(1 - u^2)$$

$$|u| \leq 1$$

Tri-cube

$$k(u) = \frac{70}{81}(1 - |u^3|)^3$$

$$|u| \leq 1$$

Gaussian

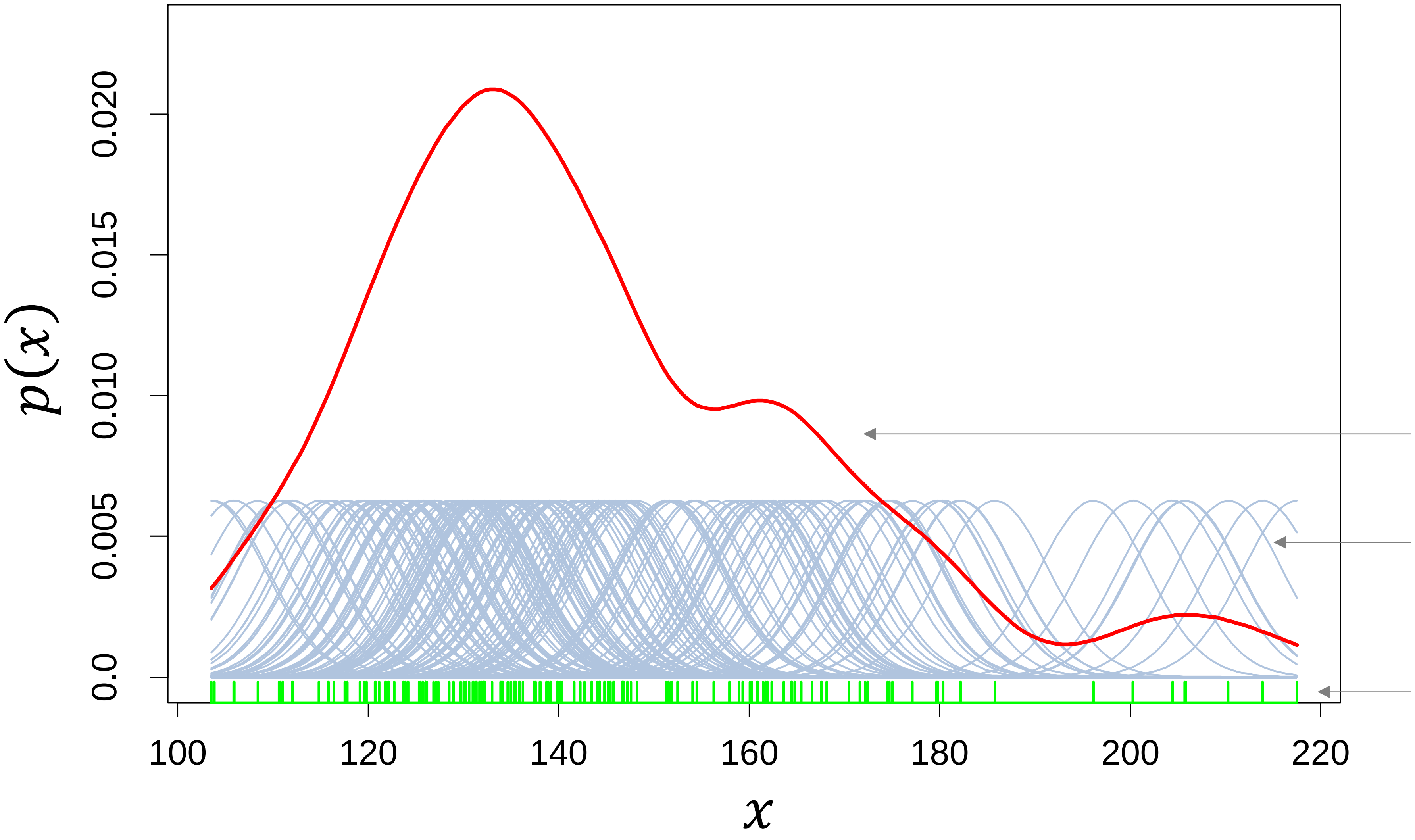
$$k(u) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}u^2}$$

$$-\infty < u < \infty$$

Hastie, Tibshirani, and Friedman, The Elements of Statistical learning, 2001

Kernel Density Estimation

a. k. a. Parzen Window Density Estimation



Density estimate

Kernel functions
(Gaussian kernel shown)

Observations (x-values)

1. Center a kernel function at each observation (x-value)

2. Estimate $p(x)$ by averaging across all the kernels at x

Hastie, Tibshirani, and Friedman, The Elements of Statistical learning, 2001

Kernel Density Estimation

Center the kernel function at each x-value in the dataset:

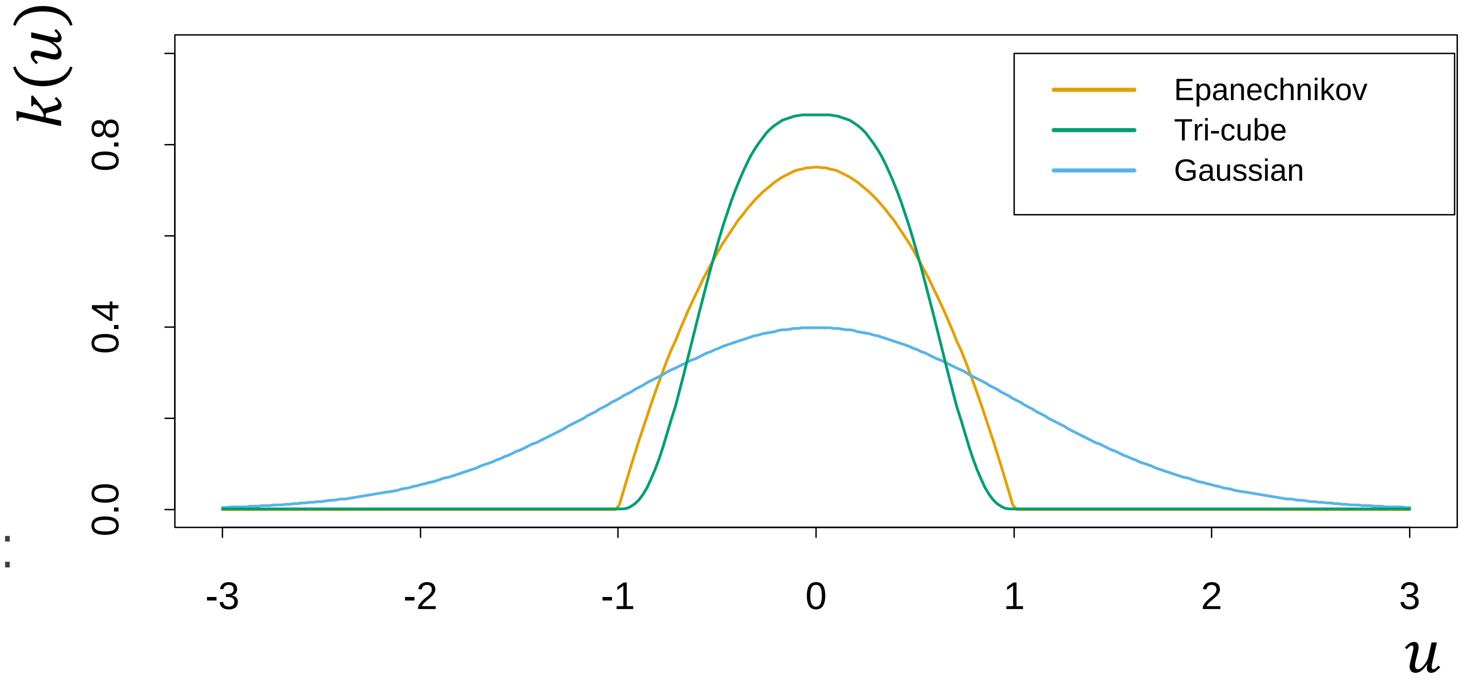
$$k(x - x_n) \qquad n = 1,2,\dots,N$$

Average over all of the kernel functions to get the density estimate:

$$p(x) = \frac{1}{N}\sum_{n=1}^{N} k(x - x_n)$$

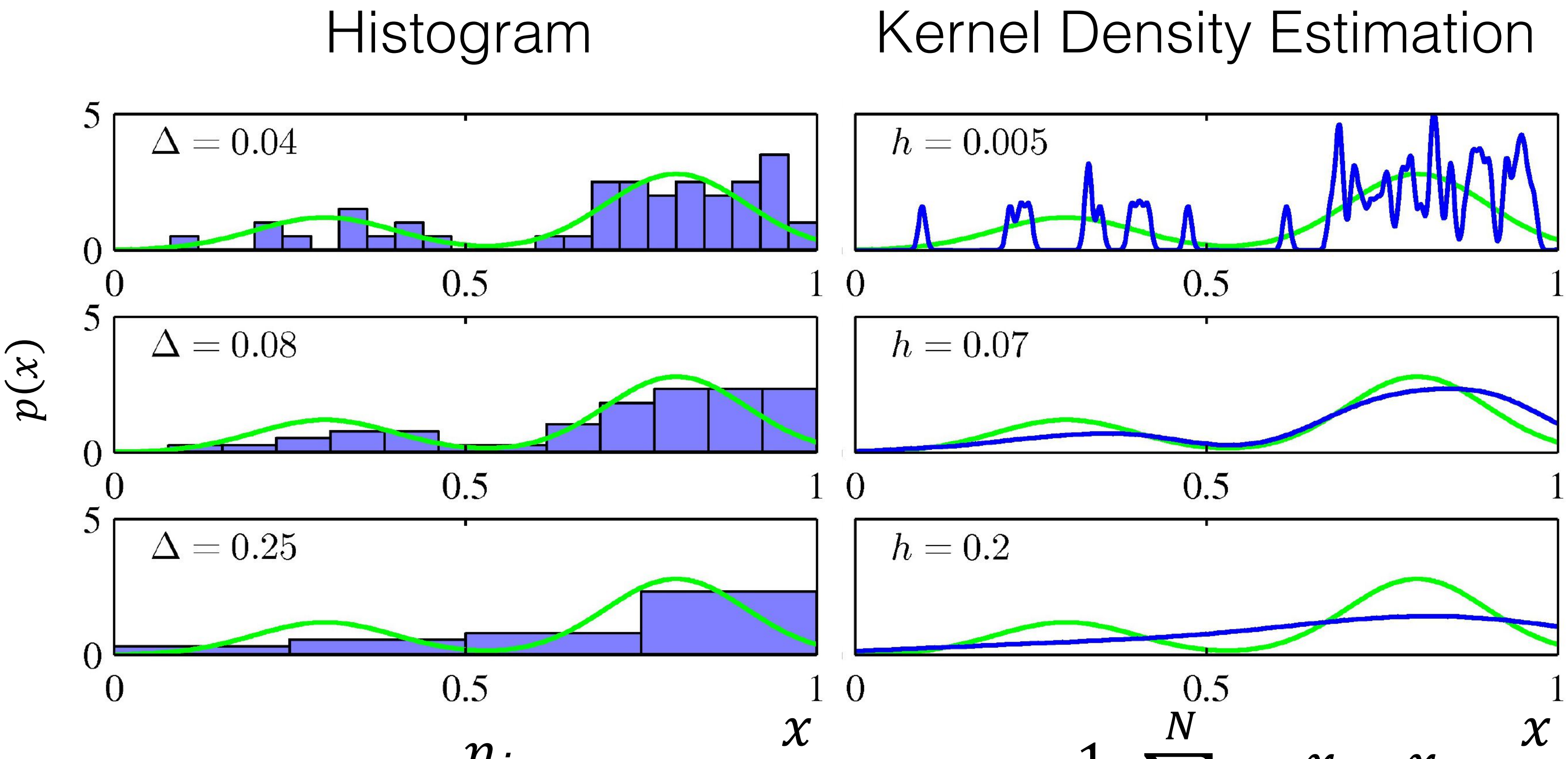
Note: we can scale the width of the kernel function with a scale factor, h:

$$k\left(\frac{x - x_n}{h}\right)$$



For kernel functions with **finite domains**, this means that each observation, x, will only affect the density estimate in a **neighborhood** close to the center of the kernel

Kernel Density Estimation



Requires tuning h, the kernel width parameter

Computational cost of evaluating this density grows linearly with the size of the data

$$p(x) = \frac{n_i}{N\Delta_i}$$

$$p(x) = \frac{1}{Nh}\sum_{n=1}^{N} k\left(\frac{x - x_n}{h}\right)$$

n_i = # observations of x falling in bin i
N = total # observations
Δ_i = width of bin i

x_n = The nth observation of x
k = kernel function
h = width of the kernel

Bishop, Pattern Recognition and Machine Learning, 2006

Density estimation uses

Describing the distribution of data and its characteristics

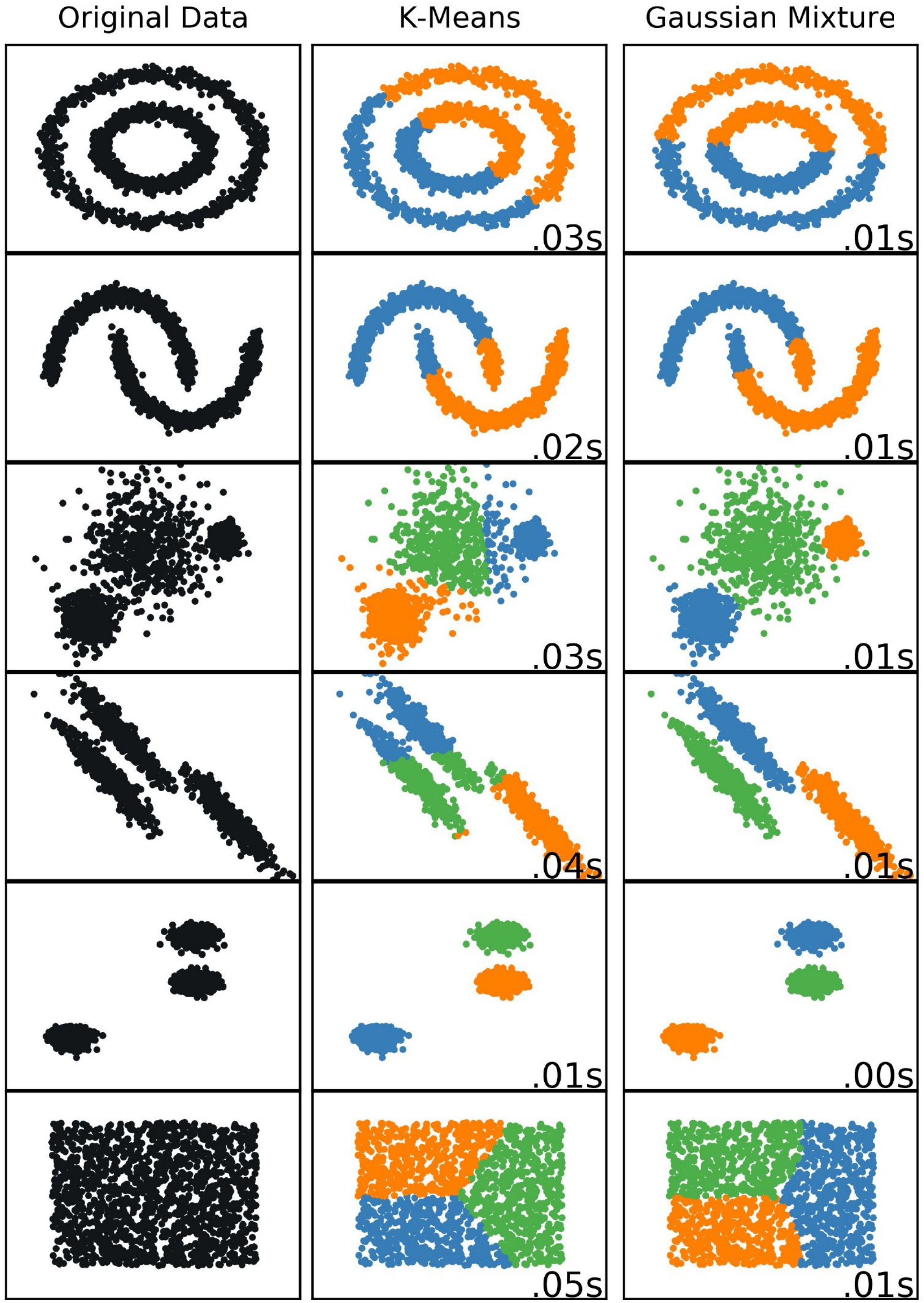
Can be used for anomaly/outlier detection

- If a new sample has a low “probability” given the distribution of the data, then it may be anomalous

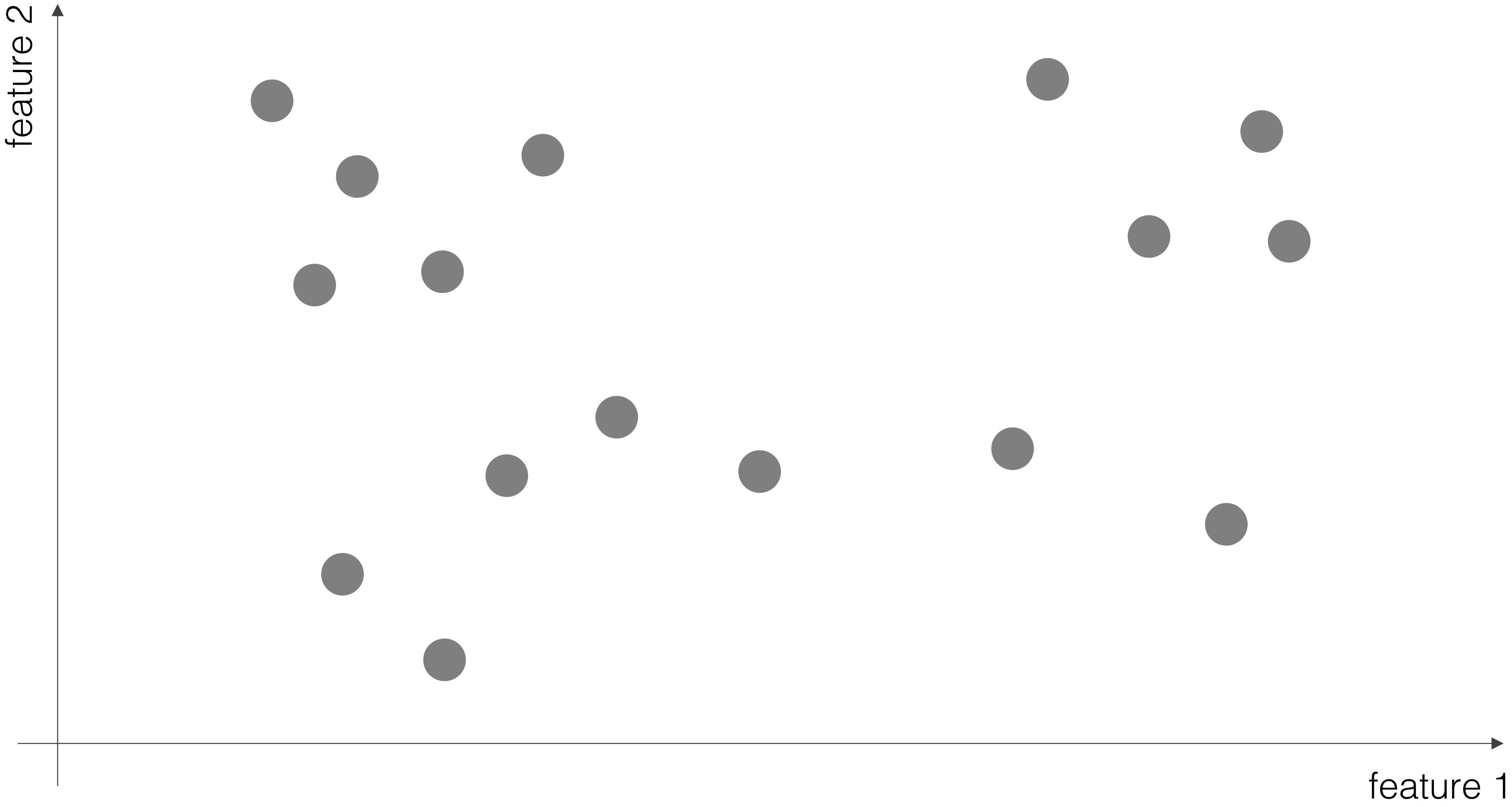
Clustering

K-Means + Gaussian Mixture Models (GMMS)

Clustering and Density Estimation (GMMS)

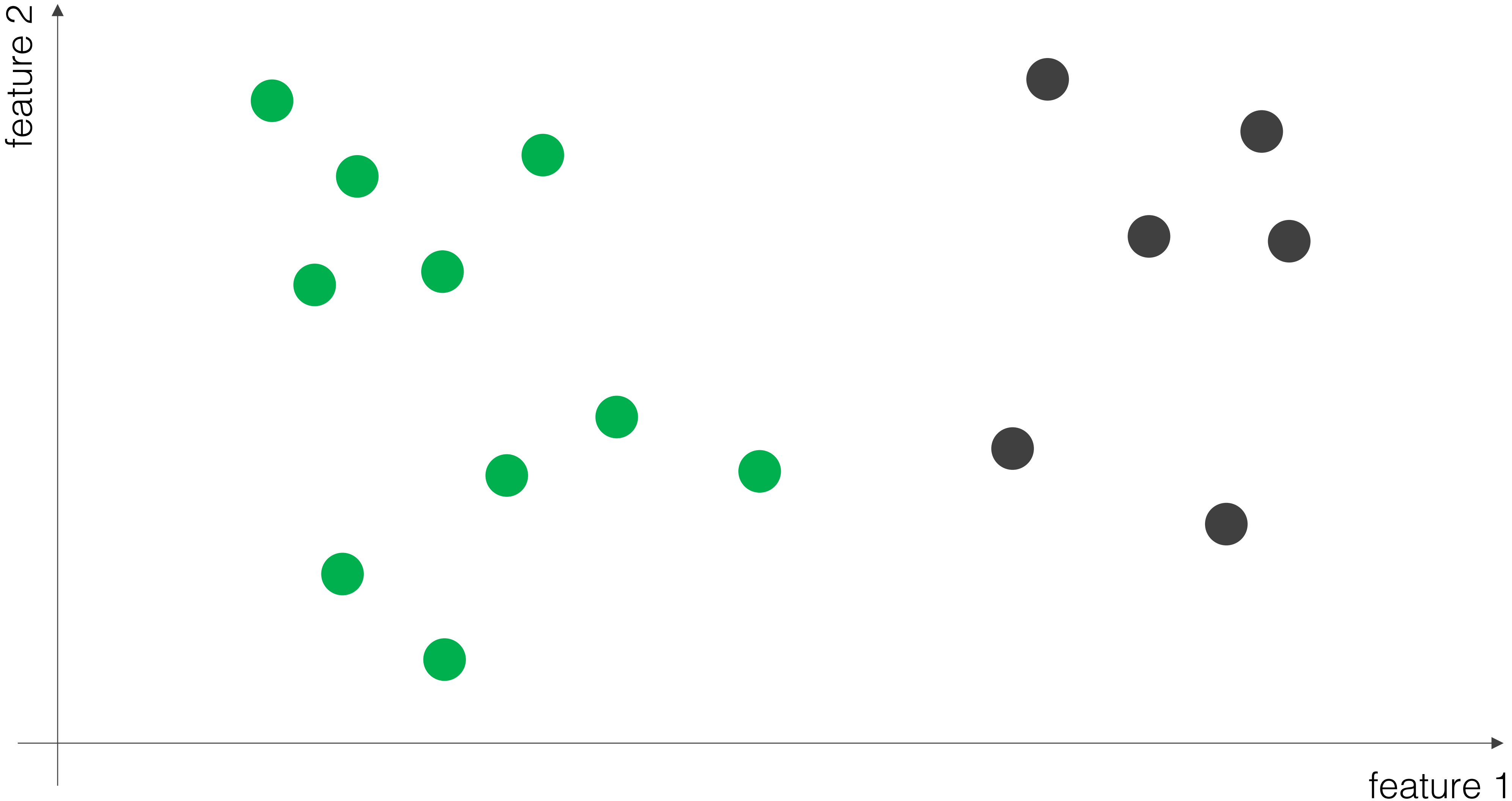


Clustering



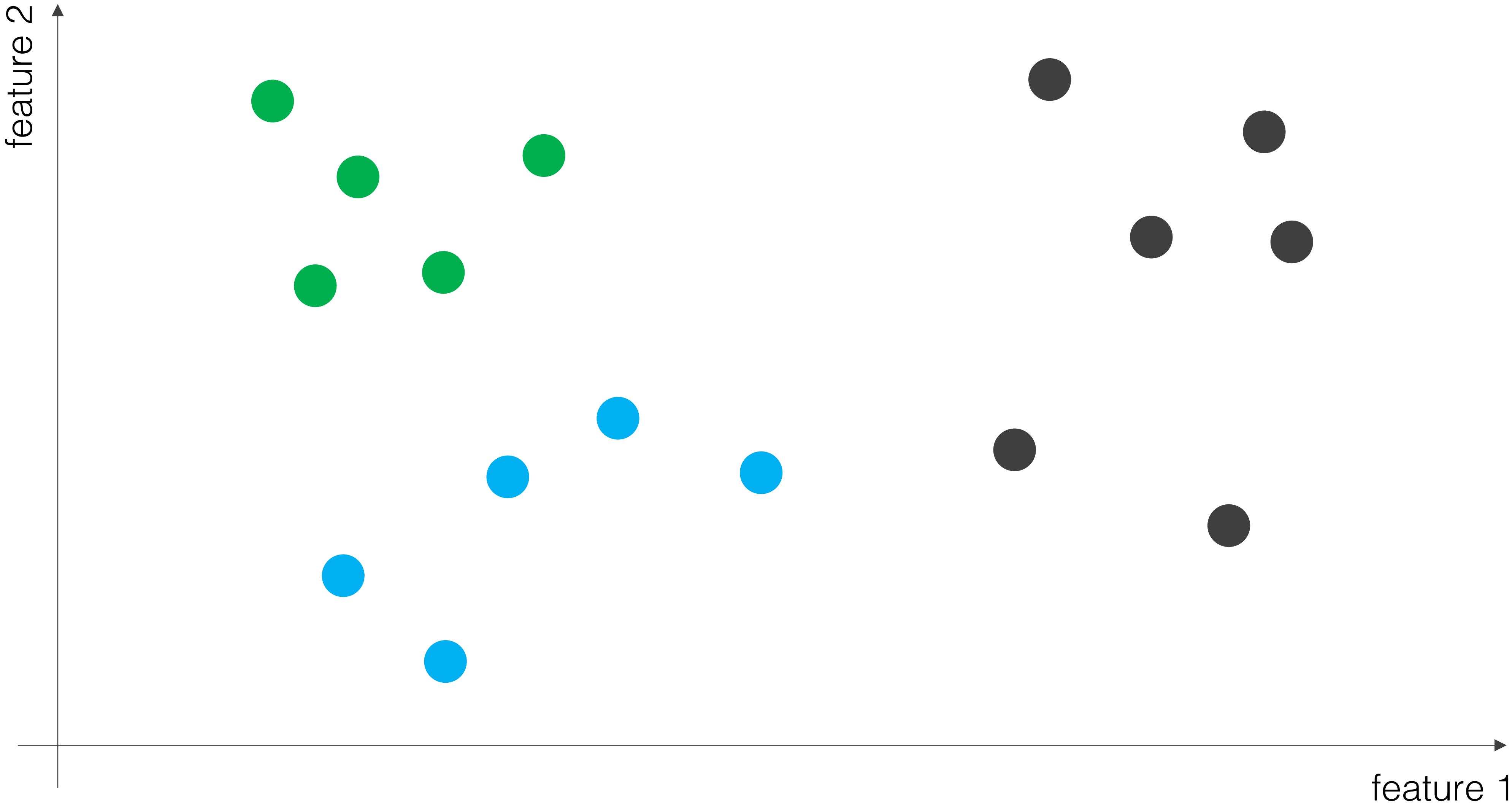
Clustering

Looks like 2 clusters…



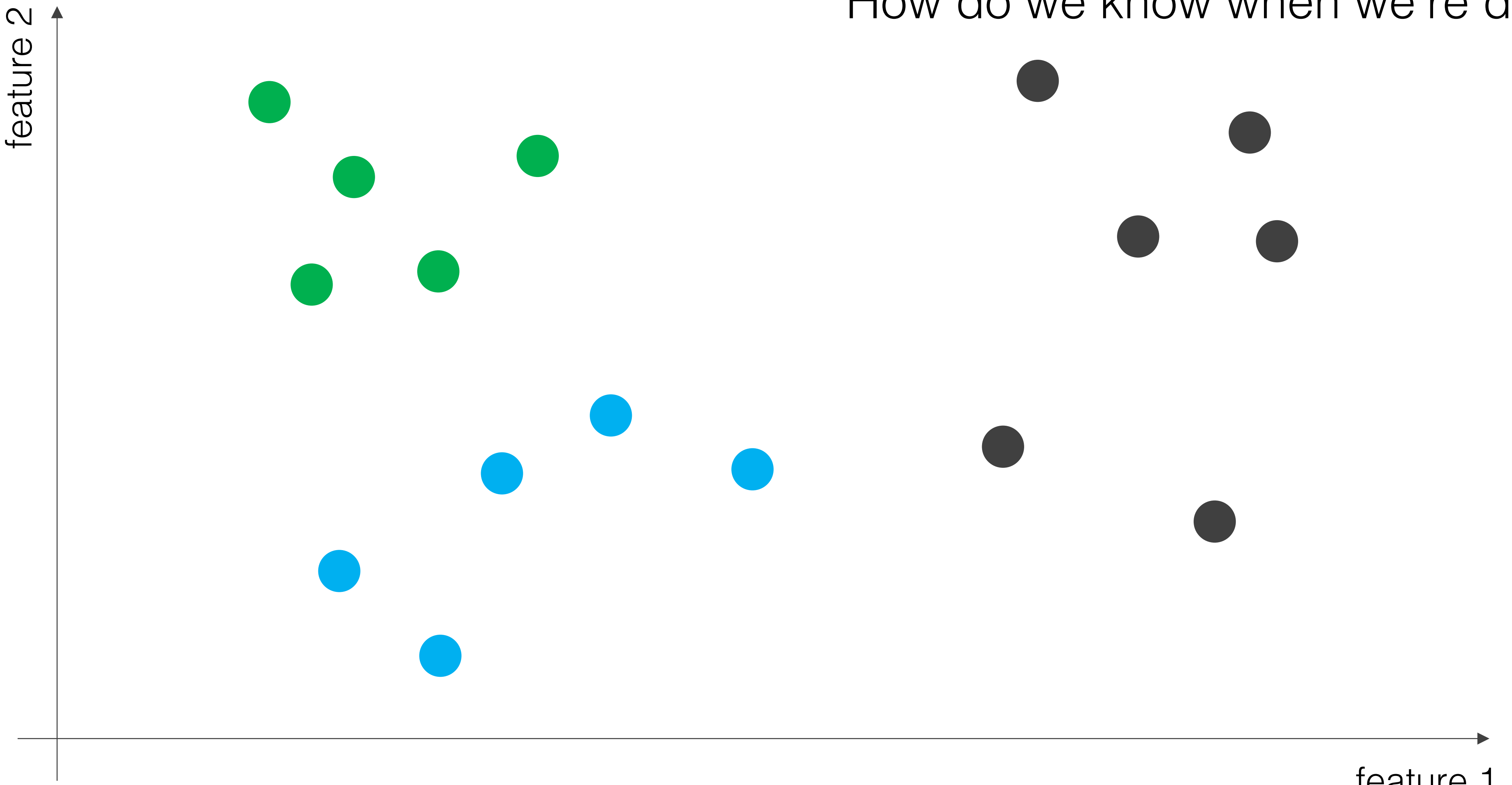
Clustering

… or maybe 3?



Clustering

How do we define “similarity”?
How do we choose the number of clusters?
How do we know when we’re doing well?



Applications

Differentiating tissue types in PET scans

Customer segmentation for market research

Social network analysis and identifying communities

Crime tracking to identify hot spots for certain types of crimes

Types of clustering algorithms

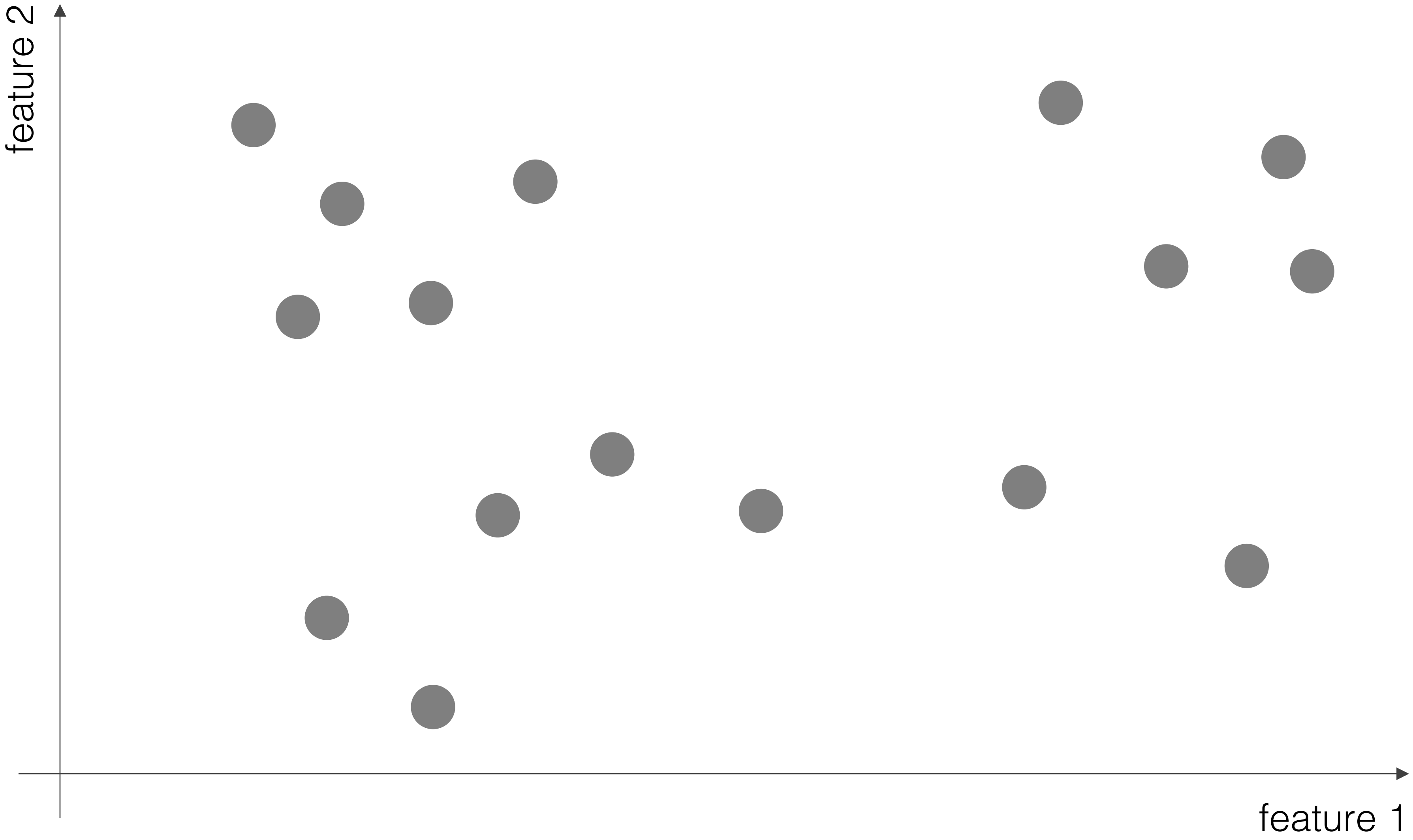
Methods

Centroid-based clustering (e.g. K-Means)
Distribution-based clustering (e.g. Gaussian mixture model)
Density-based clustering (e.g. DBSCAN)
Hierarchical clustering (e.g. agglomerative clustering)
a.k.a. connectivity-based clustering

Cluster assignment

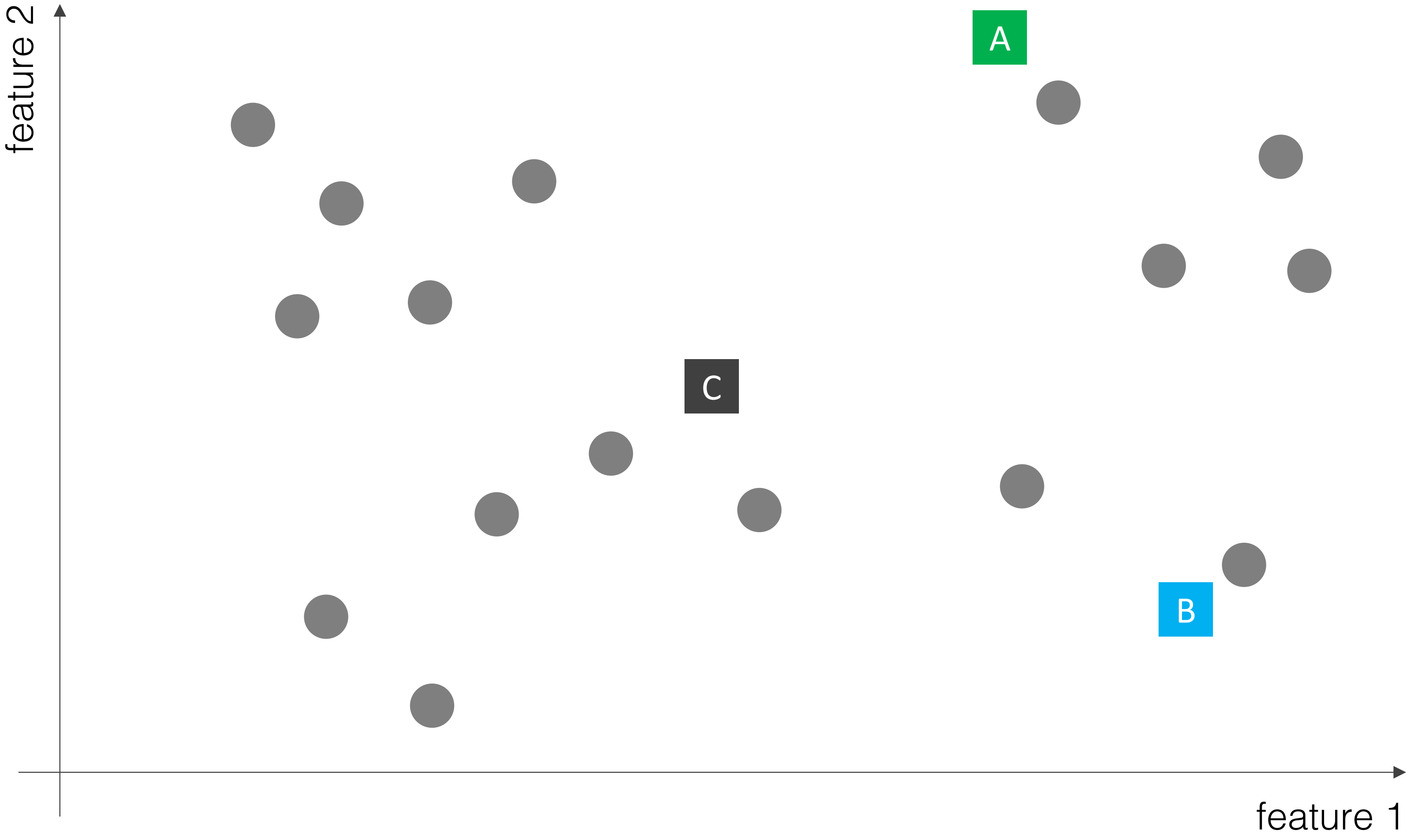
Hard clustering
Soft clustering (a.k.a. fuzzy clustering)

K-means clustering



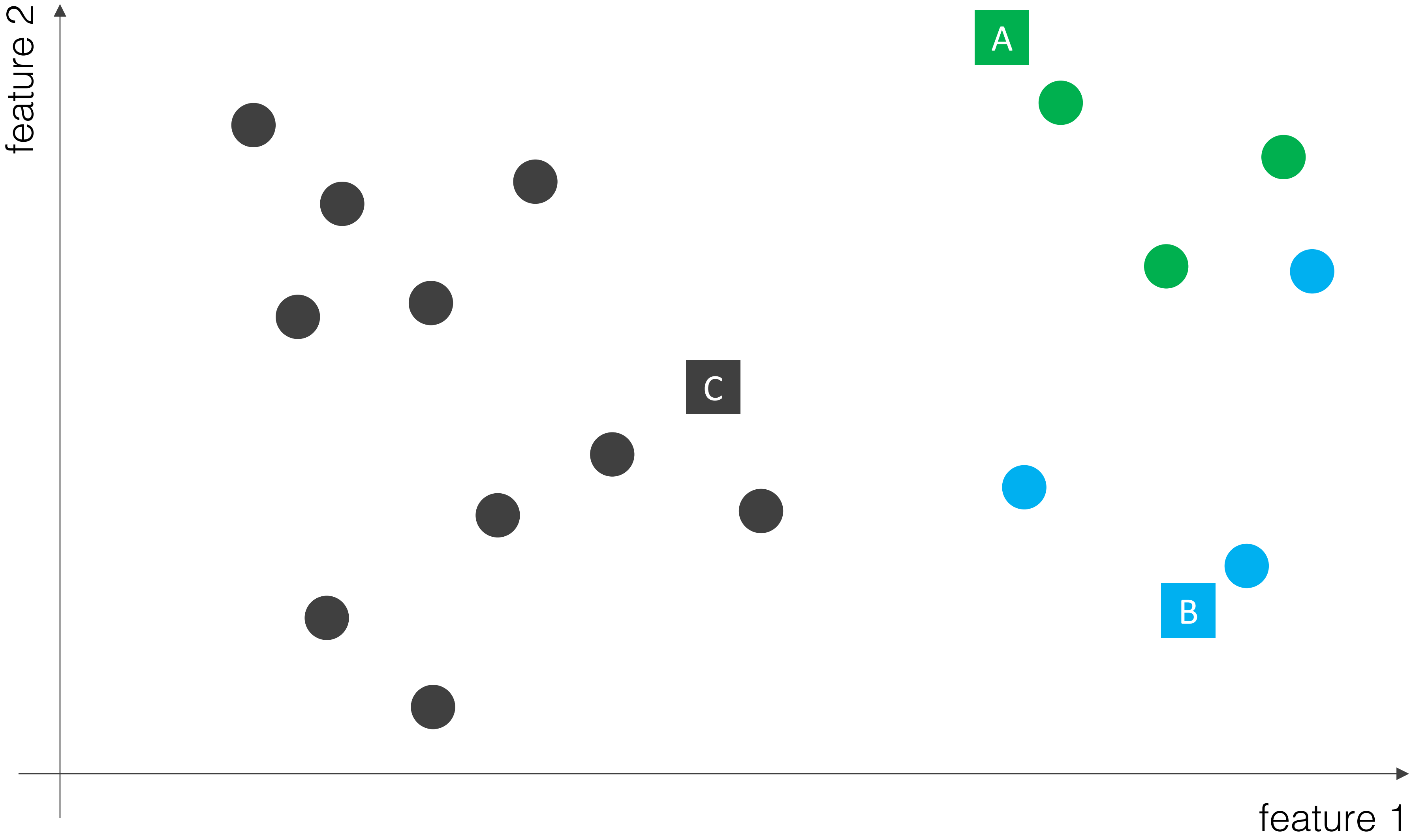
K-means clustering

1 Select k and randomly initialize k mean values

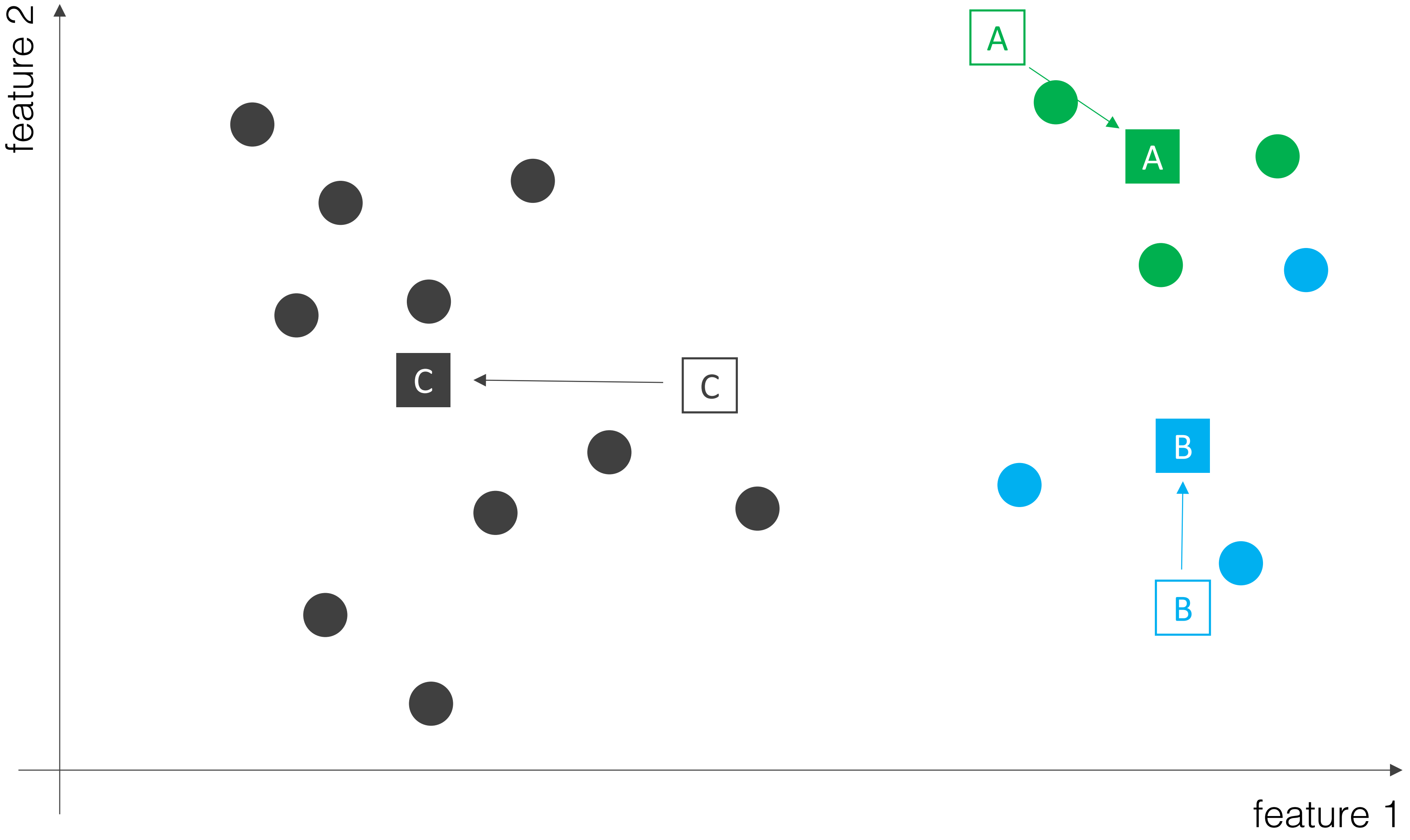


K-means clustering

1. Select k and randomly initialize k mean values
2. Assign observations to the nearest mean

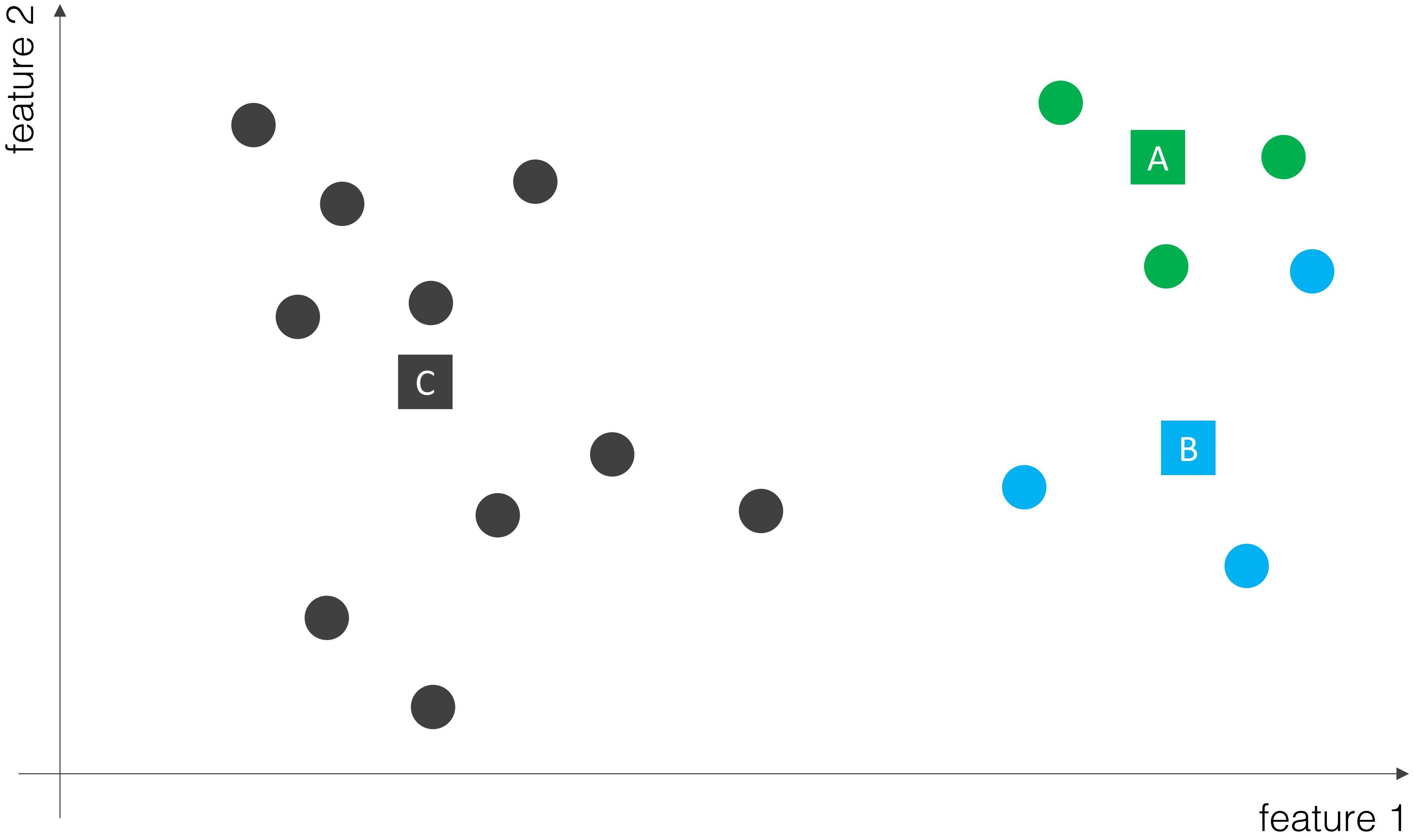


K-means clustering



1. Select k and randomly initialize k mean values
2. Assign observations to the nearest mean
3. Update the mean to be the centroid of the labeled data

K-means clustering



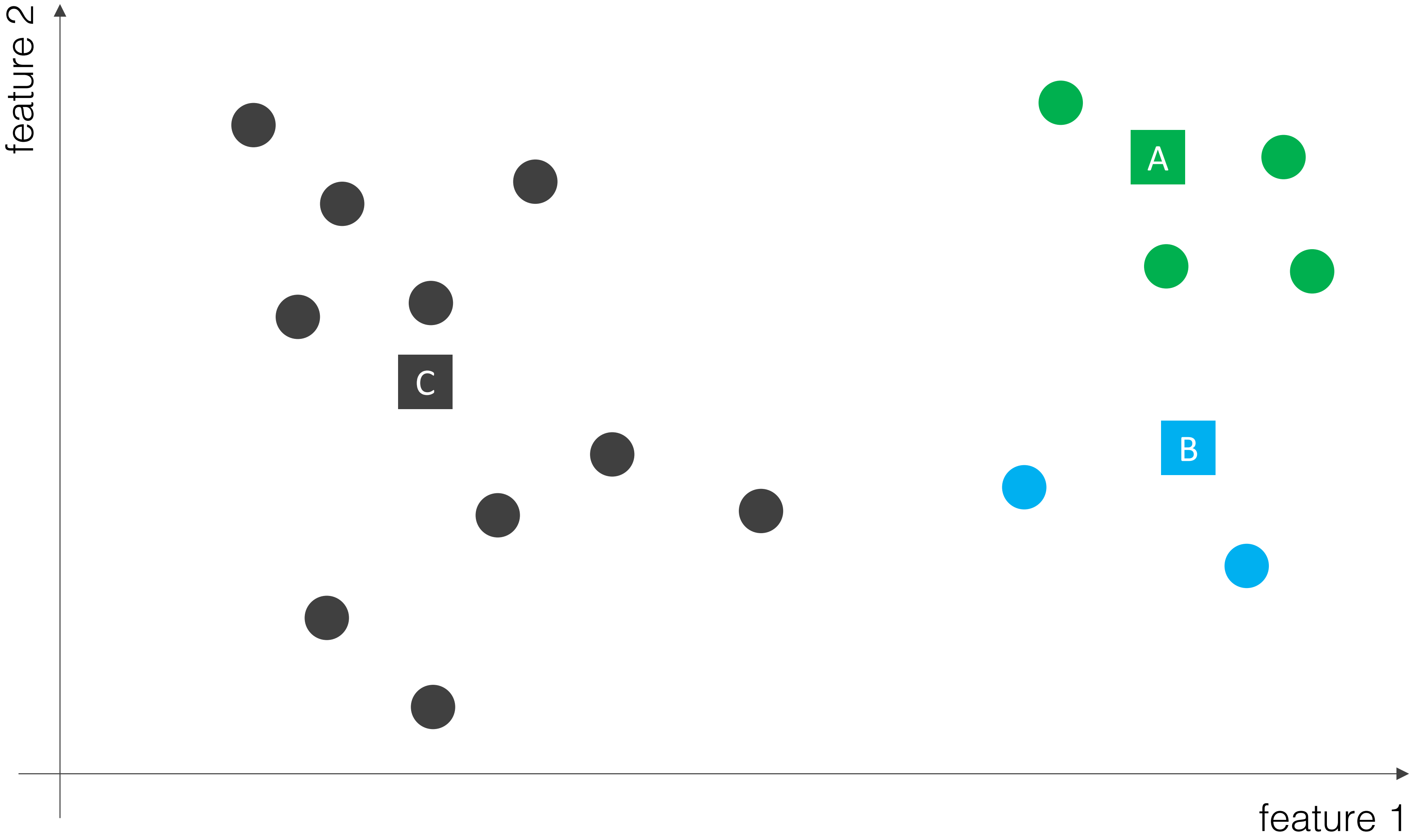
1. Select k and randomly initialize k mean values

2. Assign observations to the nearest mean

3. Update the mean to be the centroid of the labeled data

4. Repeat steps 2 and 3 until convergence

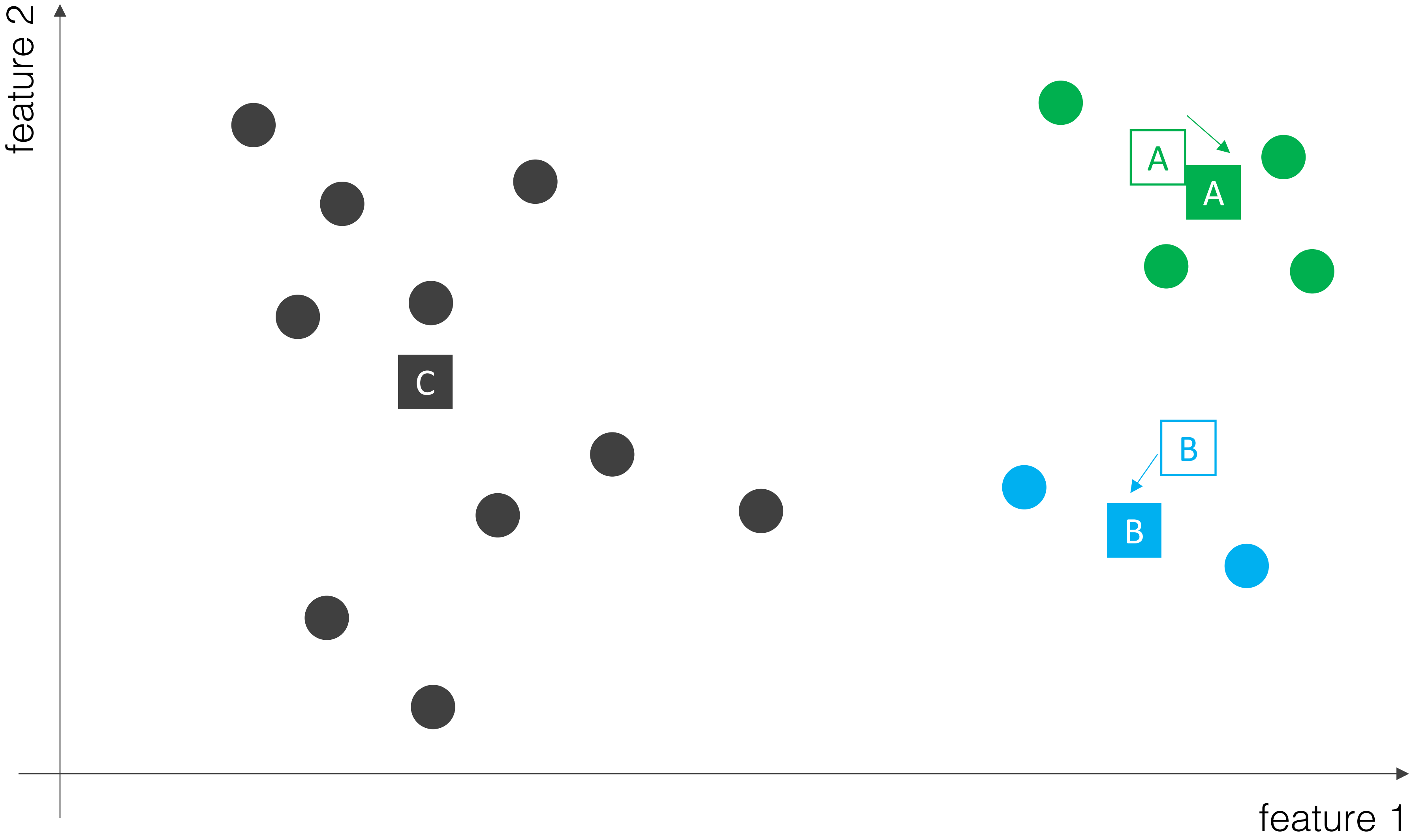
K-means clustering



1. Select k and randomly initial k mean values
2. Assign observations to the nearest mean
3. Update the mean to be the centroid of the labeled data
4. Repeat steps 2 and 3 until convergence

…Iteration 2

K-means clustering



1 Select k and randomly initialize k mean values

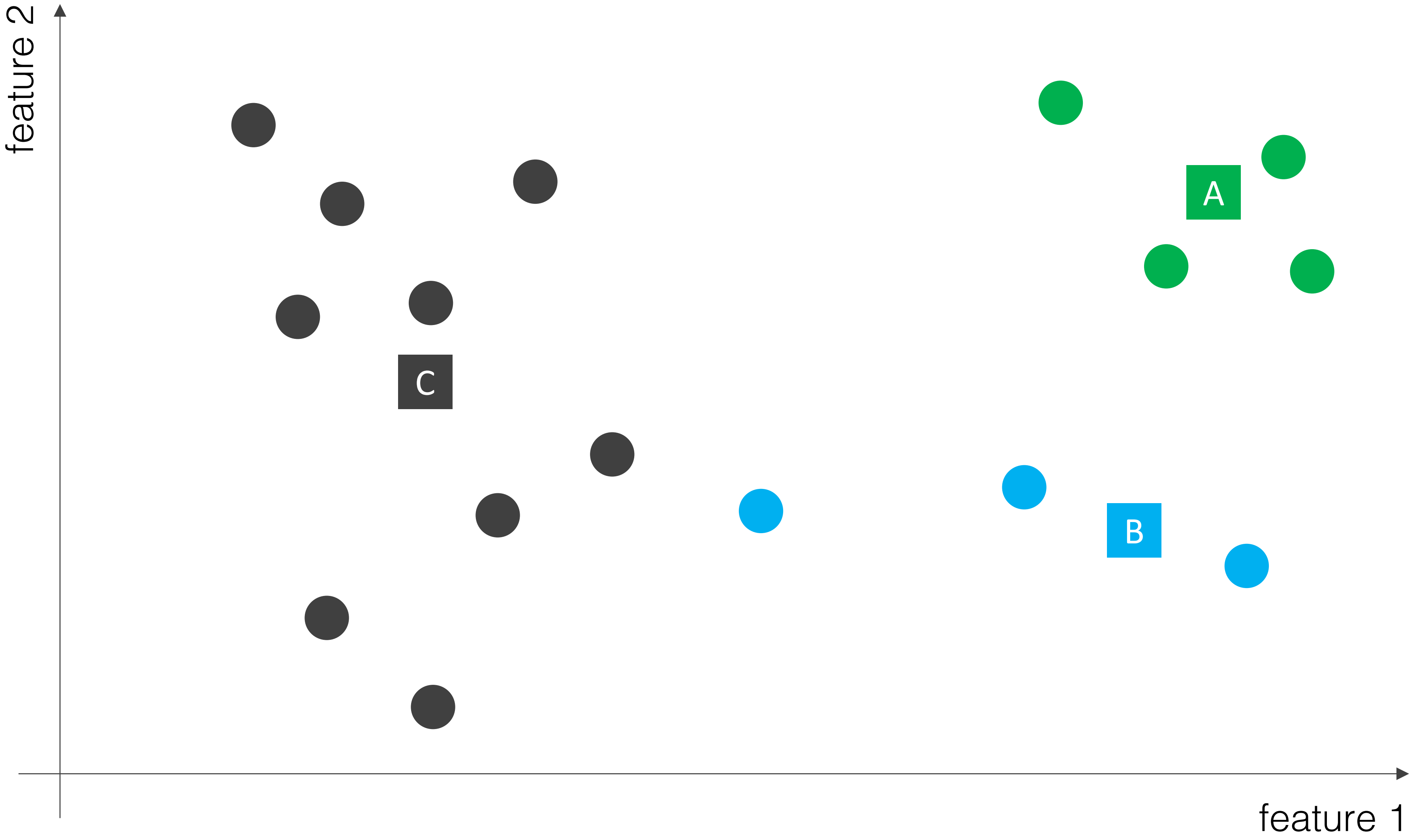
2 Assign observations to the nearest mean

3 Update the mean to be the centroid of the labeled data

4 Repeat steps 2 and 3 until convergence

…Iteration 2

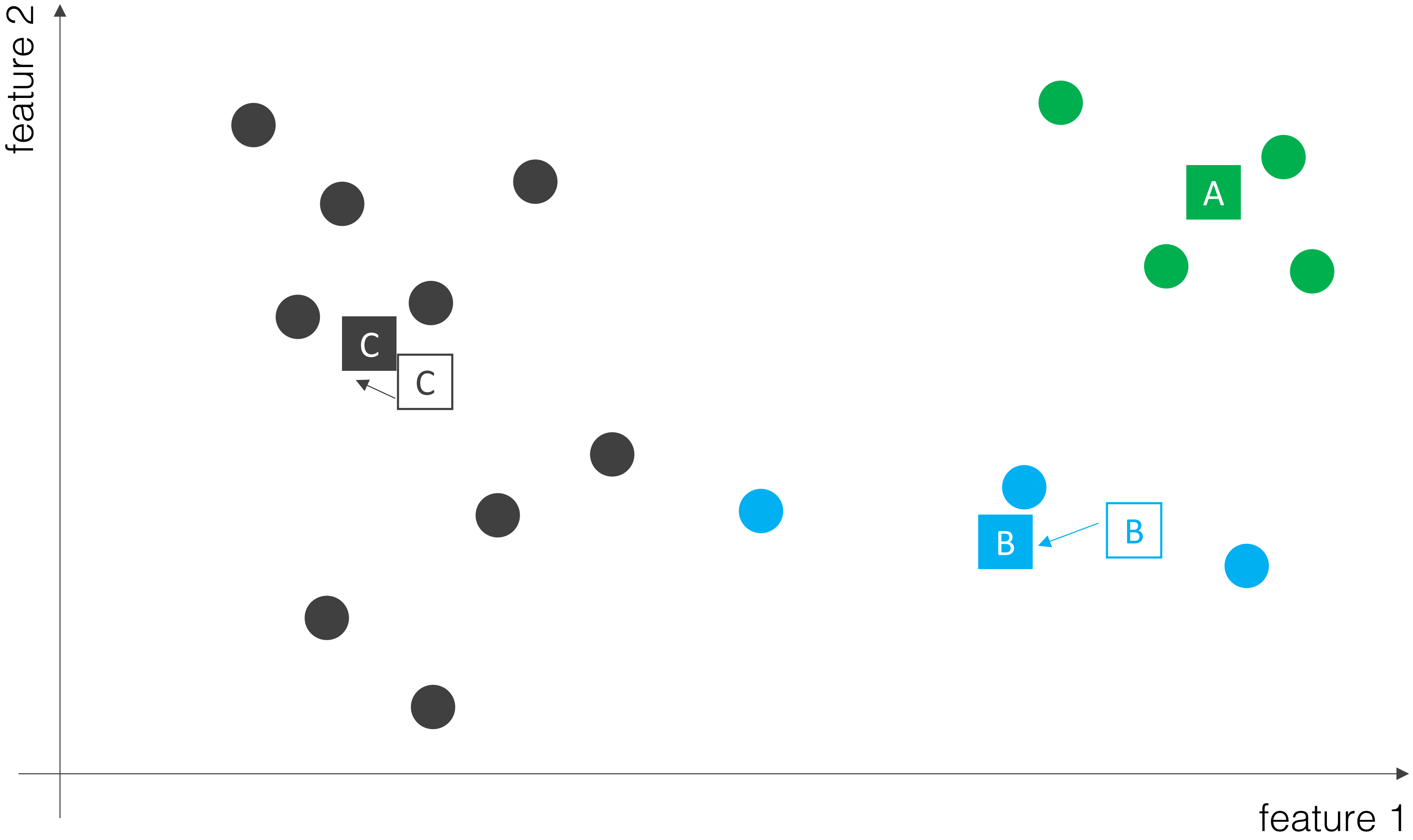
K-means clustering



1. Select k and randomly initialize k mean values
2. Assign observations to the nearest mean
3. Update the mean to be the centroid of the labeled data
4. Repeat steps 2 and 3 until convergence

…Iteration 3

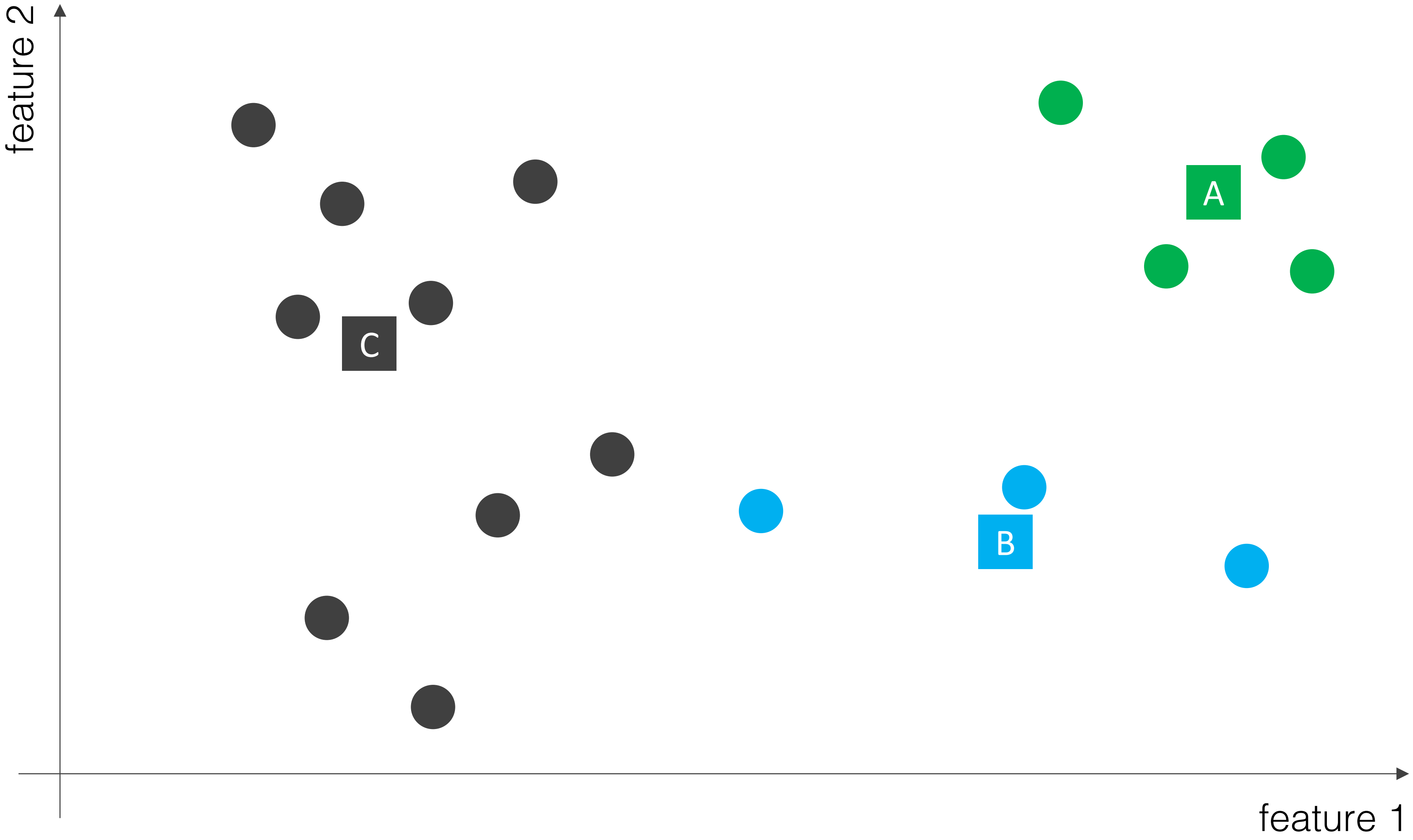
K-means clustering



1. Select k and randomly initialize k mean values
2. Assign observations to the nearest mean
3. Update the mean to be the centroid of the labeled data
4. Repeat steps 2 and 3 until convergence

…Iteration 3

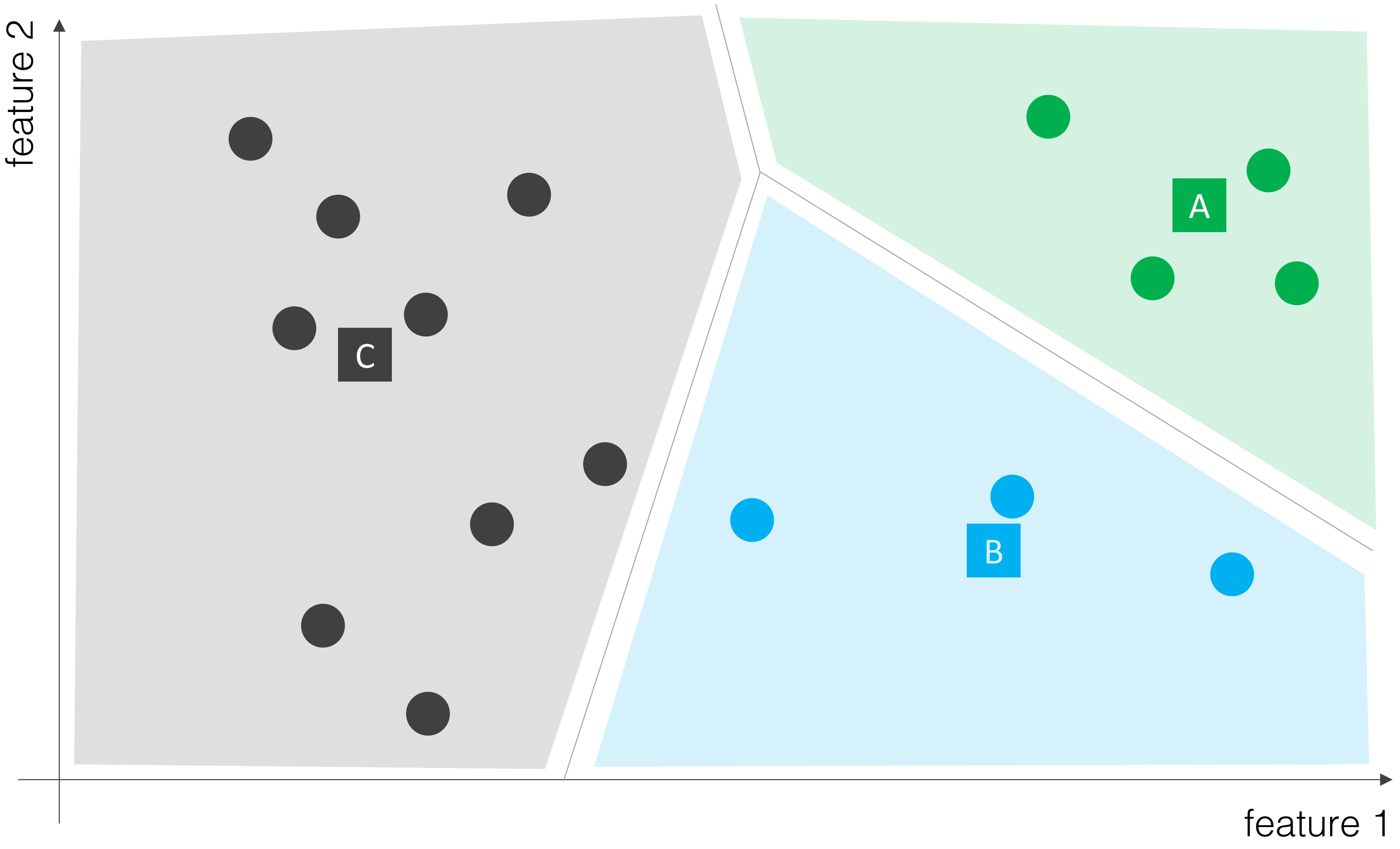
K-means clustering



1. Select k and randomly initialize k mean values
2. Assign observations to the nearest mean
3. Update the mean to be the centroid of the labeled data
4. Repeat steps 2 and 3 until convergence

…converged

K-means partitions the space into Voronoi cells



Under the hood, we minimize a cost function

Objective: For our N samples, identify K means, $\boldsymbol{\mu}_k$, such that the set of closest points in feature space are the minimum distance away.

$$r_{ik} = \begin{cases} 1 \text{ if } \boldsymbol{x}_i \text{ is closest to the kth mean } \boldsymbol{\mu}_k \\ 0 \text{ else} \end{cases}$$

responsibility

L_2 norm

$$C(\boldsymbol{x}_i, \boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_K) = \sum_{i=1}^{N} \sum_{k=1}^{K} r_{ik} \|\boldsymbol{x}_i - \boldsymbol{\mu}_k\|_2^2$$

1. E-step

Re-evaluate r_{ik}

$$r_{ik} = \begin{cases} 1 \text{ if } \boldsymbol{x}_i \text{ is closest to the kth mean } \boldsymbol{\mu}_i \\ 0 \text{ else} \end{cases}$$

Assign new “expected” cluster assignments

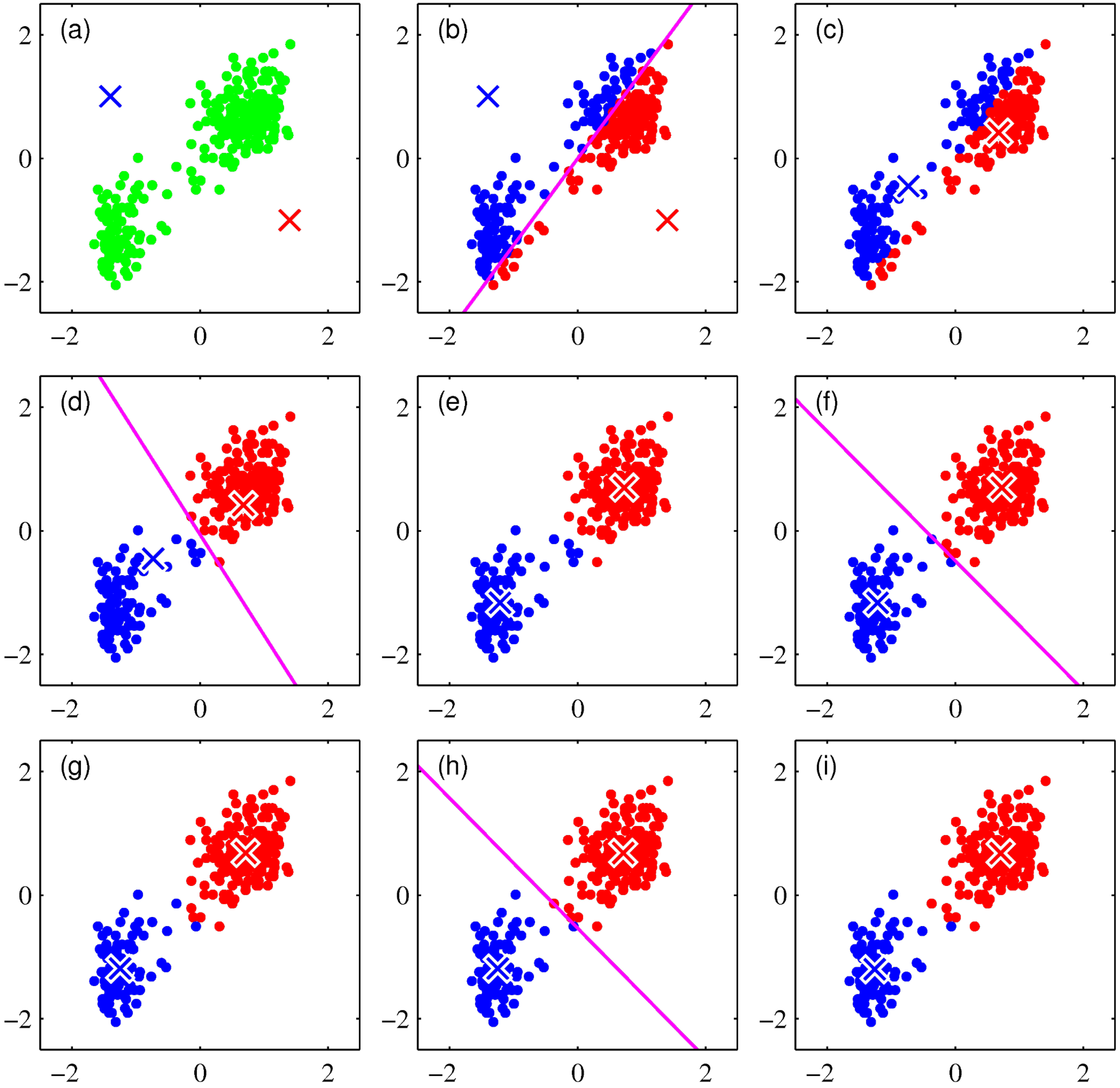
2. M-step

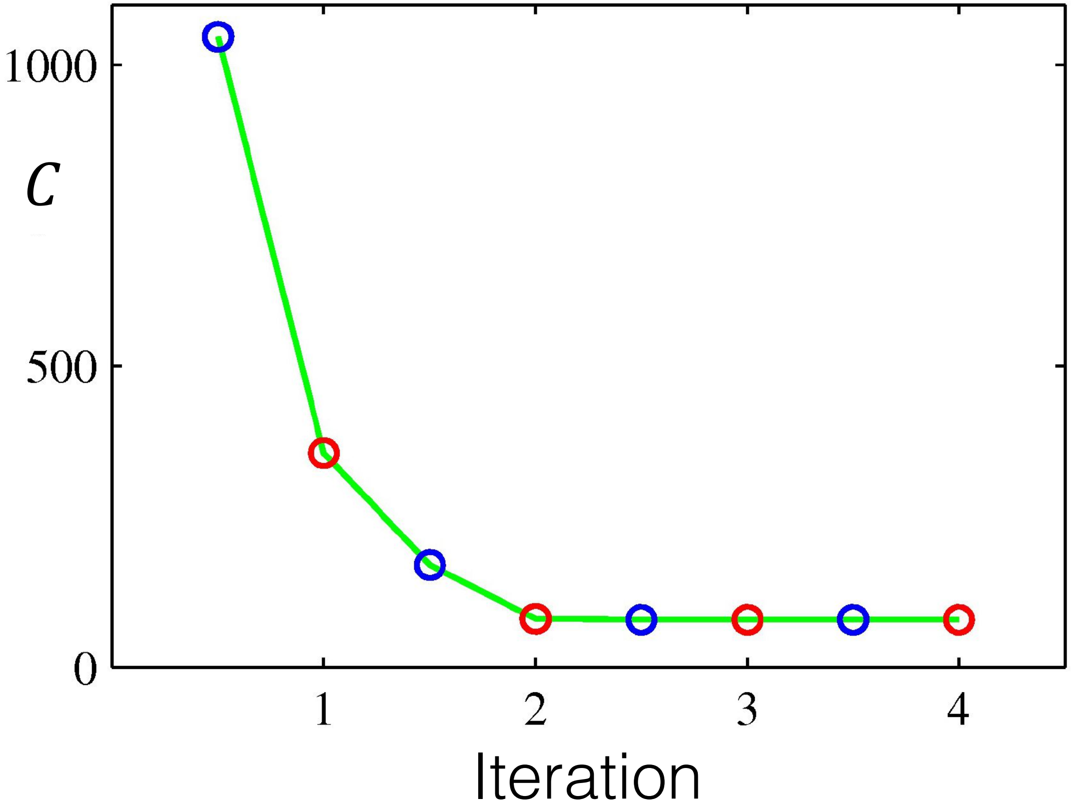
Minimize C wrt $\boldsymbol{\mu}_i$

$$\boldsymbol{\mu}_k = \frac{\sum_i r_{ik} \, \boldsymbol{x}_i}{\sum_i r_{ik}}$$

Update the cluster means to maximize the likelihood

Convergence





Bishop, Pattern Recognition, 2006

How to choose k: Elbow method

Run k-means for various k

Choose the value of k at the “elbow” of the curve

Increasing k will improve the fit, but at the cost of potentially overfitting the data

Other approach: silhouette (graphical approach to evaluating cluster fit)

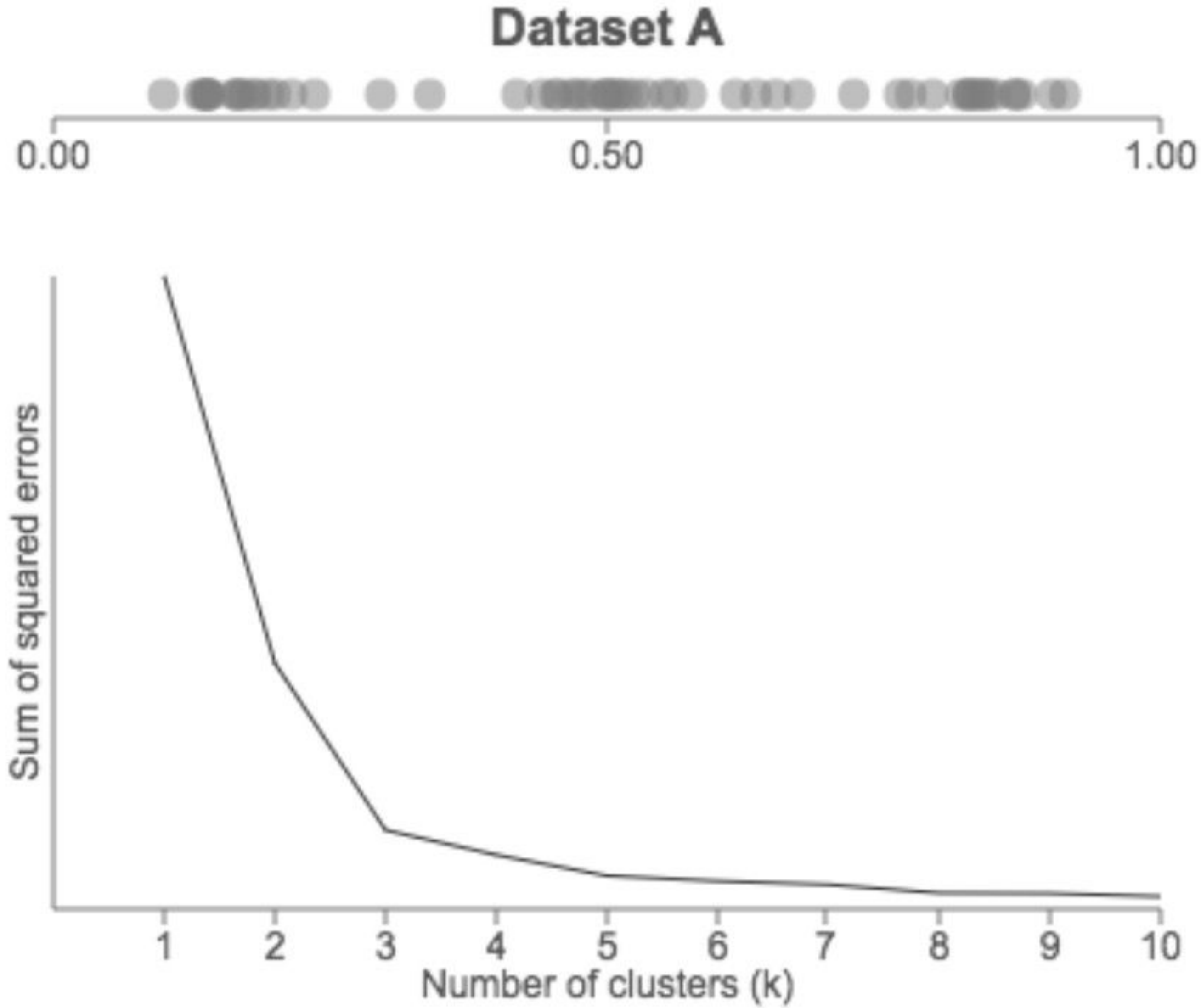
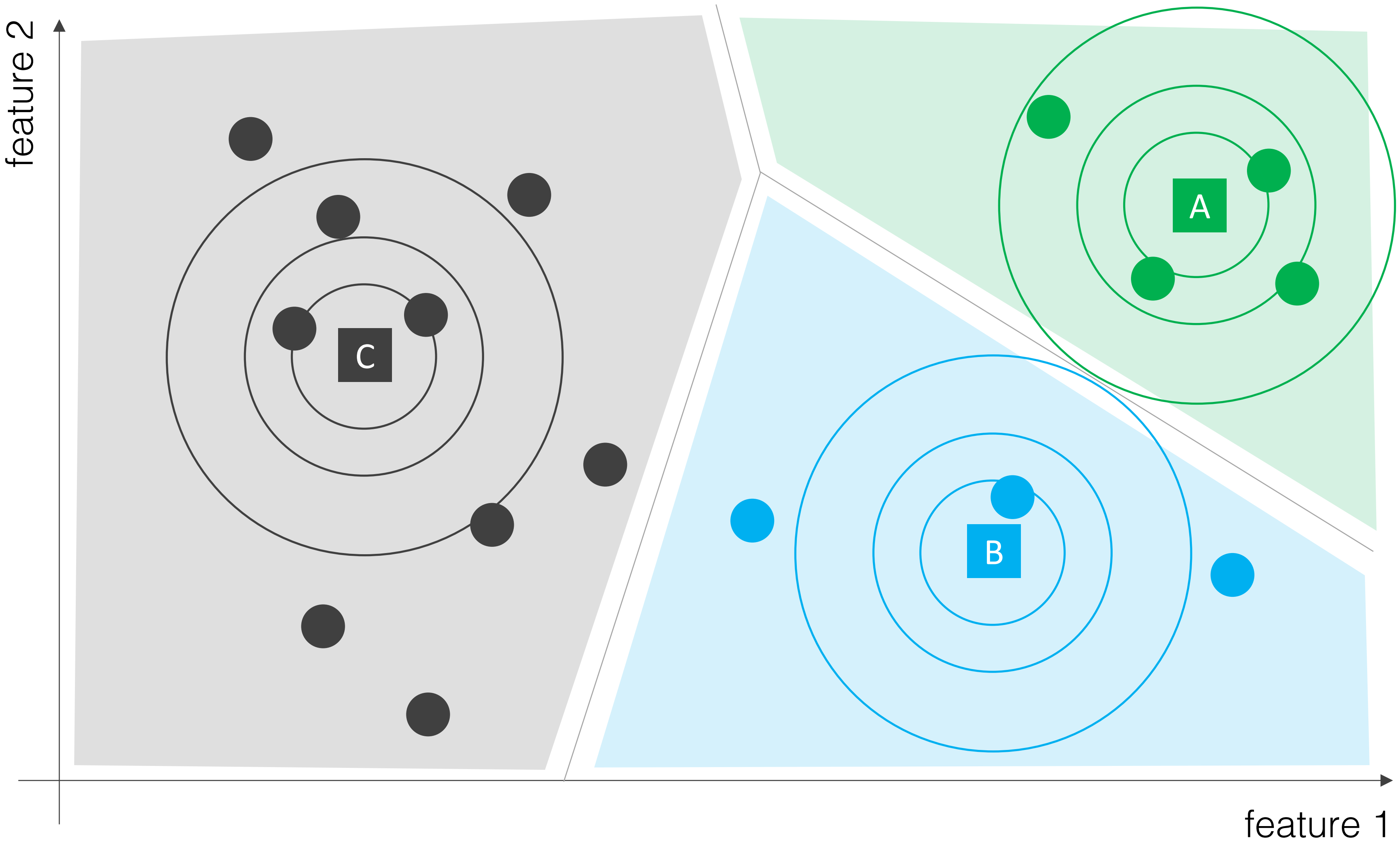


Image by Robert Gove: https://bl.ocks.org/rpgove/0060ff3b656618e9136b

Relationship to Gaussian distributions



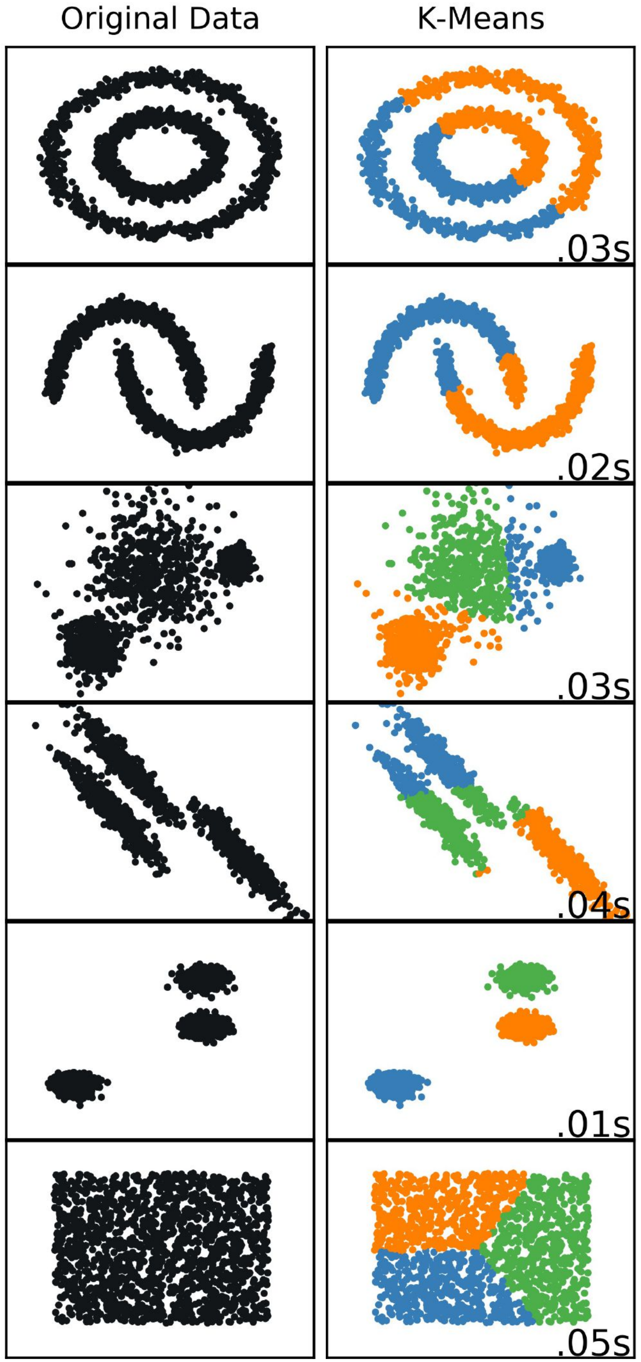
Assumes the clusters are **Gaussians** centered at the mean, each with **identical covariance matrices**, where all the features are independent:

$$\boldsymbol{\Sigma}_{\mathrm{k}} = \sigma^2 \boldsymbol{I} = \begin{bmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{bmatrix}$$

Examples: K-Means

Converges very quickly

Sensitive to initialization of means



Struggles when there are **nonlinear** boundaries between clusters

Struggles in situations with **variation in cluster variance** and **correlation between features**

Excels with clusters of **equal variance**

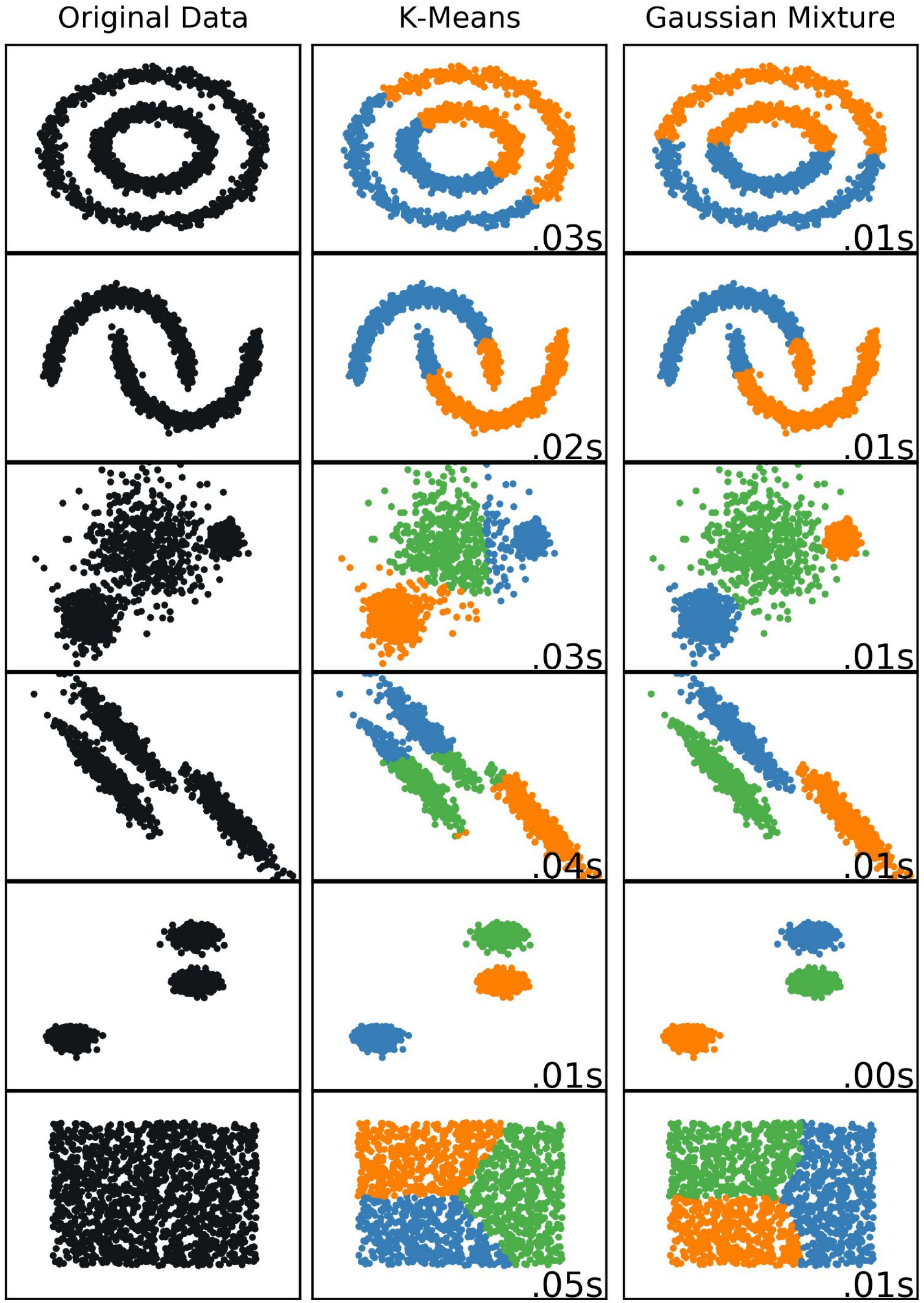
Will divide into k clusters even when there are not k

K-Means

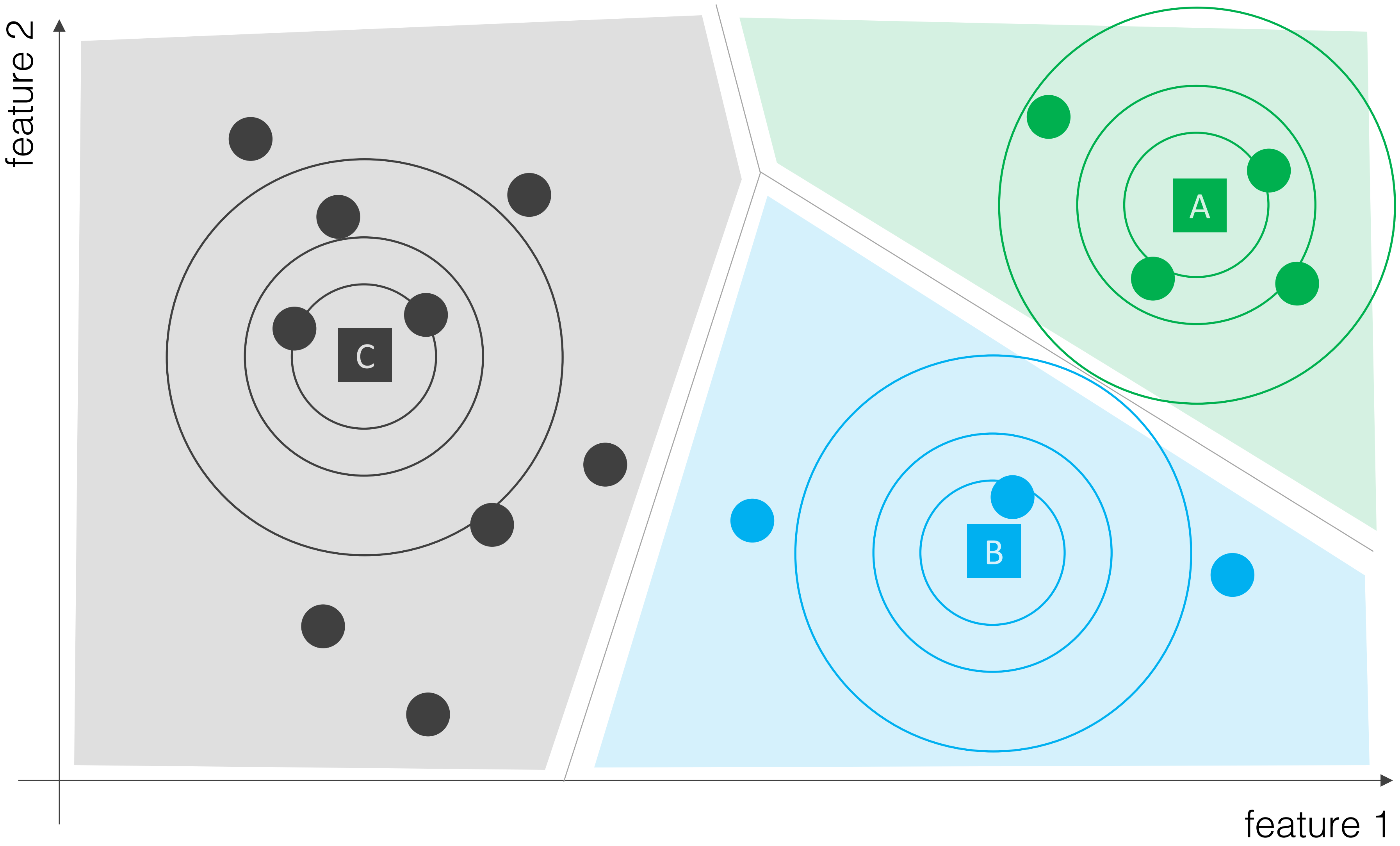
+

Gaussian Mixture Models (GMMS)

Clustering and Density Estimation (GMMS)



Relaxing our assumptions on covariance…

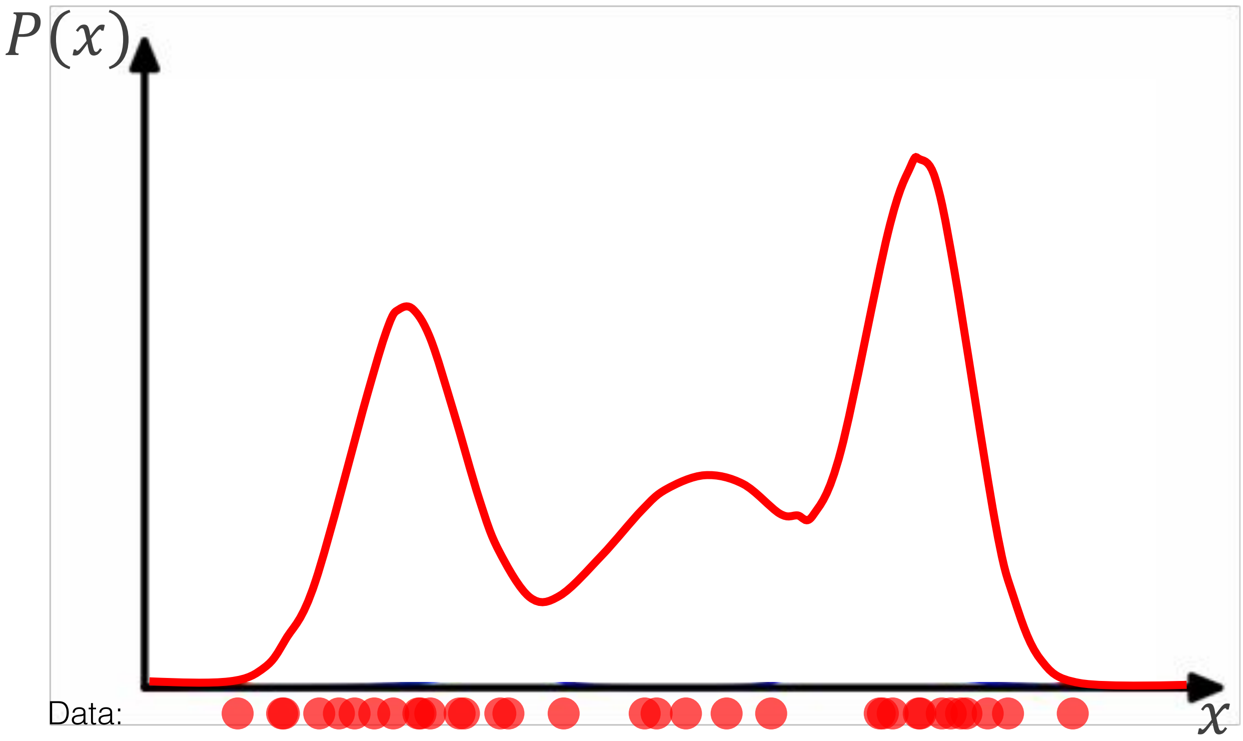


What if we **don't** assume the Gaussian clusters have **identical, diagonal covariance matrices**?

Gaussian Mixture Models

For **clustering** and **density estimation**

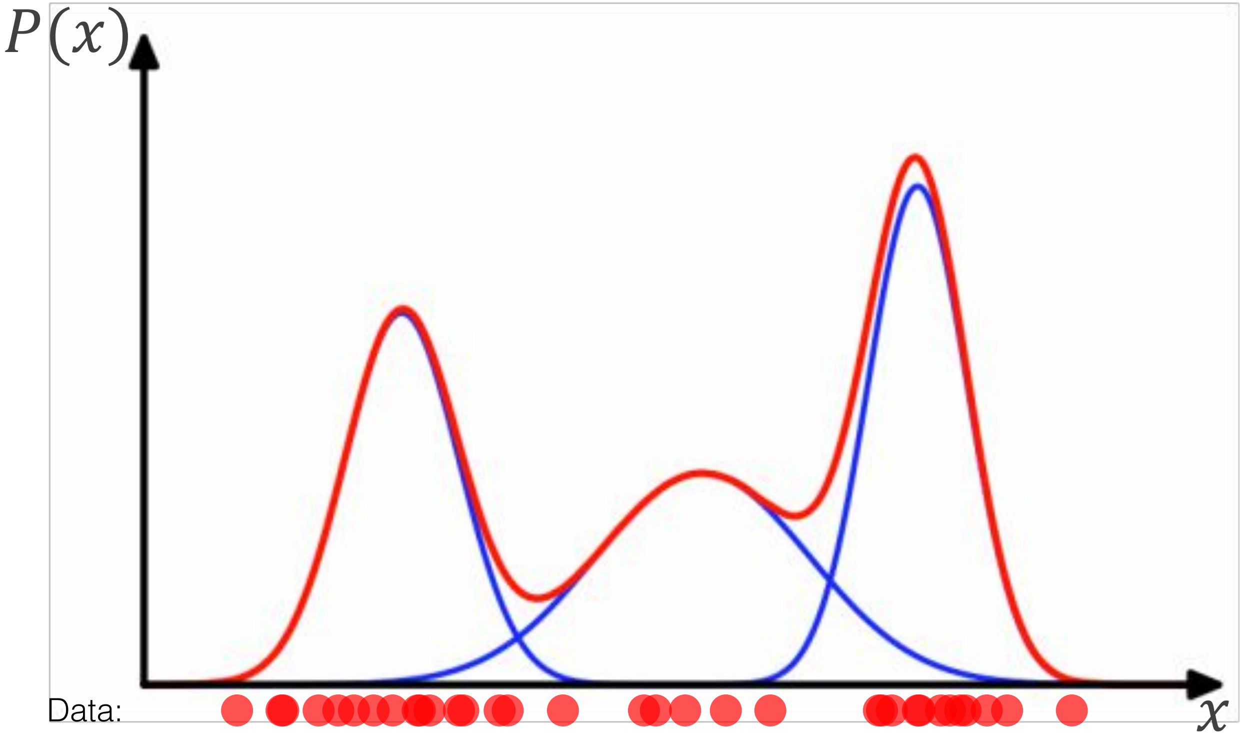
Mixture model



We can estimate the distribution density of our data…

Image from Shaun Dowling

Mixture model

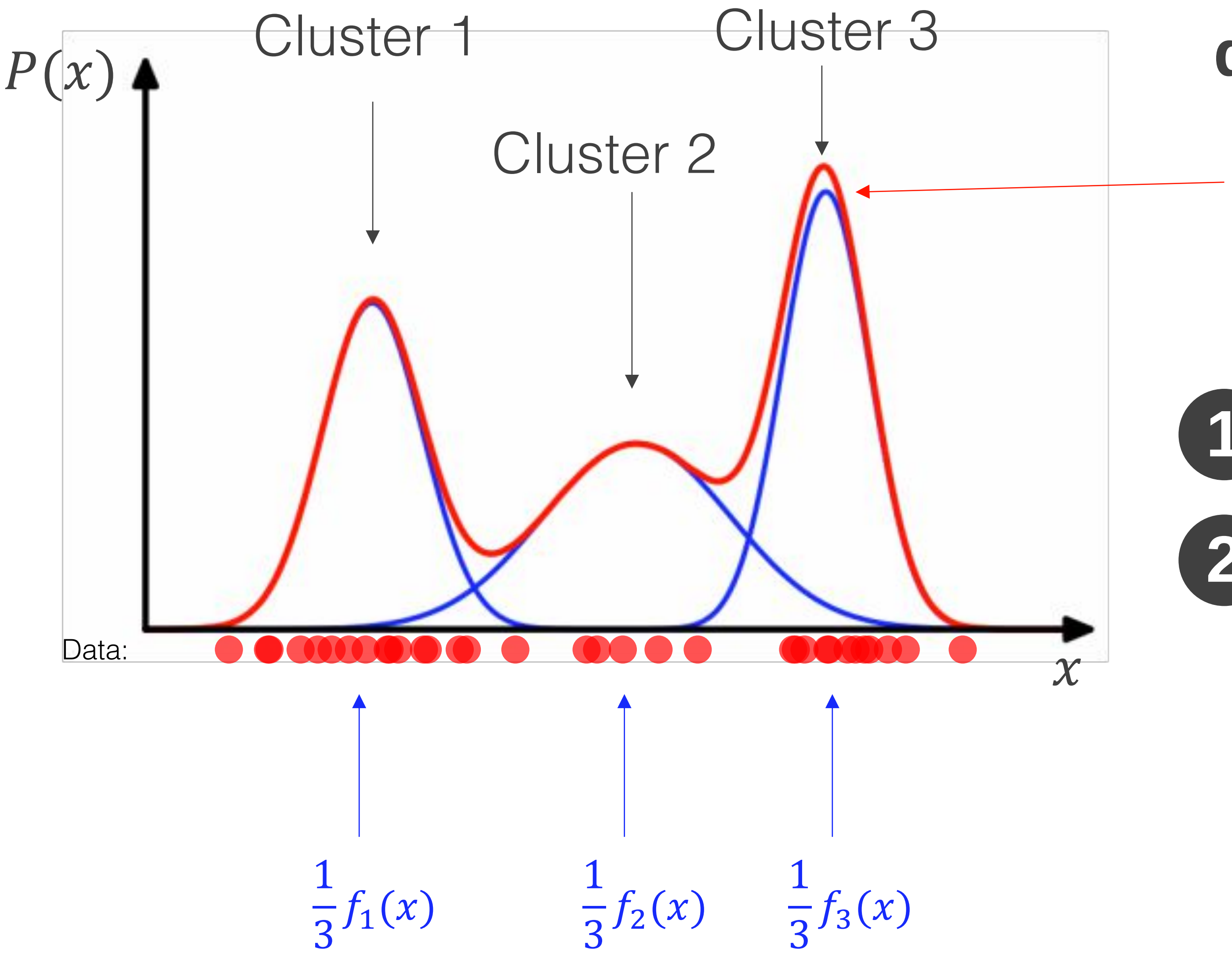


We can estimate the distribution density of our data…

…using a mixture of distributions

Image from Shaun Dowling

Mixture model



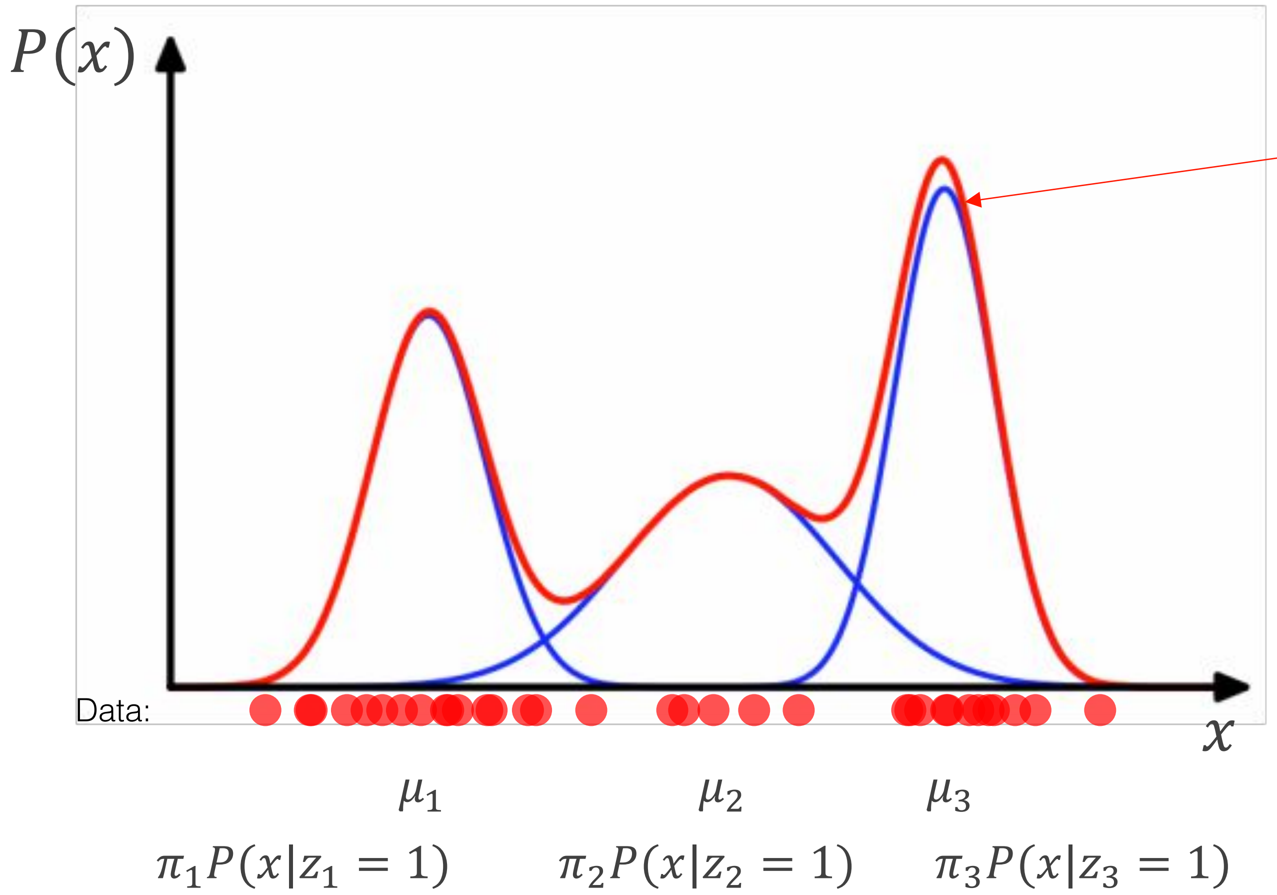
A weighted average of density functions

$$P(x) = \frac{1}{3}f_1(x) + \frac{1}{3}f_2(x) + \frac{1}{3}f_3(x)$$

1. Fit the model to the data
2. Use the model to assign clusters

Image from Shaun Dowling

Gaussian mixture model



A mixture model is represented as:

$$P(x) = \sum_{k=1}^{K} P(z_k = 1) P(x|z_k = 1)$$

If we assume this is Gaussian, it becomes a Gaussian mixture model (GMM)

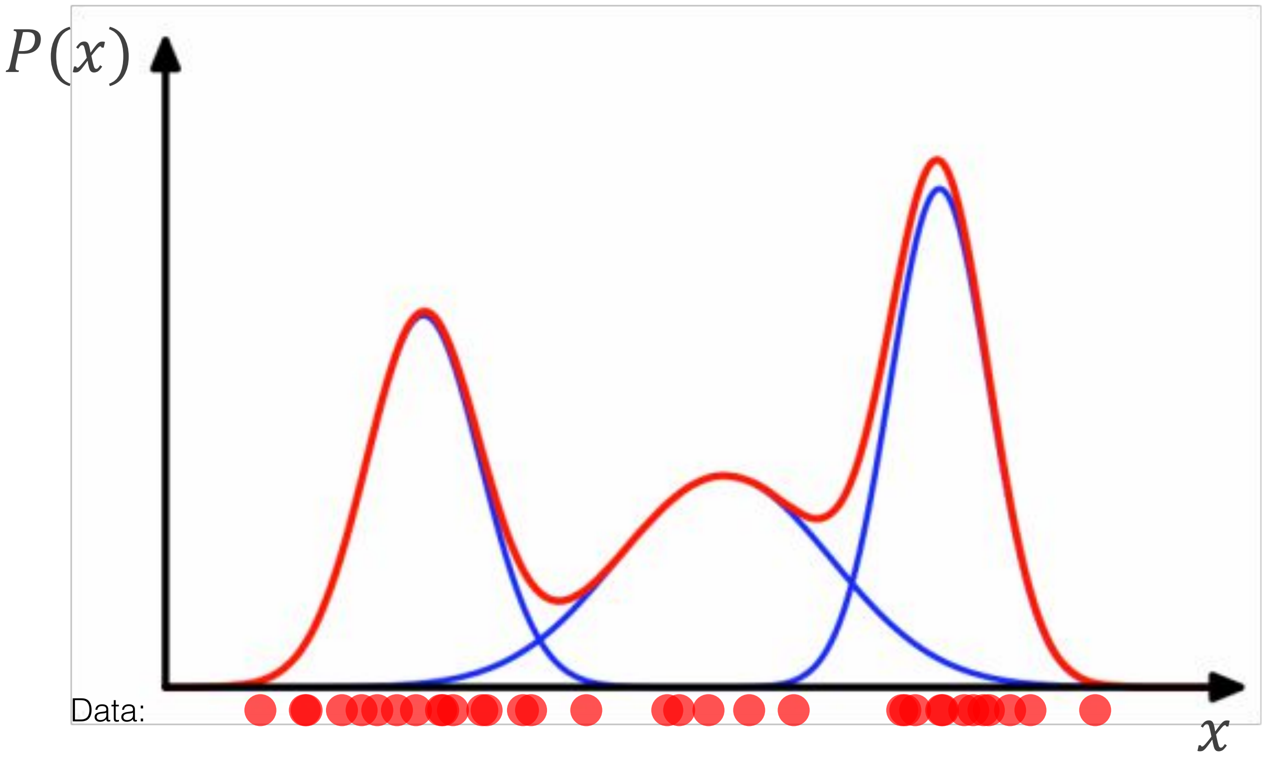
The mixing coefficients $\pi_k = P(z_k = 1)$ need to sum to 1 for a valid distribution

$$\sum_{k=1}^{K} \pi_k = 1$$

z_k = binary variable that represents cluster membership

Image from Shaun Dowling

Gaussian mixture model

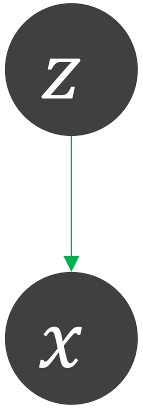


μ_1 μ_2 μ_3

$\pi_1 P(x|z_1 = 1)$ $\pi_2 P(x|z_2 = 1)$ $\pi_3 P(x|z_3 = 1)$

$$P(x) = \sum_{k=1}^{K} P(z_k = 1) P(x|z_k = 1)$$

Here we assume z is a **latent** (hidden / unobservable) variable



Hidden

This variable controls which of the k mixture components a sample is drawn from. We don't DIRECTLY see this.

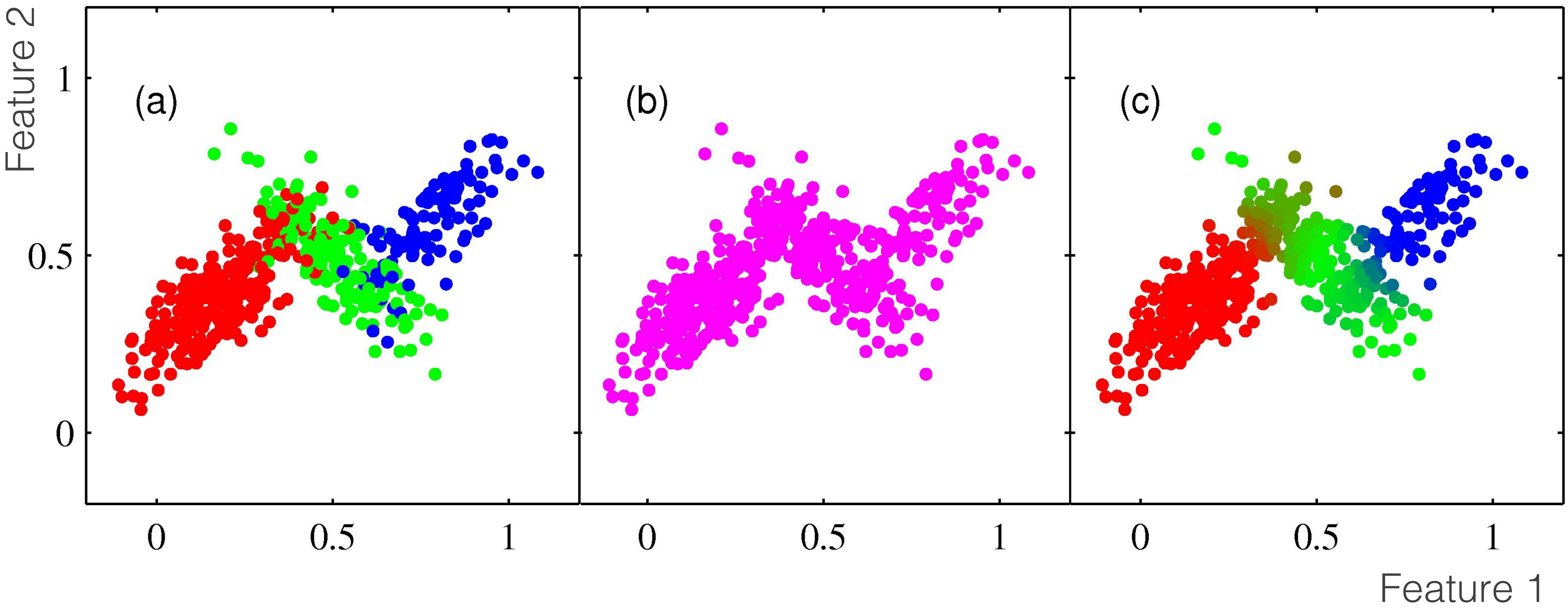
Observable

Given z, we assume a sample is drawn from $P(x|z_k = 1)$

Note: We can use these terms to compute the posterior probability $P(z_k|x)$

Image from Shaun Dowling

Gaussian Mixture Model Latent Variables



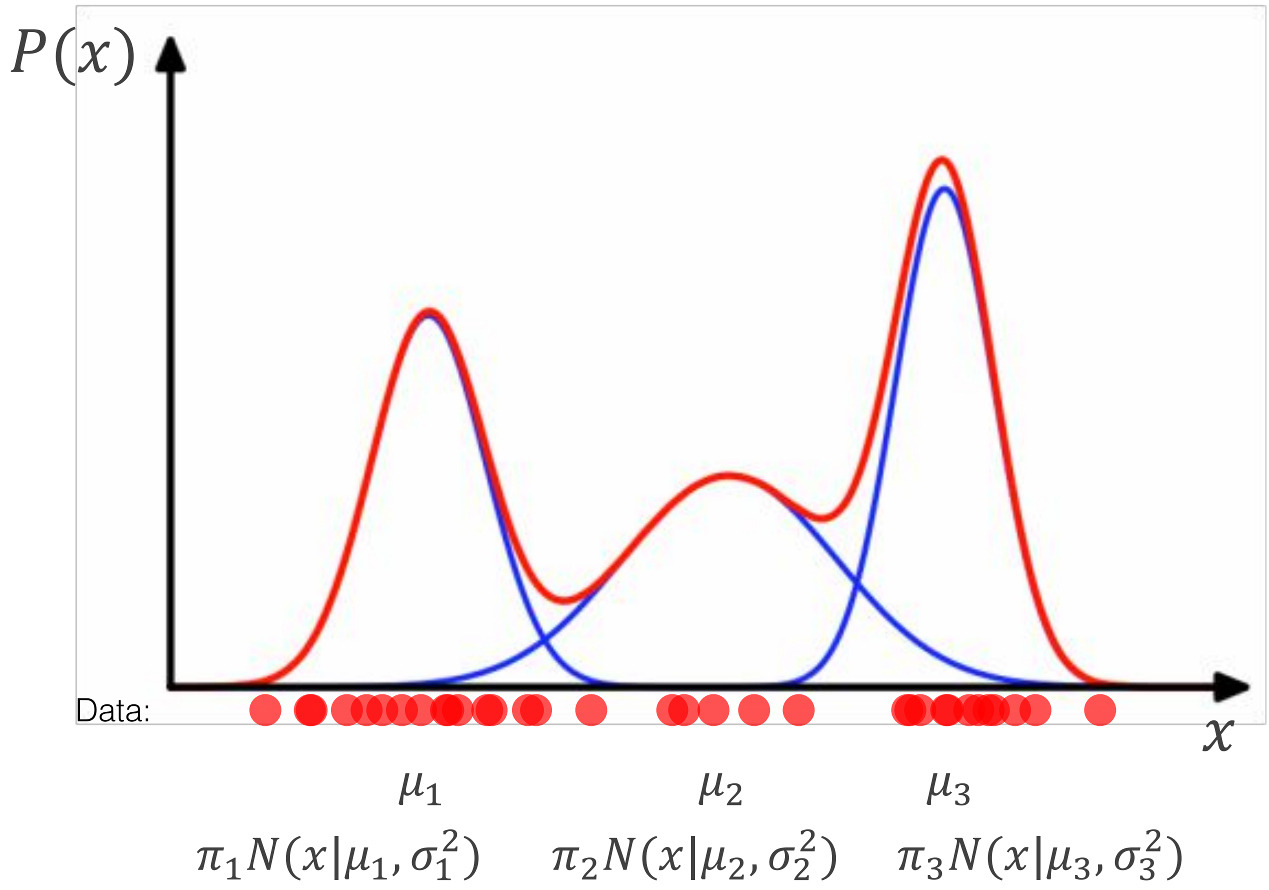
Complete data with latent variable “labels” z

Incomplete data without latent variable labels

Posterior probabilities, a.k.a. responsibilities

Image from Bishop, Pattern Recognition, 2006

Gaussian mixture model



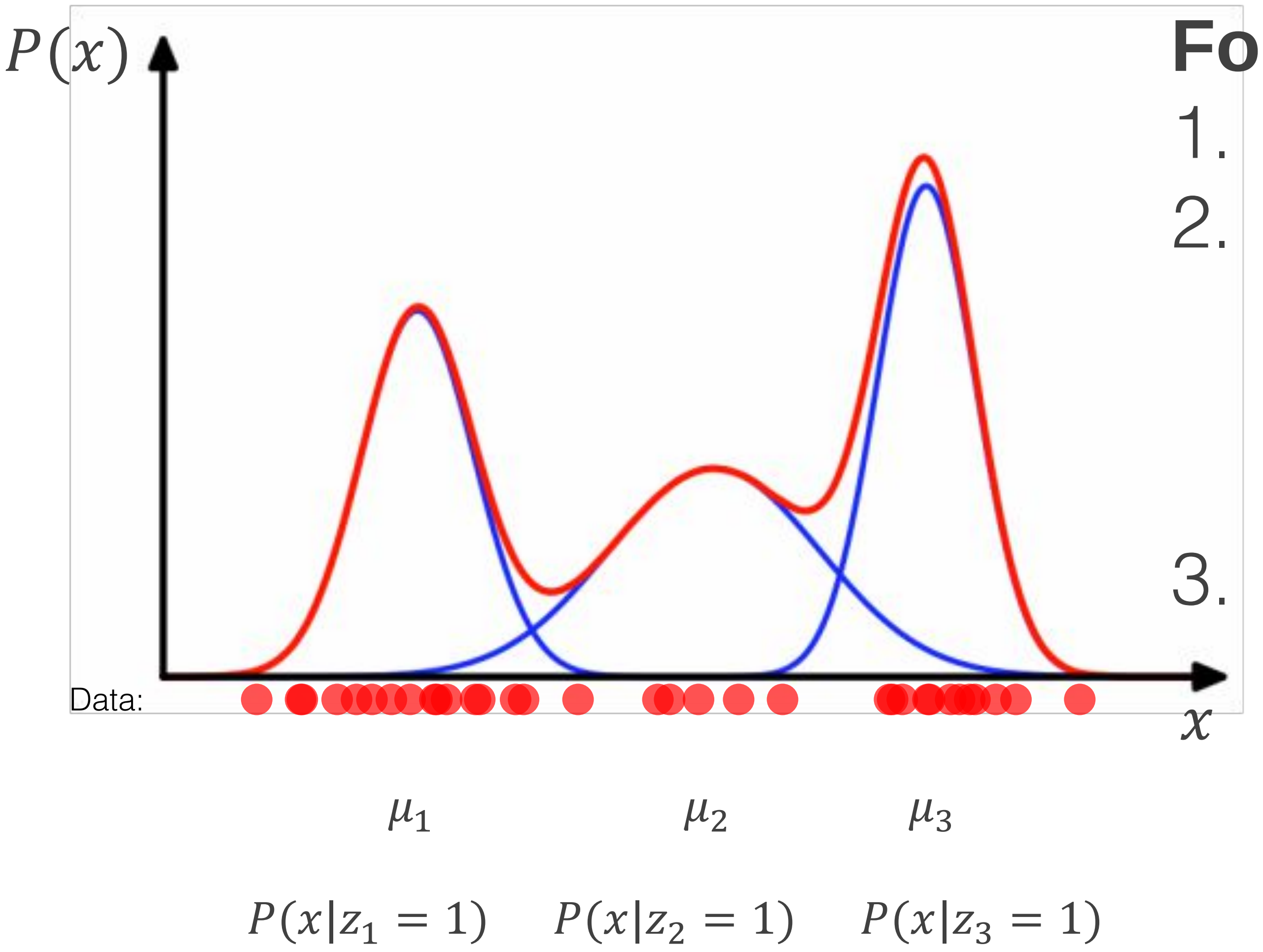
The Gaussian mixture model is represented as:

$$P(x) = \sum_{k=1}^{K} \pi_k N(x|\mu_k, \sigma_k^2)$$

where $$\sum_{k=1}^{K} \pi_k = 1$$

Image from Shaun Dowling

Gaussian mixture model



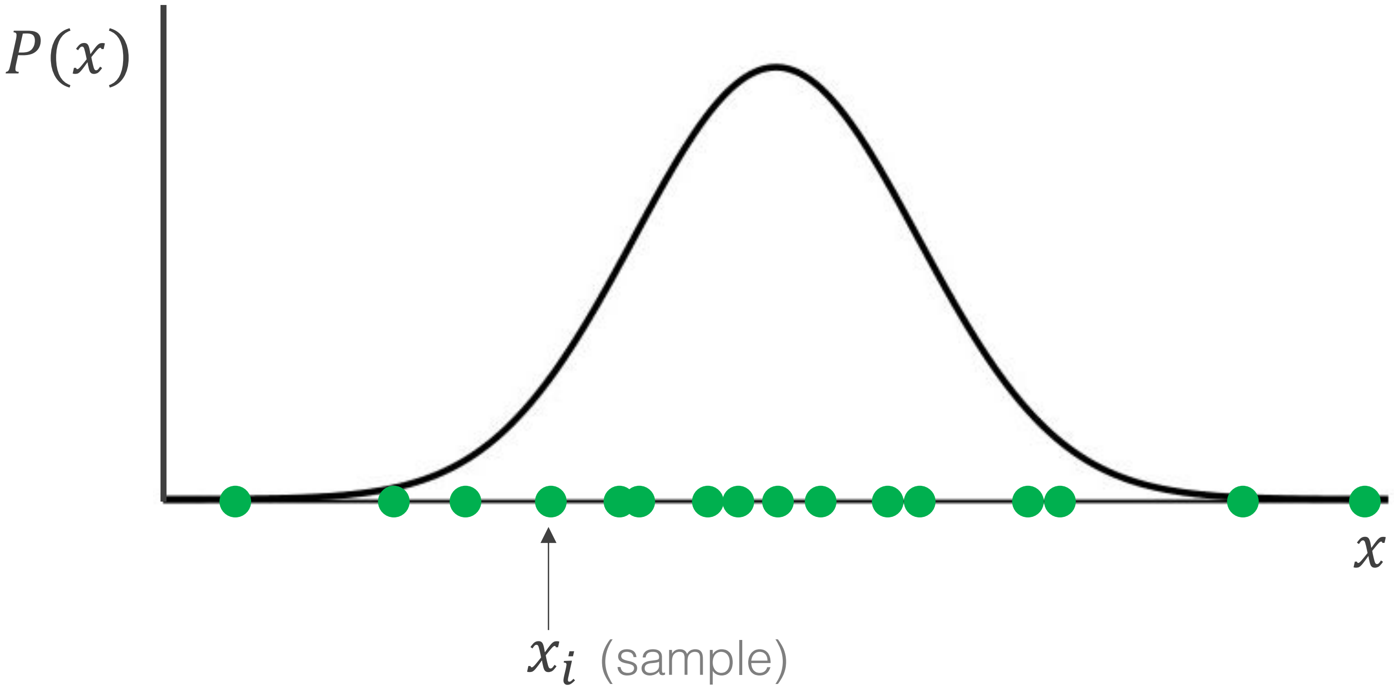
For clustering:

1. Pick a number of clusters, K
2. Fit a GMM to the data (estimate π_k, μ_k, σ_k^2 for $k = 1, \dots, K$ to maximize the likelihood of the data given the model)
3. Pick the cluster, z_k, that each data point was most likely to come from

Image from Shaun Dowling

Density estimation for a single mixture component

a.k.a. model fitting



Likelihood of one sample given the model

$$P(x_i|\mu,\sigma^2) = N(x_i|\mu,\sigma^2)$$

$$= \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$$

Assuming independent samples, the likelihood of the data given the model is:

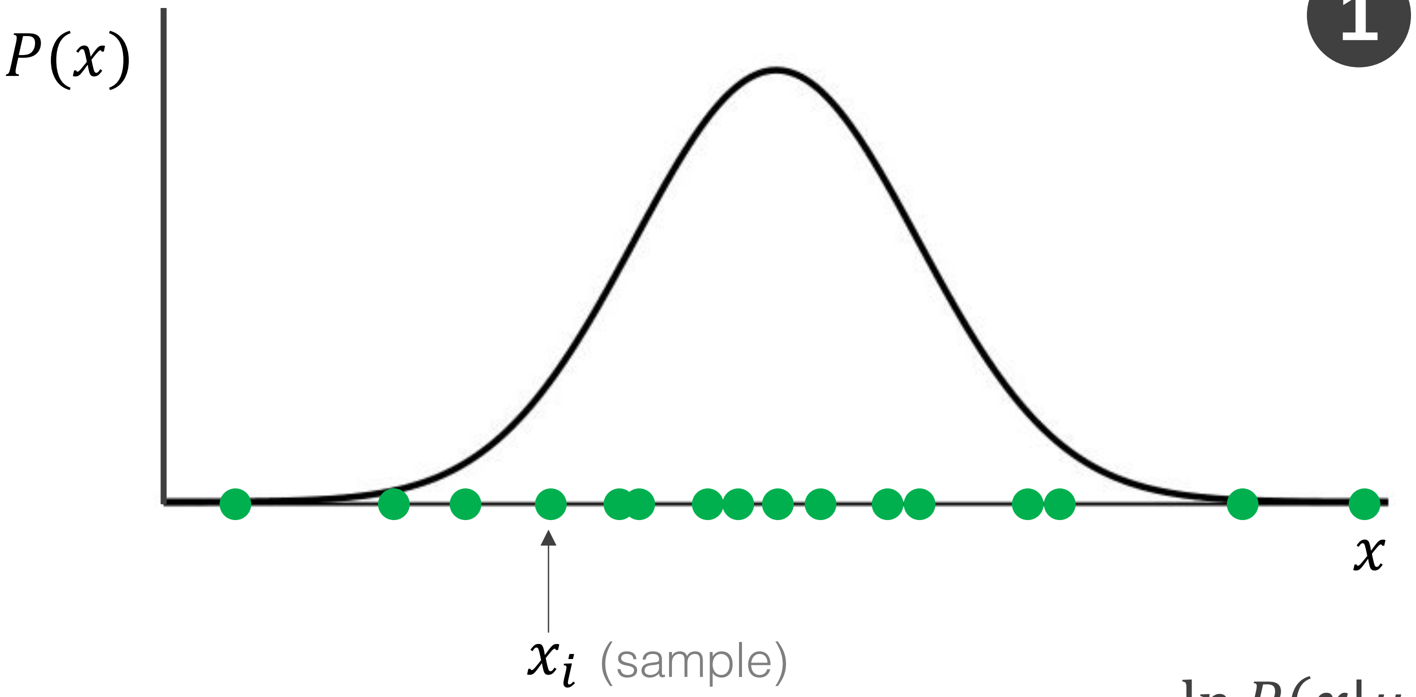
$$P(\boldsymbol{x}|\mu,\sigma^2)$$

$$= \prod_{i=1}^{N} P(x_i|\mu,\sigma^2)$$

$$= \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$$

Density estimation for a single mixture component

a.k.a. model fitting



1 We follow our familiar pattern: maximize the likelihood of the data by choosing our model parameters: μ, σ^2

$$P(\boldsymbol{x}|\mu,\sigma^2) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$$

2 Calculate the log likelihood:

$$\ln P(\boldsymbol{x}|\mu,\sigma^2) = -\frac{N}{2}\ln 2\pi\sigma^2 - \frac{1}{2\sigma^2}\sum_{i=1}^{N}(x_i-\mu)^2$$

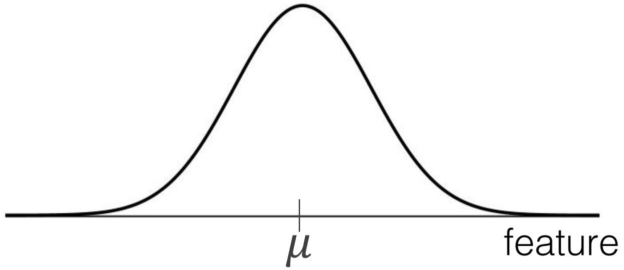
3 Take the derivative of the log likelihood w.r.t. each parameter (μ, σ^2), set equal to zero, solve for μ, σ^2

$$\hat{\mu} = \frac{1}{N}\sum_{i=1}^{N} x_i \qquad \hat{\sigma}^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i-\hat{\mu})^2$$

From a univariate to a multivariate Gaussian

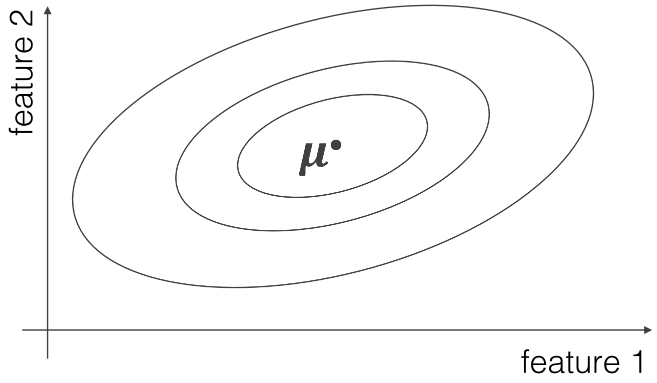
Univariate Normal density

$$N(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left\{-\frac{(x_i-\mu)^2}{2\sigma^2}\right\}$$



Multivariate Normal density

$$N(\boldsymbol{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{\sqrt{2\pi|\Sigma|}}\exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right\}$$



From a univariate to a multivariate Gaussian

Univariate Normal MLE parameter estimates:

$$\hat{\mu} = \frac{1}{N}\sum_{i=1}^{N} x_i \qquad \hat{\sigma}^2 = \frac{1}{N}\sum_{i=1}^{N} (x_i - \hat{\mu})^2$$

Multivariate Normal MLE parameter estimates:

$$\hat{\boldsymbol{\mu}} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}_i \qquad \hat{\boldsymbol{\Sigma}} = \frac{1}{N}\sum_{i=1}^{N} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}})(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}})^T$$

Moving from a single Gaussian to a mixture of Gaussians

Density estimation for a Gaussian mixture model

0 We define the likelihood of one observation given our model with parameters $\boldsymbol{\pi}_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ for $k = 1, \dots, K$

$$P(\boldsymbol{x}_i | \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{k=1}^{K} \pi_k N(\boldsymbol{x}_i | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

1 We assume the observations are independent and calculate the likelihood for all our data

$$P(\boldsymbol{X} | \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \prod_{i=1}^{N} \sum_{k=1}^{K} \pi_k N(\boldsymbol{x}_i | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

2 Calculate the log likelihood:

$$\ln P(\boldsymbol{X} | \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{i=1}^{N} \ln \left[\sum_{k=1}^{K} \pi_k N(\boldsymbol{x}_i | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right]$$

3 Take the derivative of the log likelihood w.r.t. each parameter ($\boldsymbol{\pi}_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ for $k = 1, \dots, K$), set equal to zero, solve for the parameters

Density estimation for a Gaussian mixture model

Log likelihood of the data given the model parameters

$$\ln P(\boldsymbol{X}|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma}) = \sum_{i=1}^{N} \ln\left[\sum_{k=1}^{K} \pi_k N(\boldsymbol{x}_i|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)\right]$$

There is no **closed-form solution** that maximizes this.

We could use gradient descent BUT this approach can suffer from **severe overfitting**

Example: $k = 2$ mixture components

$$\ln P(\boldsymbol{X}|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma}) = \sum_{i=1}^{N} \ln[\pi_1 N(\boldsymbol{x}_i|\boldsymbol{\mu}_1,\boldsymbol{\Sigma}_1) + \pi_2 N(\boldsymbol{x}_i|\boldsymbol{\mu}_2,\boldsymbol{\Sigma}_2)]$$

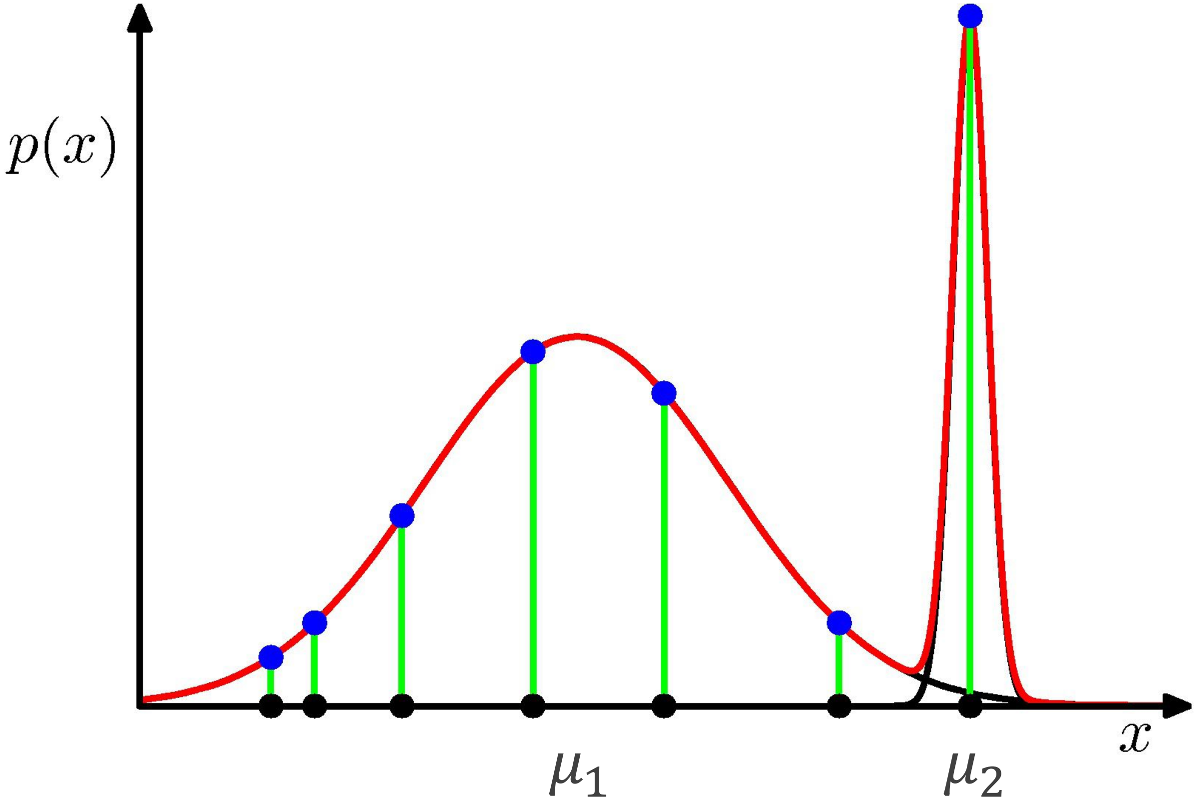
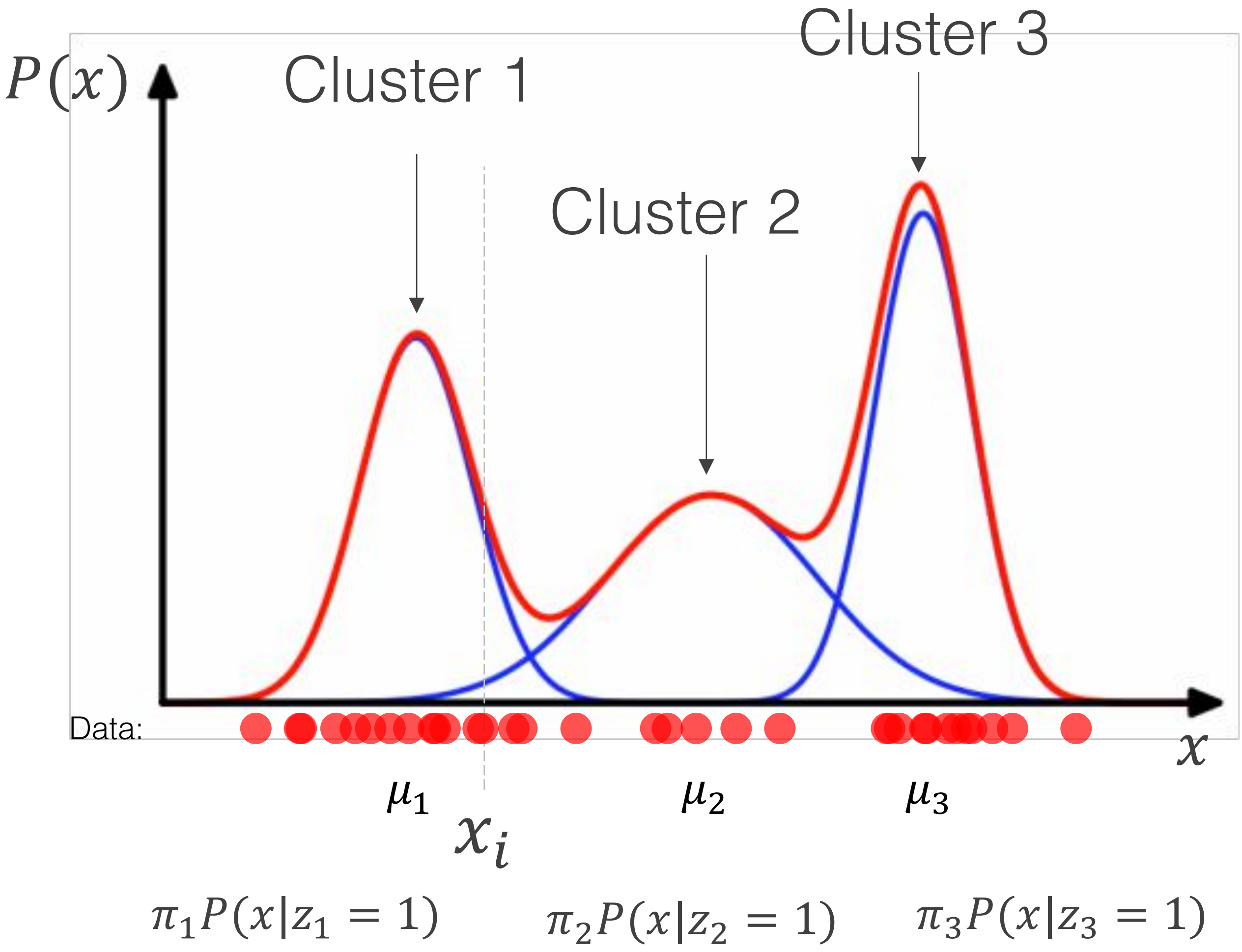


Image from Bishop, Pattern Recognition, 2006

How do we assign a cluster?

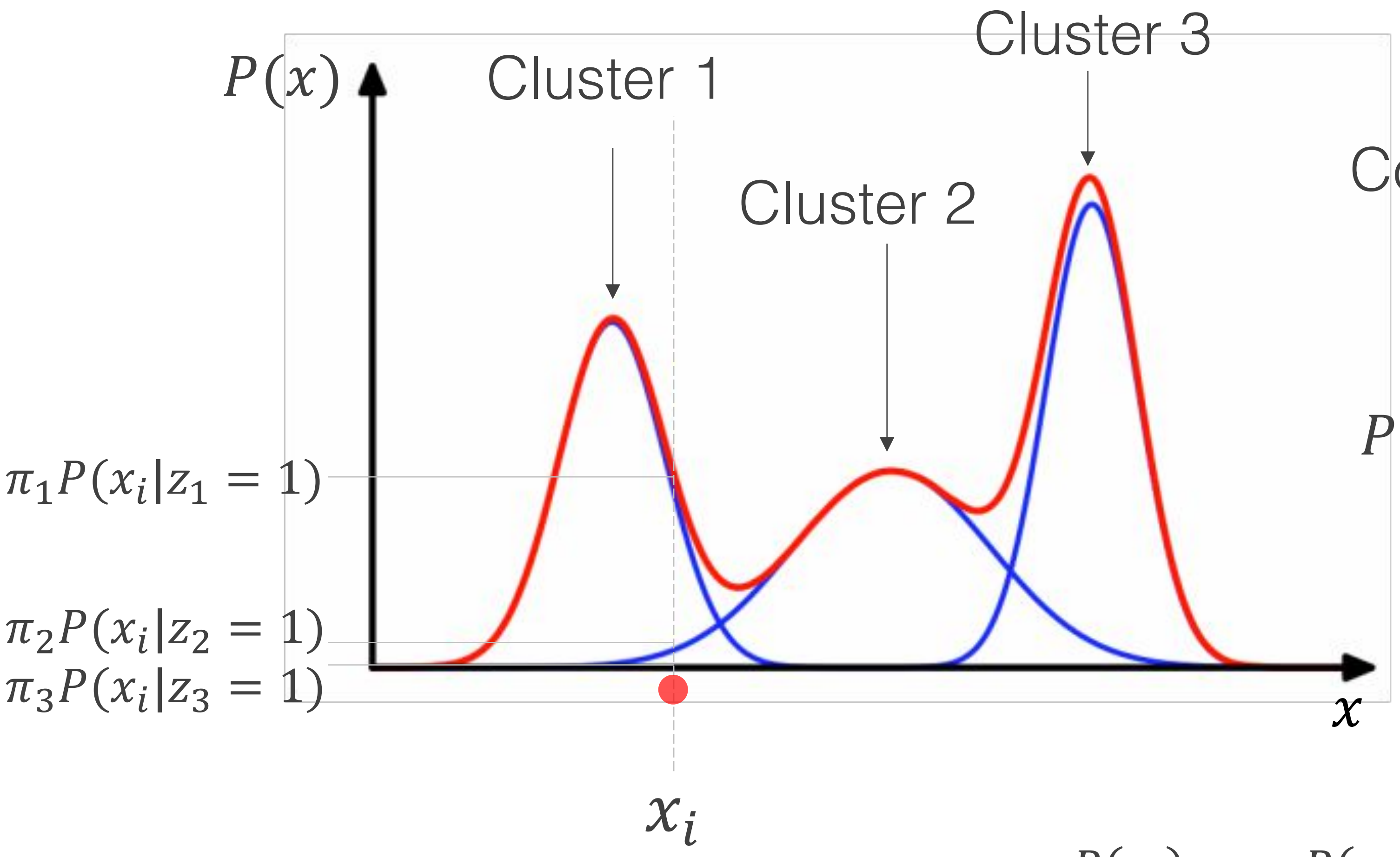


The probability of x_i is “explained” most by cluster 1, a little by cluster 2, and very little by cluster 3

We assign the cluster, z_k so that $P(z_k = 1|x)$ is the largest for all the k’s

We need an expression for: $P(z_k = 1|x)$

How do we assign a cluster?



Consider observation x_i

normal distribution for the kth cluster

π_k

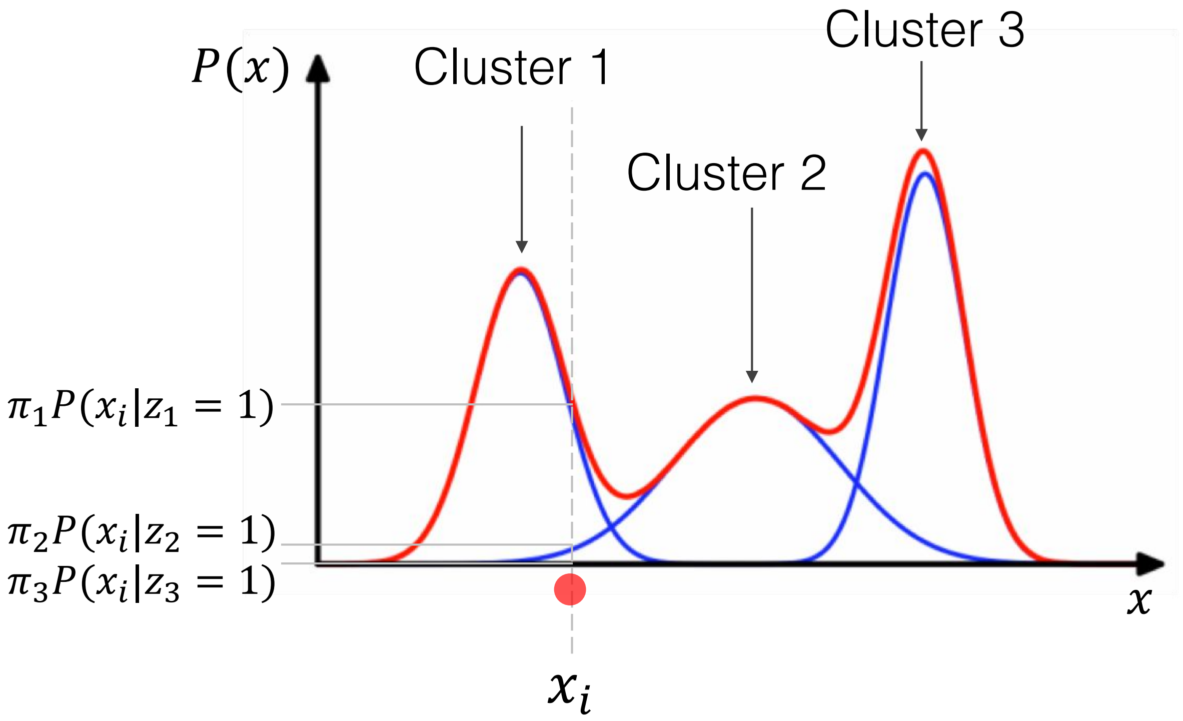
$$P(z_k = 1|x_i) = \frac{P(x_i|z_k = 1)P(z_k = 1)}{P(x_i)}$$

by Bayes' Rule

$$P(x_i) = \pi_1 P(x_i|z_1 = 1) + \pi_2 P(x_i|z_2 = 1) + \pi_3 P(x_i|z_3 = 1)$$

normalizes the probability, $P(z_k = 1|x_i)$, to add to one when summed over k

Posterior probabilities / “responsibilities”



Another interpretation of this quantity is what “fraction” of an observation is assigned to this cluster (“fuzzy” or ”soft” clustering)

$N(\boldsymbol{x}_i|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ π_k

$$r(z_{ik}) \triangleq P(z_k = 1|x_i) = \frac{P(x_i|z_k = 1)P(z_k = 1)}{\sum_{k=1}^{K} P(x_i|z_k = 1)P(z_k = 1)}$$

$$= \frac{\pi_k N(\boldsymbol{x}_i|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{k=1}^{K} \pi_k N(\boldsymbol{x}_i|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}$$

Define $N_k = \sum_{i=1}^{N} r(z_{ik})$

Expected number of samples per cluster

Expectation Maximization for a GMM

Note: EM is a general technique for finding maximum likelihood solutions for models with latent variables

Goal: maximize the log likelihood of the data given the model parameters:

$$\ln P(\boldsymbol{X}|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma}) = \sum_{i=1}^{N} \ln\left[\sum_{k=1}^{K} \pi_k N(\boldsymbol{x}_i|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)\right]$$

0. Initialization

Initialize all the parameters
(often K-means is used for this purpose)

1. Expectation-step

Calculate the “responsibilities” based on the model parameters

$$r(z_{ik}) \triangleq P(z_k = 1|x_i)$$

$$= \frac{\pi_k N(\boldsymbol{x}_i|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)}{\sum_{k=1}^{K} \pi_k N(\boldsymbol{x}_i|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)}$$

2. Maximization-step

Use the “responsibilities” to update the model parameters to maximize the log likelihood

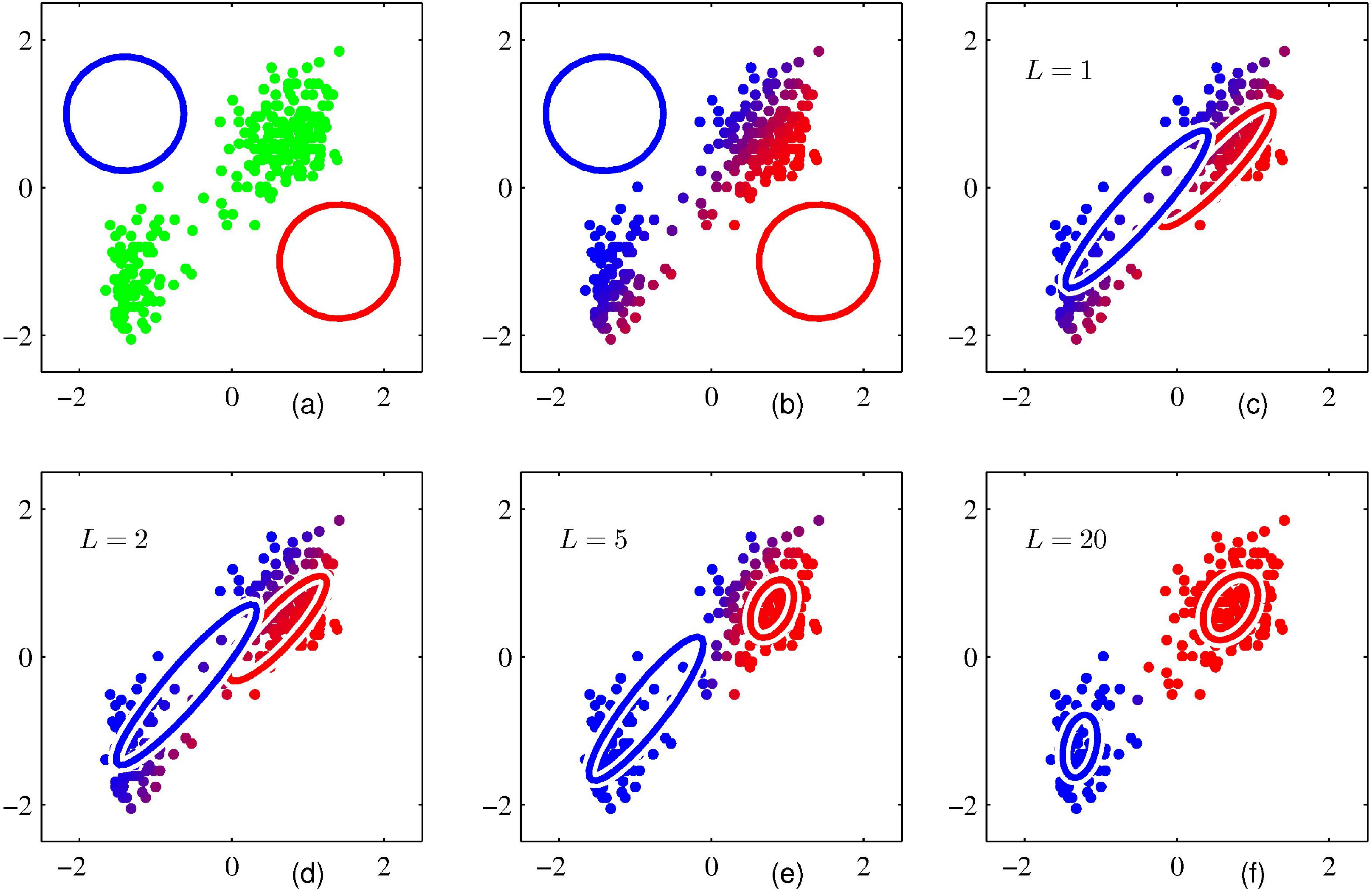
$$\boldsymbol{\mu}_k^{new} = \frac{1}{N_k}\sum_{i=1}^{N} r(z_{ik})\boldsymbol{x}_i$$

$$\boldsymbol{\Sigma}_k^{new} = \frac{1}{N_k}\sum_{i=1}^{N} r(z_{ik})(\boldsymbol{x}_i - \boldsymbol{\mu}_k^{new})(\boldsymbol{x}_i - \boldsymbol{\mu}_k^{new})^T$$

$$\pi_k^{new} = \frac{N_k}{N}$$

Where $N_k = \sum_{i=1}^{N} r(z_{ik})$

Expectation Maximization for GMM Example



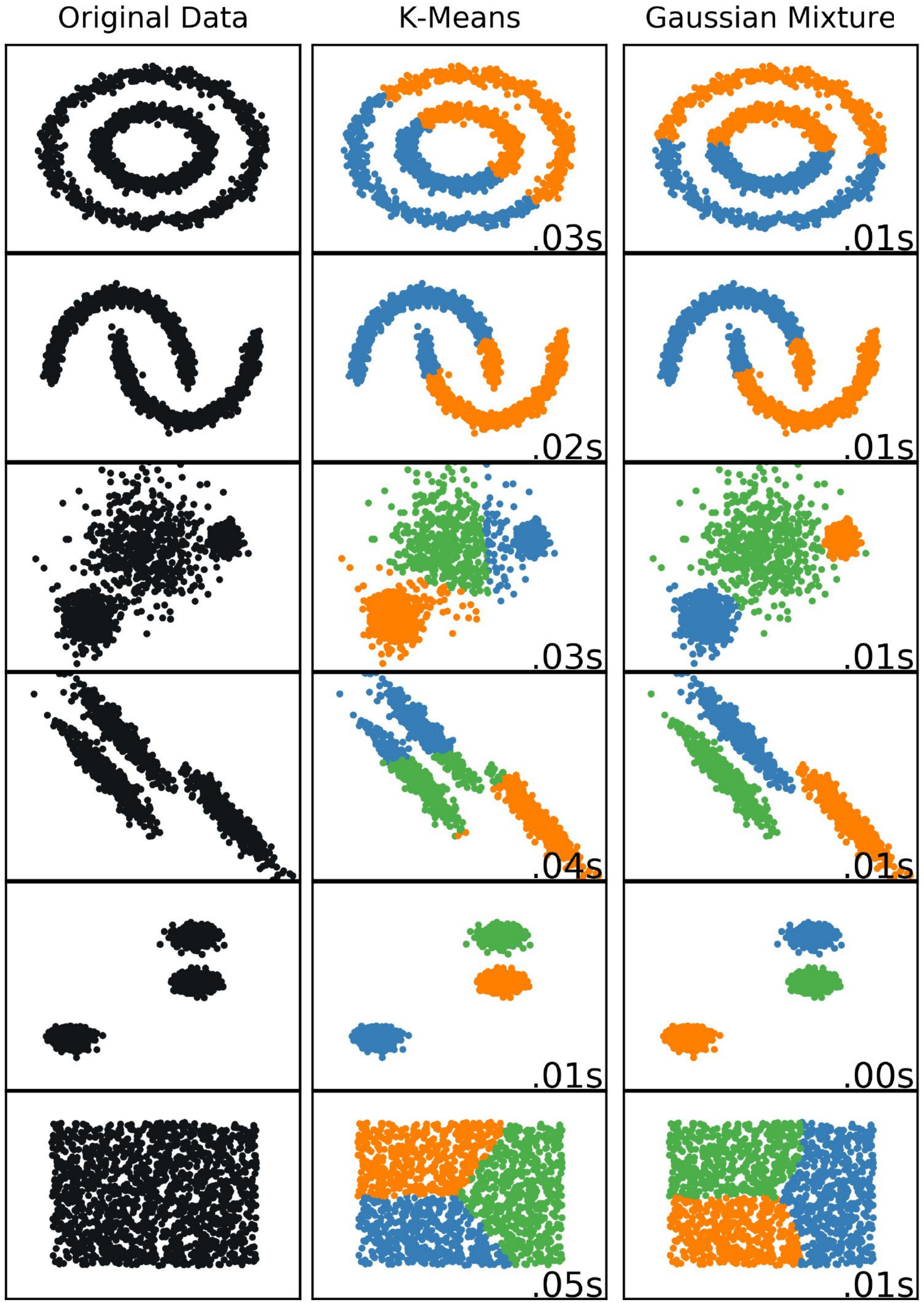
L = number of EM cycles

Image from Bishop, Pattern Recognition, 2006

Examples: GMM

Can produce soft clustering

Estimates the density / distribution of the data



Struggles when the clusters are not approximately Gaussian

Excels in situations with **variation in cluster variance** and **correlation between features**

Excels with clusters of **equal variance**

Will divide into k clusters even when there are not k

Gaussian Mixture Models

Generative models: model $P(\boldsymbol{X}|\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ are the model parameters

Very useful for density estimation

Produce hard or soft (fuzzy) clustering

When you restrict the covariance matrix to be diagonal and equal for all clusters, the GMM and K-means algorithm become the same

Expectation Maximization

Iterative method to find maximum likelihood parameter estimates when the model depends on unobserved latent variables, when this can’t be solved directly

The E-step updates the latent variable distribution estimates, so that we can calculate the likelihood function given the current parameter values

The M-step identifies the parameters that maximize the likelihood

Types of clustering algorithms

Methods

Centroid-based clustering (e.g. **K-Means**)
Distribution-based clustering (e.g. **Gaussian mixture model**)
Density-based clustering (e.g. DBSCAN)
Hierarchical clustering (e.g. agglomerative clustering)
Graph-based clustering (e.g. spectral clustering)

Cluster assignment

Hard clustering
Soft clustering (a.k.a. fuzzy clustering)