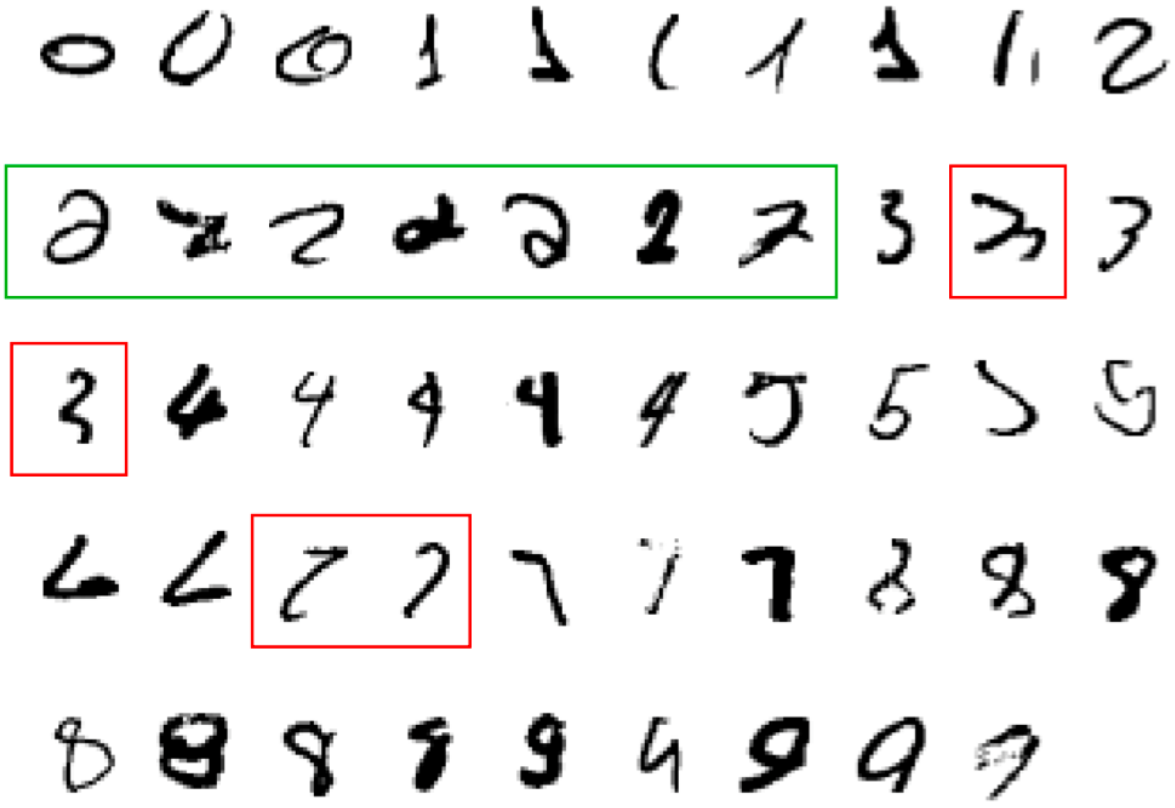
# Unsupervised Learning I: Clustering

# Example: MNIST hand written dataset



# Clustering

## Clustering

- We assume that the data was generated from a number of different classes. The aim is to cluster data from the same class together.
  - How do we decide the number of classes?
  - Why not put each datapoint into a separate class?
- What is the objective function that is optimized by "sensible" clusterings?

# Clustering algorithms

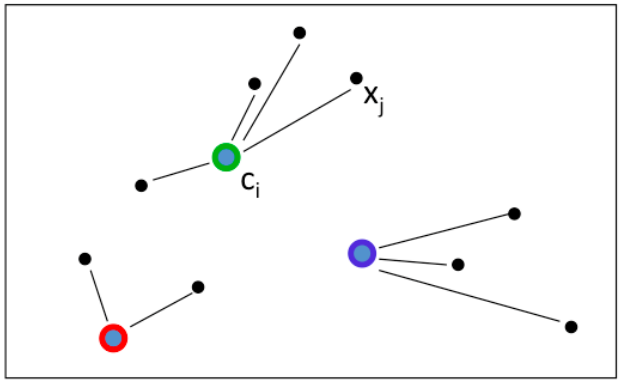
- Centroid models (K-Means) represents each cluster by a single vector.
- Connectivity models (Hierarchical Clustering) builds models based on distance connectivity between data points.
- Distribution models (Mixtures of Gaussian) fits the data using statistical distributions (e.g. one Gaussian per clustering) using the Expectation-Maximisation algorithm.

# Unsupervised Learning I: Clustering: K-Means

# K-Means Algorithm

K-means (MacQueen, 1967) is one of the simplest unsupervised learning algorithms that solve the well known clustering problem.

- Assume the data lives in a Euclidean space.
- Assume we want K classes.
- We start with randomly initialized cluster centers $\mathbf{c}_k$




Minimize the squared distance:

$$E = \sum_{j=1}^{M} \sum_{i=1}^{K} p_{ji} ||\mathbf{x}_i - \mathbf{c}_i||^2$$

Responsibility/Assignment variable $p_{ji}$ (this is what we want to learn)

$p_{ji} = 1$, if $\mathbf{x}_j$ assigned to cluster i

$p_{ji} = 0$, else

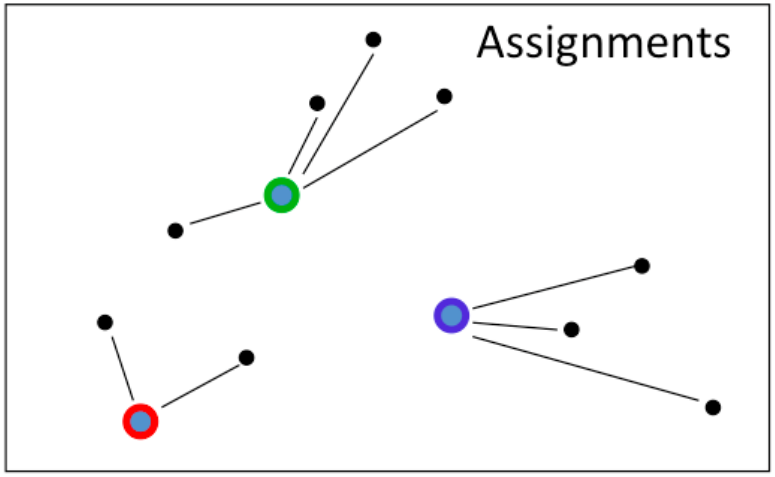
- Assume the data lives in a Euclidean space.
- Assume we want K classes.
- We start with randomly initialized cluster centers $\mathbf{c}_k$

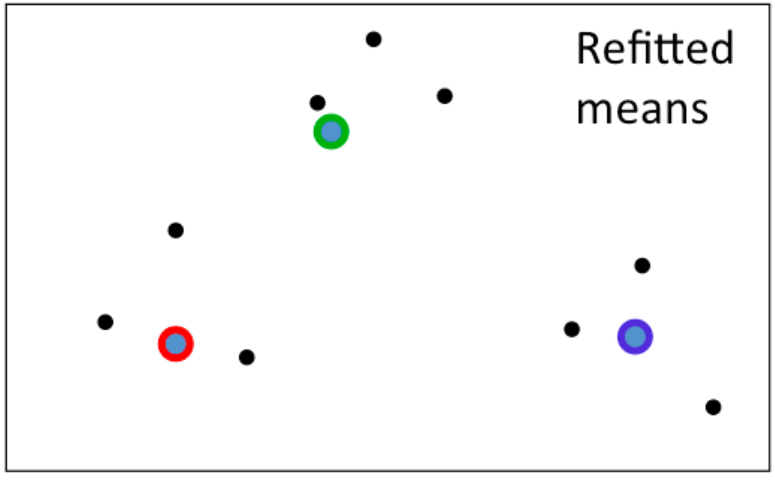
The algorithm alternates between two steps

Assignment step: Assign each data point to the closest cluster $c_k$.

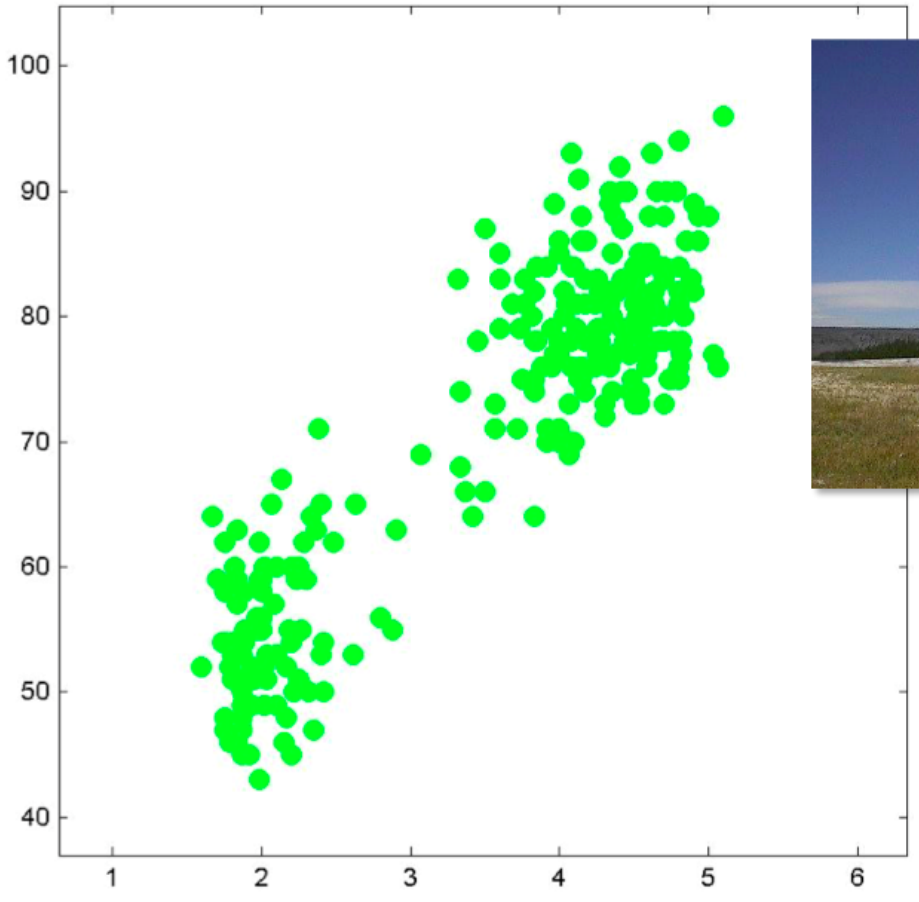
Refitting step: Move each cluster center $c_i$ to the center of gravity of the data assigned to it:

$$\mathbf{c}_k \leftarrow \frac{1}{N_k} \sum_{i \in C_k} \mathbf{x}_i$$



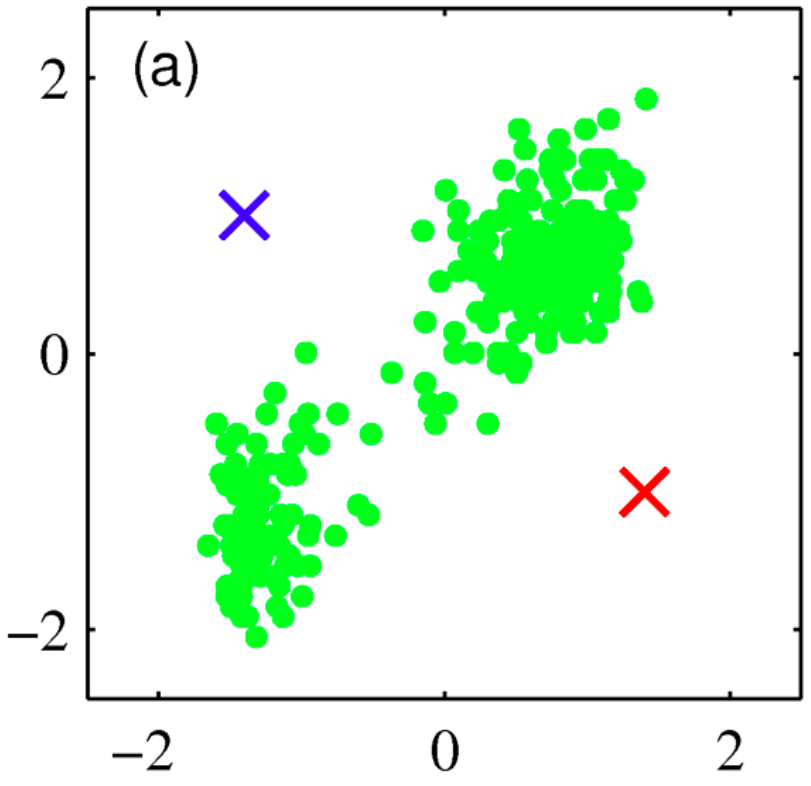



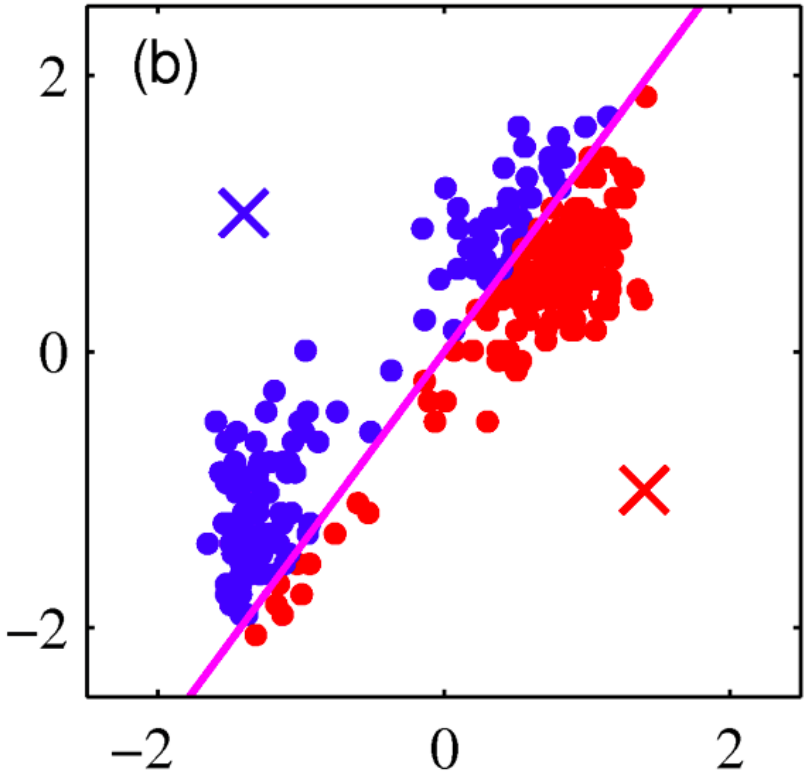
Time between eruptions (minutes)

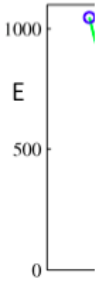


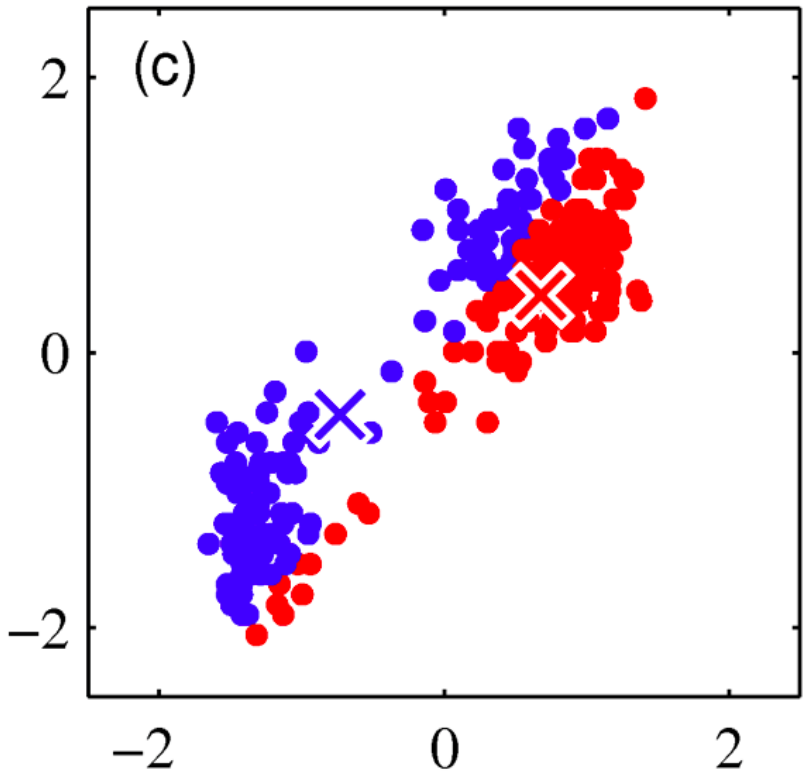

Duration of eruption (minutes)

Old Faithful Geysir,
Yellowstone National Park
(Wyoming, USA)

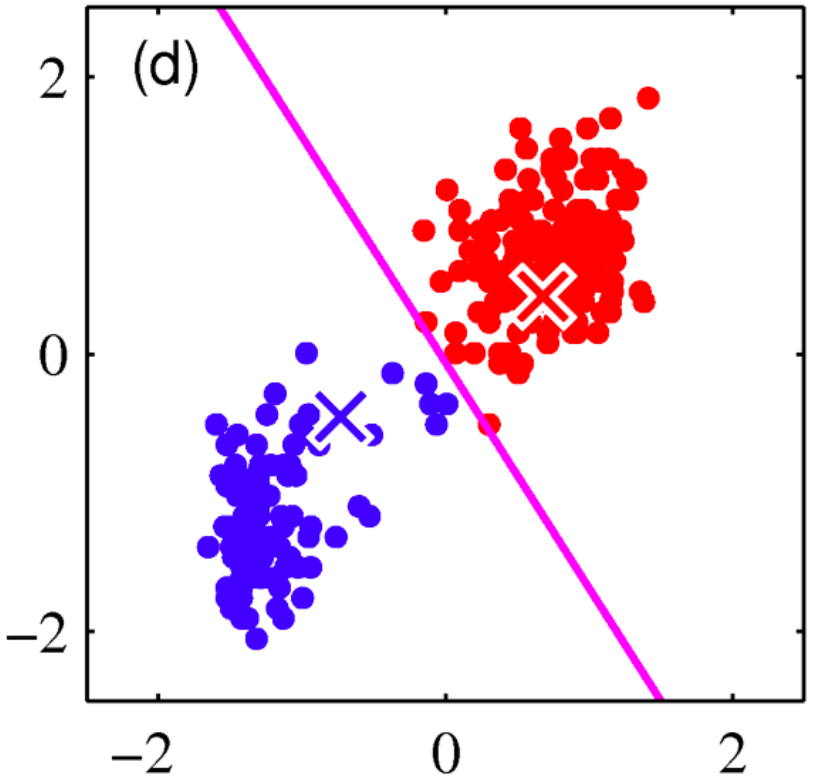

(a)
2
0
−2
−2
0
2

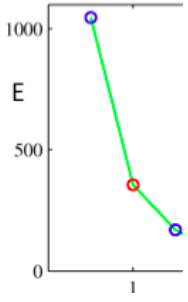

(b)
2
0
−2
−2
0
2

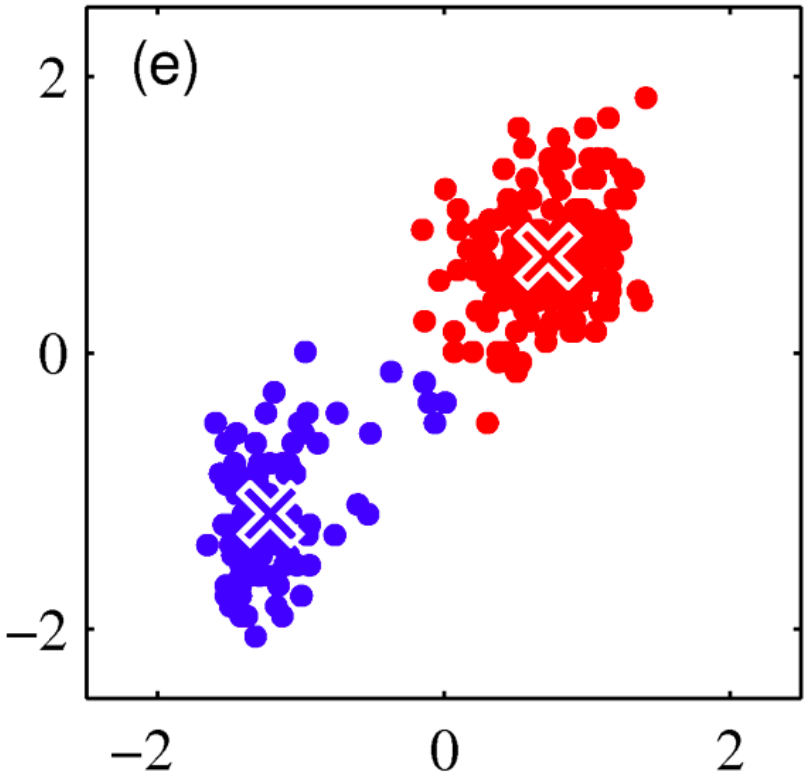

1000
E
500
0

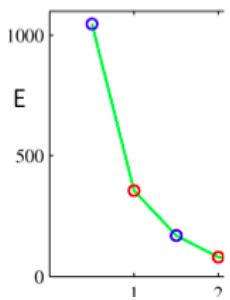

(c)
2
0
−2
−2
0
2

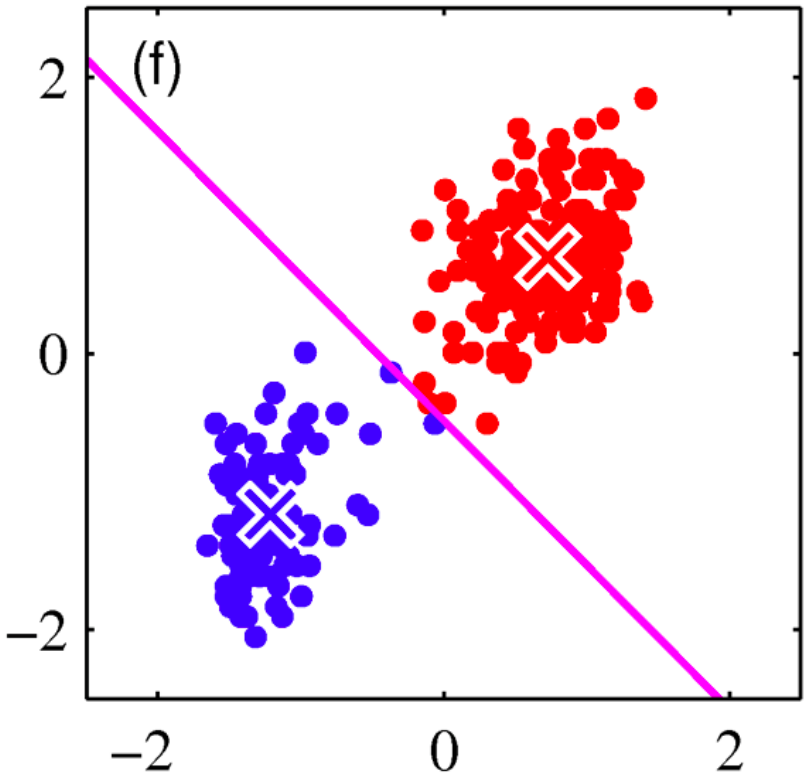

1000
E
500
0
1

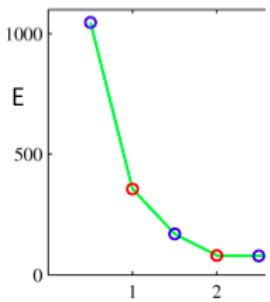

(d)
2
0
−2
−2
0
2

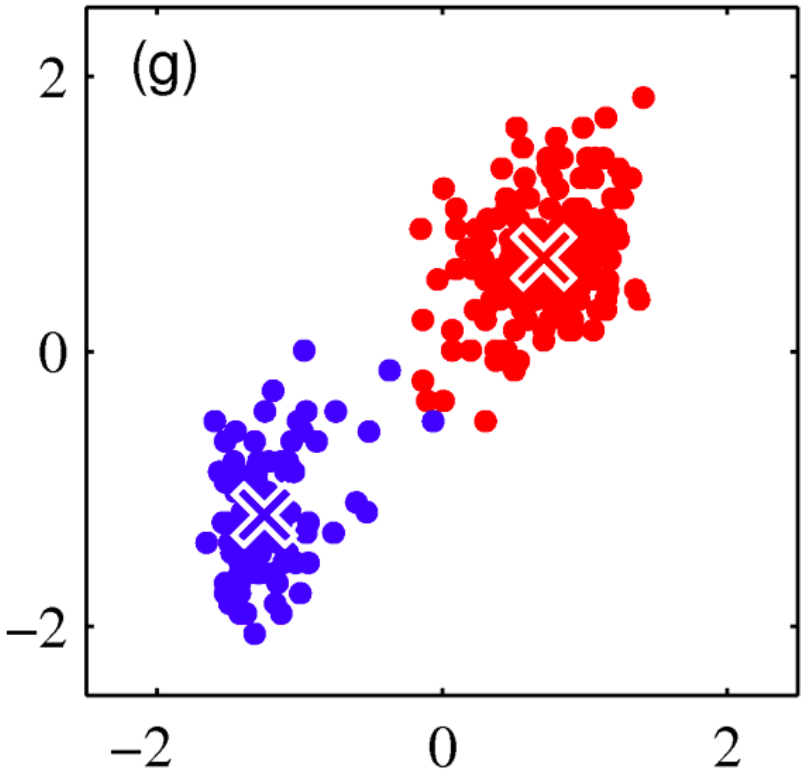

1000
E
500
0
1

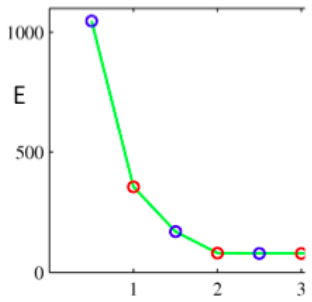

(e)
2
0
−2
−2
0
2

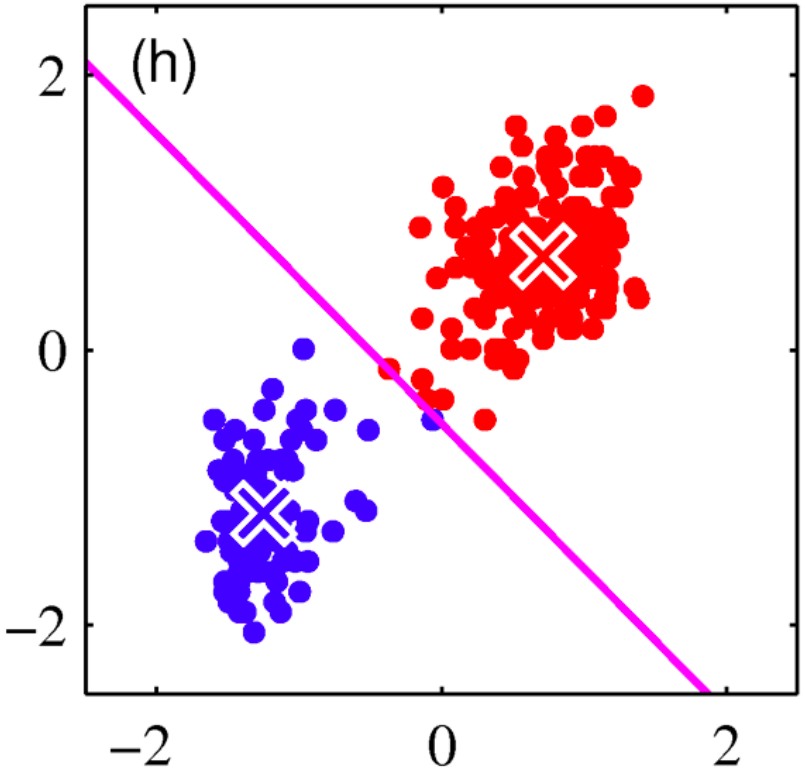

1000
E
500
0
1
2

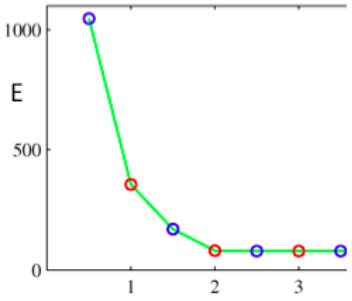

(f)
2
0
−2
−2
0
2

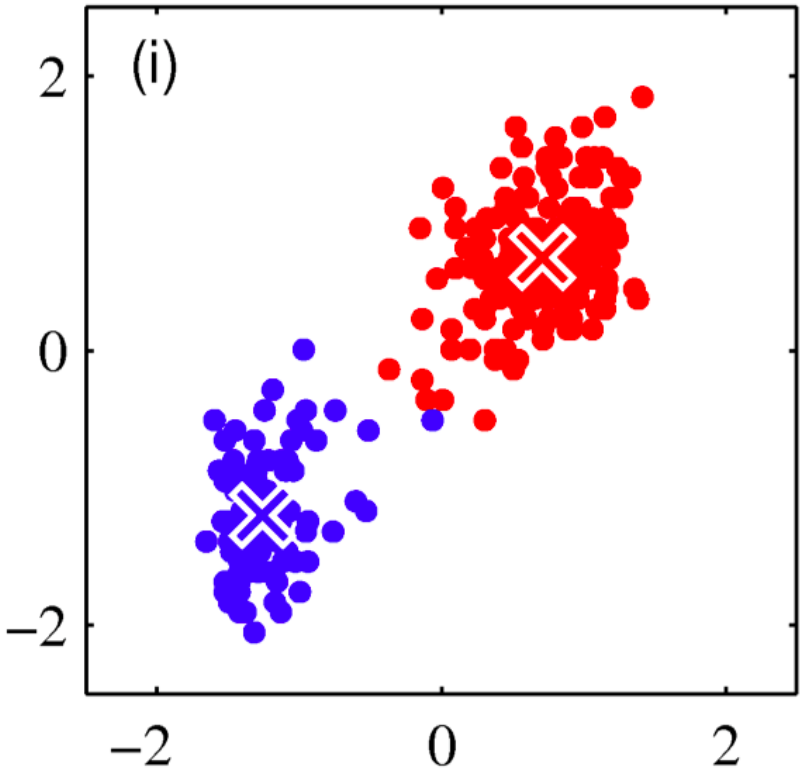

1000
E
500
0
1
2

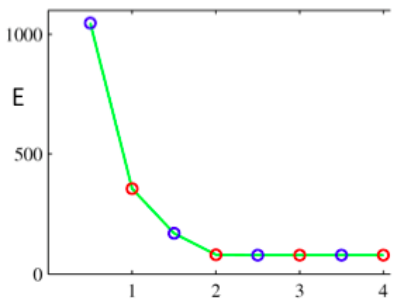

(g)
2
0
−2
−2
0
2


1000
E
500
0
1
2
3


(h)
2
0
−2
−2
0
2


1000
E
500
0
1
2
3


(i)
2
0
−2
−2
0
2


1000
E
500
0
1
2
3
4

# Convergence of K-Means

- Assign Step:

  Whenever an assignment is changed, the sum squared distances of data points from their assigned cluster centers is reduced.

- Refitting Step:

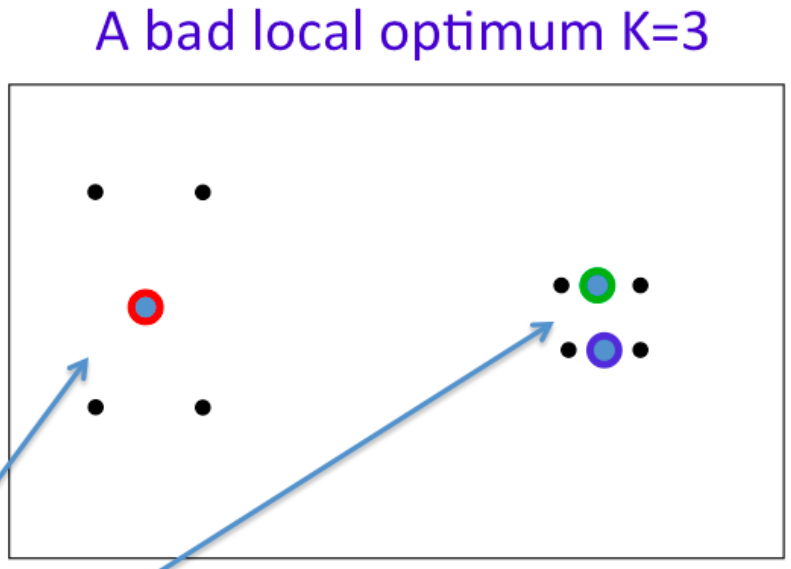
  Whenever a cluster center is moved the sum squared distances of the data points from their currently assigned cluster centers is reduced.

E is **always** reduced on each step.

- Test for convergence: If the assignments $p_{ji}$ do not change in the assignment step, we have converged.

# Local minima and hacks

- There is nothing to prevent k-means getting stuck at local minima.
- We could try many random initializations for the $\mathbf{c}_i$
- We could try non-local split-and-merge moves: Simultaneously split a big cluster into two and merge two nearby clusters and.



# k-means as a means for lossy data compression

This is also an illustration of image segmentation

Original image

$K = 10$

$K = 3$

$K = 2$

Each pixel in this 240x180 image has 3 color dimensions (red,green,blue).
Each color dimension has 8 bit precision (values 0-255).
Each pixel is a 3D color data point, all pixels in the image are the training data set.

By how many bits can we compress this image using?

# Limitations of K-means

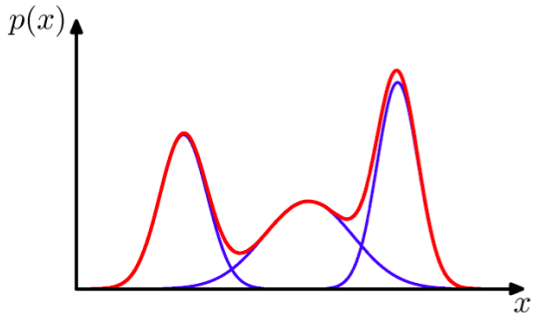
- Hard assignments of data points to clusters, so small shift of a data point can flip it to a different cluster
- Not clear how to choose the value of $K$

# Unsupervised Learning I: Clustering: Mixtures of Gaussian

# A generative view of clustering

We need a sensible way to think of what it means to cluster the data well:

- This makes it possible to judge different methods.
- It may make it possible to decide on the number of clusters
- An obvious approach is to imagine that the data was produced by a generative model.
- Learning becomes adjusting model parameters to maximise the posterior probability that it produced the data.

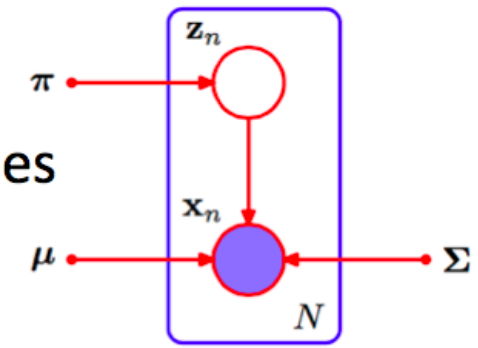



This content is covered in Bishop's book in Sections 2.3.9,9.2-9.2.2

# Mixtures of Gaussians

- Binary latent variables $\mathbf{z} = \{z_{kn}\}$ describing which component *k* generated which data point $\mathbf{x}_n$
- Conditional distribution of observed variable

$$p(\mathbf{x}|\mathbf{z}) = \prod_{k=1}^{K} \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \Sigma_k)^{z_k}$$




- Prior distribution of latent variables

$$p(\mathbf{z}) = \prod_{k=1}^{K} \pi_k^{z_k}$$

- Marginalizing over the latent variables we obtain

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \Sigma_k)$$

These are all conditioned on $\theta = \{\pi_k, \mu_k, \Sigma_k\}_K$

- In order to learn the parameters, we must first solve the inference problem: Which Gaussian generated each datapoint?
- We cannot be sure, so it is a distribution over all possibilities.
- Use Bayes theorem to get posterior probabilities

To compute the solution we will use the Expectation-Maximisation algorithm:

- E-step: Compute the posterior probability that each Gaussian generates (with current parameters) each data point.
- M-step: Assuming that the data really was generated this way, change the parameters of each Gaussian to maximize the probability that it would generate the data it is currently responsible for.

# E-Step

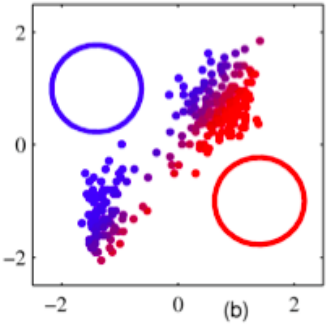
Posterior for Gaussian k

Prior for Gaussian k

Likelihood of data to come Gaussian k

$$p(k \mid \mathbf{x}_i) = \frac{p(k)p(\mathbf{x}_i \mid k)}{p(\mathbf{x}_i)}$$

Bayes' theorem




$$p(\mathbf{x}_i) = \sum_{j \in \text{All Clusters}} p(k)p(\mathbf{x}_i \mid k)$$

$$p(k) = \pi_k$$ ⟸ Mixing proportion

$$p(\mathbf{x}_i \mid k) = N(\mathbf{x}_i; c_k, \Sigma_k)$$

Responsibility that cluster k generated data i

# M-Step I

Each Gaussian gets a certain amount of posterior probability for each data point.

The optimal mixing proportion to use (given these posterior probabilities) is just the fraction of the data that the Gaussian gets responsibility for.

Posterior for Gaussian k

Data for training case i

$$\pi_k^{new} \leftarrow \frac{\sum_{i=1}^{M} p(k \mid \mathbf{x}_i)}{M}$$

Number of training cases

# M-Step II

We just take the center-of gravity of the data that the Gaussian is responsible for.

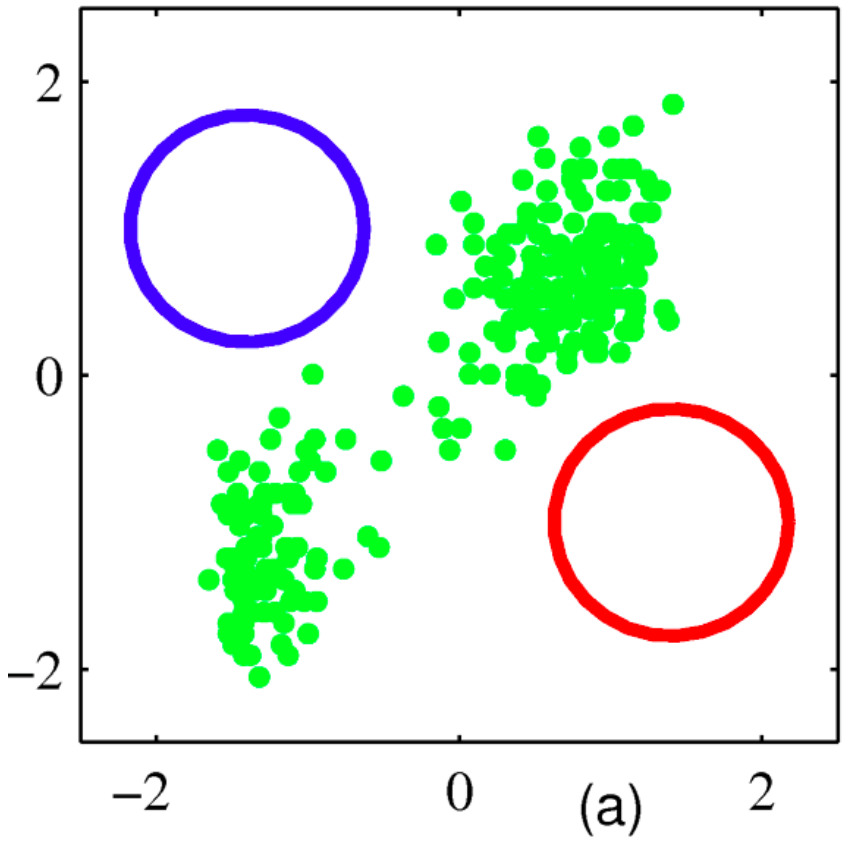
- Just like in K-means, except the data is weighted by the posterior probability of the Gaussian.
- Guaranteed to lie in the convex hull of the data
- Could be big initial jump

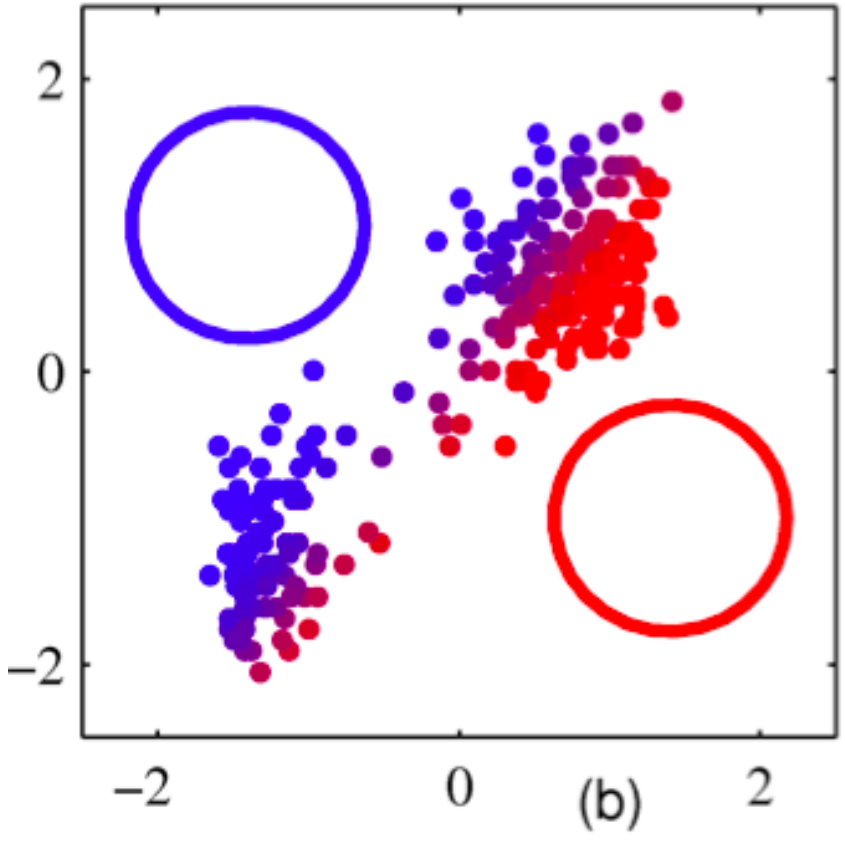
$$c_k^{new} \leftarrow \frac{\sum_{i \in \text{All Data}} p(k \mid \mathbf{x}_i)\, \mathbf{x}_i}{\sum_{i \in \text{All Data}} p(k \mid \mathbf{x}_i)}$$

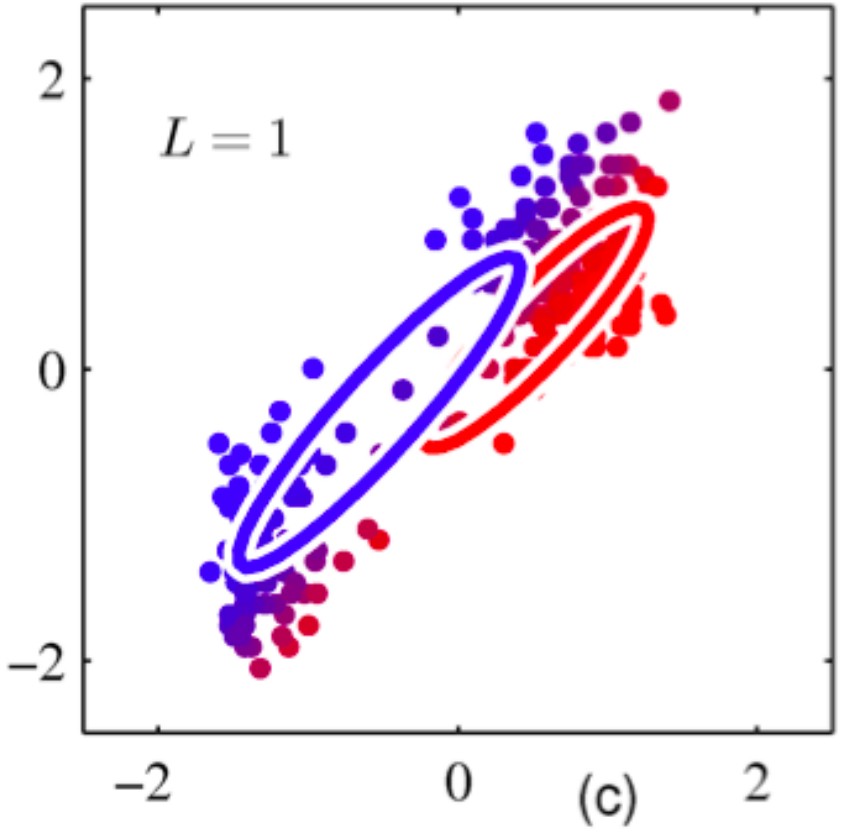
In the following we will use the cluster center $\mathbf{c}_k$ interchangeably with the mean of the Gaussian $\boldsymbol{\mu}_k$

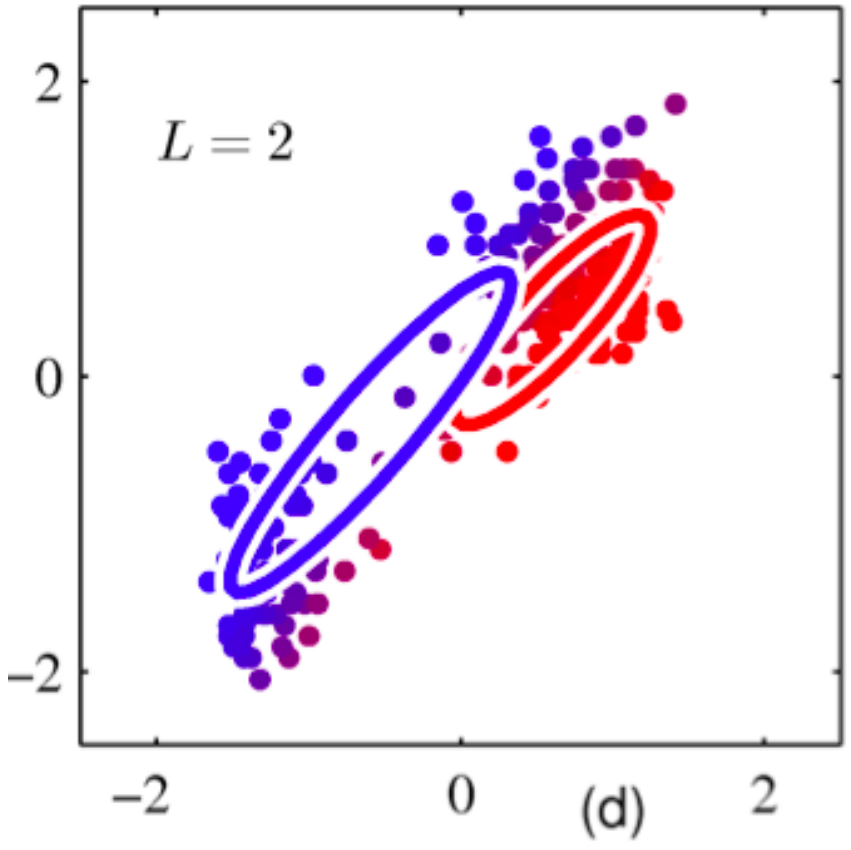
We fit the covariance matrix of each Gaussian k

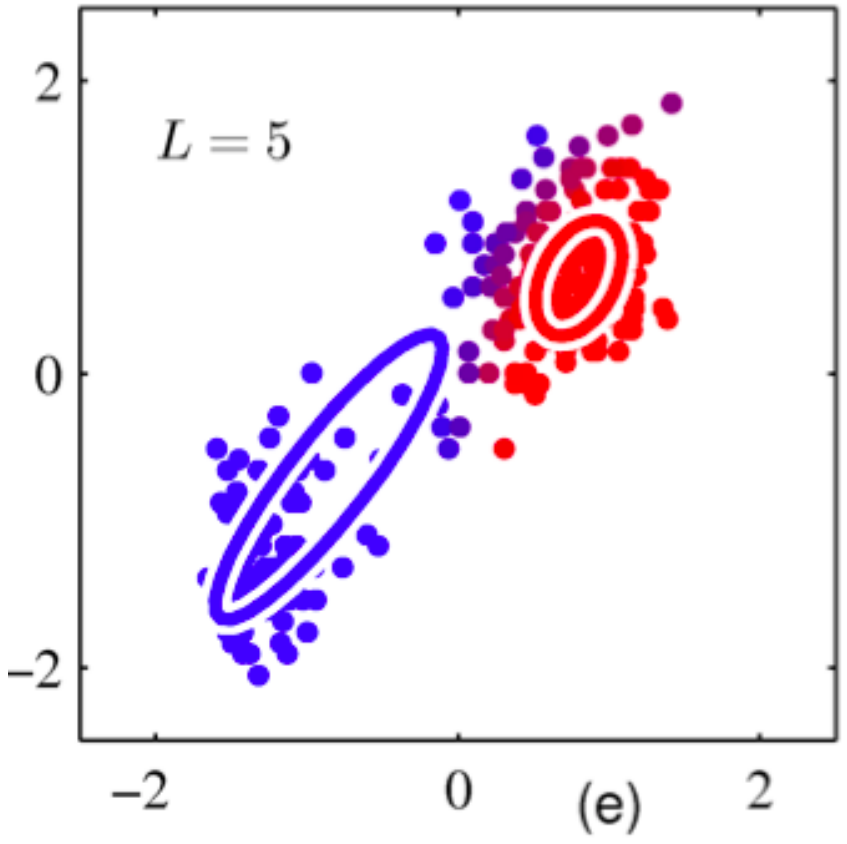
$$\boldsymbol{\Sigma}_k \leftarrow \frac{\sum_{i=1}^{M} p(k|\mathbf{x}_i)(\mathbf{x}_i - \mathbf{c}_k)(\mathbf{x}_i - \mathbf{c}_k)^T}{\sum_{i=1}^{M} p(k|\mathbf{x}_i)}$$

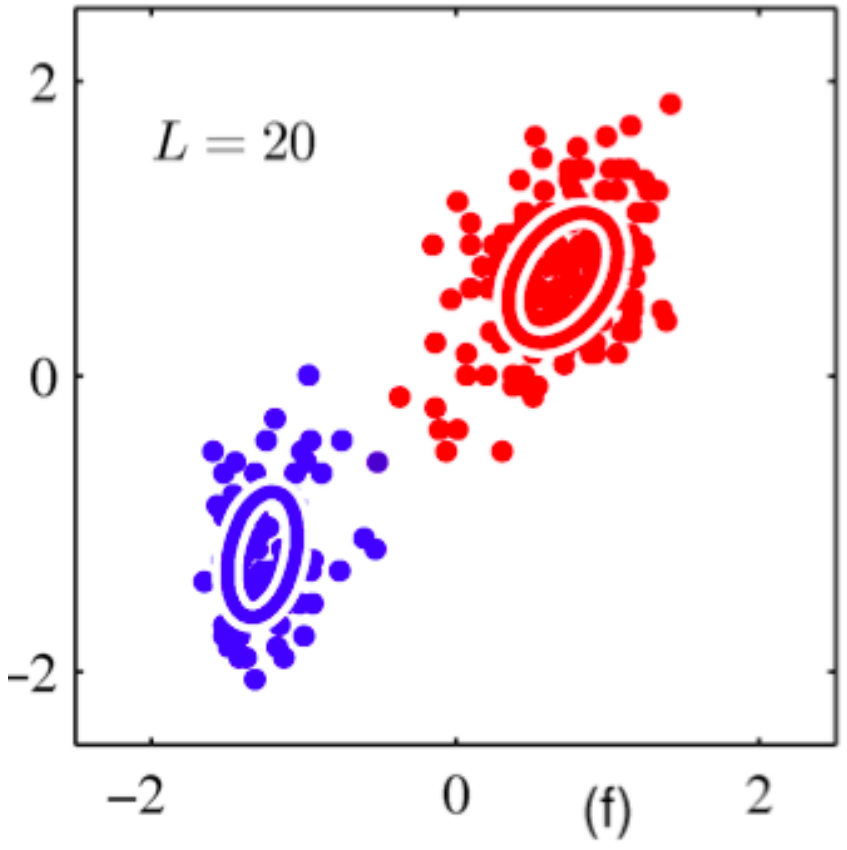

2
0
−2
−2
0
2
(a)


2
0
−2
−2
0
2
(b)


$L = 1$
2
0
−2
−2
0
2
(c)


$L = 2$
2
0
−2
−2
0
2
(d)


$L = 5$
2
0
−2
−2
0
2
(e)


$L = 20$
2
0
−2
−2
0
2
(f)

# How do we know that the EM updates improve things?

- Updating each Gaussian definitely improves the probability of generating the data if we generate it from the same Gaussians after the parameter updates.
- But we know that the posterior will change after updating the parameters.
- A good way to show that this is OK is to show that there is a single function that is improved by both the E-step and the M-step.
- This function is called Free Energy Q, one can prove (beyond the course) that EM will always improve this function.

# Unsupervised Learning I: Clustering: Information Theory [Option A]

# Information Theory I

Why is it that we can understand this sentence uttered in a loud room "Bleeze Turn phat mufic down."
How can I ensure that I achieve reliable communication over unreliable channels?

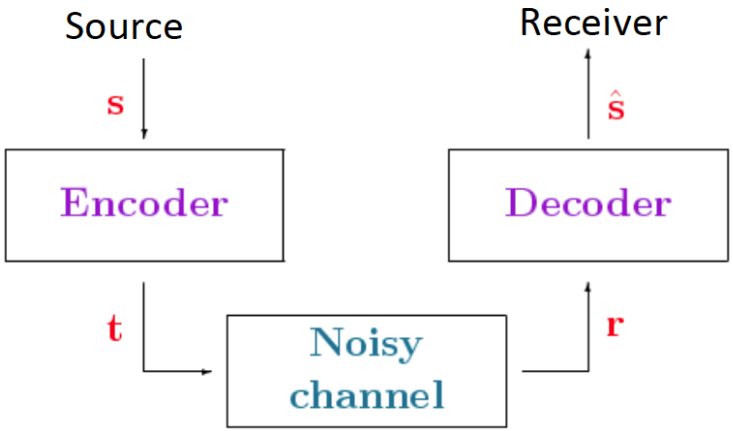
- Redundancy can improve communication reliability.
- Loss-less compression allows me to transmit the same amount of information with fewer signals by reducing reduandancy.
- Lossy compression allows me to convey the same message with less information (clustering).

The theories behind this are

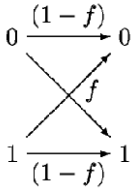
- Information Theory: How much information can I receive over a noisy channel?
- Coding Theory: How should I encode and decode signals so I reliably receive information?

These theories were invented by Claude Shannon at Bell Labs in 1948 in a single paper, including the unit of information: bit.

# Information Theory: Encode–Channel–Decoder I



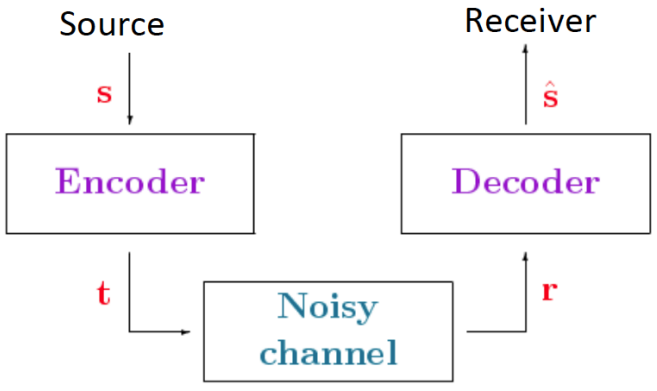




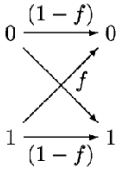


f = 0.1

# Information Theory: Clustering is lossy coding I









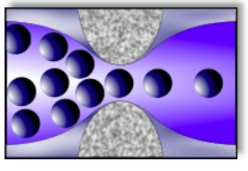
Coding theory

Add redundancy

f = 0.1

Do inference

Compression



# Entropy: Definition I

$$\mathrm{H}[x] = -\sum_{x} p(x) \log_2 p(x)$$

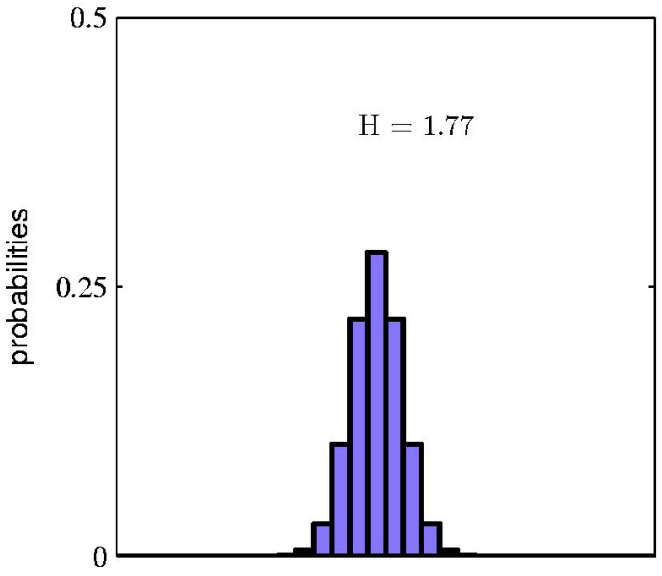
Unit "bits" (8 bits = 1 byte, 1Kb=$2^{10}$=1024 bits).

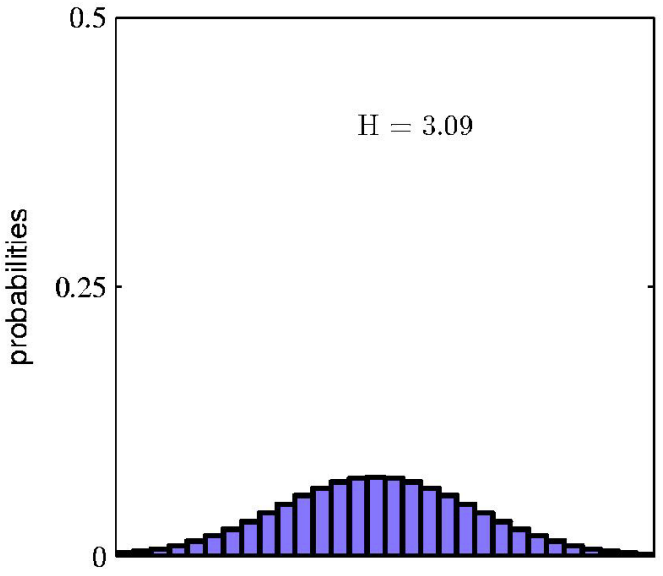
Q: How many yes-no questions to ask to know *x*?
A: H(*x*)

Important quantity in

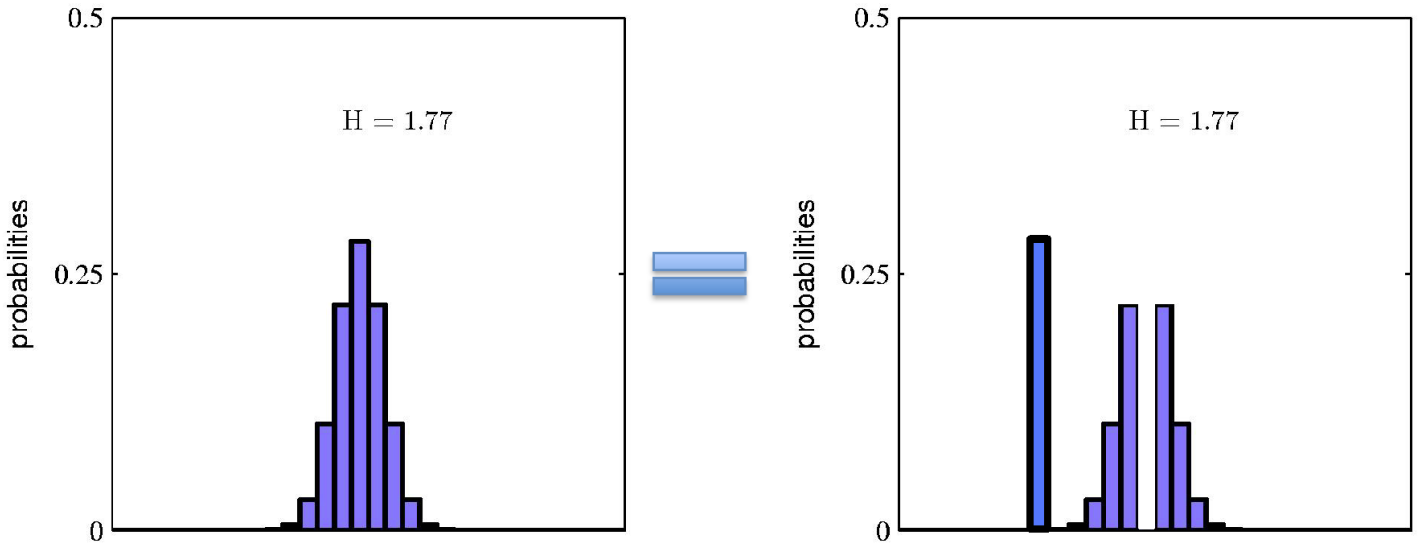
- coding theory
- statistical physics
- machine learning

# Diffuse distributions have more entropy I

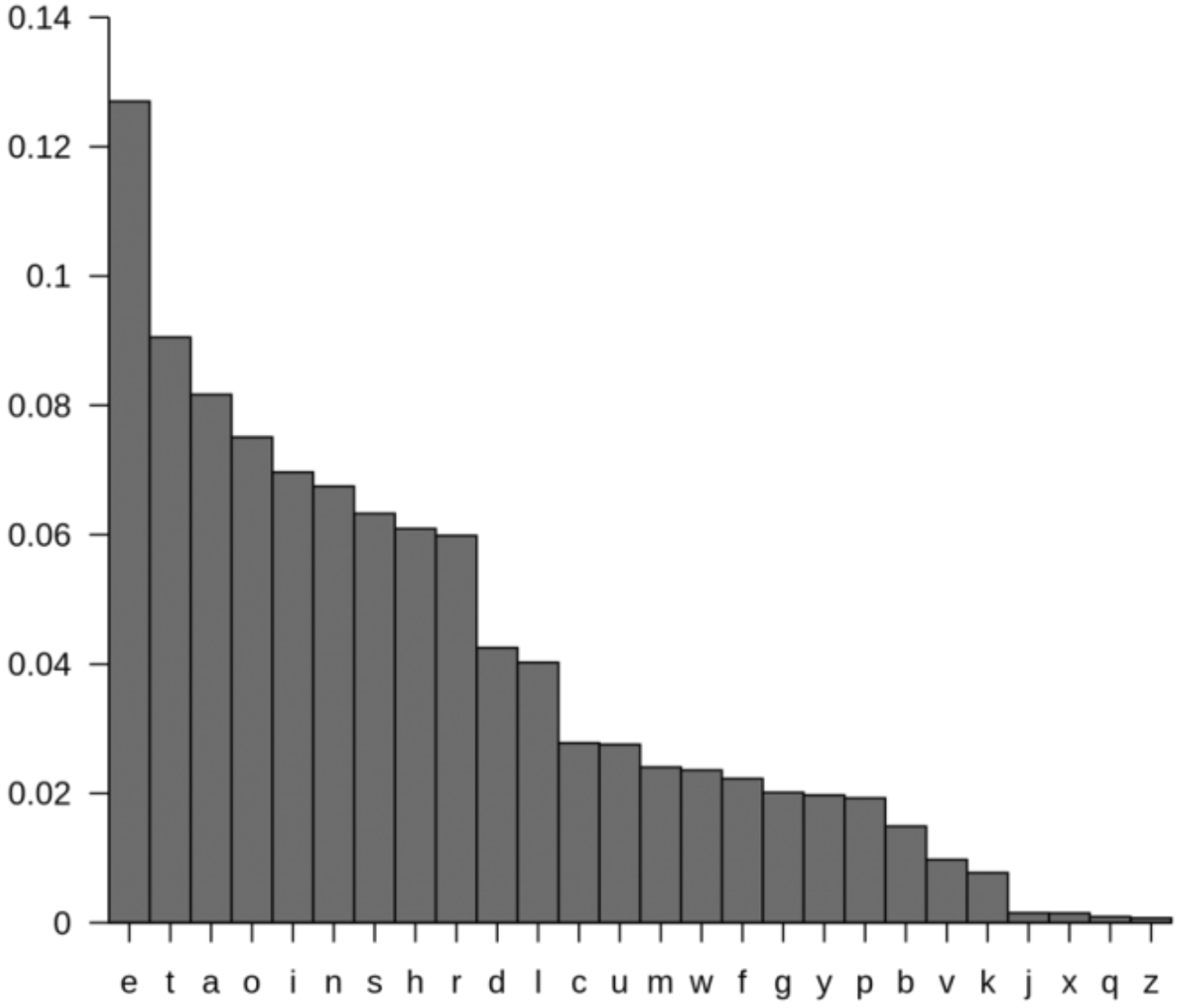


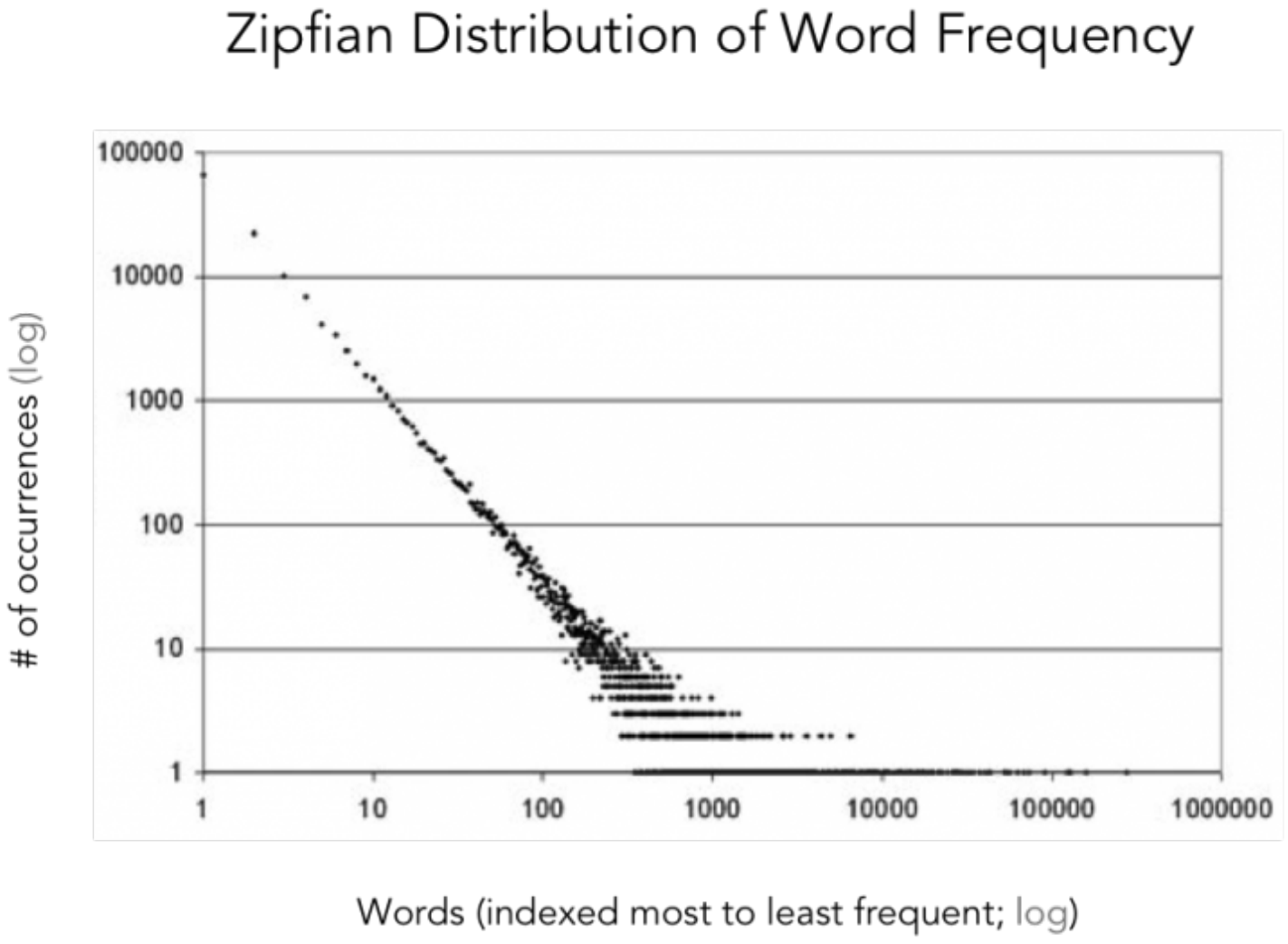
# ... but the shape is actually not so important I



# Entropy of letters in English language: $\approx 4 - 5$bits I



# Entropy of words in English language: $\approx 11 - 12$ bits I



# Coding Theory I

- Coding theory: x discrete with 8 possible states; how many bits to transmit the state of x?

- All states equally likely

$$\mathrm{H}[x] = -8 \times \frac{1}{8} \log_2 \frac{1}{8} = 3 \text{ bits.}$$

- Entropy is maximised for uniform distribution.

# Coding Theory: Arithmetic Coding I

| $x$ | a | b | c | d | e | f | g | h |
|---|---|---|---|---|---|---|---|---|
| $p(x)$ | $\frac{1}{2}$ | $\frac{1}{4}$ | $\frac{1}{8}$ | $\frac{1}{16}$ | $\frac{1}{64}$ | $\frac{1}{64}$ | $\frac{1}{64}$ | $\frac{1}{64}$ |
| code | 0 | 10 | 110 | 1110 | 111100 | 111101 | 111110 | 111111 |

This is an example of arithmetic coding.
Is this a good code?

$$\begin{aligned} \mathrm{H}[x] &= -\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{4}\log_2\frac{1}{4} - \frac{1}{8}\log_2\frac{1}{8} - \frac{1}{16}\log_2\frac{1}{16} - \frac{4}{64}\log_2\frac{1}{64} \\ &= 2 \text{ bits} \end{aligned}$$

$$\begin{aligned} \text{average code length} &= \frac{1}{2}\times 1 + \frac{1}{4}\times 2 + \frac{1}{8}\times 3 + \frac{1}{16}\times 4 + 4\times\frac{1}{64}\times 6 \\ &= 2 \text{ bits} \end{aligned}$$

# Differential Entropy I

We can define in addition two other quantities:
The conditional entropy $H(y|x)$, the joint entropy $H(x, y)$:

$$H(y|x) = -\sum_x {}_y p(x, y) \log_2 p(y|x)$$

$$H(x, y) = -\sum_x \sum_y p(x, y) \log_2 p(x, y)$$

This yields two equation linking the entropies that looks like a log-version of Bayes theorem:

$$H(x, y) = H(x|y) + H(y) = H(y|x) + H(x)$$

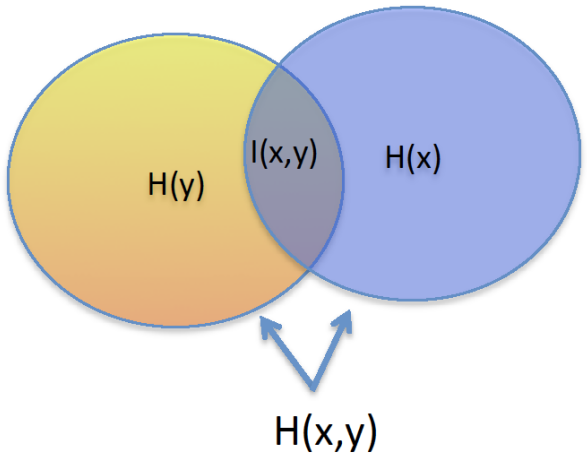
# Mutual Information I(x,y) I

How much can x tell us about y and vice versa?

$$\mathrm{I}[\mathbf{x},\mathbf{y}] = \mathrm{H}[\mathbf{x}] - \mathrm{H}[\mathbf{x}|\mathbf{y}] = \mathrm{H}[\mathbf{y}] - \mathrm{H}[\mathbf{y}|\mathbf{x}] = H(x) + H(y) - H(x,y)$$

This is how much x tells us

This is how much x tells us that we know from y already

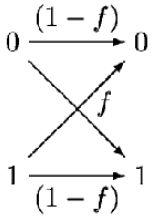



It measures how much information is flowing from source to receiver. The semantics of the message have disappeared. The only think that counts is the probability of the messages. **"Bits" are universal currency.**

Very useful to compare, say information flow in Telegraph vs Broadband vs Neurons...

$$P(X) = \left\{\frac{1}{2}, \frac{1}{2}, 0\right\}$$
$$P(X|Y=0) = \{1-p, p, 0\}$$
$$P(X|Y=1) = \{p, 1-p, 0\}$$


Source



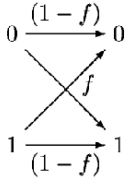

f = 0.1


Receiver

What is the mutual information between source and receiver?

$$
\begin{aligned}
P(X) &= \left\{\frac{1}{2}, \frac{1}{2}, 0\right\} \\
P(X|Y=0) &= \{1-p, p, 0\} \\
P(X|Y=1) &= \{p, 1-p, 0\}
\end{aligned}
$$


Source

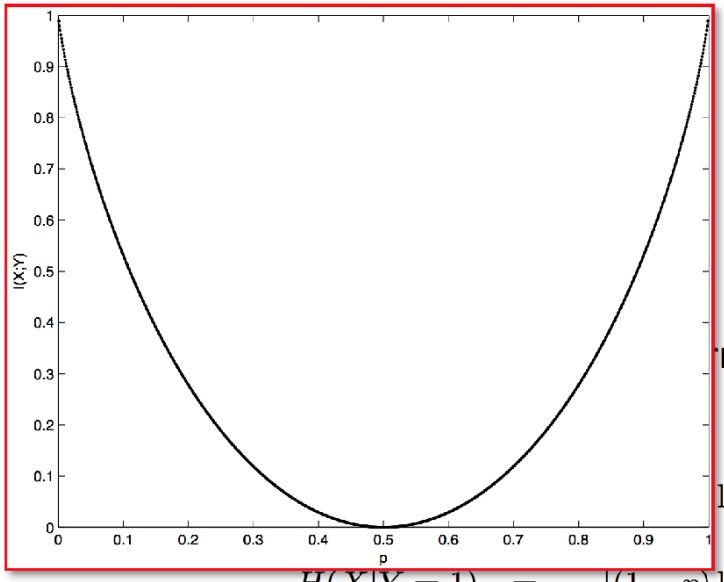



f = 0.1


Receiver

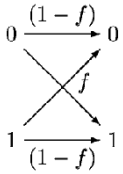
What is the mutual information between source and receiver?

$$
\begin{aligned}
H(X) &= 1 \\
H(X|Y=0) &= -\left[(1-p)\log_2(1-p) + p\log_2(p)\right] = (1-p)\log_2\frac{1}{1-p} + p\log_2\frac{1}{p} \\
H(X|Y=1) &= -\left[(1-p)\log_2(1-p) + p\log_2(p)\right] = (1-p)\log_2\frac{1}{1-p} + p\log_2\frac{1}{p} \\
I(X;Y) &= H(X) - \left[P(Y=0)H(X|Y=0) + P(Y=1)H(Y=1)\right] \\
I(X;Y) &= 1 - \left[\left(\frac{1}{2}\right)\left((1-p)\log_2\frac{1}{1-p} + p\log_2\frac{1}{p}\right) + \left(\frac{1}{2}\right)\left((1-p)\log_2\frac{1}{1-p} + p\log_2\frac{1}{p}\right)\right] \\
I(X;Y) &= 1 - \left[(1-p)\log_2\frac{1}{1-p} + p\log_2\frac{1}{p}\right]
\end{aligned}
$$





Source




f = 0.1


Receiver

mation between source and receiver?

$$\log_2(1-p) + p\log_2(p)] = (1-p)\log_2\frac{1}{1-p} + p\log_2\frac{1}{p}$$

$$H(X|Y=1) = -[(1-p)\log_2(1-p) + p\log_2(p)] = (1-p)\log_2\frac{1}{1-p} + p\log_2\frac{1}{p}$$

$$I(X;Y) = H(X) - [P(Y=0)H(X|Y=0) + P(Y=1)H(Y=1)]$$

$$I(X;Y) = 1 - \left[\left(\frac{1}{2}\right)\left((1-p)\log_2\frac{1}{1-p} + p\log_2\frac{1}{p}\right) + \left(\frac{1}{2}\right)\left((1-p)\log_2\frac{1}{1-p} + p\log_2\frac{1}{p}\right)\right]$$

$$I(X;Y) = 1 - \left[(1-p)\log_2\frac{1}{1-p} + p\log_2\frac{1}{p}\right]$$

# Differential Entropy: Entropy for continuous probabilities I

For a continuous random variable $X$ with a probability density function $p(X)$ the differential entropy $h(X)$ is defined as

$$h(X) = -\int_X p(x) \log p(x)\, dx$$

as with its discrete analog, the units of differential entropy depend on the base of the logarithm, which is usually 2 (i.e., the units are bits).
But, take care in trying to apply properties of discrete entropy to differential entropy, since probability density functions can be greater than 1. For example, the uniform distribution $U(0, \frac{1}{2})$ has "negative" differential entropy $\int_0^{\frac{1}{2}} -2\log(2)\, dx = -\log(2)$ Thus, differential entropy does not share all properties of discrete entropy:

- Its values are not always nonnegative.
- It is not invariant with respect to change of variables
- It is not derived from information-theoretic first principles; it is merely defined in the limit to the discrete entropy (more cumbersome but cleaner definitions exist (check out Renyi and Kolmogorov entropy definitions)

# Differential Entropy: Entropy for continuous probabilities II

Fortunately, the concept of mutual information I(X;Y) has the distinction of retaining its fundamental significance for both continuous and discrete entropies.
We can define in addition two other quantities:
The conditional entropy $H(y|x)$, the joint entropy $H(x, y)$:

$$H(y|x) = -\int\int p(x, y) \log_2 p(y|x) dx\ dy$$

$$H(x, y) = -\int\int p(x, y) \log_2 p(x, y) dx\ dy$$

This yields an equation linking the entropies:

$$H(x, y) = -\int\int p(x, y) \log_2 p(x, y) dx\ dy$$

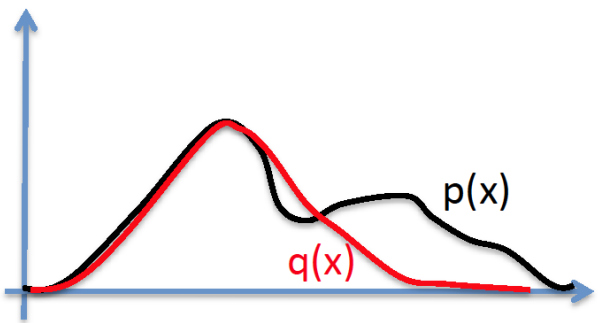
# Kullback-Leibler Divergence I

$$\begin{aligned} \mathrm{KL}(p\|q) &= -\int p(\mathbf{x}) \ln q(\mathbf{x})\, \mathrm{d}\mathbf{x} - \left(-\int p(\mathbf{x}) \ln p(\mathbf{x})\, \mathrm{d}\mathbf{x}\right) \\ &= -\int p(\mathbf{x}) \ln \left\{\frac{q(\mathbf{x})}{p(\mathbf{x})}\right\} \mathrm{d}\mathbf{x} \end{aligned}$$

$$\mathrm{KL}(p\|q) \geqslant 0$$

$$\mathrm{KL}(p\|q) \not\equiv \mathrm{KL}(q\|p)$$

It is strictly not a measure

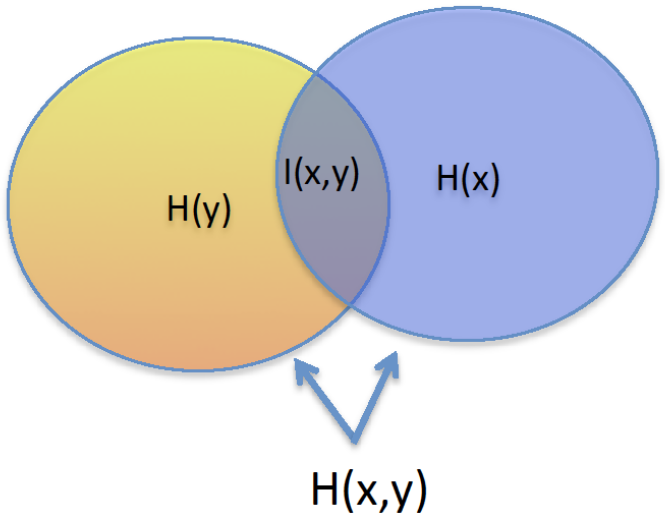



Note: KL divergence is measured often with both natural logarithm (ln) or base 2 ($\log_2$). When we talk information use base 2.

# Mutual Information as KL divergence I

How much can x tell us about y and vice versa?

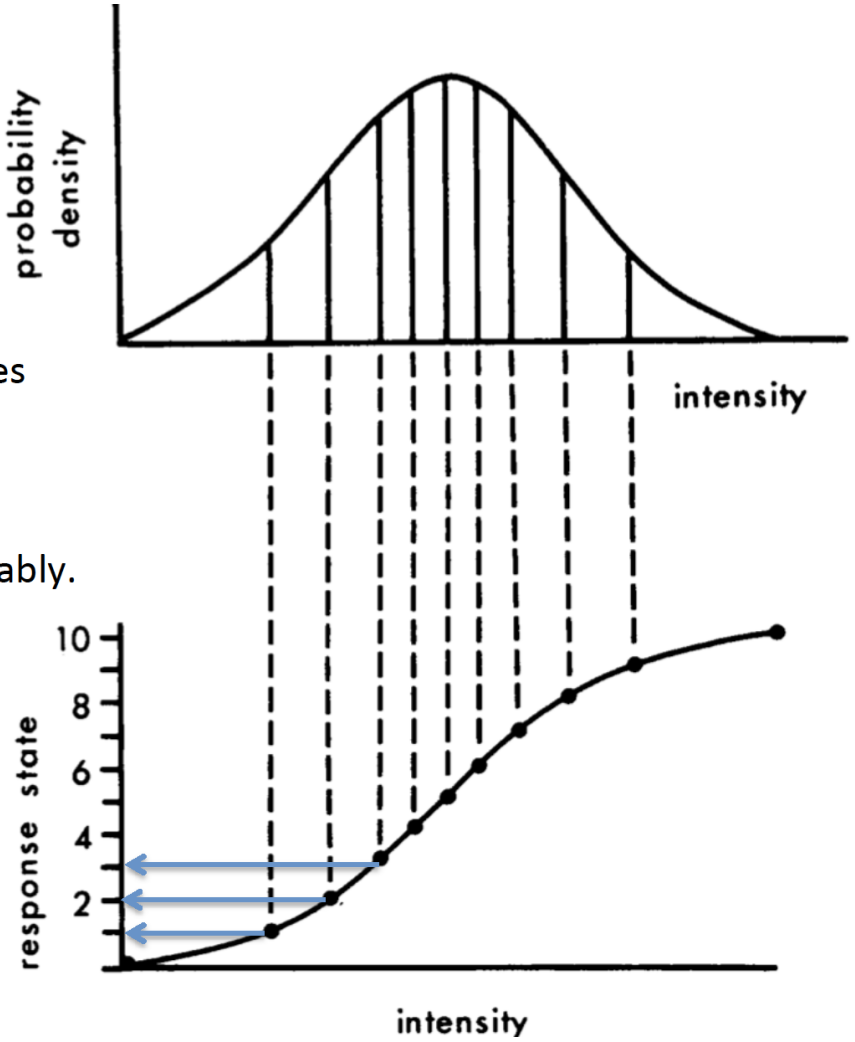
$$\mathrm{I}[\mathbf{x},\mathbf{y}] = \mathrm{H}[\mathbf{x}] - \mathrm{H}[\mathbf{x}|\mathbf{y}] = \mathrm{H}[\mathbf{y}] - \mathrm{H}[\mathbf{y}|\mathbf{x}] = H(x) + H(y) - H(x,y)$$




$$\begin{aligned}\mathrm{I}[\mathbf{x},\mathbf{y}] &\equiv \mathrm{KL}(p(\mathbf{x},\mathbf{y})\|p(\mathbf{x})p(\mathbf{y})) \\ &= -\iint p(\mathbf{x},\mathbf{y}) \ln\left(\frac{p(\mathbf{x})p(\mathbf{y})}{p(\mathbf{x},\mathbf{y})}\right) \mathrm{d}\mathbf{x}\,\mathrm{d}\mathbf{y}\end{aligned}$$

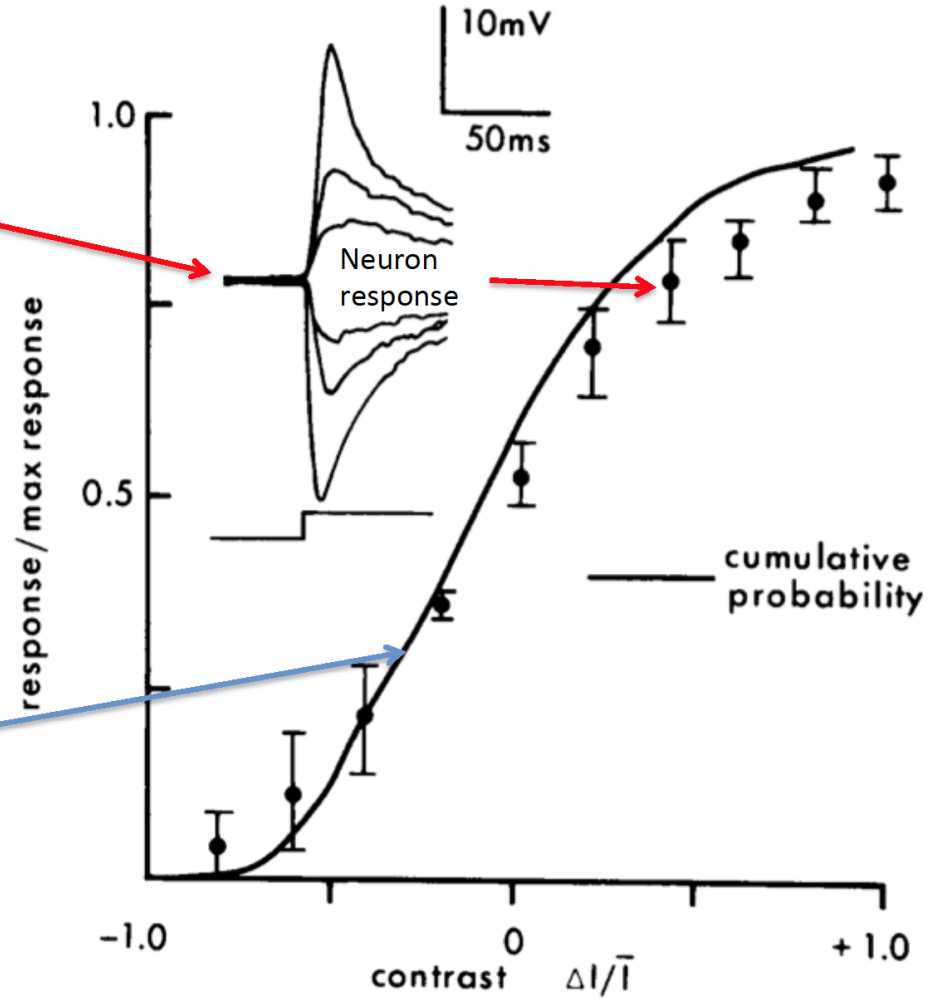
Problem: Sensory world has more states than our representation.

Solution: “Infomax” Map input to each state so that each state is equally probably.

Fly photoreceptor response

Contrast statistics




Laughlin (1981, Z. f. Naturforsch.)

# How to guess I

Suppose you have been hired by company Bo's Burgers, to analyse their worldwide sales. You visit all Bo's Burgers restaurants all over the world, and determine that, on average, people are paying 1.75 for their meals. As part of Carnivore's commitment to global homogeneity, the price of each meal is exactly the same in every restaurant (after local currencies are converted to dollars). The prices are 1 for the burger meal, 2 for the chicken meal, and 3 for the fish meal.
Your supervisors ask about the probabilities of a customer ordering each of the three value meals, but you do not have that data. You have to make the best estimate of the probabilities p(B), p(C), and p(F) consistent with 2 things you know:

$$1 = p(B) + p(C) + p(F)$$
$$1.75 = 1.00 \times p(B) + 2.00 \times p(C) + 3.00 \times p(F)$$

Since you have three unknowns and only two equations, there is not enough information to solve for the unknowns. The amount of your uncertainty about the probability distribution
$H = p(B) \log p(B) + p(C) \log p(C) + p(F) \log p(F)$ is the entropy.
For example:

# How to guess II

- if your average had been 2.00 rather than 1.75, you could have met both of your constraints by assuming that everybody bought the chicken meal. Then your uncertainty would have been 0 bits.
- or you could have assumed that half the orders were for burgers and half for fish, and the uncertainty would have been 1 bit.
- neither of these assumptions seems particularly appropriate, because each goes beyond what you know.

# Principle of Maximum Entropy I

The Principle of Maximum Entropy: when estimating the probability distribution, select that probability distribution which leaves you the largest remaining uncertainty (i.e., the maximum entropy) consistent with your constraints. That way you have not introduced any additional assumptions or biases into your calculations.

- The uniform distribution on the interval $[a, b]$ is the maximum entropy distribution among all continuous distributions which are supported in the interval $[a, b]$, and thus the probability density is 0 outside of the interval.
- The exponential distribution, for which the density function is

$$p(x|\lambda) = \begin{cases} \lambda e^{-\lambda x} & x \geq 0, \\ 0 & x < 0, \end{cases}$$

is the maximum entropy distribution among all continuous distributions supported in $[0, \infty]$ that have a specified mean of $\frac{1}{\lambda}$.

# Principle of Maximum Entropy II

- The normal distribution $N(\mu, \sigma^2)$, for which the density function is

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

has maximum entropy among all real-valued distributions supported on $(-\infty, \infty)$ with a specified variance $\sigma^2$. Note, that in engineering the variance is nothing else but a measure of the power or energy in a signal, all physical systems have energy-constrained signals, therefore ...

# Unsupervised Learning I: Clustering: Hierarchical Clustering [Option B]

# Hierarchical Clustering I

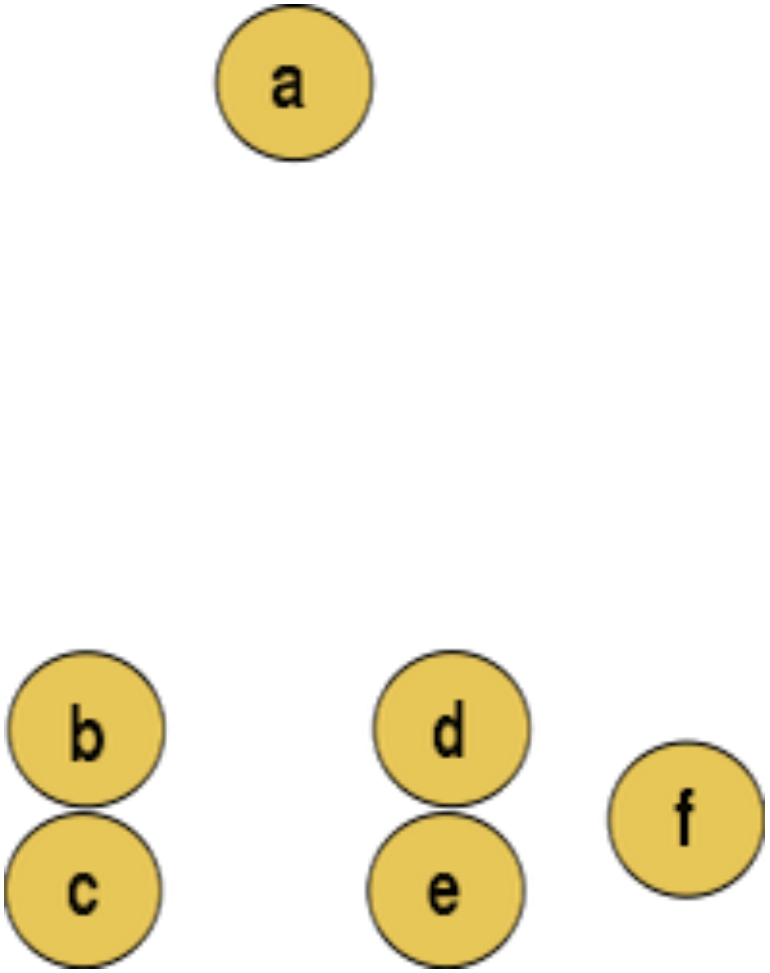
Hierarchical clustering is a method of cluster analysis which seeks to build a hierarchy of clusters. It is agglomerative, in that each data points starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy.

1. At the beginning of the process, each element is in a cluster of its own. The clusters are then sequentially combined into larger clusters until all elements end up being in the same cluster.
2. At each step, the two clusters separated by the 'shortest distance' are combined.

The definition of 'shortest distance' is what differentiates between the different agglomerative clustering methods and is defined by the distance metric and the linkage criterion.

# Hierarchical Clustering II

Example data set



# Hierarchical Clustering III

Distance metrics include:

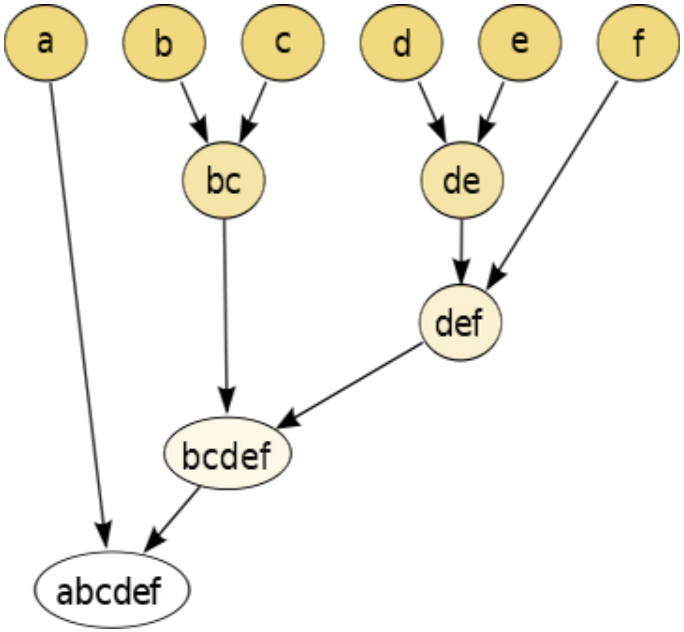
- Euclidean Distance $d(x, y) = \|x - y\|_2 = \sqrt{\sum_i (x_i - y_i)^2}$
- Manhattan or city block distance $d(x, y) = \|x - y\|_1 = sum|x_i - y_i|$
- Maximum distance $d(x, y) = \|x - y\|_\infty = max_i|x_i - b_i|$

Linkage criteria include:

- $max\{d(x, y) : x \in cluster\ X, y \in cluster\ Y\}$ complete-linkage clustering or farthest neighbour clustering
- $min\{d(x, y) : x \in cluster\ X, y \in cluster\ Y\}$ single-linkage clustering or nearest neighbour clustering.
- Ward's method: Merge clusters so that the total within-cluster variance is minimised (req. euclidean distance metrics)

# Hierarchical Clustering IV

The Dendrogram is a tree-structure visualisation of hierarchical clustering for our example:




Notes: 1. Be careful when clustering data vectors that have different units and magnitudes (why?).
2. Agglomerative clustering is slow for large number $n$ of data points: $\mathcal{O}(n^2 \log(n))$.