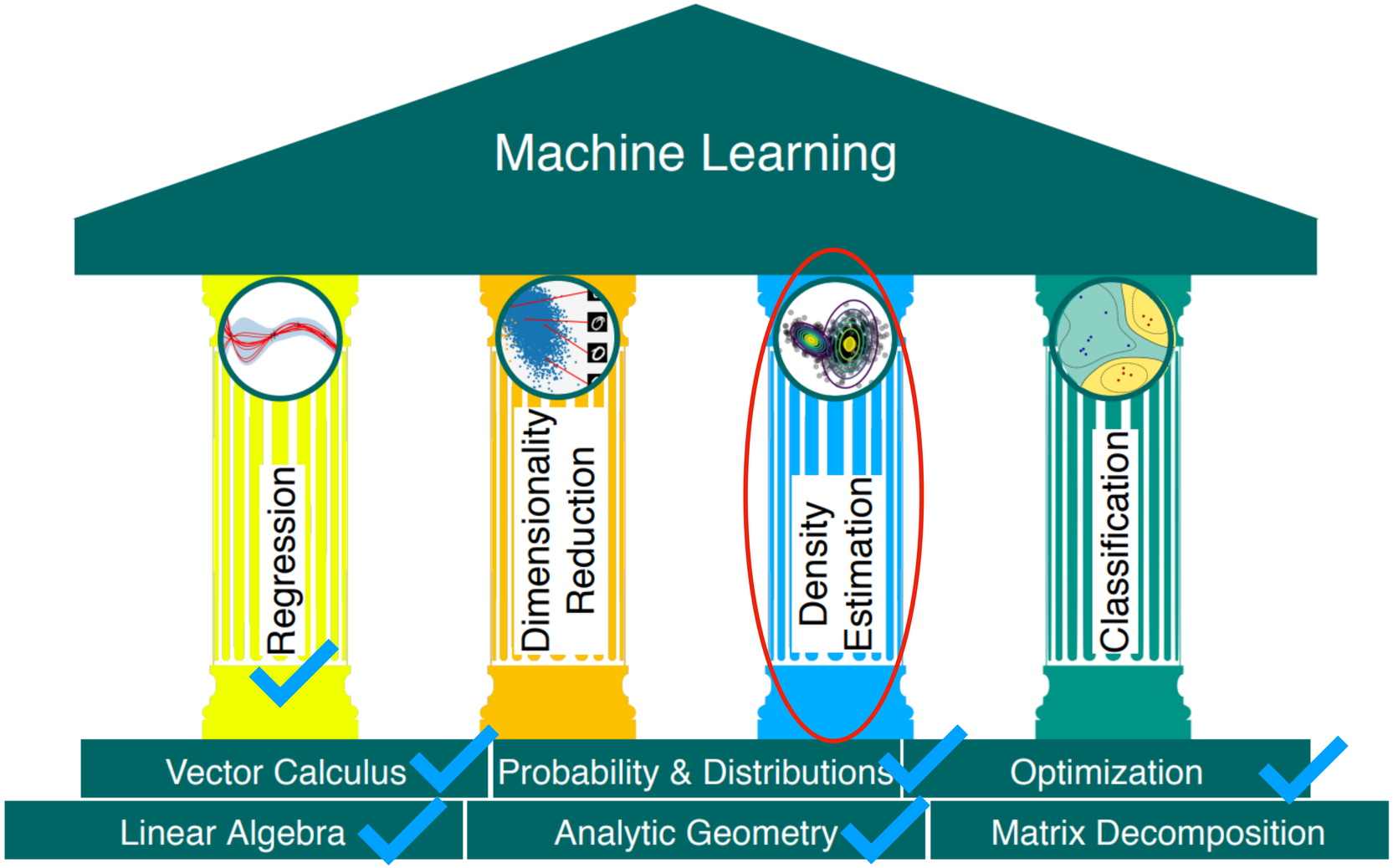
Density estimation with Gaussian mixture models

Week 8 - Introduction to ML / Thang Bui / ANU / 2023 S2

Housekeeping

- Assignment 3 is now available
- [Centrally invigilated, written] exam timetable will be available soon
- Please use Ed!
- Next Monday is a public holiday
  - Lecture: Dunbar Lecture Theatre, Physics Bldg 39A, Thursday 8:30 - 10 am
  - Tutorial: Comp Lab 1.24, Hanna Neumann Bldg 145, Tuesday 5 pm

Foundations of ML



This week:
Density estimation using Gaussian mixture models

Source: Maths for ML Textbook

Recap: Linear regression

Point estimate

$$\theta_{\mathrm{ML}} = (X^\intercal X)^{-1} X^\intercal \mathbf{y} \text{ or } \theta_{\mathrm{MAP}} = (X^\intercal X + N\lambda \mathrm{I})^{-1} X^\intercal \mathbf{y}$$

Exact posterior

$$p(\theta \mid \mathscr{D}) = \mathcal{N}(\theta; \mu, \Sigma)$$

Mean = MAP

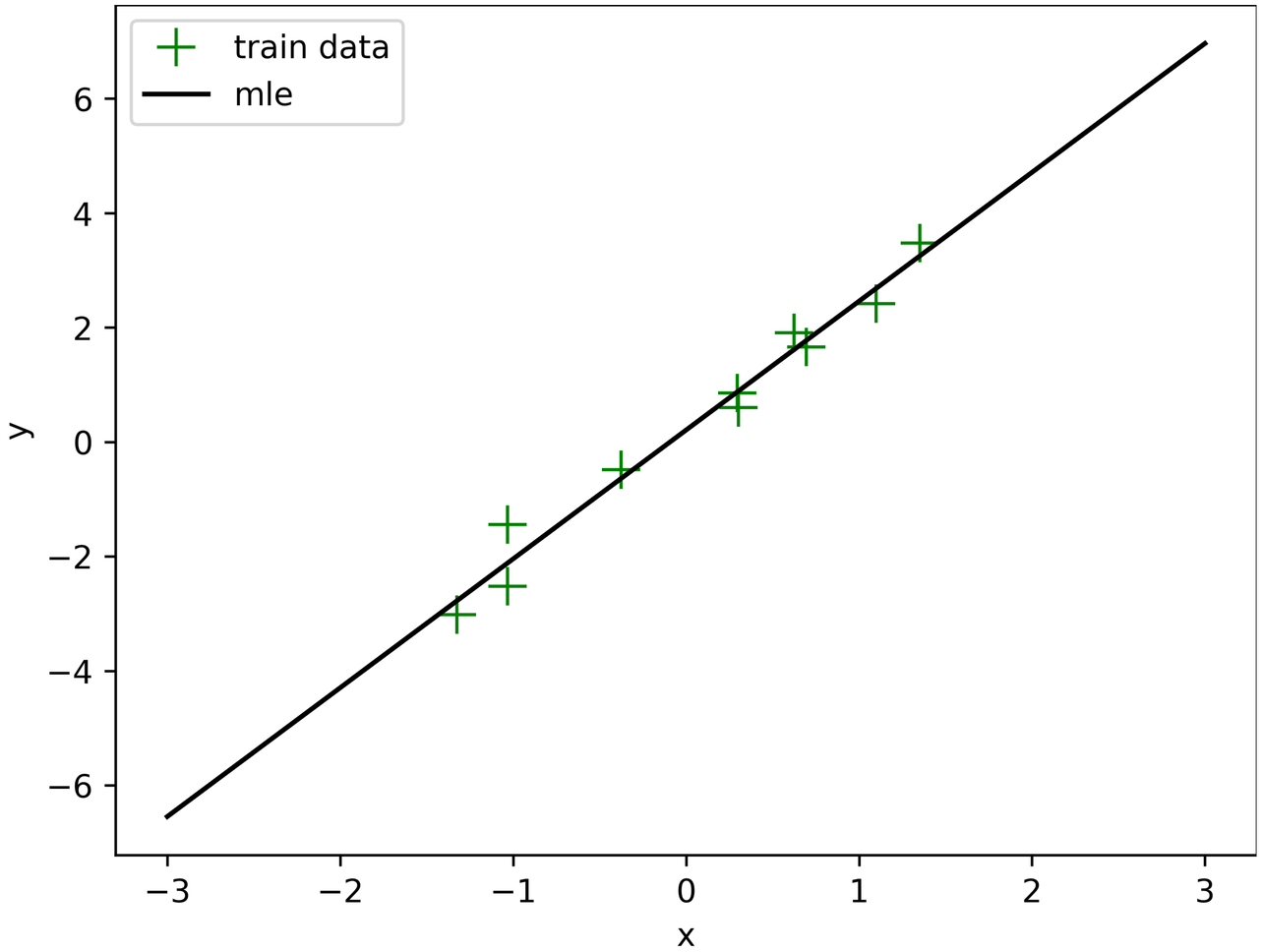
$$\mu = \Sigma \sigma^{-2} \mathbf{X}^\intercal \mathbf{y}$$

$$\Sigma = (\sigma^{-2} \mathbf{X}^\intercal \mathbf{X} + \sigma_o^{-2} \mathrm{I}_D)^{-1}$$

Recap: Linear regression

Point estimate

$$\theta_{\text{ML}} = (X^\intercal X)^{-1} X^\intercal \mathbf{y} \text{ or } \theta_{\text{MAP}} = (X^\intercal X + N\lambda \mathbf{I})^{-1} X^\intercal \mathbf{y}$$



Exact posterior

$$p(\theta \mid \mathcal{D}) = \mathcal{N}(\theta; \mu, \Sigma)$$

Mean = MAP

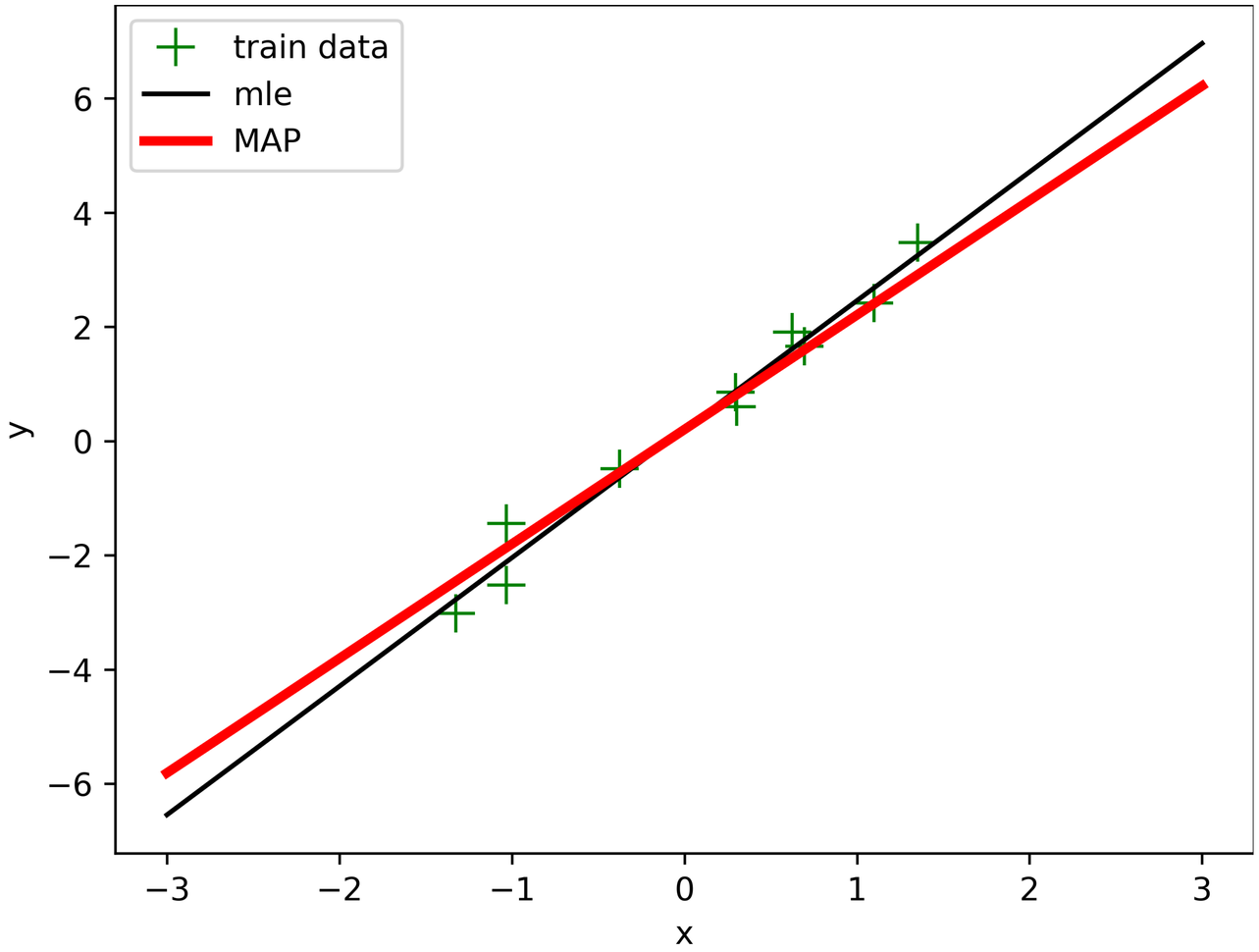
$$\mu = \Sigma \sigma^{-2} \mathbf{X}^\intercal \mathbf{y}$$

$$\Sigma = (\sigma^{-2} \mathbf{X}^\intercal \mathbf{X} + \sigma_o^{-2} \mathbf{I}_D)^{-1}$$

Recap: Linear regression

Point estimate

$$\theta_{\text{ML}} = (X^\intercal X)^{-1} X^\intercal \mathbf{y} \text{ or } \theta_{\text{MAP}} = (X^\intercal X + N\lambda \mathbf{I})^{-1} X^\intercal \mathbf{y}$$



Exact posterior

$$p(\theta \mid \mathcal{D}) = \mathcal{N}(\theta; \mu, \Sigma)$$

Mean = MAP

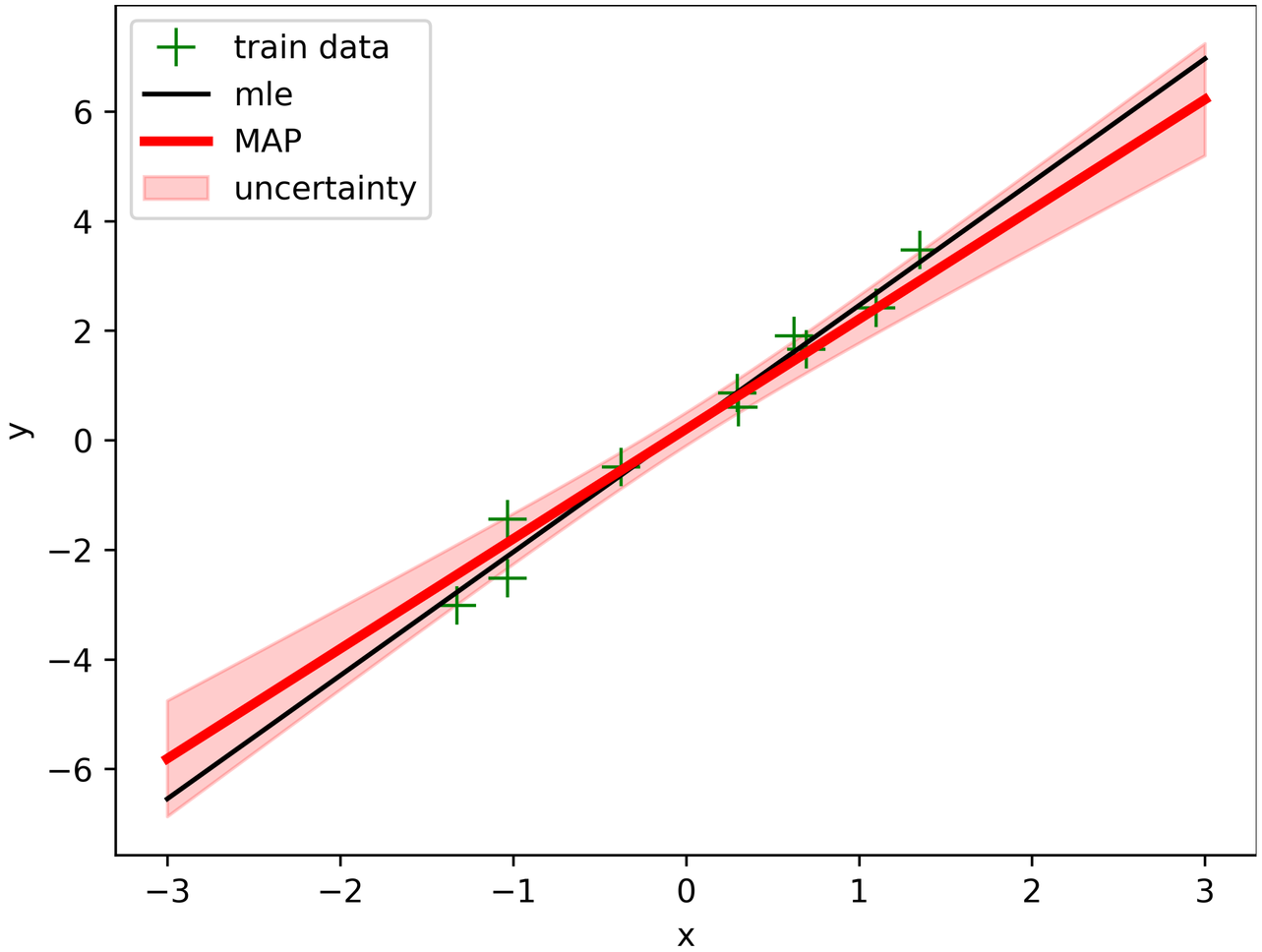
$$\mu = \Sigma \sigma^{-2} \mathbf{X}^\intercal \mathbf{y}$$

$$\Sigma = (\sigma^{-2} \mathbf{X}^\intercal \mathbf{X} + \sigma_o^{-2} \mathbf{I}_D)^{-1}$$

Recap: Linear regression

Point estimate

$$\theta_{\mathrm{ML}} = (X^\intercal X)^{-1} X^\intercal \mathbf{y} \text{ or } \theta_{\mathrm{MAP}} = (X^\intercal X + N\lambda \mathrm{I})^{-1} X^\intercal \mathbf{y}$$



Exact posterior

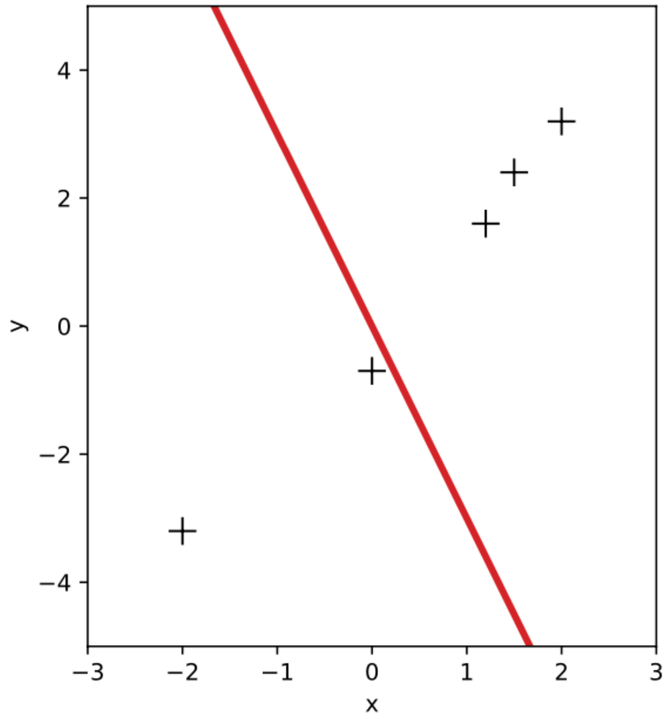
$$p(\theta \mid \mathcal{D}) = \mathcal{N}(\theta; \mu, \Sigma)$$

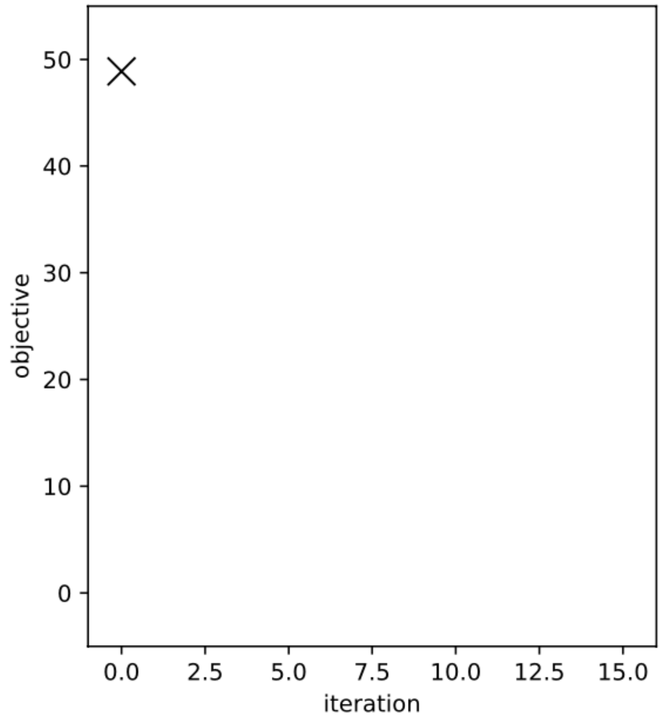
Mean = MAP

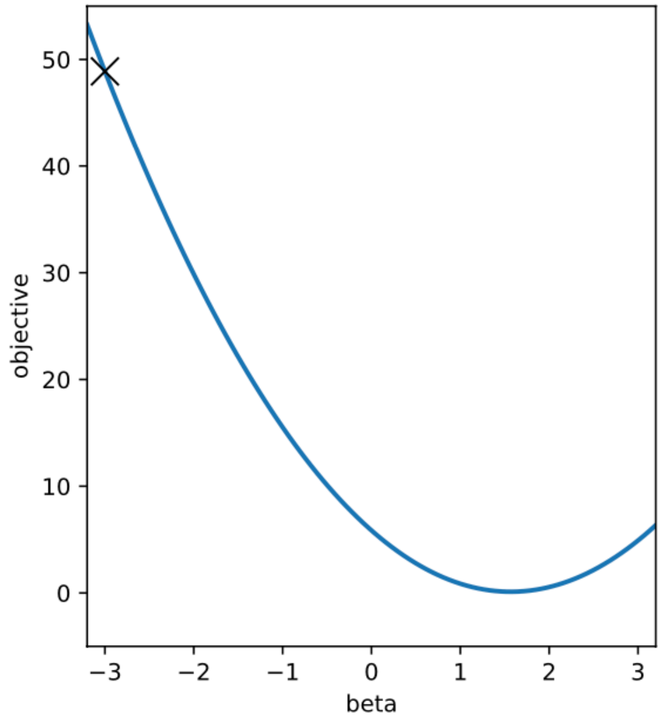
$$\mu = \Sigma \sigma^{-2} \mathbf{X}^\intercal \mathbf{y}$$

$$\Sigma = (\sigma^{-2} \mathbf{X}^\intercal \mathbf{X} + \sigma_o^{-2} \mathrm{I}_D)^{-1}$$

Recap: Optimisation for linear regression

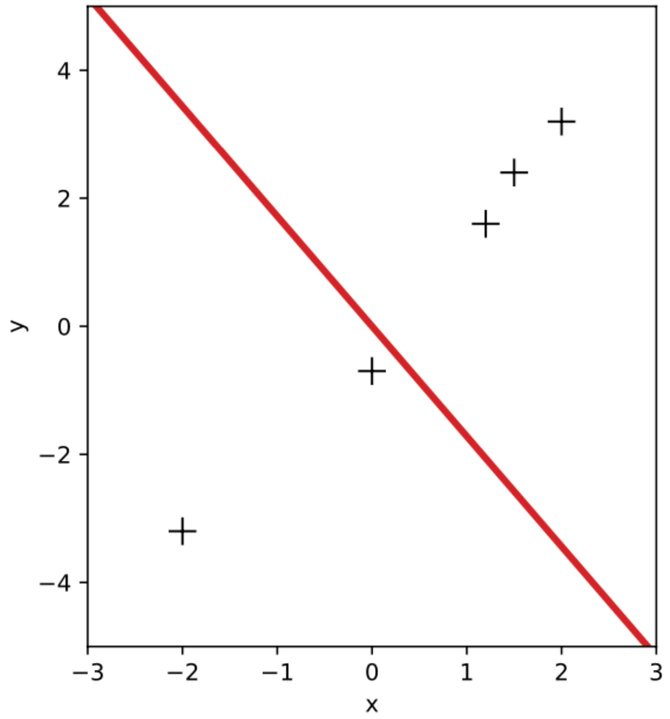


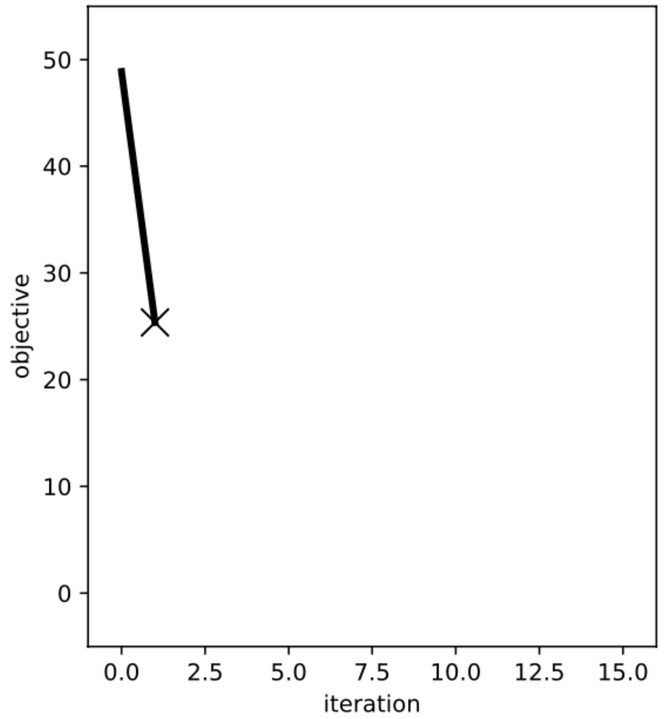


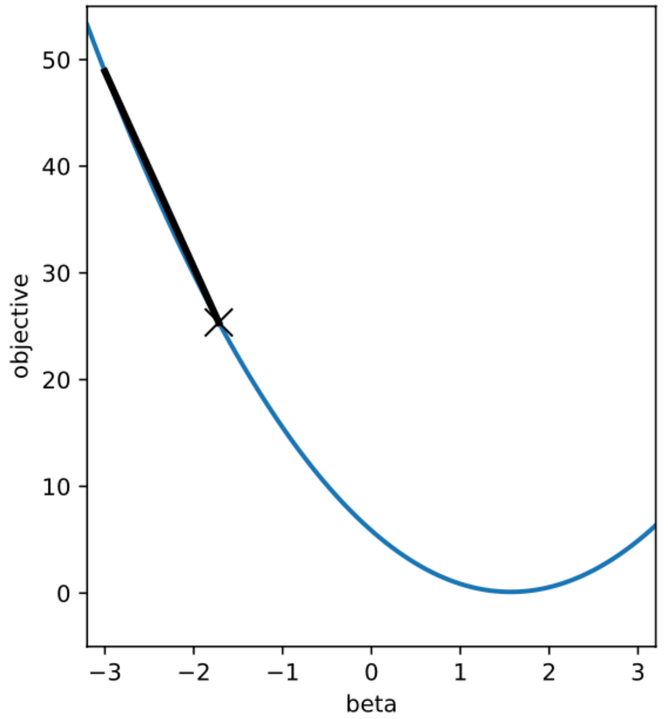


Another example: https://playground.tensorflow.org/

Recap: Optimisation for linear regression

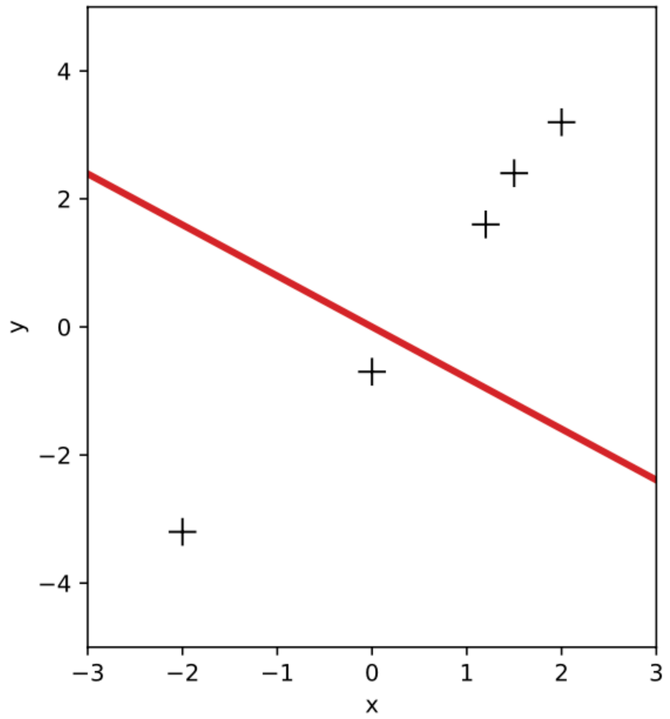


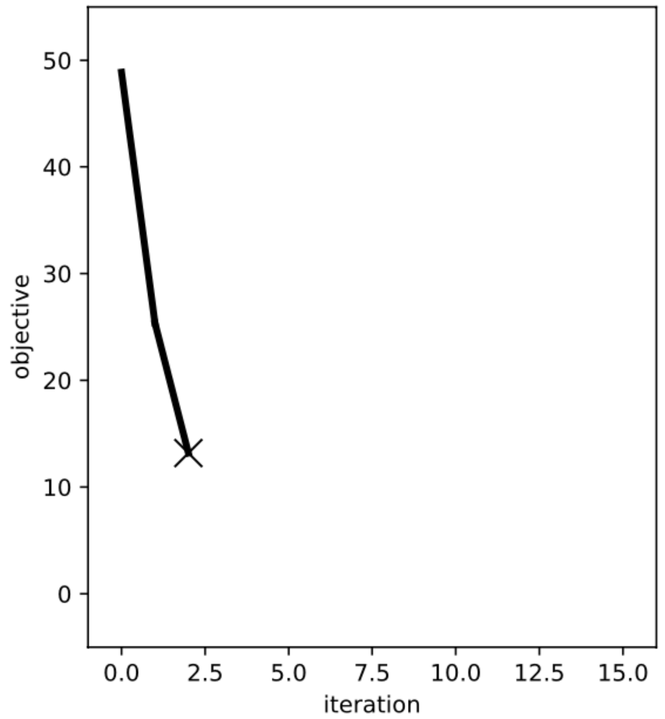


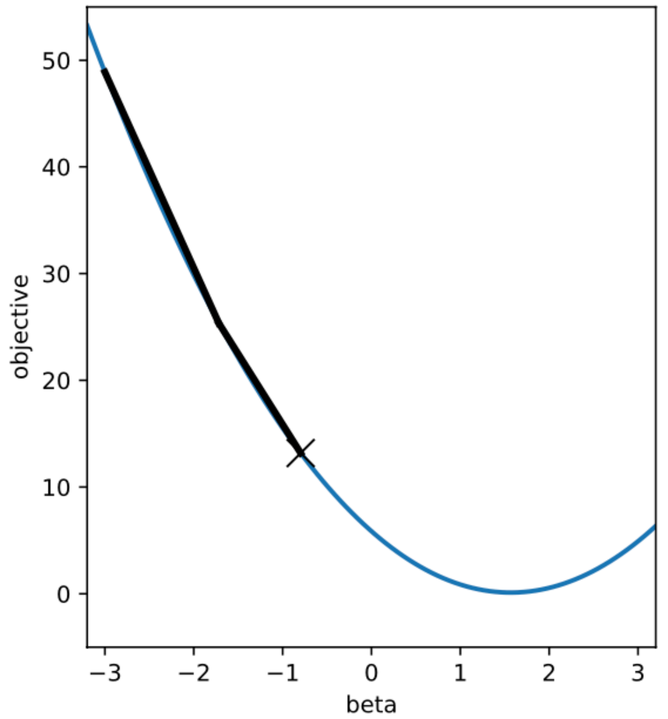


Another example: https://playground.tensorflow.org/

Recap: Optimisation for linear regression

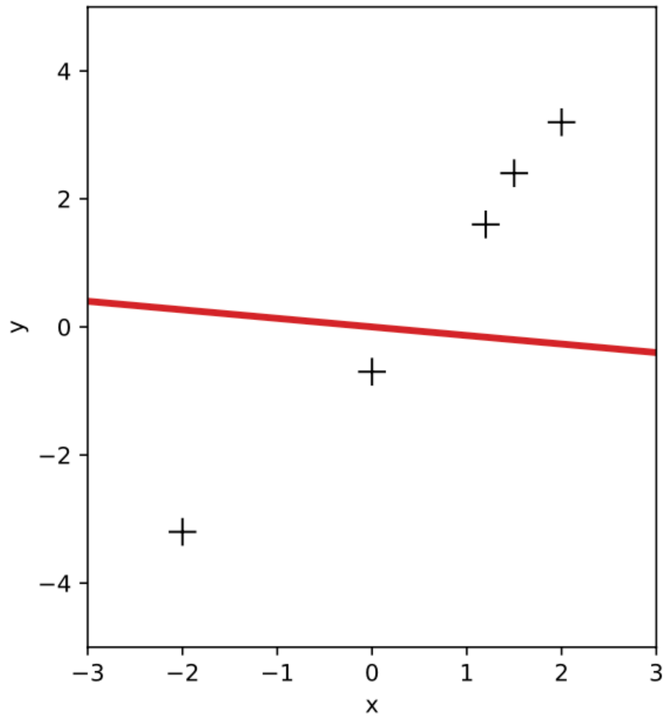


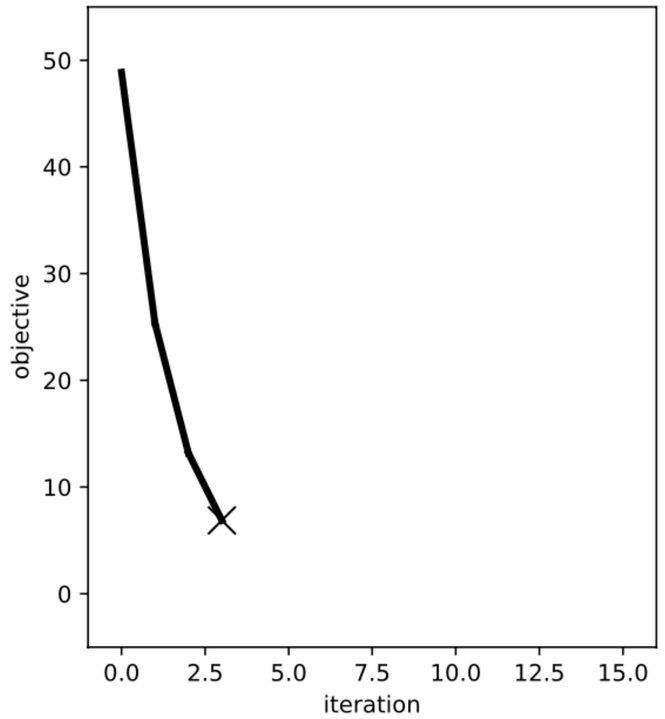


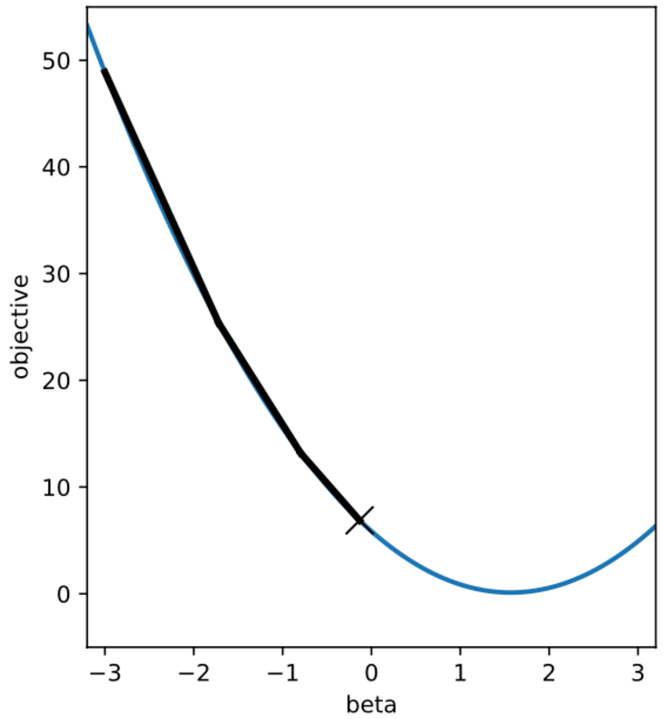


Another example: https://playground.tensorflow.org/

Recap: Optimisation for linear regression

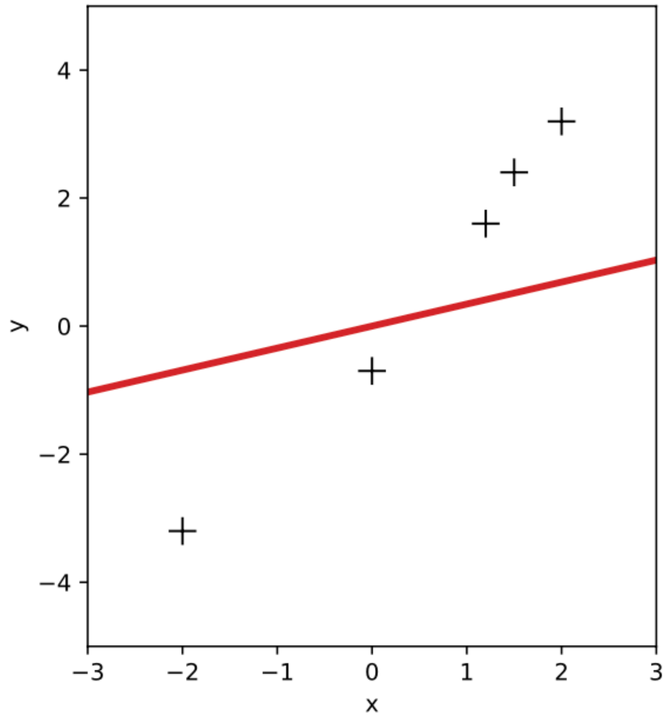


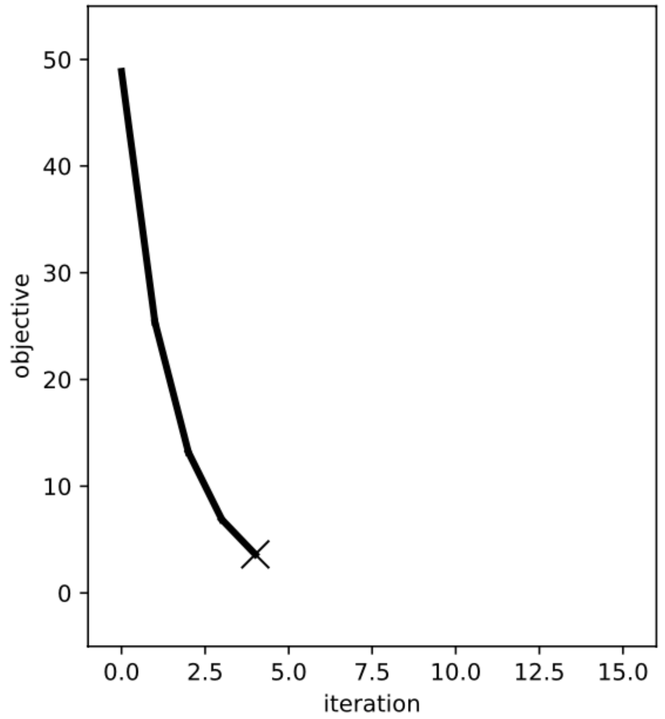


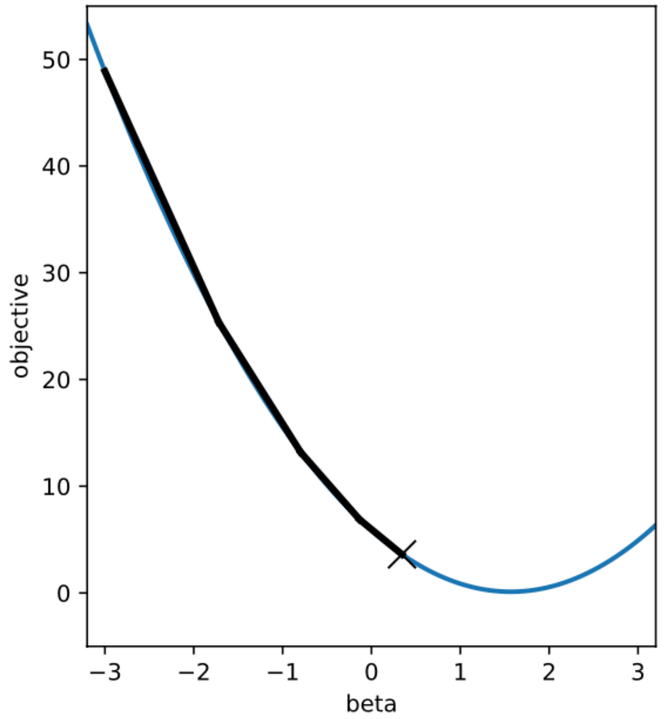


Another example: https://playground.tensorflow.org/

Recap: Optimisation for linear regression

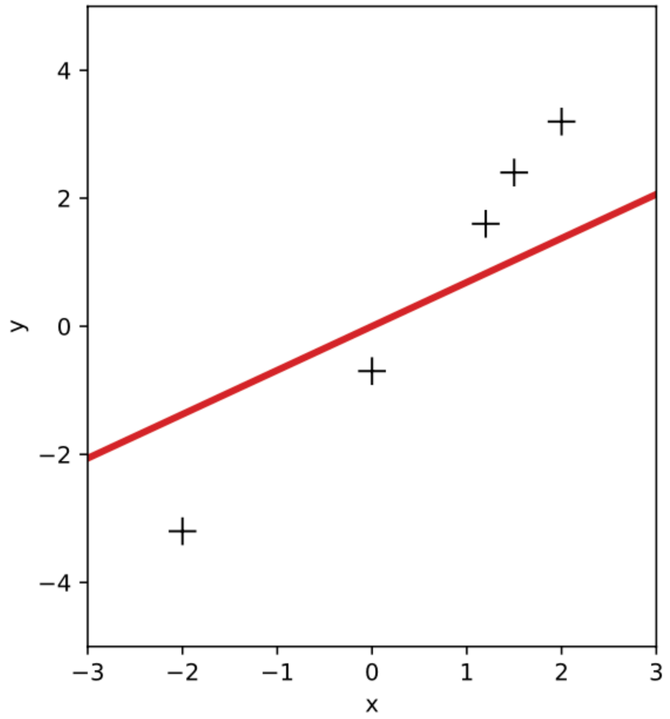


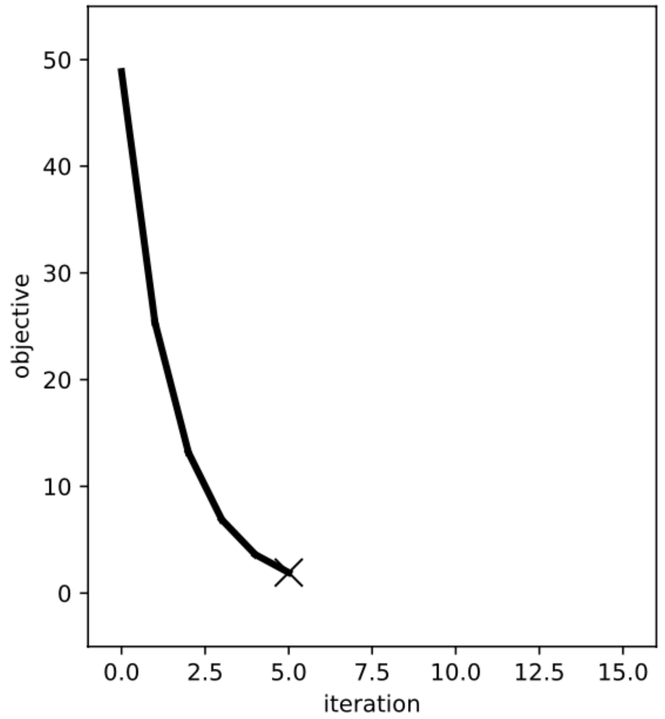


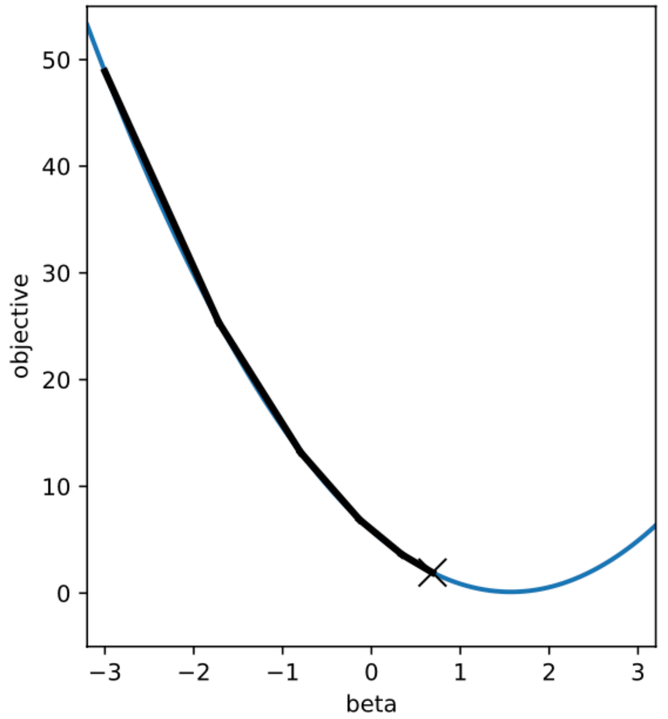


Another example: https://playground.tensorflow.org/

Recap: Optimisation for linear regression

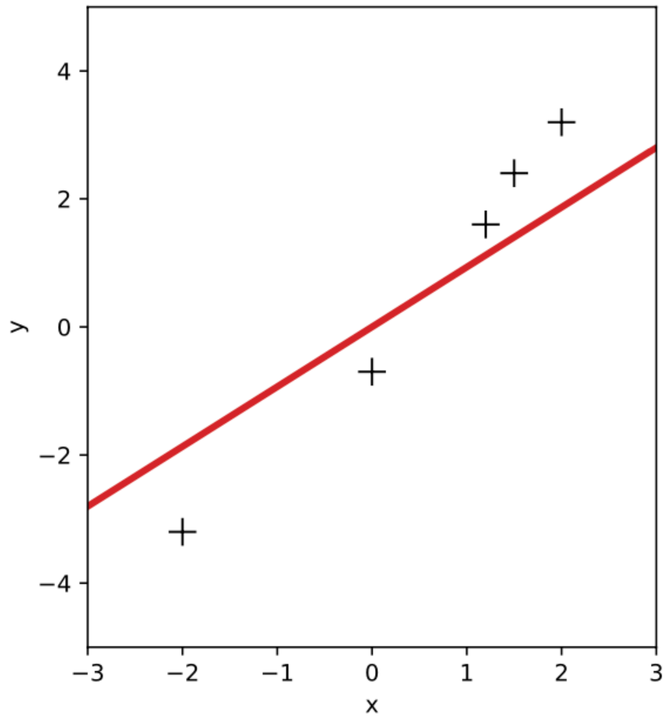


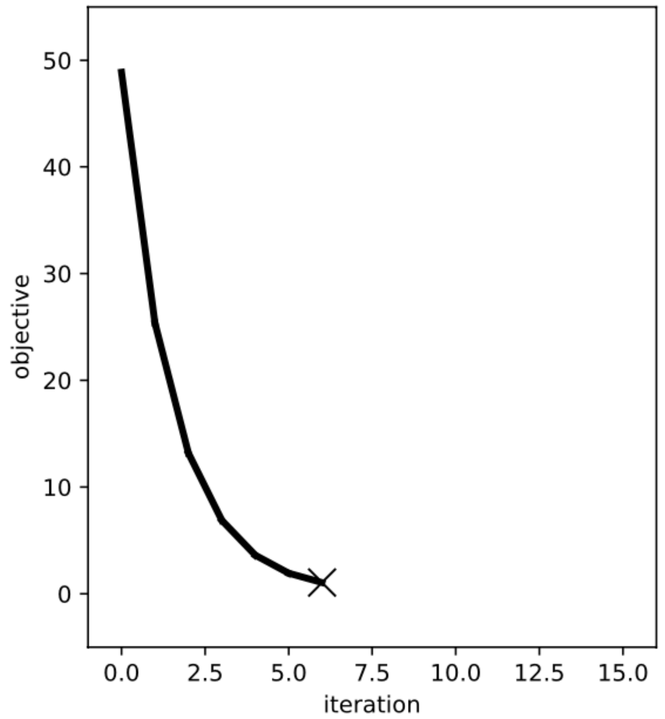


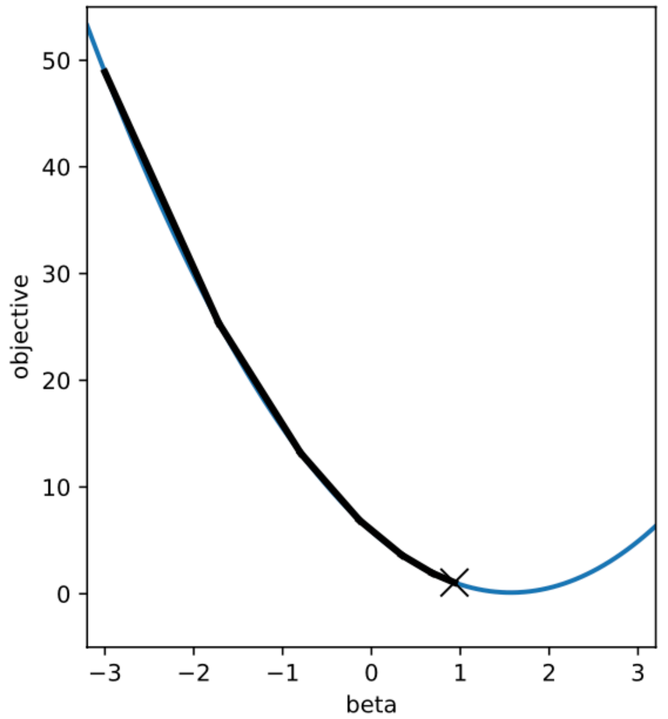


Another example: https://playground.tensorflow.org/

Recap: Optimisation for linear regression

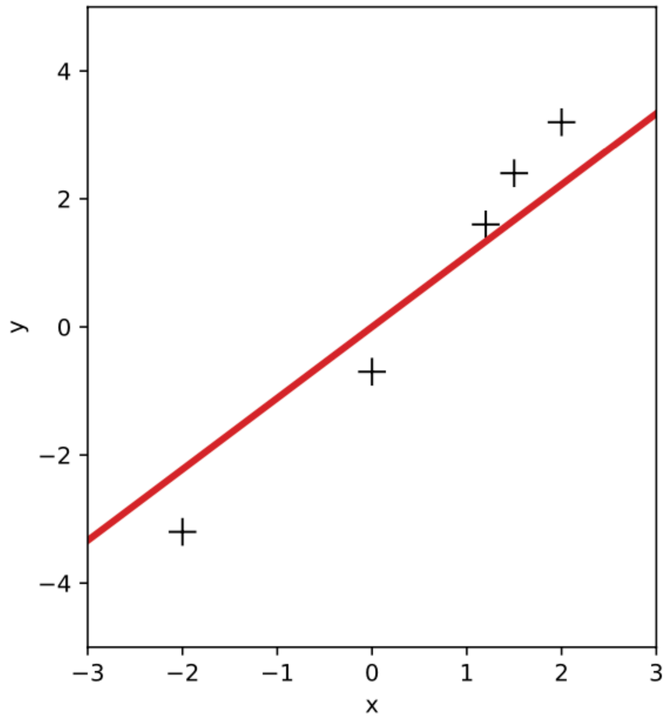


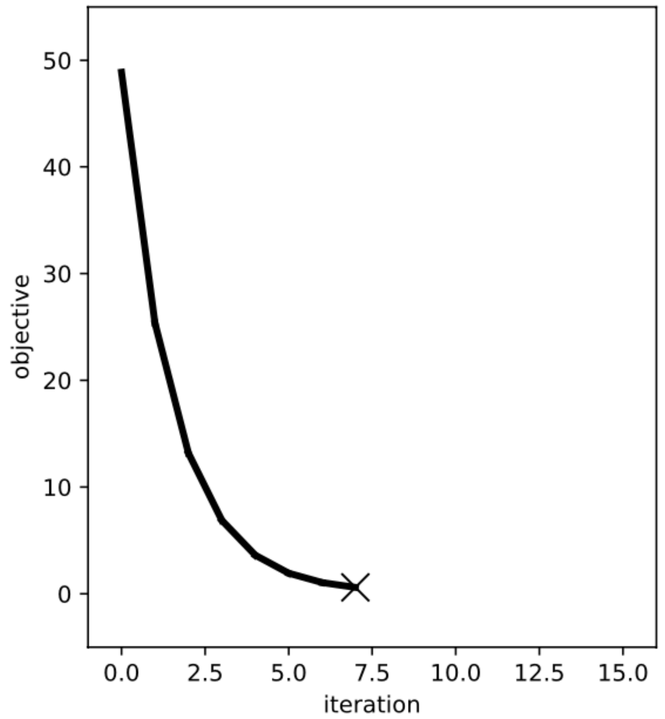


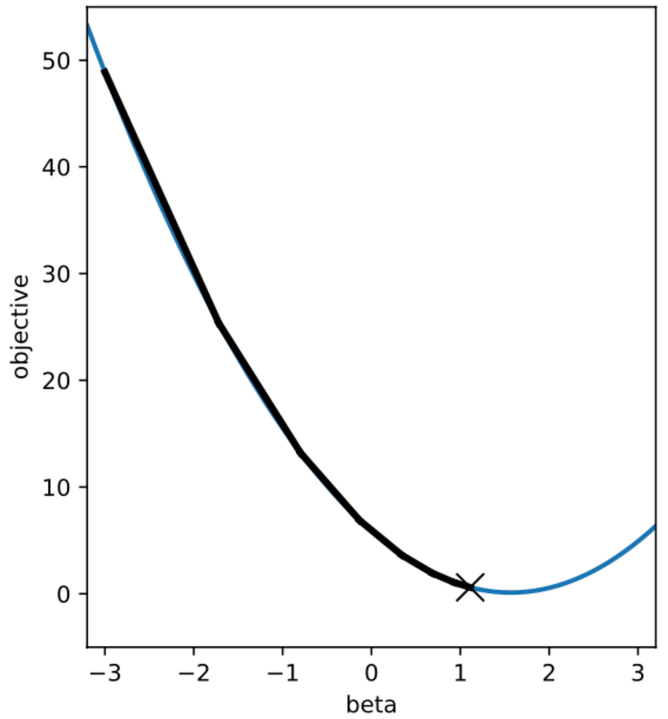


Another example: https://playground.tensorflow.org/

Recap: Optimisation for linear regression

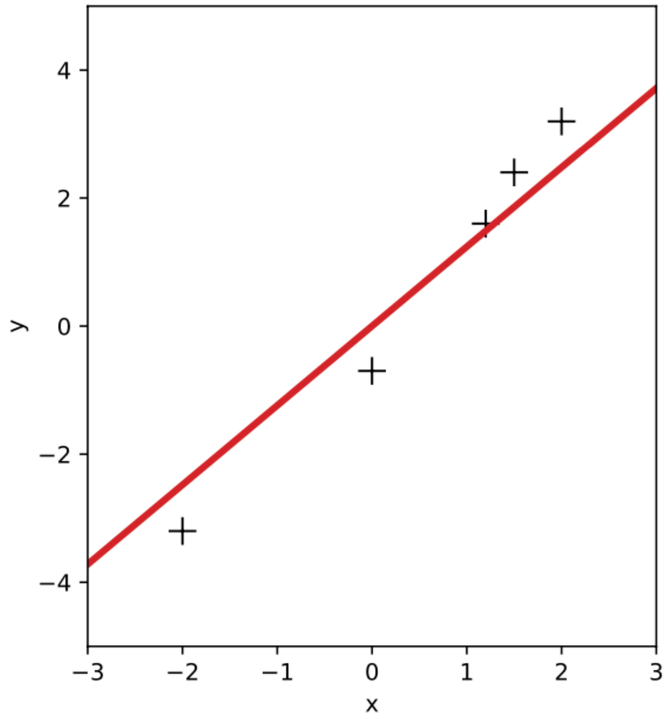


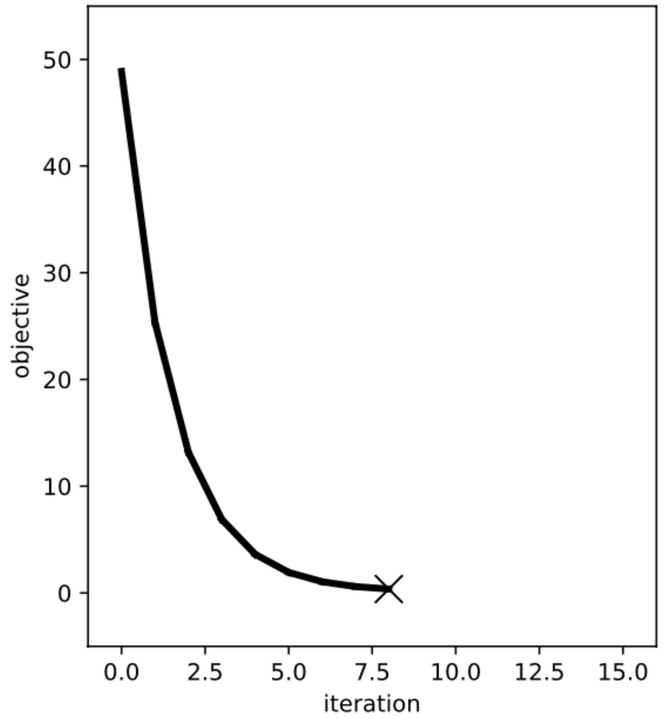


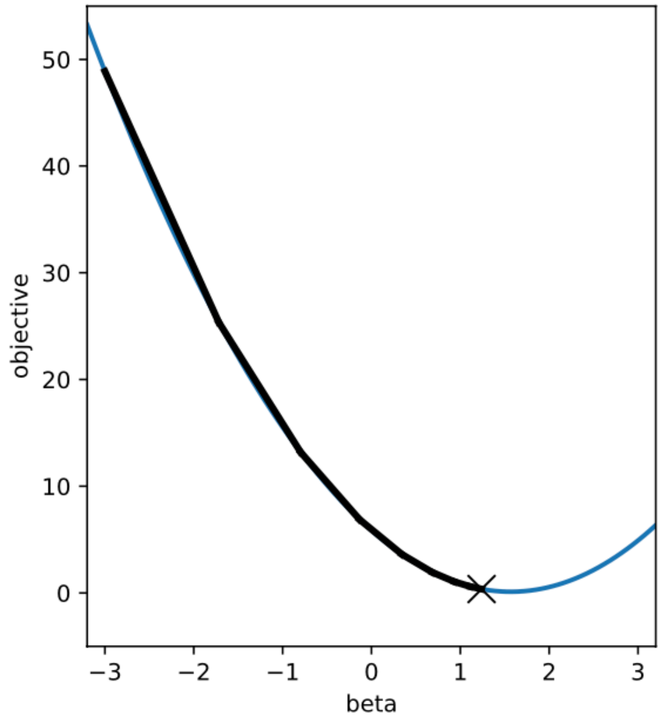


Another example: https://playground.tensorflow.org/

Recap: Optimisation for linear regression

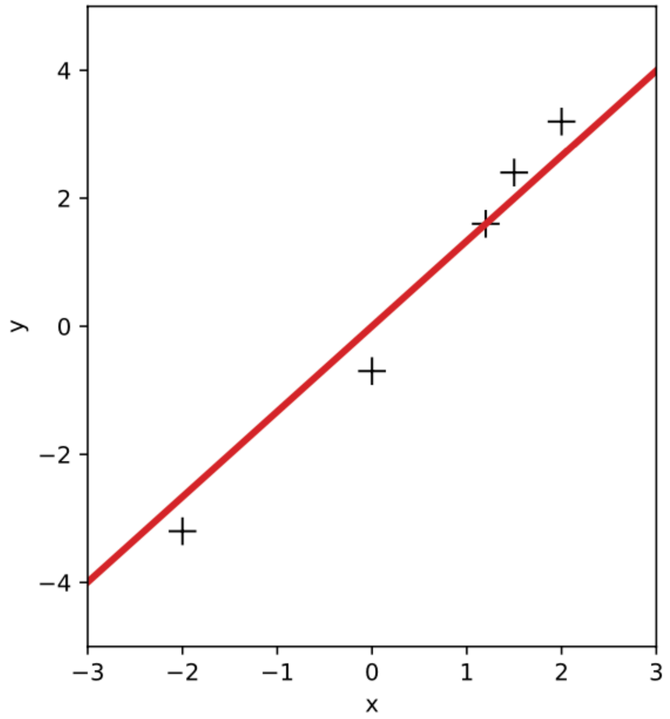


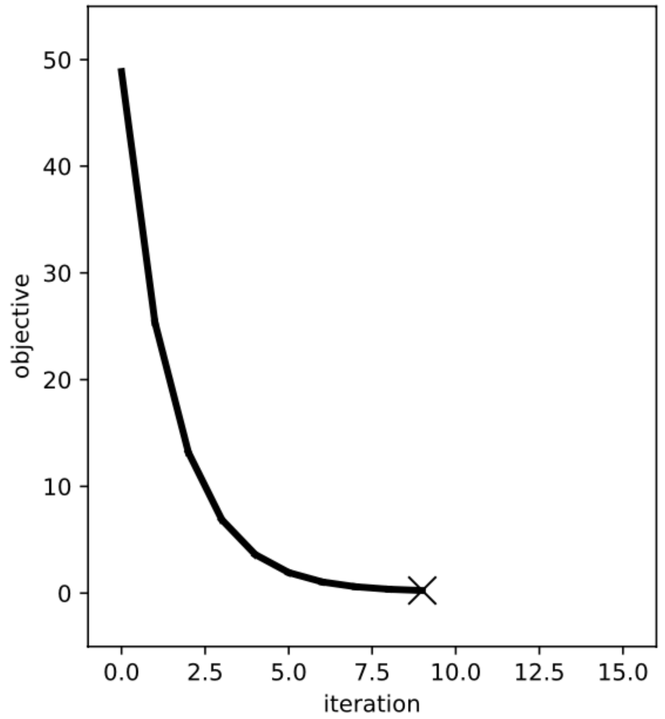


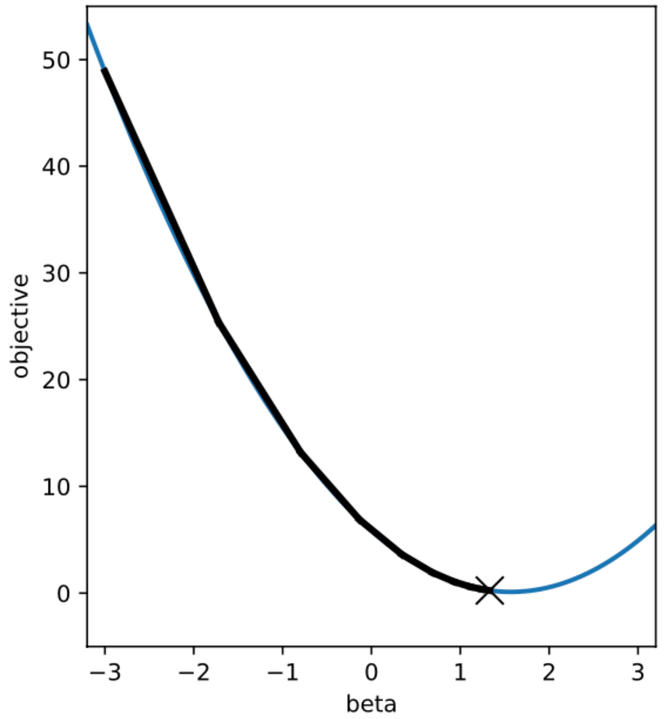


Another example: https://playground.tensorflow.org/

Recap: Optimisation for linear regression

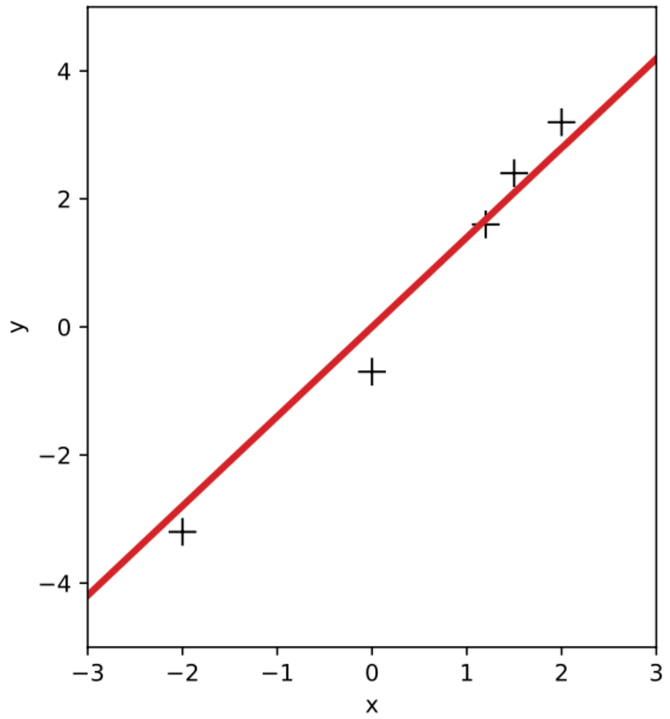


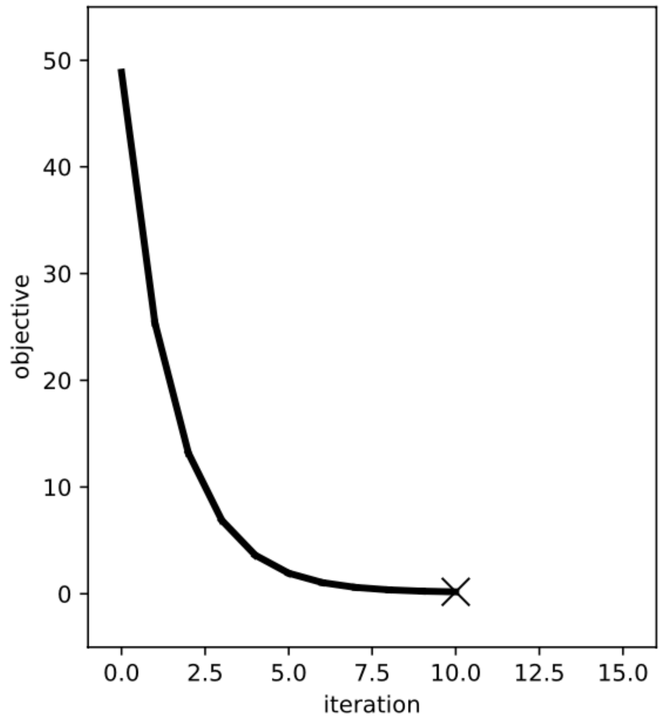


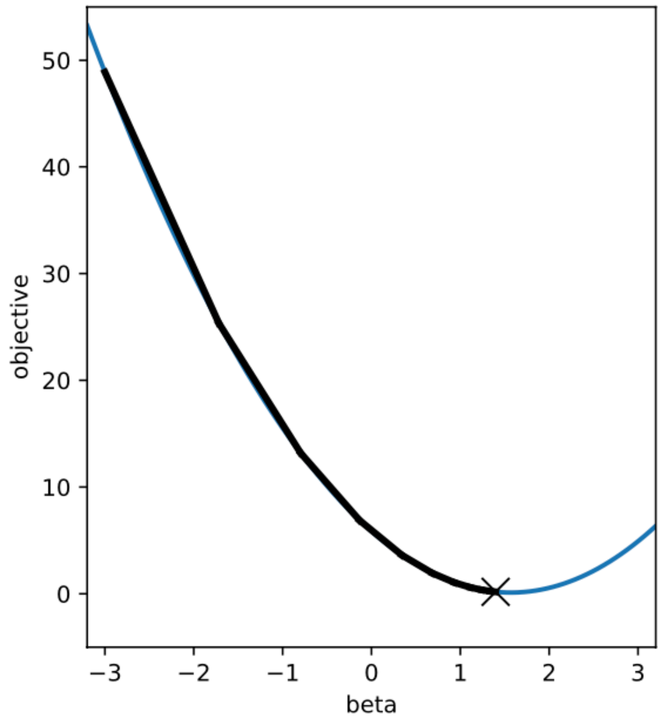


Another example: https://playground.tensorflow.org/

Recap: Optimisation for linear regression

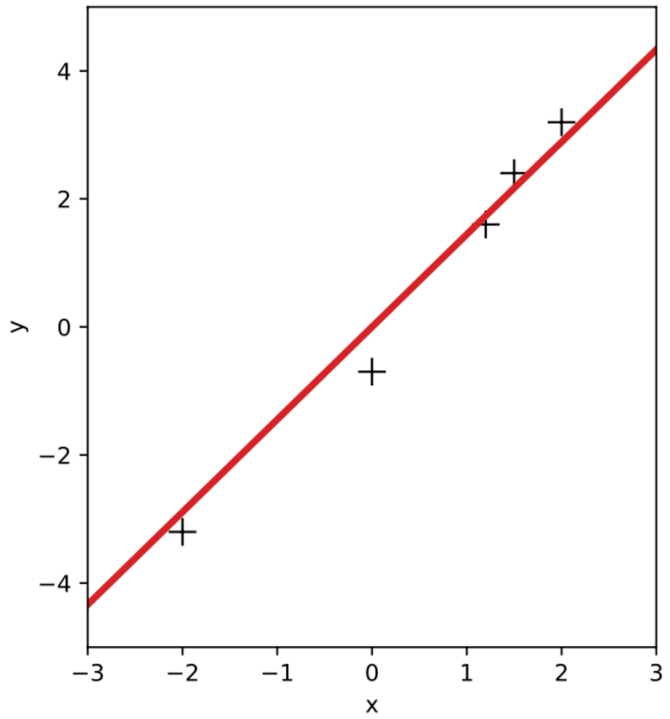


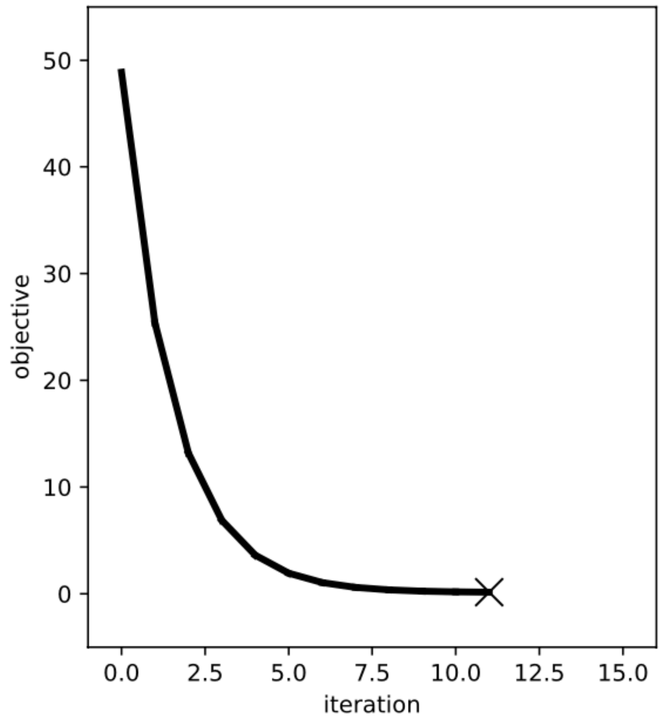


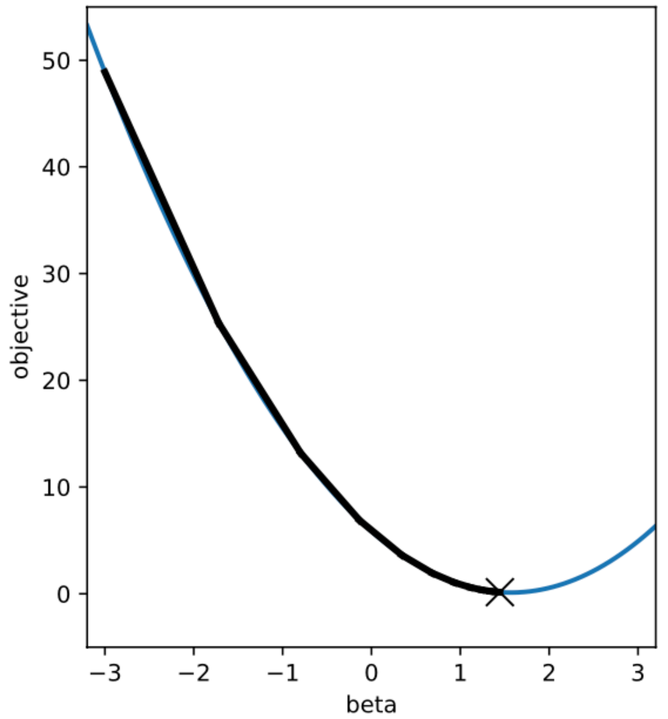


Another example: https://playground.tensorflow.org/

Recap: Optimisation for linear regression

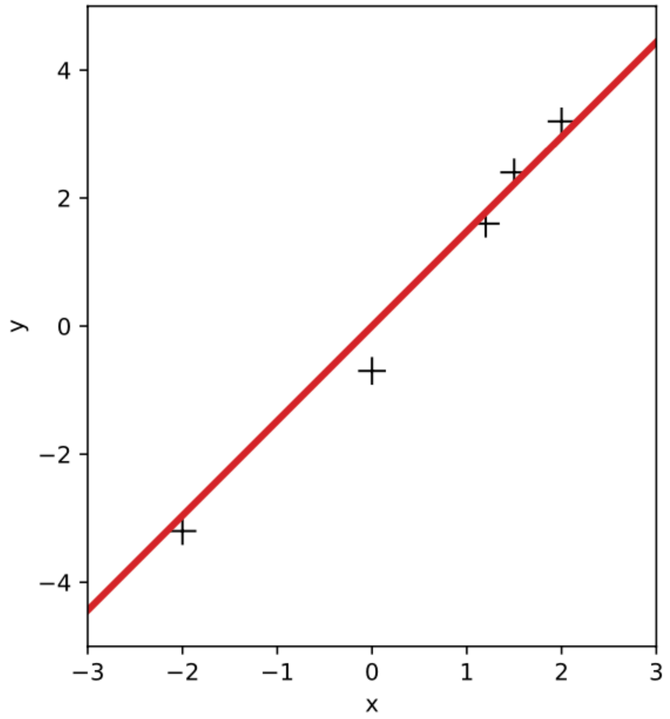


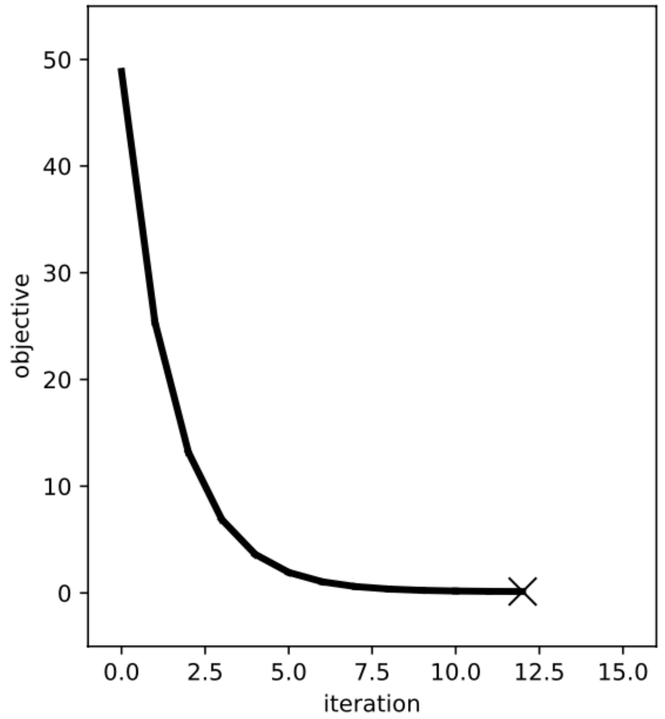


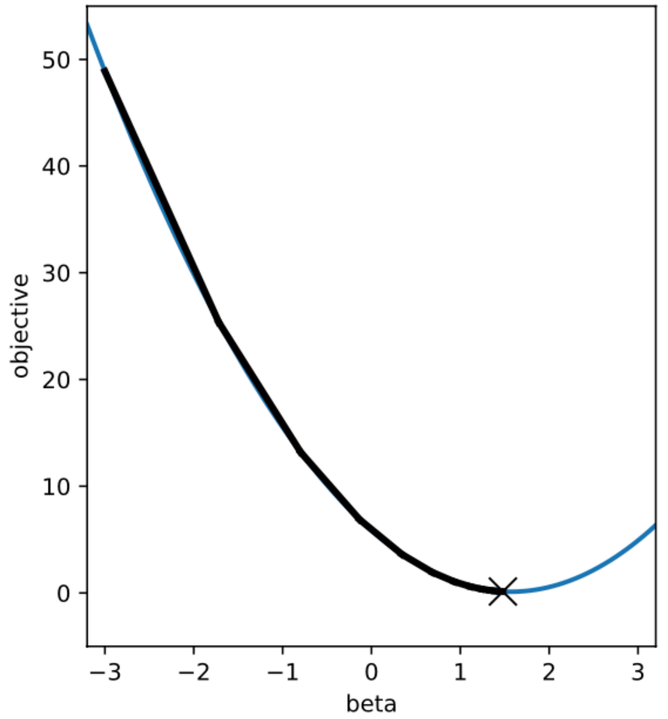


Another example: https://playground.tensorflow.org/

Recap: Optimisation for linear regression

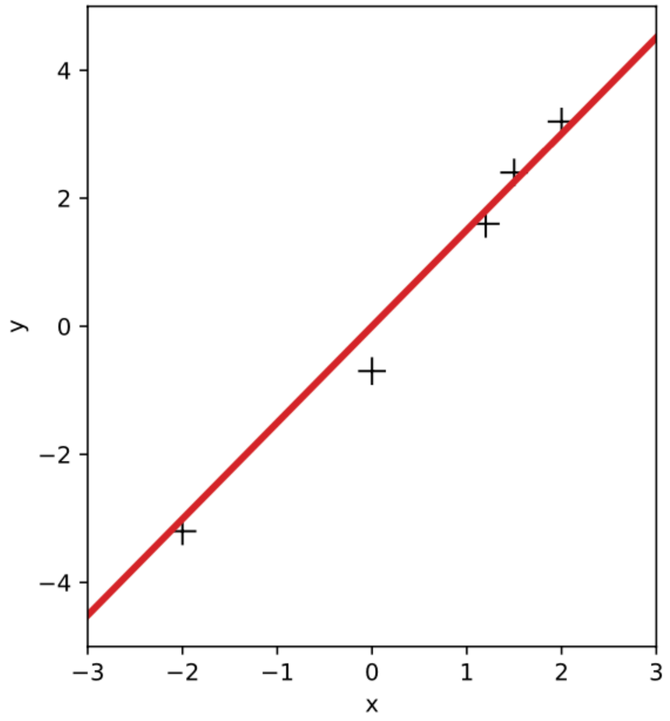


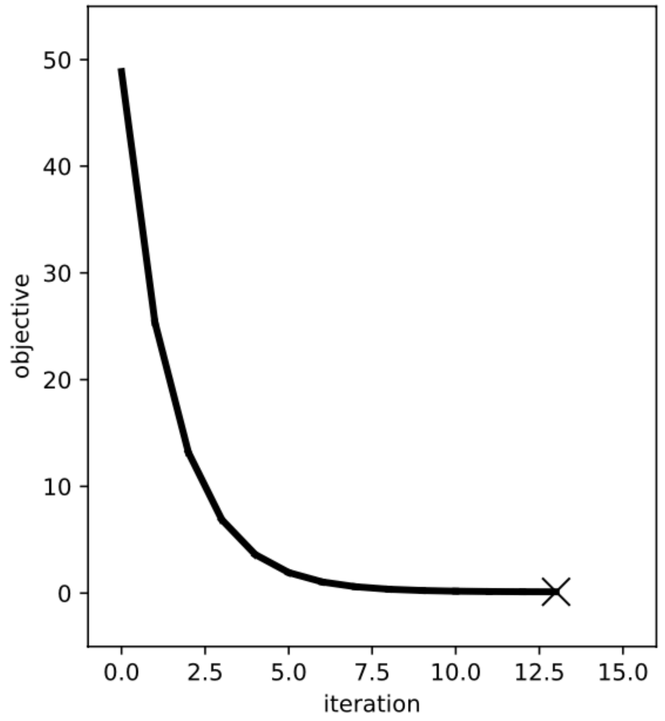


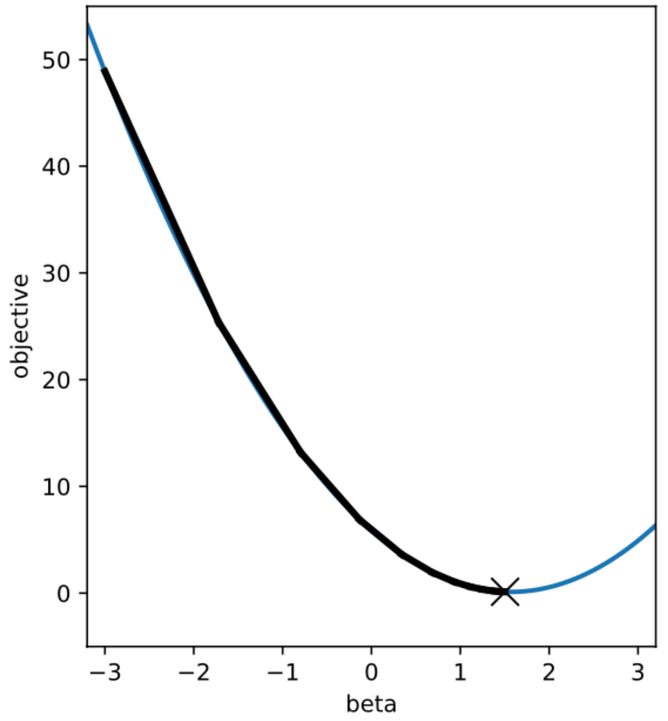


Another example: https://playground.tensorflow.org/

Recap: Optimisation for linear regression

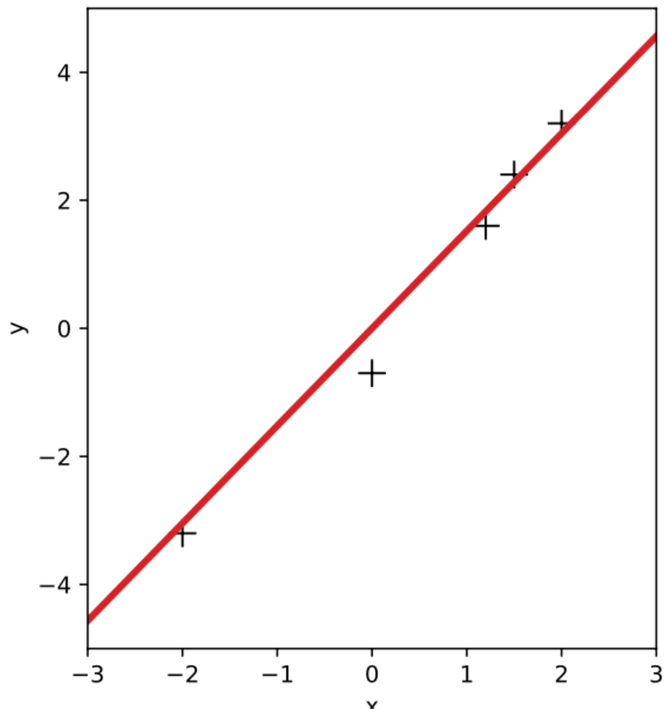


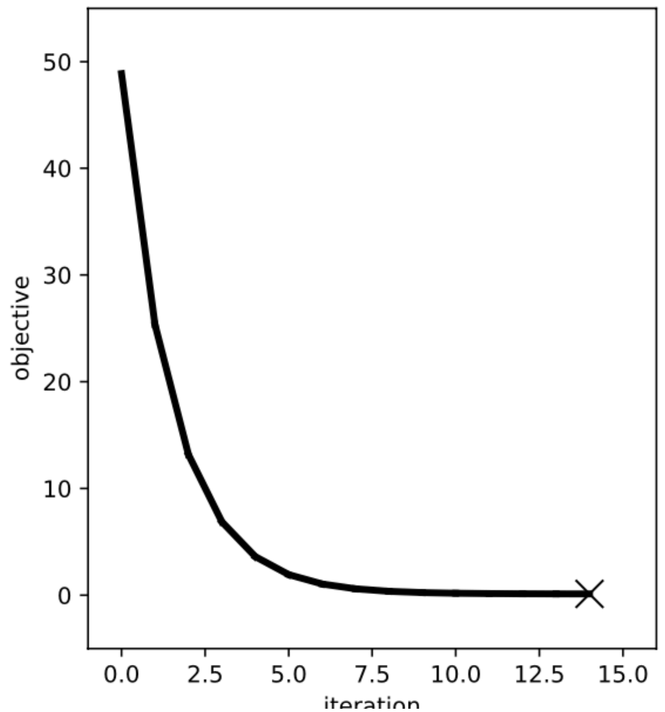


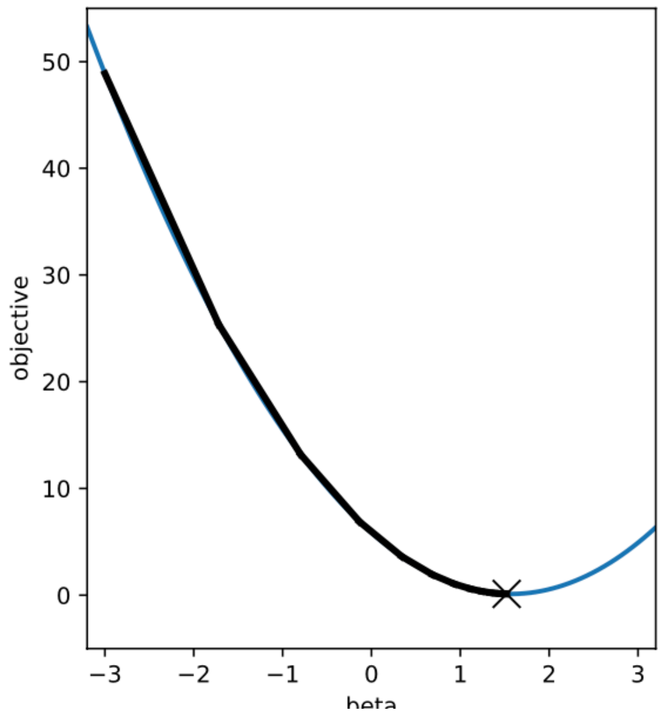


Another example: https://playground.tensorflow.org/

Recap: Optimisation for linear regression







Another example: https://playground.tensorflow.org/

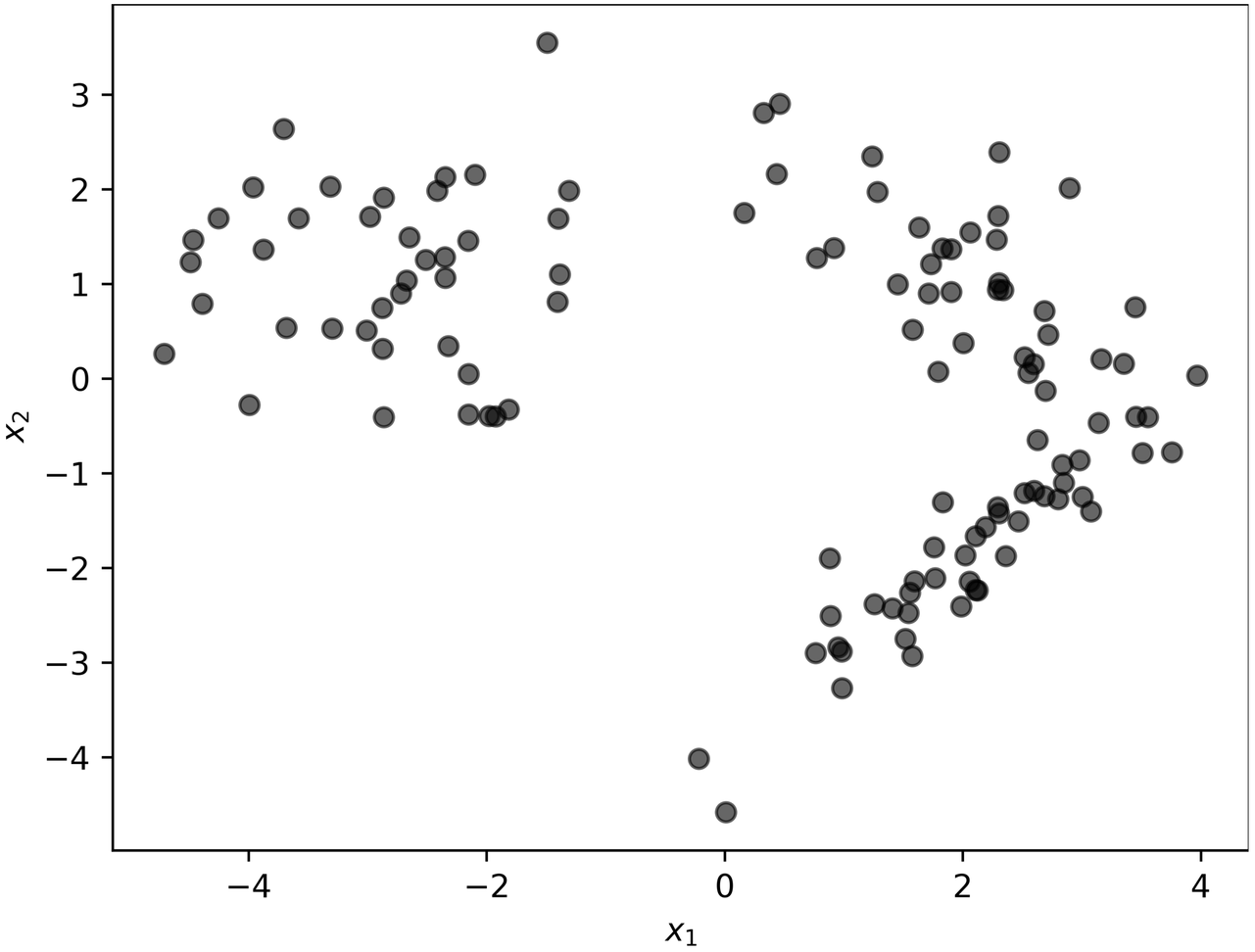
What is density estimation?

What is density estimation?

Goal: estimate the distribution your data come from, aka estimate $p(x)$

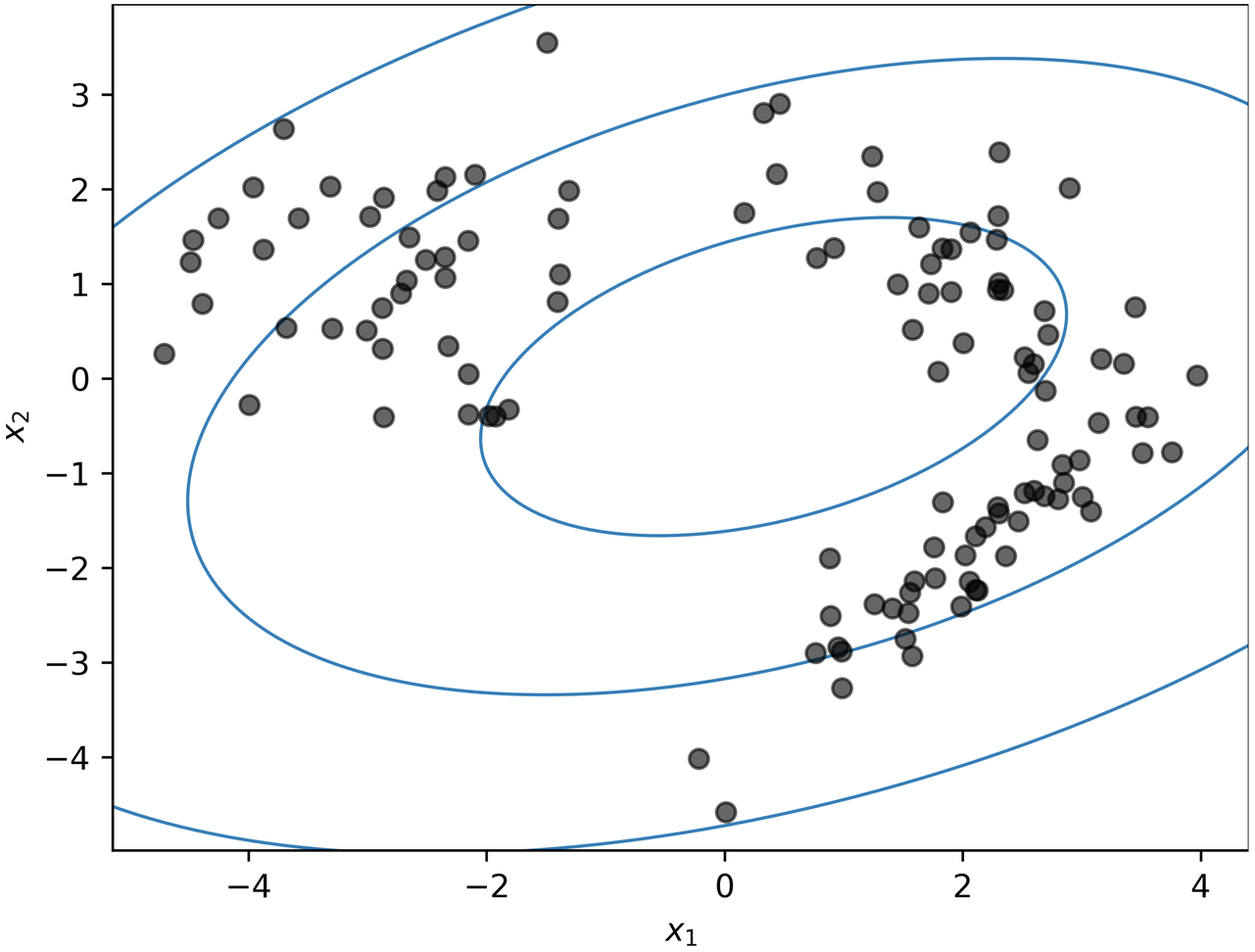
What is density estimation?

Goal: estimate the distribution your data come from, aka estimate $p(x)$



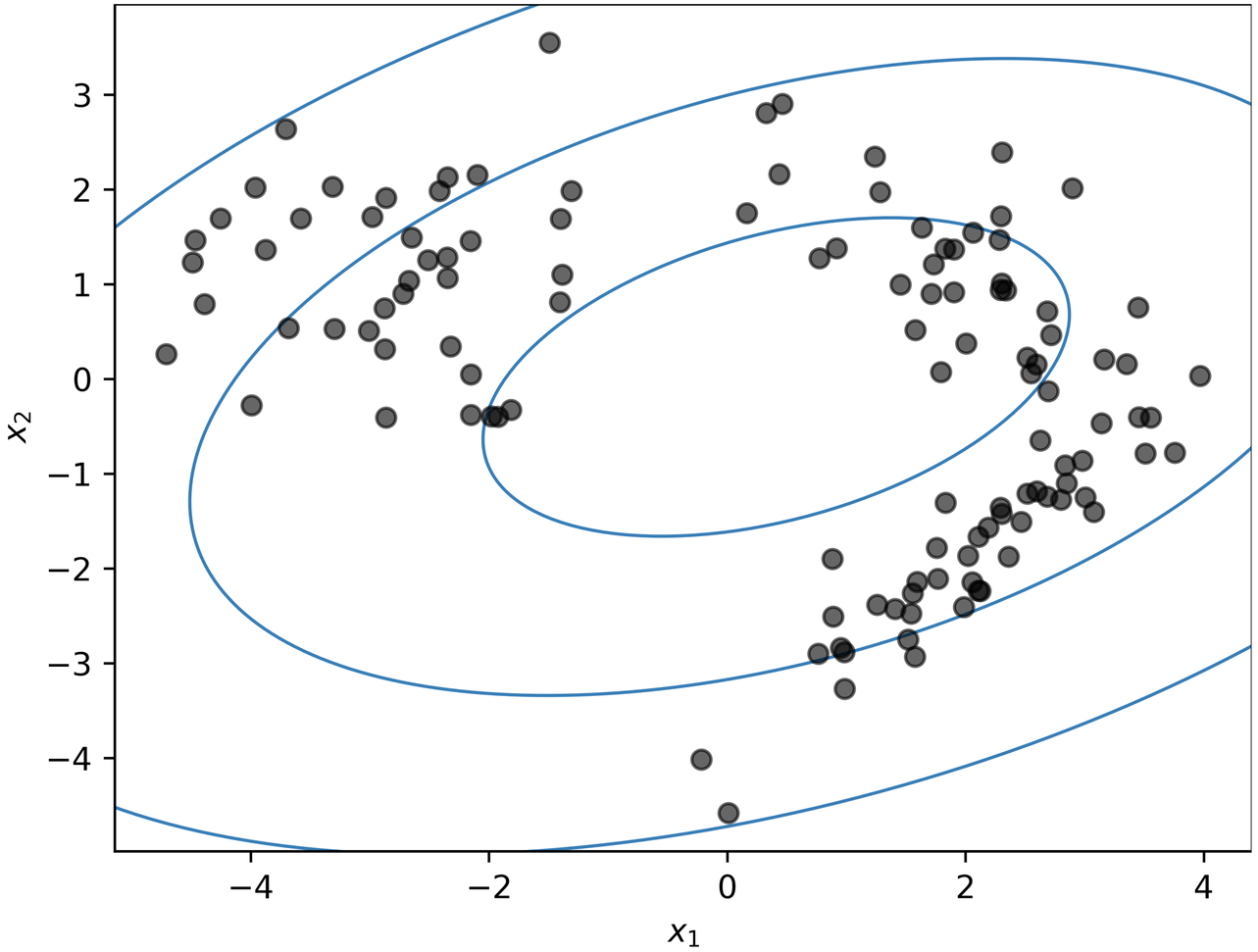
What is density estimation?

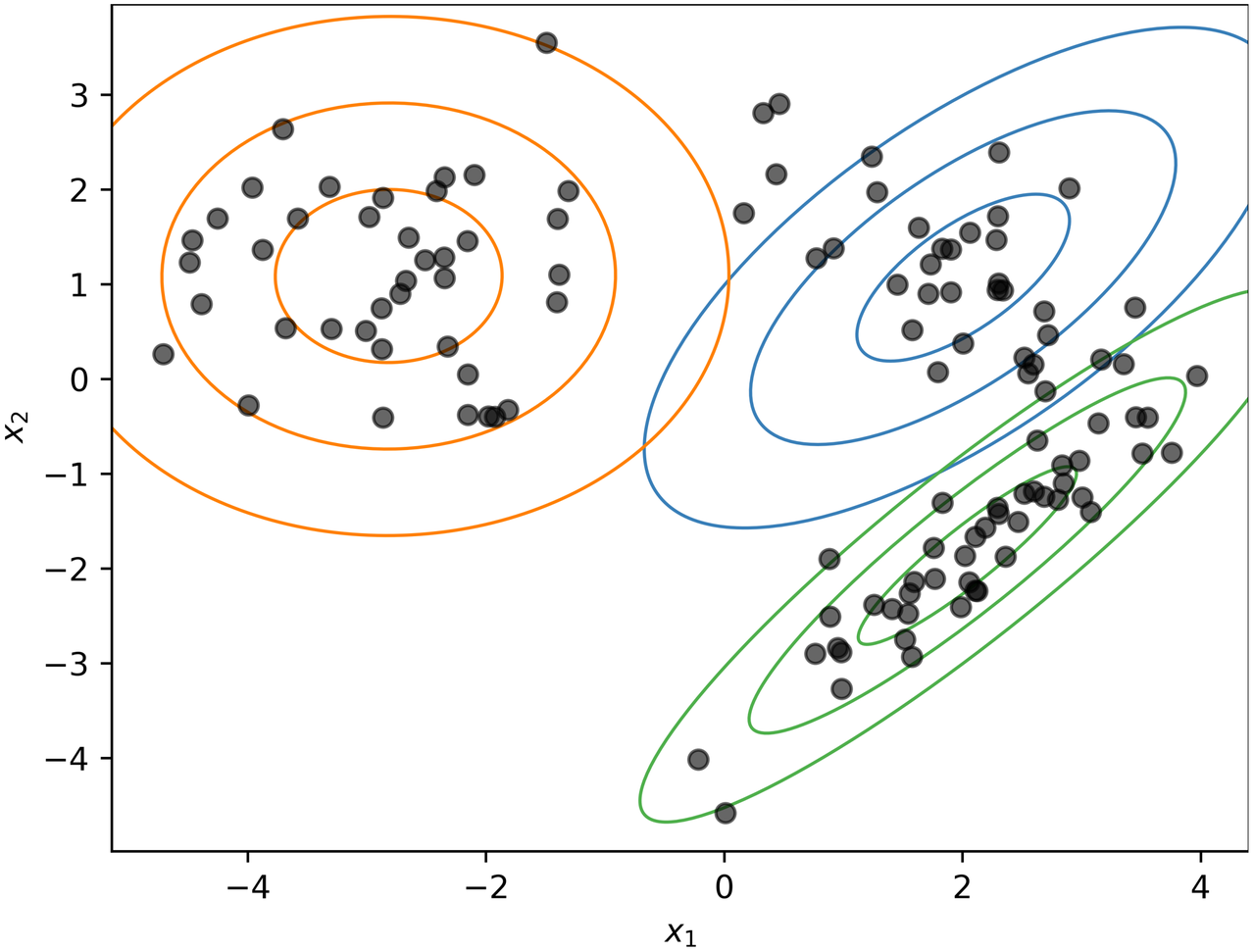
Goal: estimate the distribution your data come from, aka estimate $p(x)$



What is density estimation?

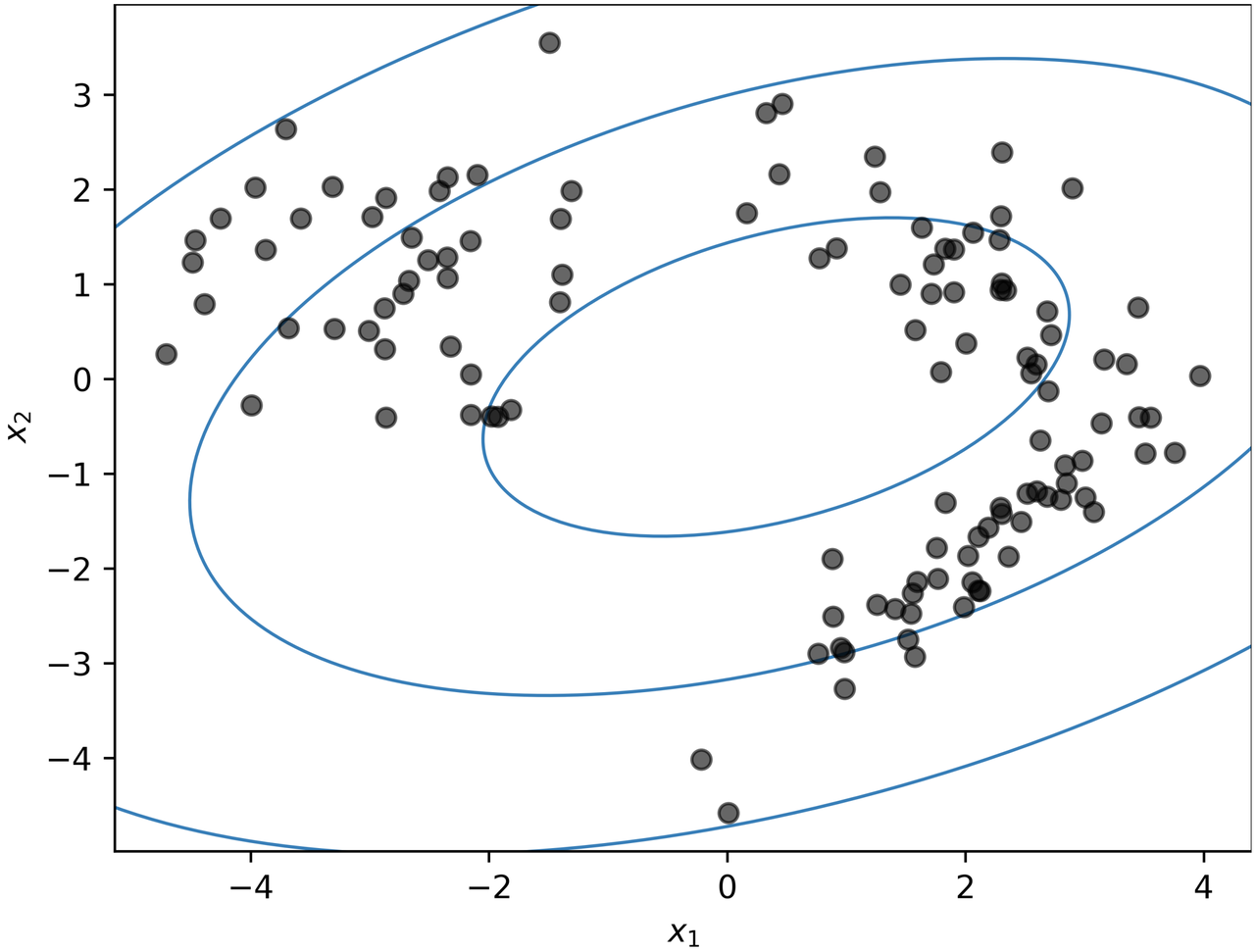
Goal: estimate the distribution your data come from, aka estimate $p(x)$

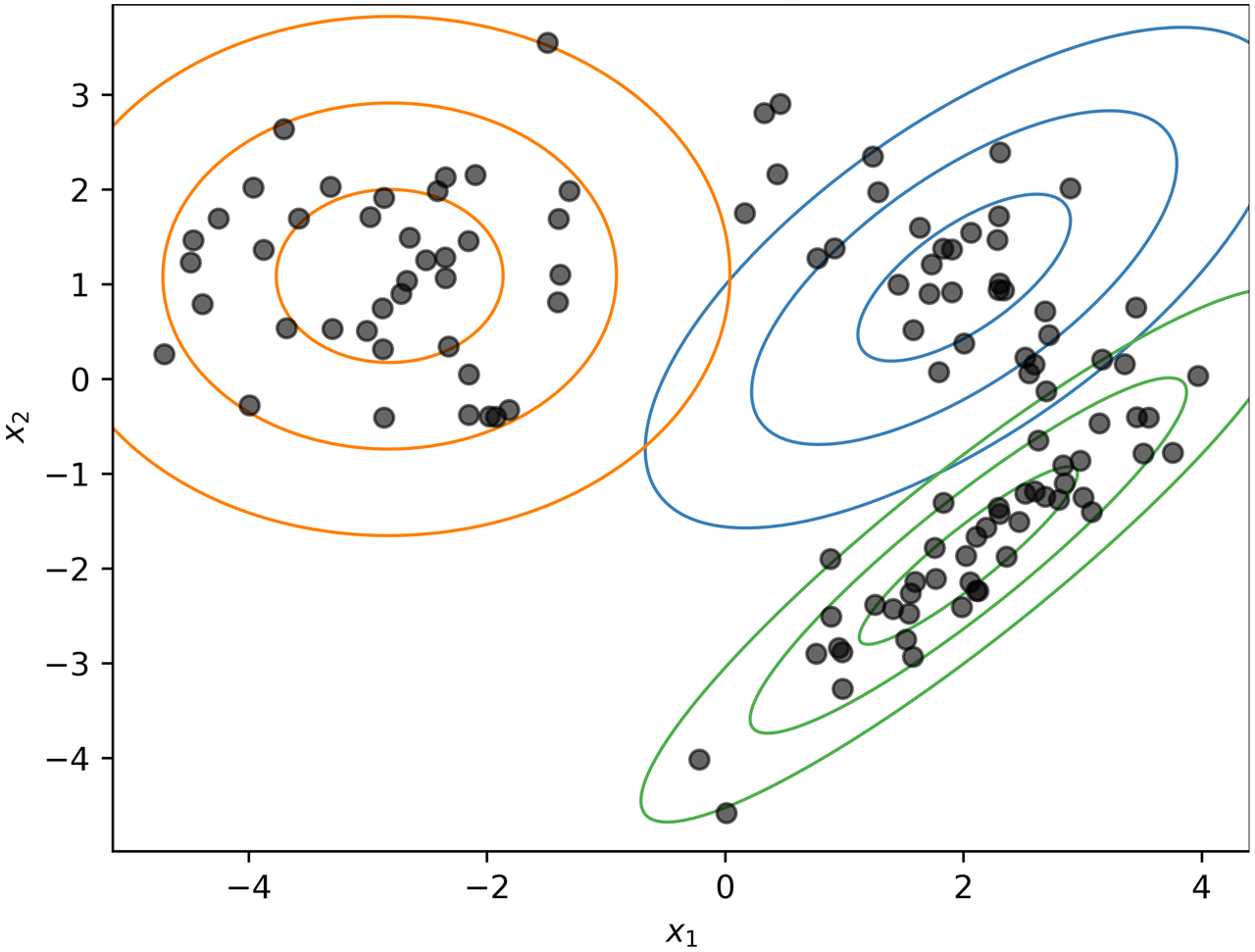




What is density estimation?

Goal: estimate the distribution your data come from, aka estimate $p(x)$



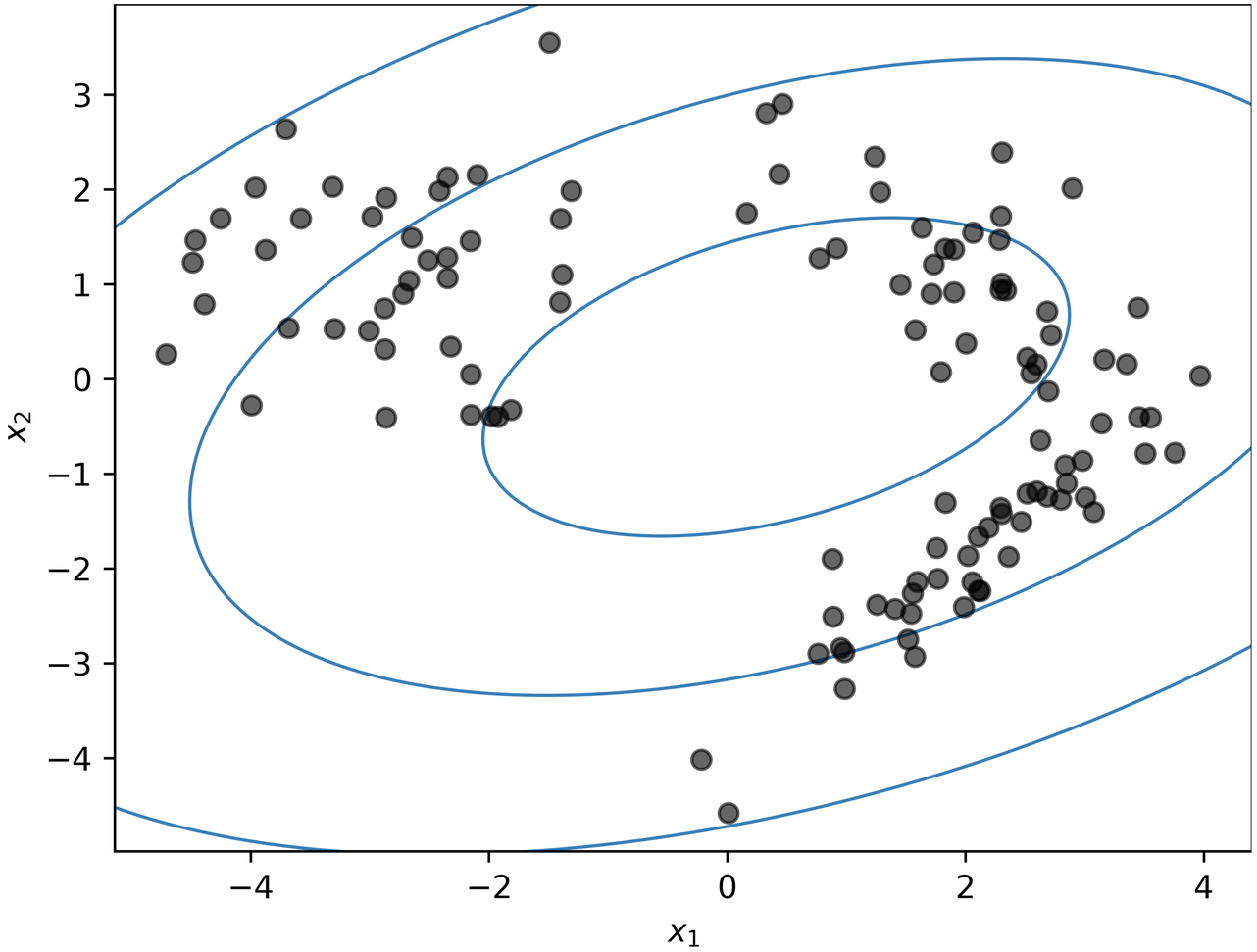


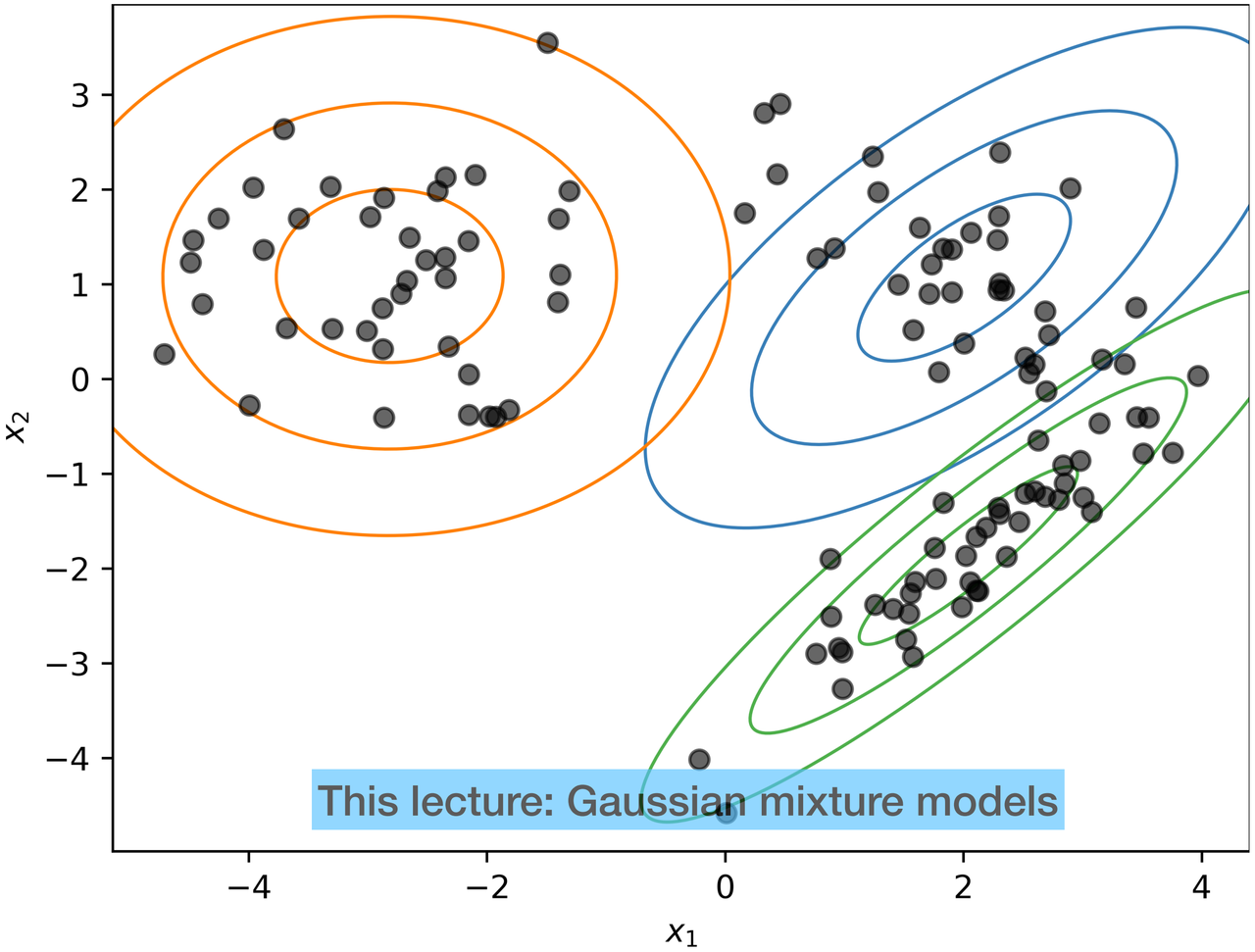
Applications: (1) exploratory analyses, (2) evaluation, (3) synthetic data generation

Are the data bimodal or skewed? Do datapoints come from the same density? Generate new samples that have the same characteristics

What is density estimation?

Goal: estimate the distribution your data come from, aka estimate $p(x)$





Applications: (1) exploratory analyses, (2) evaluation, (3) synthetic data generation

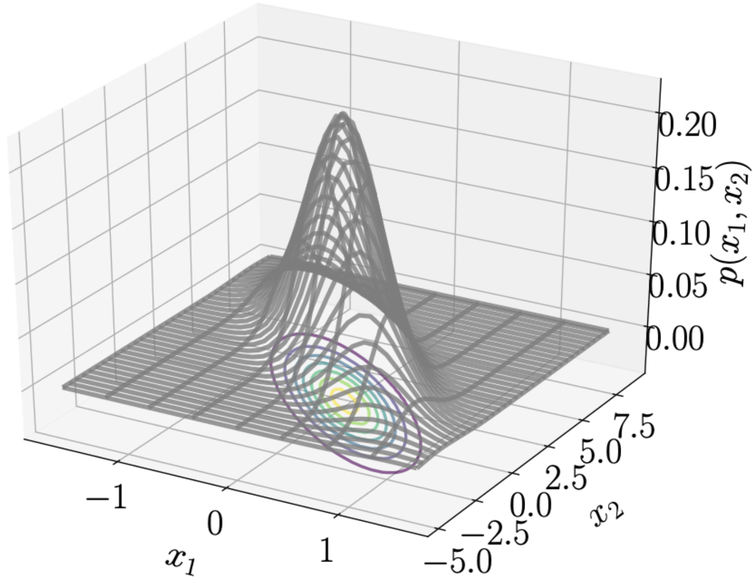
Are the data bimodal or skewed? Do datapoints come from the same density? Generate new samples that have the same characteristics

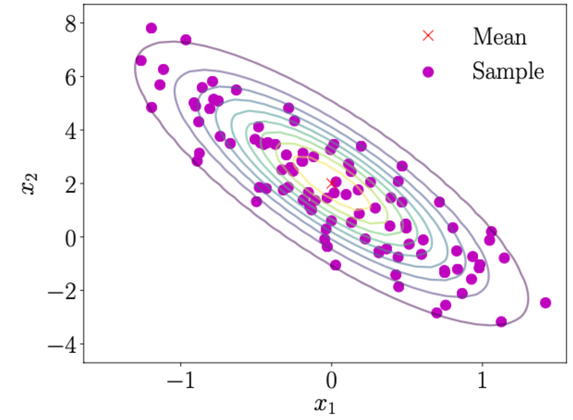
Recap: multivariate Gaussian distributions (W6)

A D-dimensional multivariate **Gaussian** or **normal** distribution is characterised by a *mean* vector μ and a *covariance matrix* Σ, with

pdf: $p(\mathbf{x} \mid \mu, \Sigma) = (2\pi)^{-\frac{D}{2}} \mid \Sigma \mid^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^T \Sigma^{-1}(\mathbf{x}-\mu)\right)$

We often write: $p(\mathbf{x} \mid \mu, \Sigma) = \mathcal{N}(\mathbf{x}; \mu, \Sigma)$ or $\mathbf{x} \sim \mathcal{N}(\mu, \Sigma)$



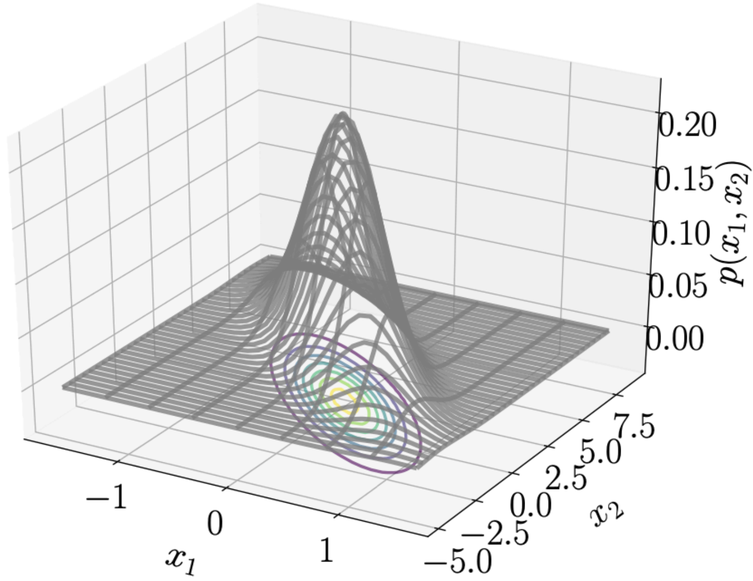


(b) Multivariate (two-dimensional) Gaussian, viewed from top. The red cross shows the mean and the colored lines show the contour lines of the density.

Recap: multivariate Gaussian distributions (W6)

A D-dimensional multivariate **Gaussian** or **normal** distribution is characterised by a *mean* vector μ and a *covariance matrix* Σ, with

pdf: $p(\mathbf{x}\,|\,\mu,\Sigma) = (2\pi)^{-\frac{D}{2}}\,|\,\Sigma\,|^{-\frac{1}{2}}\exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\right)$

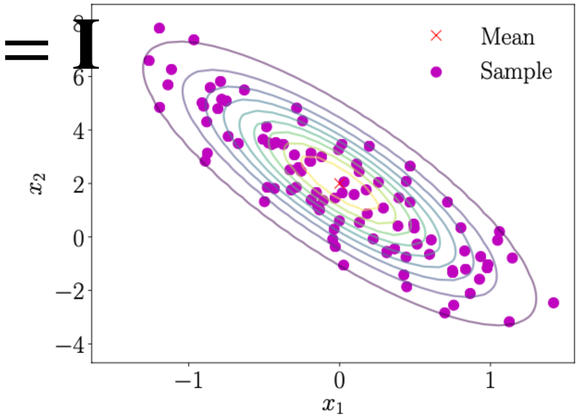


We often write: $p(\mathbf{x}\,|\,\mu,\Sigma) = \mathcal{N}(\mathbf{x};\mu,\Sigma)$ or $\mathbf{x} \sim \mathcal{N}(\mu,\Sigma)$

Standard multivariate **Gaussian** or **normal** distribution: $\mu = \mathbf{0}$ and $\Sigma = \mathbf{I}$

When Σ is a diagonal matrix, $p(\mathbf{x}\,|\,\mu,\Sigma) = \prod_{d=1}^{D} p(x_d\,|\,\mu_d,\Sigma_{d,d})$

Diagonal Gaussian, dimensions are independent

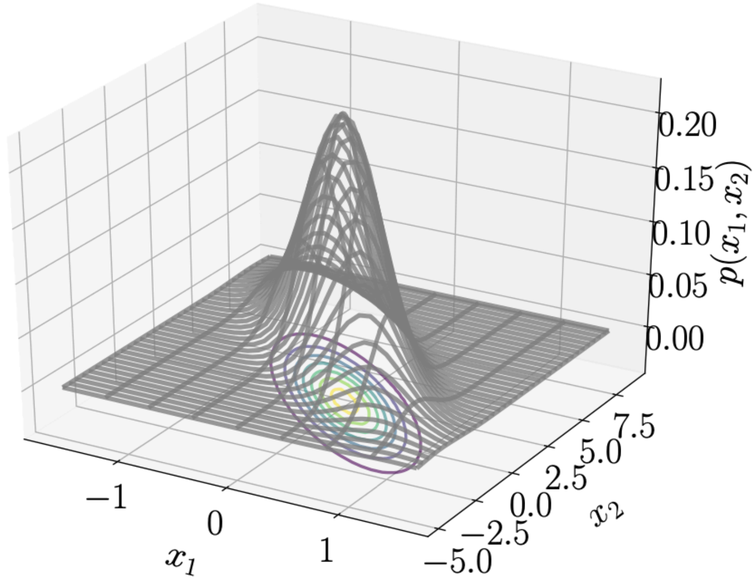


(b) Multivariate (two-dimensional) Gaussian, viewed from top. The red cross shows the mean and the colored lines show the contour lines of the density.

Recap: multivariate Gaussian distributions (W6)

A D-dimensional multivariate **Gaussian** or **normal** distribution is characterised by a *mean* vector μ and a *covariance matrix* Σ, with

pdf: $p(\mathbf{x}\,|\,\mu,\Sigma) = (2\pi)^{-\frac{D}{2}}\,|\,\Sigma\,|^{-\frac{1}{2}}\exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\right)$



We often write: $p(\mathbf{x}\,|\,\mu,\Sigma) = \mathcal{N}(\mathbf{x};\mu,\Sigma)$ or $\mathbf{x} \sim \mathcal{N}(\mu,\Sigma)$

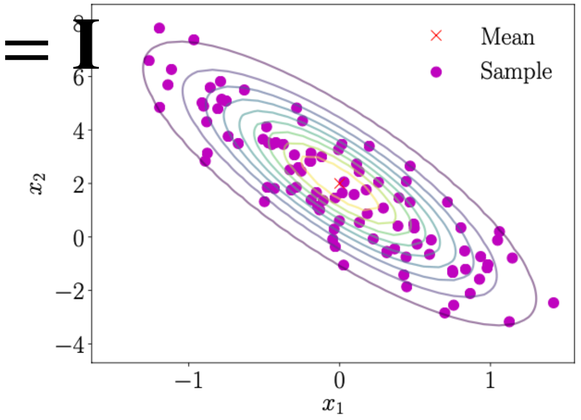
Standard multivariate **Gaussian** or **normal** distribution: $\mu = \mathbf{0}$ and $\Sigma = \mathbf{I}$

When Σ is a diagonal matrix, $p(\mathbf{x}\,|\,\mu,\Sigma) = \prod_{d=1}^{D} p(x_d\,|\,\mu_d,\Sigma_{d,d})$

Diagonal Gaussian, dimensions are independent

Alternative parameterisation, $\eta_1 = \Sigma^{-1}\mu$ and $\eta_2 = \Sigma^{-1}$ [precision]

Easier for multiplication/division and identifying conditional independence



(b) Multivariate (two-dimensional) Gaussian, viewed from top. The red cross shows the mean and the colored lines show the contour lines of the density.

(Gaussian) mixture models: from one to multiple densities

(Gaussian) mixture models: from one to multiple densities

Mixture models can be used to describe a distribution $p(x)$ by a *convex* combination of K simple (base) distributions

$$p(x) = \sum_{k=1}^{K} \pi_k p_k(x) \qquad 0 \leq \pi_k \leq 1, \sum_{k=1}^{K} \pi_k = 1$$

Why?

The components $p_k(x)$ are, typically, members of a family of basic distributions, e.g., Gaussians, Bernoullis, or Gammas, and π_k's are *mixture weights*.

(Gaussian) mixture models: from one to multiple densities

Mixture models can be used to describe a distribution $p(x)$ by a *convex* combination of K simple (base) distributions

$$p(x) = \sum_{k=1}^{K} \pi_k p_k(x) \qquad 0 \leq \pi_k \leq 1, \sum_{k=1}^{K} \pi_k = 1$$

Why?

The components $p_k(x)$ are, typically, members of a family of basic distributions, e.g., Gaussians, Bernoullis, or Gammas, and π_k's are *mixture weights*.

When the base distributions are Gaussian distributions $p_k(x) = \mathcal{N}(x; \mu_k, \Sigma_k)$, we have a **Gaussian mixture model (GMM)**:

$$p_\theta(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k) \qquad 0 \leq \pi_k \leq 1, \sum_{k=1}^{K} \pi_k = 1$$

$\theta = \{\pi_k, \mu_k, \Sigma_k \text{ for } k = 1,2,\ldots,K\}$ are the parameters of the GMM.

Gaussian mixture models: check your understanding

Model: $$p_\theta(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k) \qquad 0 \leq \pi_k \leq 1, \sum_{k=1}^{K} \pi_k = 1$$

where we defined $\theta = \{\pi_k, \mu_k, \Sigma_k \text{ for } k = 1,2,\ldots,K\}$ as the parameters of the GMM.

- Given a dataset generated by a mixture of 3 Gaussians, when we randomly sample a data point, it has the probability of 1/3 belonging to each Gaussian.
- A GMM is a linear combination of several Gaussian distributions.
- In GMMs, K (number of Gaussians) is a hyper-parameter.
- If a dataset is not generated by Gaussian distributions, it cannot be modelled by a GMM.

Gaussian mixture models: check your understanding

Model: $$p_\theta(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k) \qquad 0 \leq \pi_k \leq 1, \sum_{k=1}^{K} \pi_k = 1$$

where we defined $\theta = \{\pi_k, \mu_k, \Sigma_k \text{ for } k = 1,2,\ldots,K\}$ as the parameters of the GMM.

- Given a dataset generated by a mixture of 3 Gaussians, when we randomly sample a data point, it has the probability of 1/3 belonging to each Gaussian. No
- A GMM is a linear combination of several Gaussian distributions.
- In GMMs, K (number of Gaussians) is a hyper-parameter.
- If a dataset is not generated by Gaussian distributions, it cannot be modelled by a GMM.

Gaussian mixture models: check your understanding

Model: $$p_\theta(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k) \qquad 0 \leq \pi_k \leq 1, \sum_{k=1}^{K} \pi_k = 1$$

where we defined $\theta = \{\pi_k, \mu_k, \Sigma_k \text{ for } k = 1,2,\ldots,K\}$ as the parameters of the GMM.

- Given a dataset generated by a mixture of 3 Gaussians, when we randomly sample a data point, it has the probability of 1/3 belonging to each Gaussian. No
- A GMM is a linear combination of several Gaussian distributions. Yes
- In GMMs, K (number of Gaussians) is a hyper-parameter.
- If a dataset is not generated by Gaussian distributions, it cannot be modelled by a GMM.

Gaussian mixture models: check your understanding

Model: $$p_\theta(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k) \qquad 0 \leq \pi_k \leq 1, \sum_{k=1}^{K} \pi_k = 1$$

where we defined $\theta = \{\pi_k, \mu_k, \Sigma_k \text{ for } k = 1,2,\ldots,K\}$ as the parameters of the GMM.

- Given a dataset generated by a mixture of 3 Gaussians, when we randomly sample a data point, it has the probability of 1/3 belonging to each Gaussian. No
- A GMM is a linear combination of several Gaussian distributions. Yes
- In GMMs, K (number of Gaussians) is a hyper-parameter. Yes
- If a dataset is not generated by Gaussian distributions, it cannot be modelled by a GMM.

Gaussian mixture models: check your understanding

Model: $$p_\theta(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k) \qquad 0 \leq \pi_k \leq 1, \sum_{k=1}^{K} \pi_k = 1$$

where we defined $\theta = \{\pi_k, \mu_k, \Sigma_k \text{ for } k = 1,2,\ldots,K\}$ as the parameters of the GMM.

- Given a dataset generated by a mixture of 3 Gaussians, when we randomly sample a data point, it has the probability of 1/3 belonging to each Gaussian. No
- A GMM is a linear combination of several Gaussian distributions. Yes
- In GMMs, K (number of Gaussians) is a hyper-parameter. Yes
- If a dataset is not generated by Gaussian distributions, it cannot be modelled by a GMM. No

Gaussian mixture models: how to find θ

Gaussian mixture models: how to find θ

Model: $$p_\theta(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k) \qquad 0 \leq \pi_k \leq 1, \sum_{k=1}^{K} \pi_k = 1$$

where we defined $\theta = \{\pi_k, \mu_k, \Sigma_k$ for $k = 1,2,\ldots,K\}$ as the parameters of the GMM.

Gaussian mixture models: how to find θ

Model: $$p_\theta(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k) \qquad 0 \leq \pi_k \leq 1, \sum_{k=1}^{K} \pi_k = 1$$

where we defined $\theta = \{\pi_k, \mu_k, \Sigma_k \text{ for } k = 1,2,\ldots, K\}$ as the parameters of the GMM.

Data: $\mathbf{X} = \{x_1, x_2, \ldots, x_N\}$, where x_n's are drawn i.i.d. from an unknown distribution $p(x)$.

Gaussian mixture models: how to find θ

Model: $$p_\theta(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k) \qquad 0 \leq \pi_k \leq 1, \sum_{k=1}^{K} \pi_k = 1$$

where we defined $\theta = \{\pi_k, \mu_k, \Sigma_k \text{ for } k = 1,2,\ldots,K\}$ as the parameters of the GMM.

Data: $\mathbf{X} = \{x_1, x_2, \ldots, x_N\}$, where x_n's are drawn i.i.d. from an unknown distribution $p(x)$.

Objective: find a *good* approximation/representation of this unknown density $p(x)$ by means of a GMM with K components, that is finding θ.

Gaussian mixture models: how to find θ

Model: $$p_\theta(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k) \qquad 0 \le \pi_k \le 1, \sum_{k=1}^{K} \pi_k = 1$$

where we defined $\theta = \{\pi_k, \mu_k, \Sigma_k \text{ for } k = 1,2,\ldots,K\}$ as the parameters of the GMM.

Data: $\mathbf{X} = \{x_1, x_2, \ldots, x_N\}$, where x_n's are drawn i.i.d. from an unknown distribution $p(x)$.

Objective: find a *good* approximation/representation of this unknown density $p(x)$ by means of a GMM with K components, that is finding θ.

$$P(\theta \mid \mathscr{D}) = \frac{P(\theta, \mathscr{D})}{P(\mathscr{D})} = \frac{P(\theta)\, P(\mathscr{D} \mid \theta)}{P(\mathscr{D})}$$

Weeks 6 and 7

- Maximum likelihood (ML/MLE), $\text{argmax}_\theta\ p(\mathscr{D} \mid \theta)$
- Maximum a-posteriori (MAP), $\text{argmax}_\theta\ p(\mathscr{D} \mid \theta) p(\theta)$
- Exact/approximate Bayesian inference

Gaussian mixture models: how to find θ

Model: $$p_\theta(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k) \qquad 0 \leq \pi_k \leq 1, \sum_{k=1}^{K} \pi_k = 1$$

where we defined $\theta = \{\pi_k, \mu_k, \Sigma_k \text{ for } k = 1,2,\ldots,K\}$ as the parameters of the GMM.

Data: $\mathbf{X} = \{x_1, x_2, \ldots, x_N\}$, where x_n's are drawn i.i.d. from an unknown distribution $p(x)$.

Objective: find a *good* approximation/representation of this unknown density $p(x)$ by means of a GMM with K components, that is finding θ.

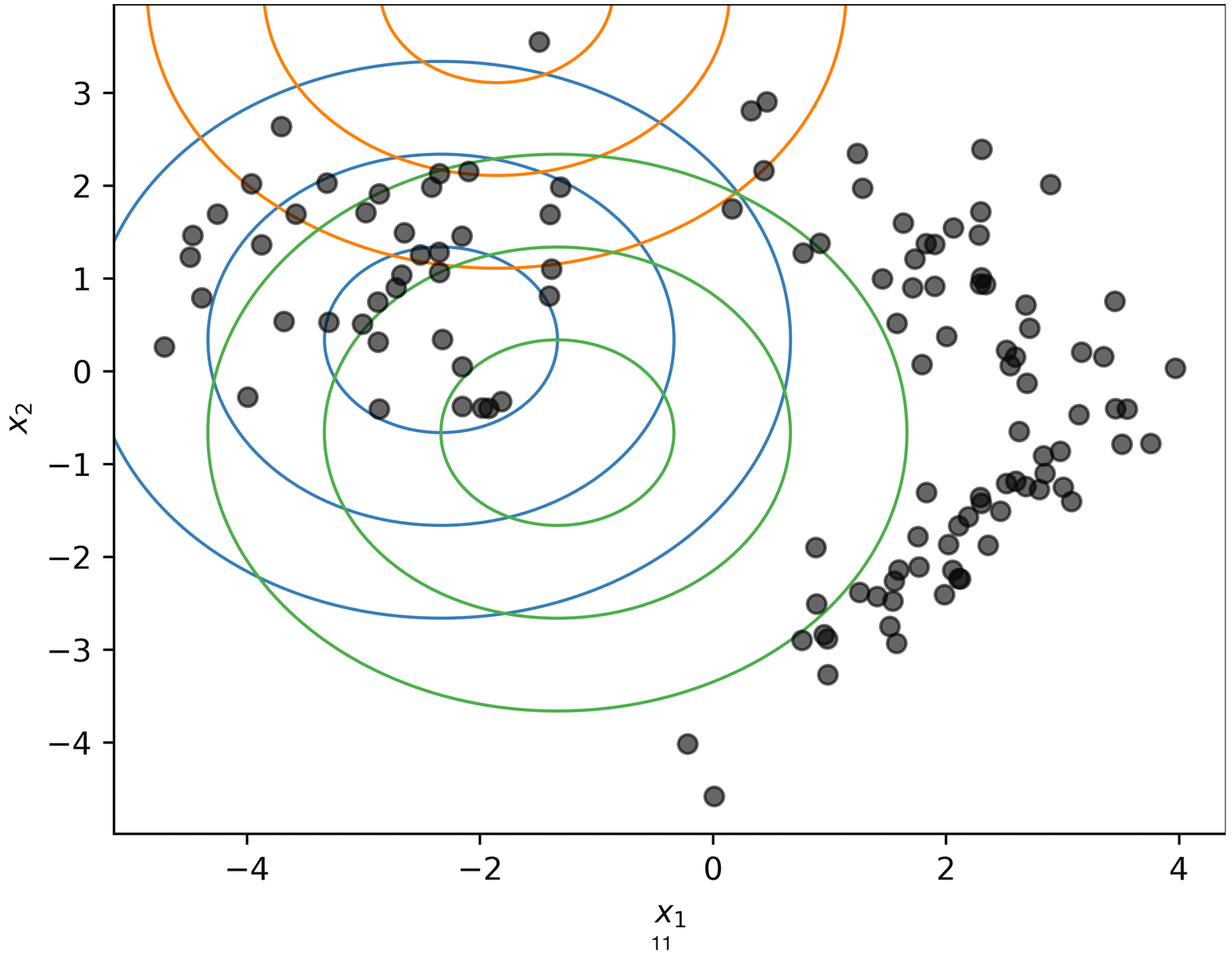
$$P(\theta \mid \mathcal{D}) = \frac{P(\theta, \mathcal{D})}{P(\mathcal{D})} = \frac{P(\theta)\, P(\mathcal{D} \mid \theta)}{P(\mathcal{D})}$$

Weeks 6 and 7

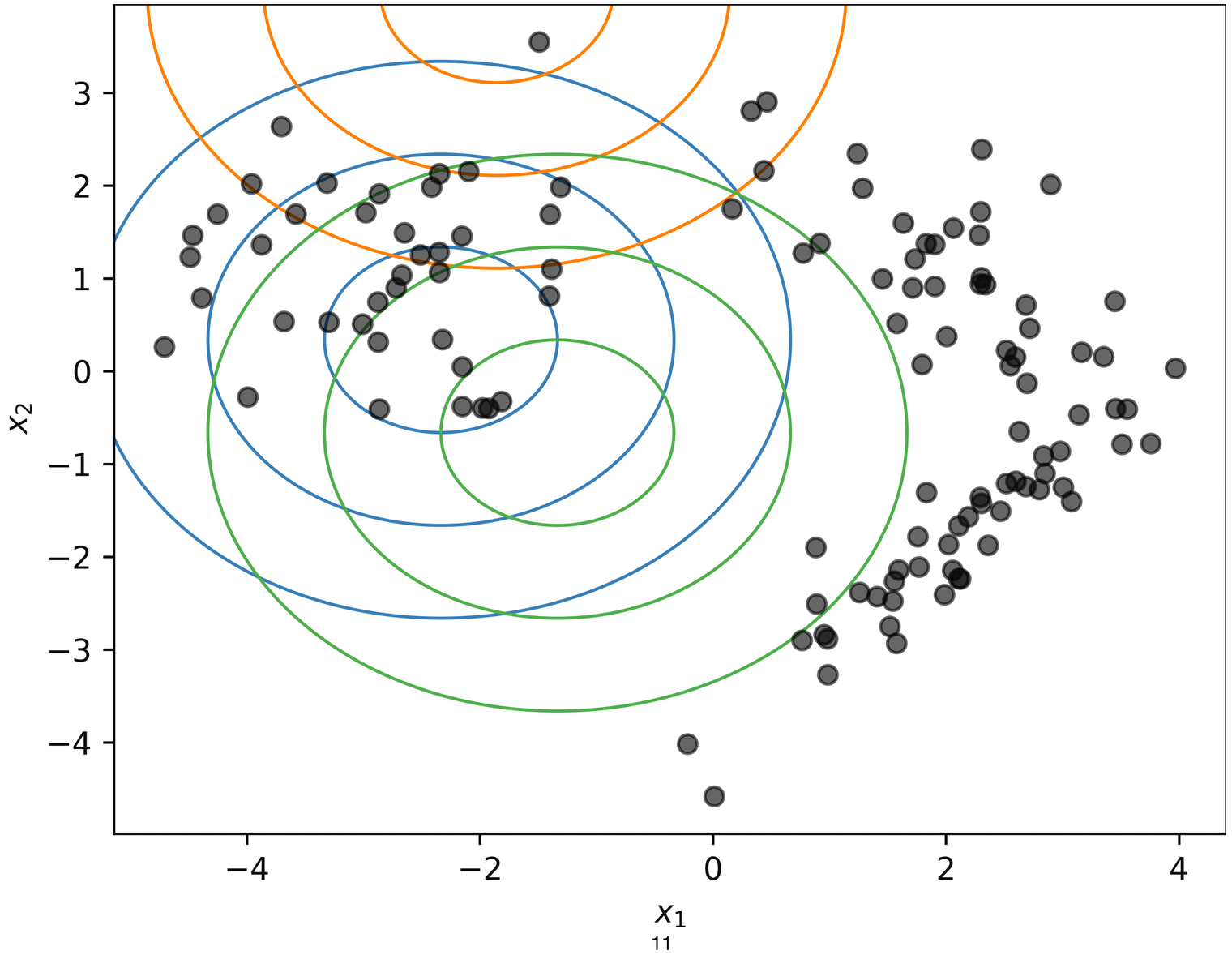
This week

- Maximum likelihood (ML/MLE), $\text{argmax}_\theta\ p(\mathcal{D} \mid \theta)$
- Maximum a-posteriori (MAP), $\text{argmax}_\theta\ p(\mathcal{D} \mid \theta)p(\theta)$
- Exact/approximate Bayesian inference

Maximum likelihood - Sneak peek



Maximum likelihood - Sneak peek



GMMs - likelihood

GMMs - likelihood

The likelihood, i.e., the predictive distribution of the training data given the parameters:

$$p(X \mid \theta) = \prod_{n=1}^{N} p(x_n \mid \theta) = \prod_{n=1}^{N} \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Factorise due to iid assumption

Each individual likelihood term is a Gaussian mixture density

GMMs - likelihood

The likelihood, i.e., the predictive distribution of the training data given the parameters:

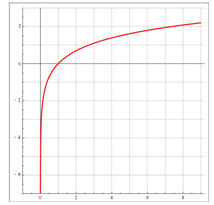
$$p(X \mid \theta) = \prod_{n=1}^{N} p(x_n \mid \theta) = \prod_{n=1}^{N} \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Factorise due to iid assumption

Each individual likelihood term is a Gaussian mixture density

Goal: find maximise likelihood, $\text{argmax}_\theta \; p(X \mid \theta)$, which is equivalent to $\text{argmax}_\theta \; \log p(X \mid \theta)$

as log is monotonically increasing



GMMs - likelihood

The likelihood, i.e., the predictive distribution of the training data given the parameters:

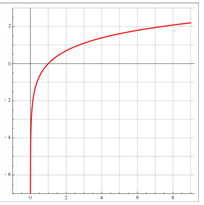
$$p(X \mid \theta) = \prod_{n=1}^{N} p(x_n \mid \theta) = \prod_{n=1}^{N} \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Factorise due to iid assumption

Each individual likelihood term is a Gaussian mixture density

Goal: find maximise likelihood, $\text{argmax}_\theta \; p(X \mid \theta)$, which is equivalent to $\text{argmax}_\theta \; \log p(X \mid \theta)$

as log is monotonically increasing



Log-likelihood: $L(\theta) = \log p(X \mid \theta) = \sum_{n=1}^{N} \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$

Strategy: find the gradients of $L(\theta)$ and try to set them to zero.

GMMs - log-likelihood gradients

Log-likelihood: $L(\theta) = \log p(X|\theta) = \sum_{n=1}^{N} \log p(x_n|\theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$

GMMs - log-likelihood gradients

Log-likelihood: $L(\theta) = \log p(X|\theta) = \sum_{n=1}^{N} \log p(x_n|\theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$

Strategy: find the gradients of $L(\theta)$ and try to set them to zero.

$$\frac{\partial}{\partial \mu_k} L = \sum_{n=1}^{N} \frac{\partial \log p(x_n|\theta)}{\partial \mu_k} = \mathbf{0}^\intercal \qquad \frac{\partial}{\partial \Sigma_k} L = \sum_{n=1}^{N} \frac{\partial \log p(x_n|\theta)}{\partial \Sigma_k} = \mathbf{0} \qquad \frac{\partial}{\partial \pi_k} L = \sum_{n=1}^{N} \frac{\partial \log p(x_n|\theta)}{\partial \pi_k} = 0$$

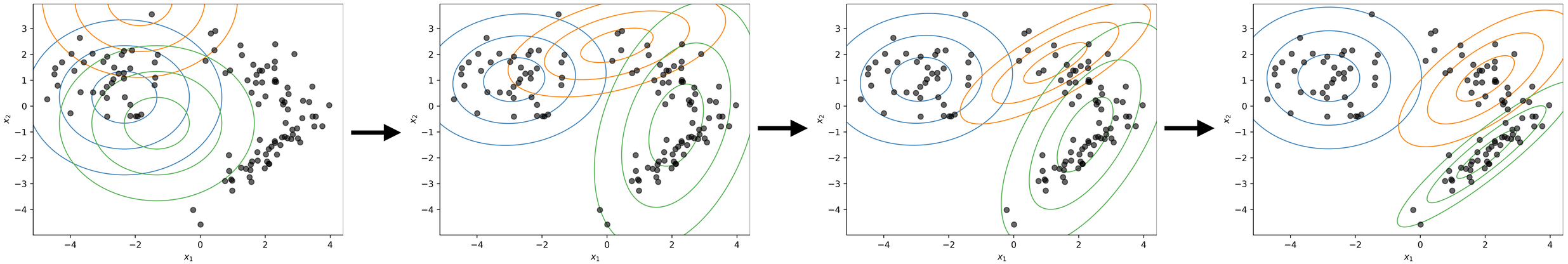
GMMs - log-likelihood gradients

Log-likelihood: $L(\theta) = \log p(X|\theta) = \sum_{n=1}^{N} \log p(x_n|\theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$

Strategy: find the gradients of $L(\theta)$ and try to set them to zero.

$$\frac{\partial}{\partial \mu_k} L = \sum_{n=1}^{N} \frac{\partial \log p(x_n|\theta)}{\partial \mu_k} = \mathbf{0}^\intercal \qquad \frac{\partial}{\partial \Sigma_k} L = \sum_{n=1}^{N} \frac{\partial \log p(x_n|\theta)}{\partial \Sigma_k} = \mathbf{0} \qquad \frac{\partial}{\partial \pi_k} L = \sum_{n=1}^{N} \frac{\partial \log p(x_n|\theta)}{\partial \pi_k} = 0$$

Unlike linear regression, the optimal parameters are not analytically tractable. We resort to an iterative optimisation procedure.



Initialisation *After 50 iterations*

GMMs - iterative procedure

Initialise $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$

Repeat until convergence:

1. **Evaluate responsibilities** r_{nk} for every data point x_n using current parameters:

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}$$

2. **Re-estimate parameters** $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$ using the current responsibilities r_{nk}:

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^N r_{nk} x_n, \quad N_k = \sum_{n=1}^N r_{nk}$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^N r_{nk} (x_n - \mu_k)(x_n - \mu_k)^\intercal$$

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

GMMs - responsibility definition

GMMs - responsibility definition

We define **responsibility** r_{nk} for every data point x_n:

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_j \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}$$

GMMs - responsibility definition

We define **responsibility** r_{nk} for every data point x_n:

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_j \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}$$

proportional to the likelihood of the k-th mixture component given the data point x_n

GMMs - responsibility definition

We define **responsibility** r_{nk} for every data point x_n:

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_j \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}$$

proportional to the likelihood of the k-th mixture component given the data point x_n

Note that $\mathbf{r}_n = [r_{n1}, r_{n2}, \ldots, r_{nK}]^\intercal \in \mathbb{R}^D$ is a probability vector, i.e., $\sum_k r_{nk} = 1$ with $r_{nk} \geq 0$

This probability vector distributes probability mass among the K mixture components, and we can think of $\mathbf{r}_n$ as a "soft assignment" of x_n to the K mixture components.

GMMs - responsibility definition

We define **responsibility** r_{nk} for every data point x_n:

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_j \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}$$

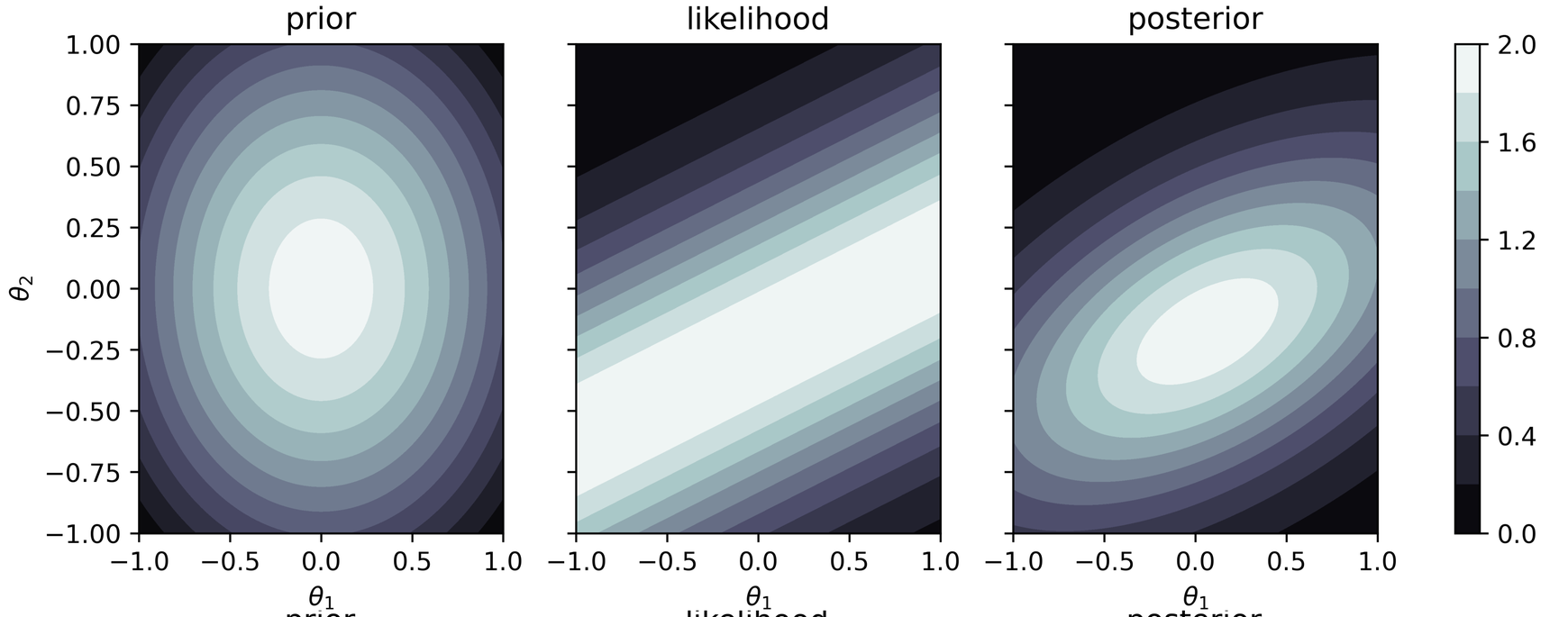
proportional to the likelihood of the k-th mixture component given the data point x_n

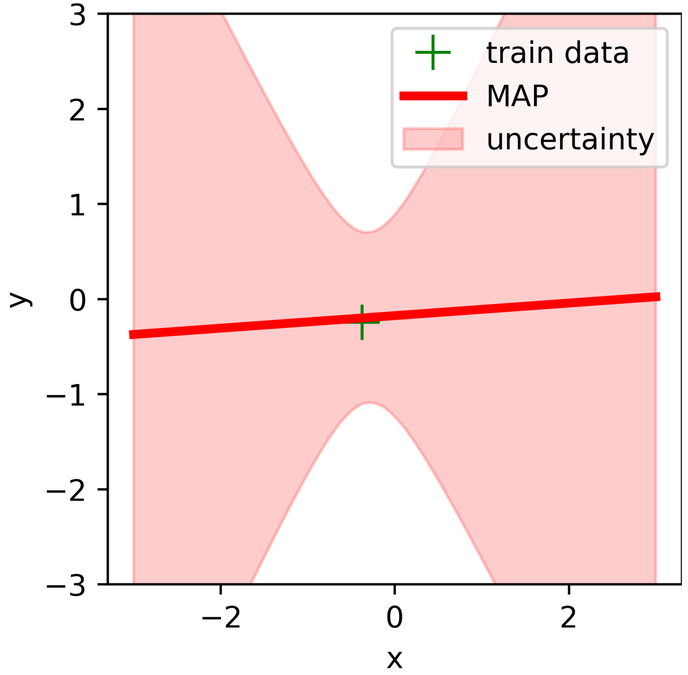
Note that $\mathbf{r}_n = [r_{n1}, r_{n2}, \ldots, r_{nK}]^\intercal \in \mathbb{R}^D$ is a probability vector, i.e., $\sum_k r_{nk} = 1$ with $r_{nk} \geq 0$

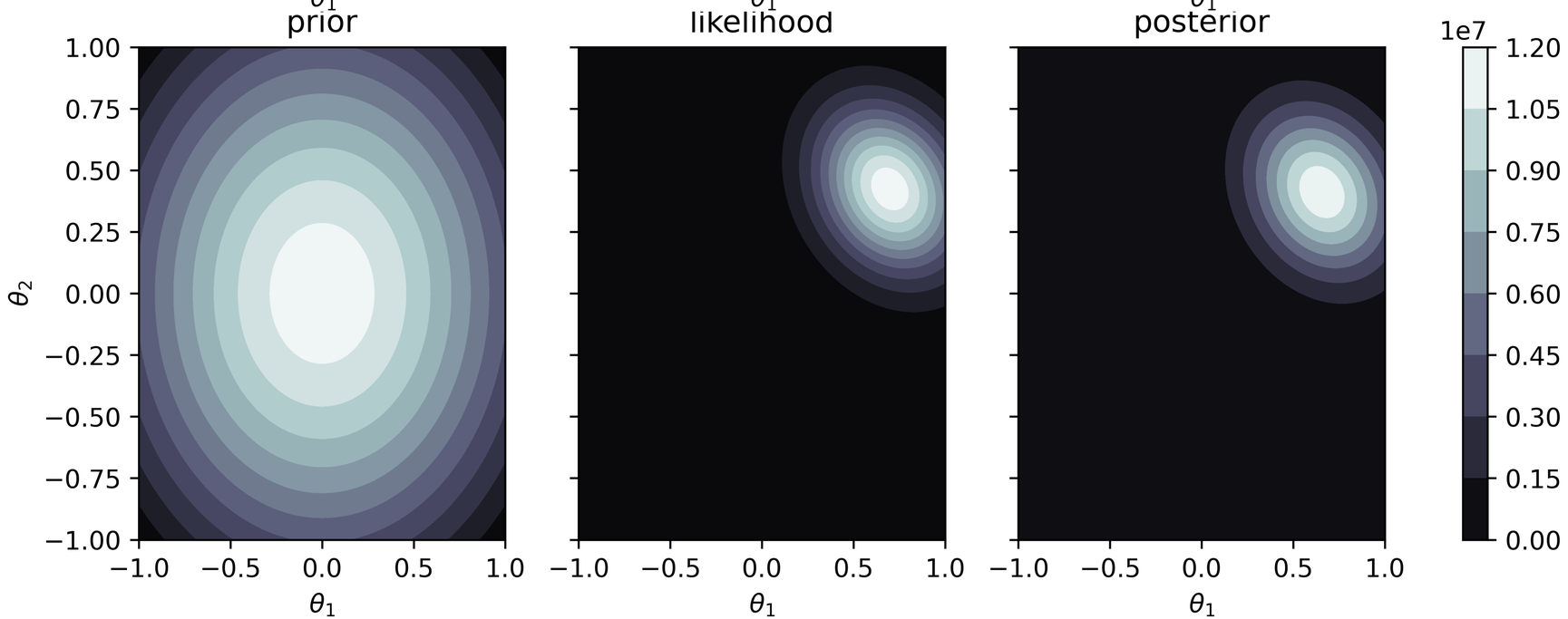
This probability vector distributes probability mass among the K mixture components, and we can think of $\mathbf{r}_n$ as a "soft assignment" of x_n to the K mixture components.

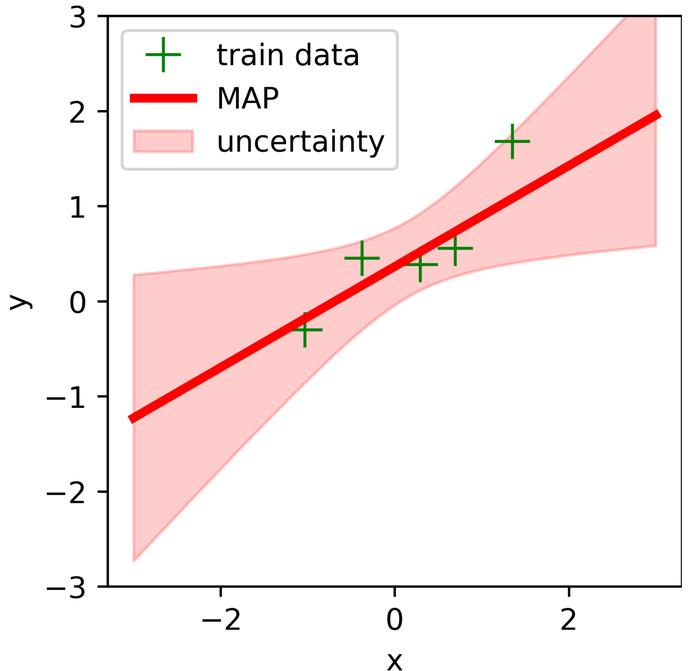
Let's now derive the parameter updates in step 2!

More on Bayesian linear regression









Overview

Last lecture:

1. **Model**: Gaussian mixture models
2. **Learning**: parameter learning using *maximum likelihood*
3. **Iterative** procedure, derive mean update

Initialise $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$

Repeat until convergence:

1. **Evaluate responsibilities** r_{nk} for every data point x_n using current parameters:

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}$$

2. **Re-estimate parameters** $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$ using the current responsibilities r_{nk}:

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^N r_{nk} x_n, \quad N_k = \sum_{n=1}^N r_{nk}$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^N r_{nk} (x_n - \mu_k)(x_n - \mu_k)^\intercal$$

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

This lecture:

1. Derive covariance and mixture weight updates
2. Issues with maximum likelihood and workarounds
3. K-means as a special case
4. Theoretical foundation: EM algorithm (1 is E step, 2 is M step)

Recap: Gaussian mixture models and how to find θ

Model: $$p_\theta(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k) \qquad 0 \leq \pi_k \leq 1, \sum_{k=1}^{K} \pi_k = 1$$

where we defined $\theta = \{\pi_k, \mu_k, \Sigma_k \text{ for } k = 1,2,\ldots,K\}$ as the parameters of the GMM.

Data: $\mathbf{X} = \{x_1, x_2, \ldots, x_N\}$, where x_n's are drawn i.i.d. from an unknown distribution $p(x)$.

Objective: find a *good* approximation/representation of this unknown density $p(x)$ by means of a GMM with K components, that is finding θ.

$$P(\theta \mid \mathcal{D}) = \frac{P(\theta, \mathcal{D})}{P(\mathcal{D})} = \frac{P(\theta)\, P(\mathcal{D} \mid \theta)}{P(\mathcal{D})}$$

Weeks 6 and 7

This week

- Maximum likelihood (ML/MLE), $\text{argmax}_\theta\ p(\mathcal{D} \mid \theta)$
- Maximum a-posteriori (MAP), $\text{argmax}_\theta\ p(\mathcal{D} \mid \theta)p(\theta)$
- Exact/approximate Bayesian inference

Recap: GMMs - iterative procedure

Initialise $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$

Repeat until convergence:

1. **Evaluate responsibilities** r_{nk} for every data point x_n using current parameters:

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}$$

2. **Re-estimate parameters** $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$ using the current responsibilities r_{nk}:

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^N r_{nk} x_n, \quad N_k = \sum_{n=1}^N r_{nk}$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^N r_{nk} (x_n - \mu_k)(x_n - \mu_k)^\intercal$$

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

Recap: Step 2, mean update

$$L(\theta) = \sum_{n} \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Log likelihood

Recap: Step 2, mean update

$$L(\theta) = \sum_n \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Log likelihood

$$\frac{\partial L(\theta)}{\partial \mu_k} = \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \frac{\partial}{\partial \mu_k} \sum_{j=1}^{K} \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)$$

gradient of log likelihood wrt mean

Recap: Step 2, mean update

$$L(\theta) = \sum_n \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Log likelihood

$$\frac{\partial L(\theta)}{\partial \mu_k} = \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \frac{\partial}{\partial \mu_k} \sum_{j=1}^{K} \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)$$

gradient of log likelihood wrt mean

$$= \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \pi_k \frac{\partial}{\partial \mu_k} \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Ignore terms that do not depend on k

Recap: Step 2, mean update

$$L(\theta) = \sum_n \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Log likelihood

$$\frac{\partial L(\theta)}{\partial \mu_k} = \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \frac{\partial}{\partial \mu_k} \sum_{j=1}^{K} \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)$$

gradient of log likelihood wrt mean

$$= \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \pi_k \frac{\partial}{\partial \mu_k} \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Ignore terms that do not depend on k

$$= \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k) \left(x_n - \mu_k\right)^\intercal \Sigma_k^{-1}$$

Gradient of Gaussian wrt mean

Recap: Step 2, mean update

$$L(\theta) = \sum_n \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Log likelihood

$$\frac{\partial L(\theta)}{\partial \mu_k} = \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \frac{\partial}{\partial \mu_k} \sum_{j=1}^{K} \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)$$

gradient of log likelihood wrt mean

$$= \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \pi_k \frac{\partial}{\partial \mu_k} \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Ignore terms that do not depend on k

$$= \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k) \left(x_n - \mu_k\right)^\intercal \Sigma_k^{-1}$$

Gradient of Gaussian wrt mean

$$= \sum_{n=1}^{N} r_{nk} \left(x_n - \mu_k\right)^\intercal \Sigma_k^{-1}$$

Rewrite using responsibility

Recap: Step 2, mean update

$$L(\theta) = \sum_n \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Log likelihood

$$\frac{\partial L(\theta)}{\partial \mu_k} = \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \frac{\partial}{\partial \mu_k} \sum_{j=1}^{K} \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)$$

gradient of log likelihood wrt mean

$$= \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \pi_k \frac{\partial}{\partial \mu_k} \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Ignore terms that do not depend on k

$$= \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k) \left(x_n - \mu_k\right)^\intercal \Sigma_k^{-1}$$

Gradient of Gaussian wrt mean

$$= \sum_{n=1}^{N} r_{nk} \left(x_n - \mu_k\right)^\intercal \Sigma_k^{-1}$$

Rewrite using responsibility

$$\frac{\partial L(\theta)}{\partial \mu_k} = \mathbf{0}^\intercal \rightarrow \mu_k = \frac{\sum_n r_{nk} x_n}{\sum_n r_{nk}}$$

Set gradient to zero

Step 2, covariance update [1]

$$L(\theta) = \sum_{n} \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Log likelihood

Step 2, covariance update [1]

$$L(\theta) = \sum_n \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Log likelihood

$$\frac{\partial L(\theta)}{\partial \Sigma_k} = \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \frac{\partial}{\partial \Sigma_k} \sum_{j=1}^{K} \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)$$

gradient of log likelihood wrt cov

$$= \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \pi_k \frac{\partial}{\partial \Sigma_k} \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Ignore terms that do not depend on k

Step 2, covariance update [1]

$$L(\theta) = \sum_n \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Log likelihood

$$\frac{\partial L(\theta)}{\partial \Sigma_k} = \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \frac{\partial}{\partial \Sigma_k} \sum_{j=1}^{K} \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)$$

gradient of log likelihood wrt cov

$$= \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \pi_k \frac{\partial}{\partial \Sigma_k} \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Ignore terms that do not depend on k

$$\frac{\partial}{\partial \Sigma_k} \mathcal{N}(x_n; \mu_k, \Sigma_k) = \frac{\partial}{\partial \Sigma_k} \left[(2\pi)^{-\frac{D}{2}} \mid \Sigma_k \mid^{-\frac{1}{2}} \exp\left(-\frac{1}{2} (x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k) \right) \right]$$

Gradient of Gaussian wrt cov

Step 2, covariance update [1]

$$L(\theta) = \sum_n \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Log likelihood

$$\frac{\partial L(\theta)}{\partial \Sigma_k} = \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \frac{\partial}{\partial \Sigma_k} \sum_{j=1}^{K} \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)$$

gradient of log likelihood wrt cov

$$= \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \pi_k \frac{\partial}{\partial \Sigma_k} \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Ignore terms that do not depend on k

$$\frac{\partial}{\partial \Sigma_k} \mathcal{N}(x_n; \mu_k, \Sigma_k) = \frac{\partial}{\partial \Sigma_k} \left[(2\pi)^{-\frac{D}{2}} \mid \Sigma_k \mid^{-\frac{1}{2}} \exp\left(-\frac{1}{2} (x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k) \right) \right]$$

Gradient of Gaussian wrt cov

$$= (2\pi)^{-\frac{D}{2}} \frac{-1}{2} \mid \Sigma_k \mid^{-\frac{1}{2}} \Sigma_k^{-1} \exp\left(-\frac{1}{2} (x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k) \right)$$

$$+ (2\pi)^{-\frac{D}{2}} \mid \Sigma_k \mid^{-\frac{1}{2}} \exp\left(-\frac{1}{2} (x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k) \right) \frac{1}{2} \Sigma_k^{-1} (x_n - \mu_k)(x_n - \mu_k)^\intercal \Sigma_k^{-1}$$

Step 2, covariance update [1]

$$L(\theta) = \sum_n \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Log likelihood

$$\frac{\partial L(\theta)}{\partial \Sigma_k} = \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \frac{\partial}{\partial \Sigma_k} \sum_{j=1}^{K} \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)$$

gradient of log likelihood wrt cov

$$= \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \pi_k \frac{\partial}{\partial \Sigma_k} \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Ignore terms that do not depend on k

$$\frac{\partial}{\partial \Sigma_k} \mathcal{N}(x_n; \mu_k, \Sigma_k) = \frac{\partial}{\partial \Sigma_k} \left[(2\pi)^{-\frac{D}{2}} |\Sigma_k|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k) \right) \right]$$

Gradient of Gaussian wrt cov

$$= (2\pi)^{-\frac{D}{2}} \frac{-1}{2} |\Sigma_k|^{-\frac{1}{2}} \Sigma_k^{-1} \exp\left(-\frac{1}{2}(x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k) \right)$$

$$+ (2\pi)^{-\frac{D}{2}} |\Sigma_k|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k) \right) \frac{1}{2} \Sigma_k^{-1} (x_n - \mu_k)(x_n - \mu_k)^\intercal \Sigma_k^{-1}$$

$$= -\frac{1}{2} \mathcal{N}(x_n; \mu_k, \Sigma_k) [\Sigma_k^{-1} - \Sigma_k^{-1} (x_n - \mu_k)(x_n - \mu_k)^\intercal \Sigma_k^{-1}]$$

Step 2, covariance update [2]

gradient of log likelihood wrt cov

$$\frac{\partial L(\theta)}{\partial \Sigma_k} = \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \pi_k \frac{-1}{2} \mathcal{N}(x_n; \mu_k, \Sigma_k)[\Sigma_k^{-1} - \Sigma_k^{-1}(x_n - \mu_k)(x_n - \mu_k)^\intercal \Sigma_k^{-1}]$$

Step 2, covariance update [2]

gradient of log likelihood wrt cov

$$\frac{\partial L(\theta)}{\partial \Sigma_k} = \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \pi_k \frac{-1}{2} \mathcal{N}(x_n; \mu_k, \Sigma_k)[\Sigma_k^{-1} - \Sigma_k^{-1}(x_n - \mu_k)(x_n - \mu_k)^\intercal \Sigma_k^{-1}]$$

$$= -\frac{1}{2} \sum_{n=1}^{N} r_{nk}[\Sigma_k^{-1} - \Sigma_k^{-1}(x_n - \mu_k)(x_n - \mu_k)^\intercal \Sigma_k^{-1}]$$

Rewrite using responsibility

Step 2, covariance update [2]

gradient of log likelihood wrt cov

$$\frac{\partial L(\theta)}{\partial \Sigma_k} = \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \pi_k \frac{-1}{2} \mathcal{N}(x_n; \mu_k, \Sigma_k)[\Sigma_k^{-1} - \Sigma_k^{-1}(x_n - \mu_k)(x_n - \mu_k)^\intercal \Sigma_k^{-1}]$$

Rewrite using responsibility

$$= -\frac{1}{2} \sum_{n=1}^{N} r_{nk}[\Sigma_k^{-1} - \Sigma_k^{-1}(x_n - \mu_k)(x_n - \mu_k)^\intercal \Sigma_k^{-1}]$$

Set gradient to zero

$$\frac{\partial L(\theta)}{\partial \Sigma_k} = \mathbf{0} \rightarrow \Sigma_k = \frac{\sum_n r_{nk}(x_n - \mu_k)(x_n - \mu_k)^\intercal}{\sum_n r_{nk}}$$

Step 2, covariance update [2]

gradient of log likelihood wrt cov

$$\frac{\partial L(\theta)}{\partial \Sigma_k} = \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \pi_k \frac{-1}{2} \mathcal{N}(x_n; \mu_k, \Sigma_k)[\Sigma_k^{-1} - \Sigma_k^{-1}(x_n - \mu_k)(x_n - \mu_k)^\intercal \Sigma_k^{-1}]$$

Rewrite using responsibility

$$= -\frac{1}{2} \sum_{n=1}^{N} r_{nk}[\Sigma_k^{-1} - \Sigma_k^{-1}(x_n - \mu_k)(x_n - \mu_k)^\intercal \Sigma_k^{-1}]$$

Set gradient to zero

$$\frac{\partial L(\theta)}{\partial \Sigma_k} = \mathbf{0} \rightarrow \Sigma_k = \frac{\sum_n r_{nk}(x_n - \mu_k)(x_n - \mu_k)^\intercal}{\sum_n r_{nk}}$$

New covariance = weighted covariance of data where weights = responsibilities

Step 2, mixture weight update

$$L(\theta) = \sum_{n} \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Log likelihood

Step 2, mixture weight update

$$L(\theta) = \sum_n \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Log likelihood

$$\hat{L}(\theta) = \sum_n \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k) + \lambda \left(\sum_{k=1}^{K} \pi_k - 1 \right)$$

Lagrangian to deal with equality constraint $\sum_k \pi_k = 1$

Covered in optimisation and advanced ML

Step 2, mixture weight update

$$L(\theta) = \sum_n \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Log likelihood

$$\hat{L}(\theta) = \sum_n \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k) + \lambda \left(\sum_{k=1}^{K} \pi_k - 1 \right)$$

Lagrangian to deal with equality constraint $\sum_k \pi_k = 1$

Covered in optimisation and advanced ML

$$\frac{\partial \hat{L}(\theta)}{\partial \pi_k} = \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \frac{\partial}{\partial \pi_k} \sum_{j=1}^{K} \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j) + \frac{\partial}{\partial \pi_k} \lambda \left(\sum_{k=1}^{K} \pi_k - 1 \right)$$

gradient of Lagrangian wrt cov

Step 2, mixture weight update

$$L(\theta) = \sum_n \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Log likelihood

$$\hat{L}(\theta) = \sum_n \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k) + \lambda \left(\sum_{k=1}^{K} \pi_k - 1 \right)$$

Lagrangian to deal with equality constraint $\sum_k \pi_k = 1$

Covered in optimisation and advanced ML

$$\frac{\partial \hat{L}(\theta)}{\partial \pi_k} = \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \frac{\partial}{\partial \pi_k} \sum_{j=1}^{K} \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j) + \frac{\partial}{\partial \pi_k} \lambda \left(\sum_{k=1}^{K} \pi_k - 1 \right)$$

gradient of Lagrangian wrt cov

$$= \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \mathcal{N}(x_n; \mu_k, \Sigma_k) \quad + \quad \lambda$$

Ignore terms that do not depend on k

Step 2, mixture weight update

$$L(\theta) = \sum_n \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Log likelihood

$$\hat{L}(\theta) = \sum_n \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k) + \lambda \left(\sum_{k=1}^{K} \pi_k - 1 \right)$$

Lagrangian to deal with equality constraint $\sum_k \pi_k = 1$

Covered in optimisation and advanced ML

$$\frac{\partial \hat{L}(\theta)}{\partial \pi_k} = \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \frac{\partial}{\partial \pi_k} \sum_{j=1}^{K} \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j) + \frac{\partial}{\partial \pi_k} \lambda \left(\sum_{k=1}^{K} \pi_k - 1 \right)$$

gradient of Lagrangian wrt cov

$$= \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \mathcal{N}(x_n; \mu_k, \Sigma_k) \quad + \quad \lambda$$

Ignore terms that do not depend on k

$$= \sum_{n=1}^{N} \frac{r_{nk}}{\pi_k} \quad + \quad \lambda$$

Rewrite using responsibility

Step 2, mixture weight update

$$L(\theta) = \sum_n \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Log likelihood

$$\hat{L}(\theta) = \sum_n \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k) + \lambda \left(\sum_{k=1}^{K} \pi_k - 1 \right)$$

Lagrangian to deal with equality constraint $\sum_k \pi_k = 1$

Covered in optimisation and advanced ML

$$\frac{\partial \hat{L}(\theta)}{\partial \pi_k} = \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \frac{\partial}{\partial \pi_k} \sum_{j=1}^{K} \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j) + \frac{\partial}{\partial \pi_k} \lambda \left(\sum_{k=1}^{K} \pi_k - 1 \right)$$

gradient of Lagrangian wrt cov

$$= \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \mathcal{N}(x_n; \mu_k, \Sigma_k) \quad + \quad \lambda$$

Ignore terms that do not depend on k

$$= \sum_{n=1}^{N} \frac{r_{nk}}{\pi_k} \quad + \quad \lambda$$

Rewrite using responsibility

$$\frac{\partial \hat{L}(\theta)}{\partial \lambda} = \sum_{k=1}^{K} \pi_k - 1$$

gradient wrt Lagrange multiplier

Step 2, mixture weight update

$$L(\theta) = \sum_n \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Log likelihood

$$\hat{L}(\theta) = \sum_n \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k) + \lambda \left(\sum_{k=1}^{K} \pi_k - 1 \right)$$

Lagrangian to deal with equality constraint $\sum_k \pi_k = 1$

Covered in optimisation and advanced ML

$$\frac{\partial \hat{L}(\theta)}{\partial \pi_k} = \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \frac{\partial}{\partial \pi_k} \sum_{j=1}^{K} \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j) + \frac{\partial}{\partial \pi_k} \lambda \left(\sum_{k=1}^{K} \pi_k - 1 \right)$$

gradient of Lagrangian wrt cov

$$= \sum_{n=1}^{N} \frac{1}{p(x_n \mid \theta)} \mathcal{N}(x_n; \mu_k, \Sigma_k) \quad + \quad \lambda$$

Ignore terms that do not depend on k

$$= \sum_{n=1}^{N} \frac{r_{nk}}{\pi_k} \quad + \quad \lambda$$

Rewrite using responsibility

$$\frac{\partial \hat{L}(\theta)}{\partial \lambda} = \sum_{k=1}^{K} \pi_k - 1$$

gradient wrt Lagrange multiplier

$$\frac{\partial \hat{L}(\theta)}{\partial \pi_k} = 0 \quad \text{and} \quad \frac{\partial \hat{L}(\theta)}{\partial \lambda} = 0 \rightarrow \pi_k = \frac{\sum_n r_{nk}}{N}$$

Set gradients to zero

Overview

Last lecture:

1. **Model**: Gaussian mixture models
2. **Learning**: parameter learning using *maximum likelihood*
3. **Iterative** procedure, derive mean update

Initialise $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$

Repeat until convergence:

1. **Evaluate responsibilities** r_{nk} for every data point x_n using current parameters:

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}$$

2. **Re-estimate parameters** $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$ using the current responsibilities r_{nk}:

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^N r_{nk} x_n, \quad N_k = \sum_{n=1}^N r_{nk}$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^N r_{nk} (x_n - \mu_k)(x_n - \mu_k)^\intercal$$

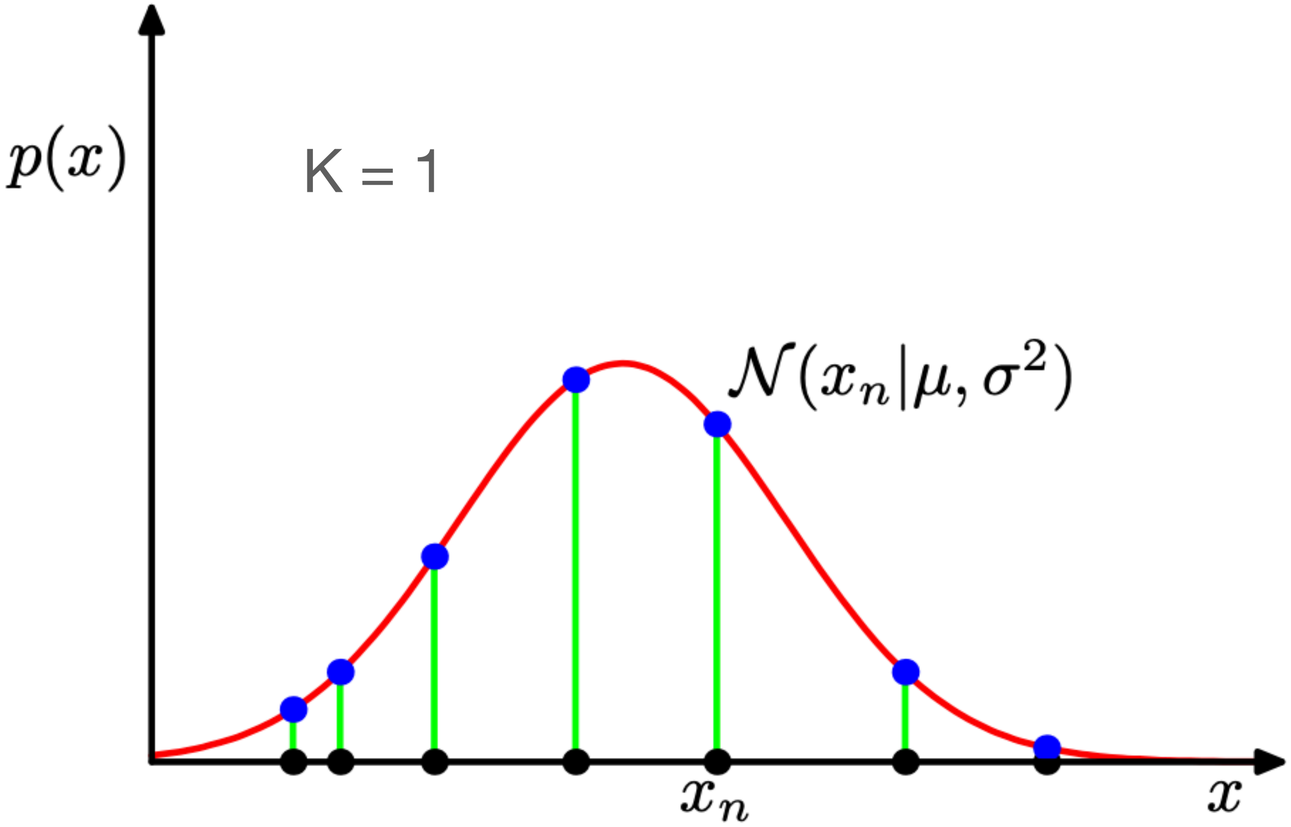
$$\pi_k = \frac{N_k}{N}$$

This lecture:

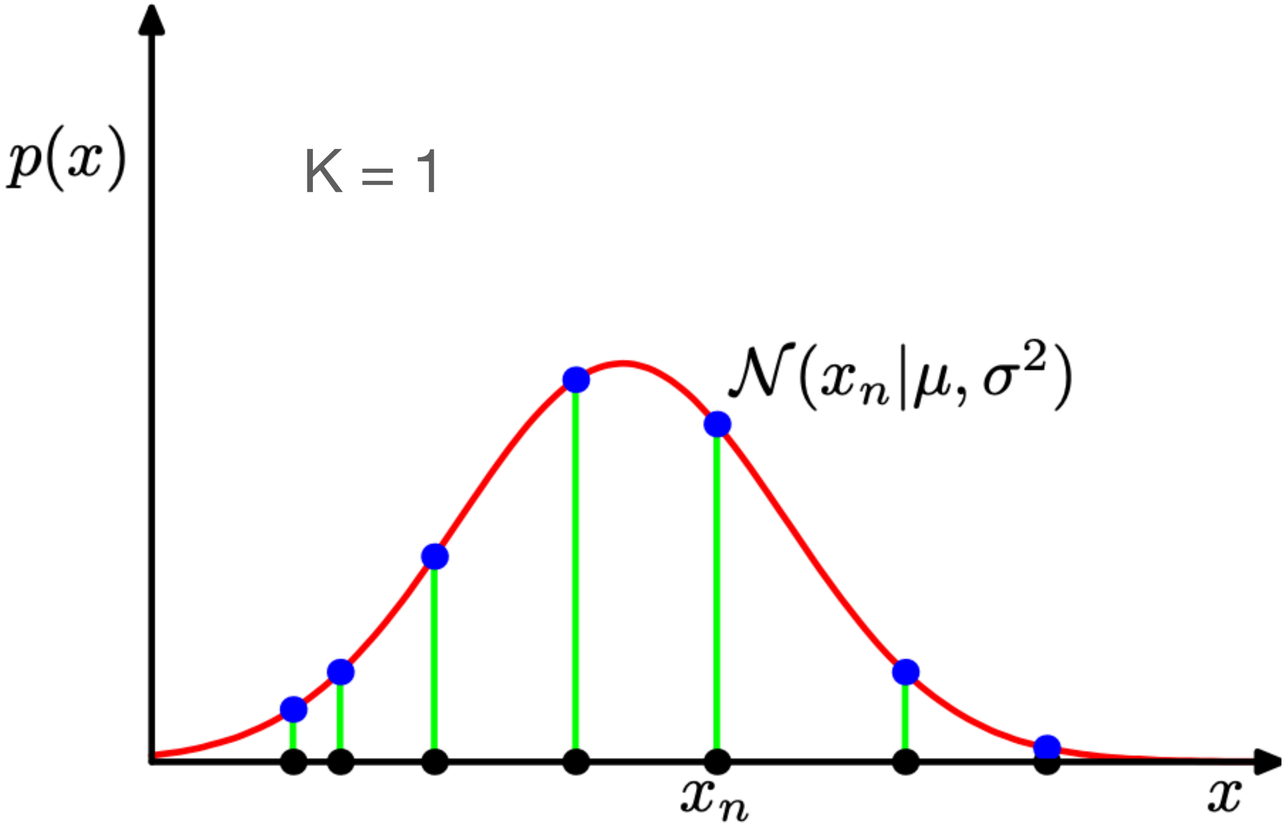
1. Derive covariance and mixture weight updates
2. Issues with maximum likelihood and workarounds
3. K-means as a special case
4. Theoretical foundation*: EM algorithm (1 is E step, 2 is M step)

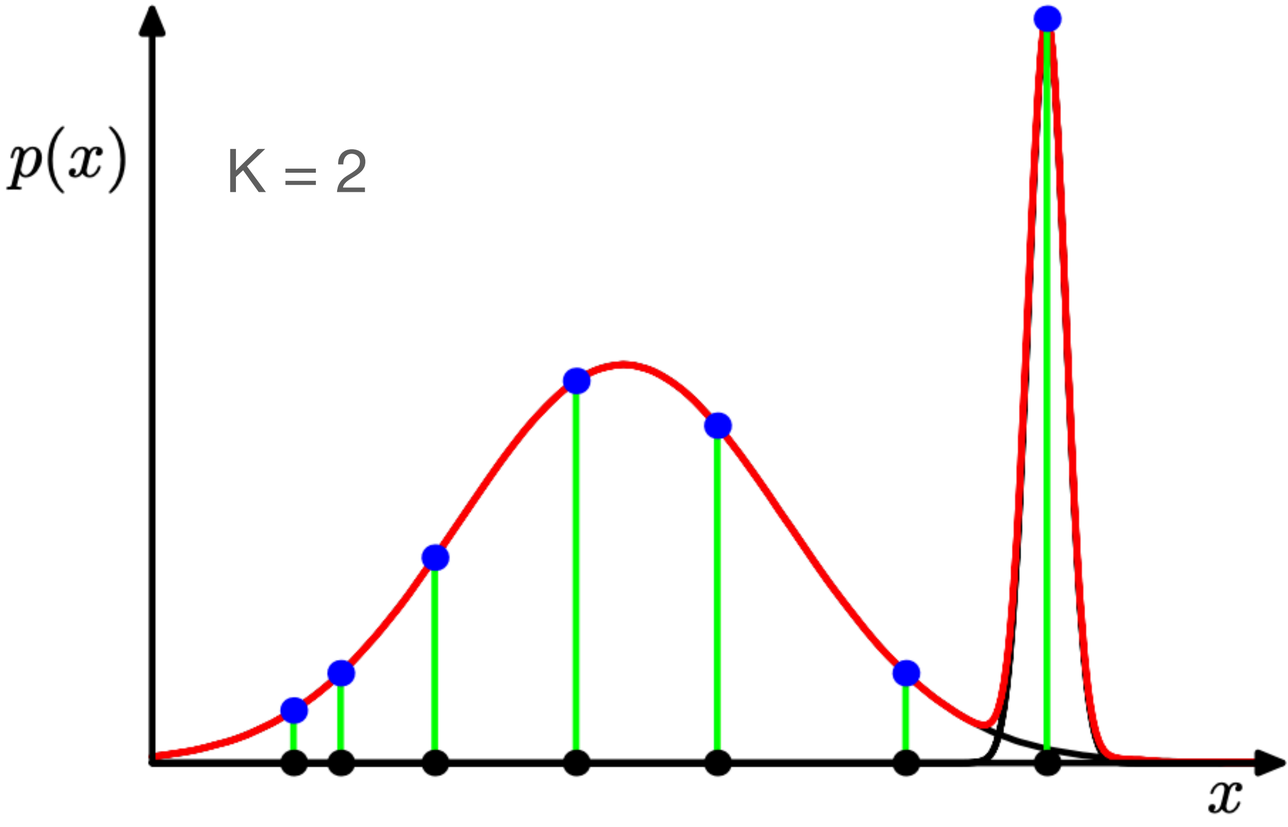
* not in assignment/exam

Maximum likelihood - overfitting

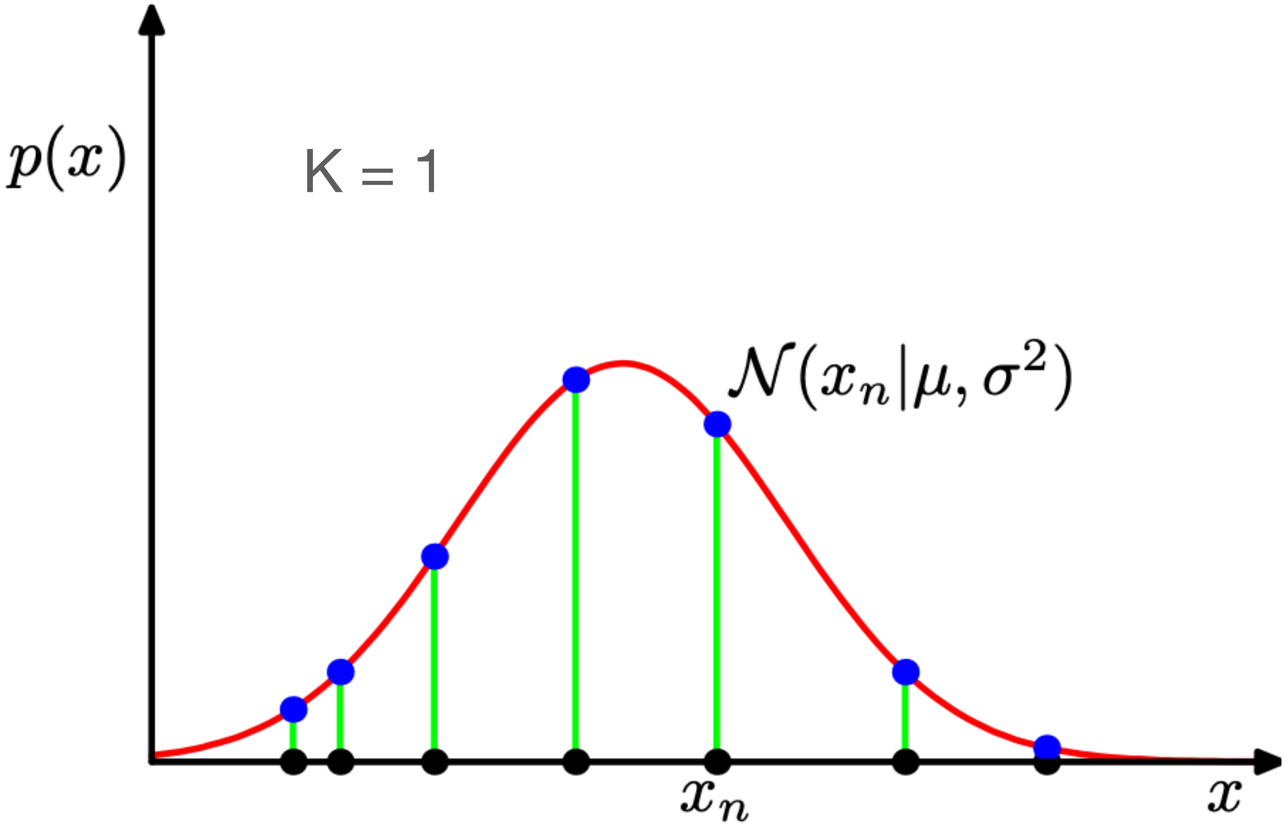


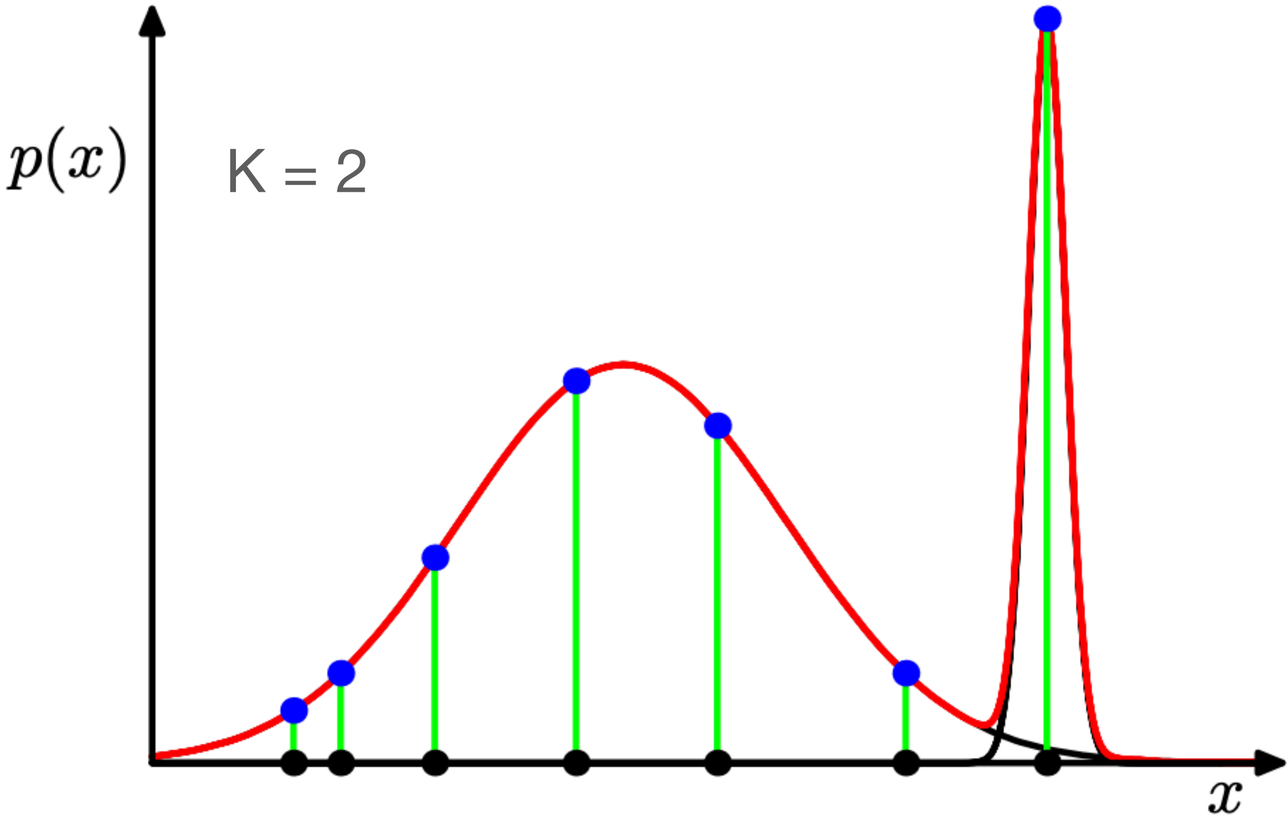
Maximum likelihood - overfitting





Maximum likelihood - overfitting

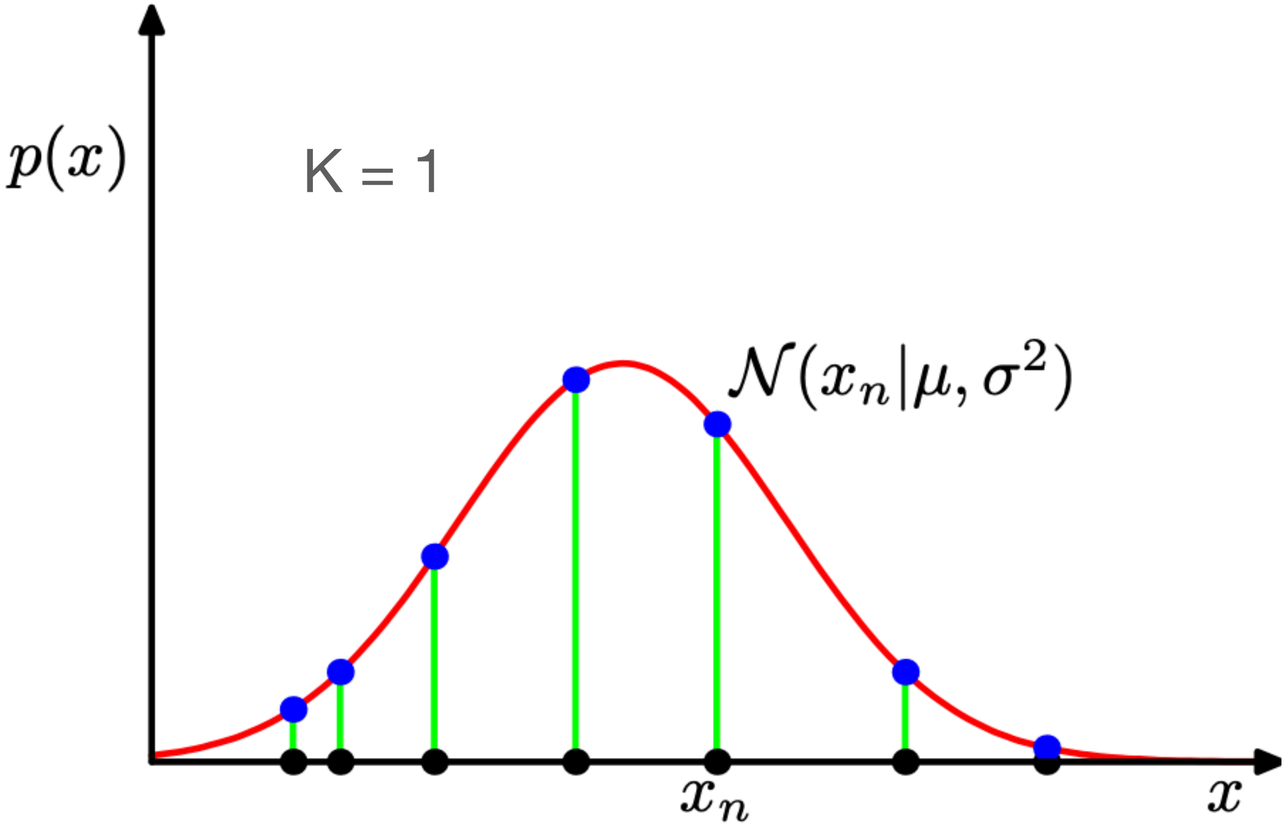


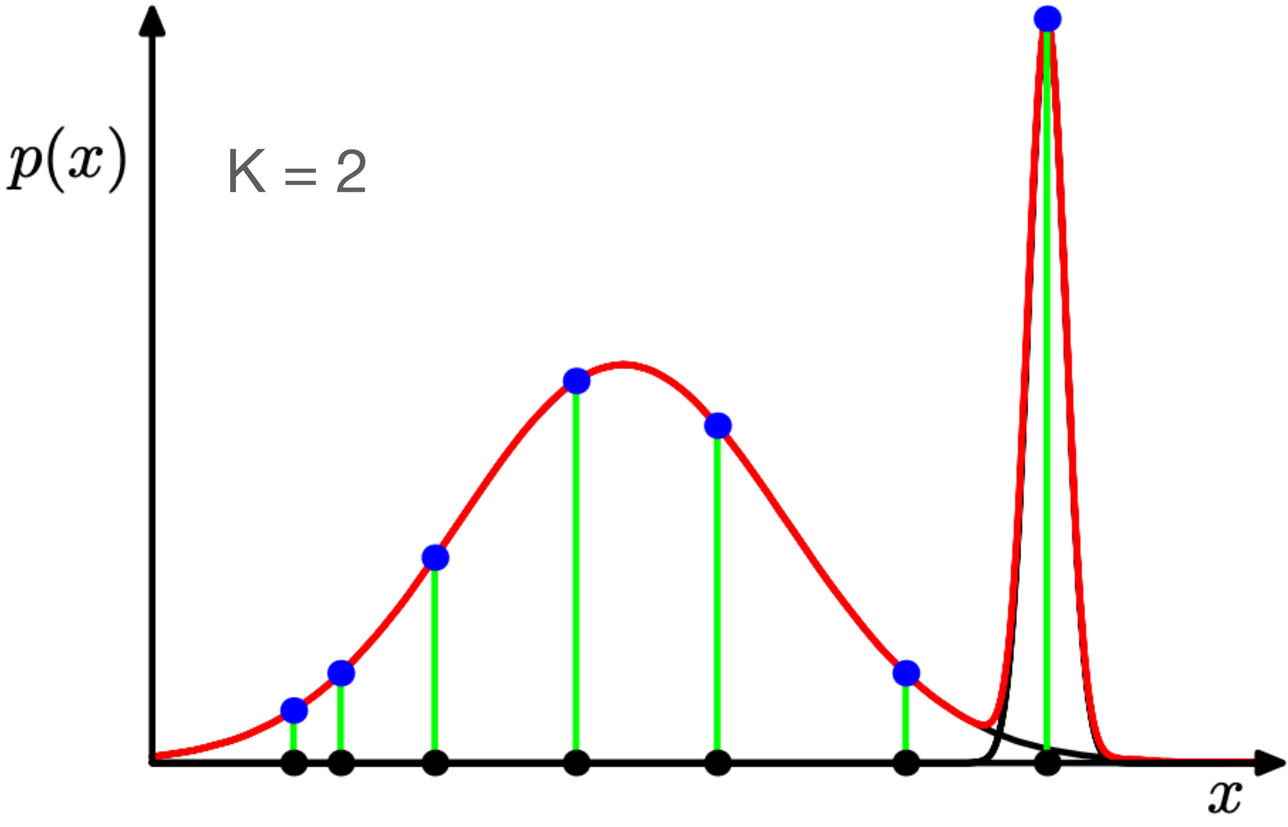


$$p_2(x_7 \,|\, \mu_2, \sigma_2^2) = \mathcal{N}(x_7; x_7, \sigma_2^2) = \frac{1}{\sqrt{2\pi}} \frac{1}{\sigma_2}$$

One component collapses to a data point!

Maximum likelihood - overfitting



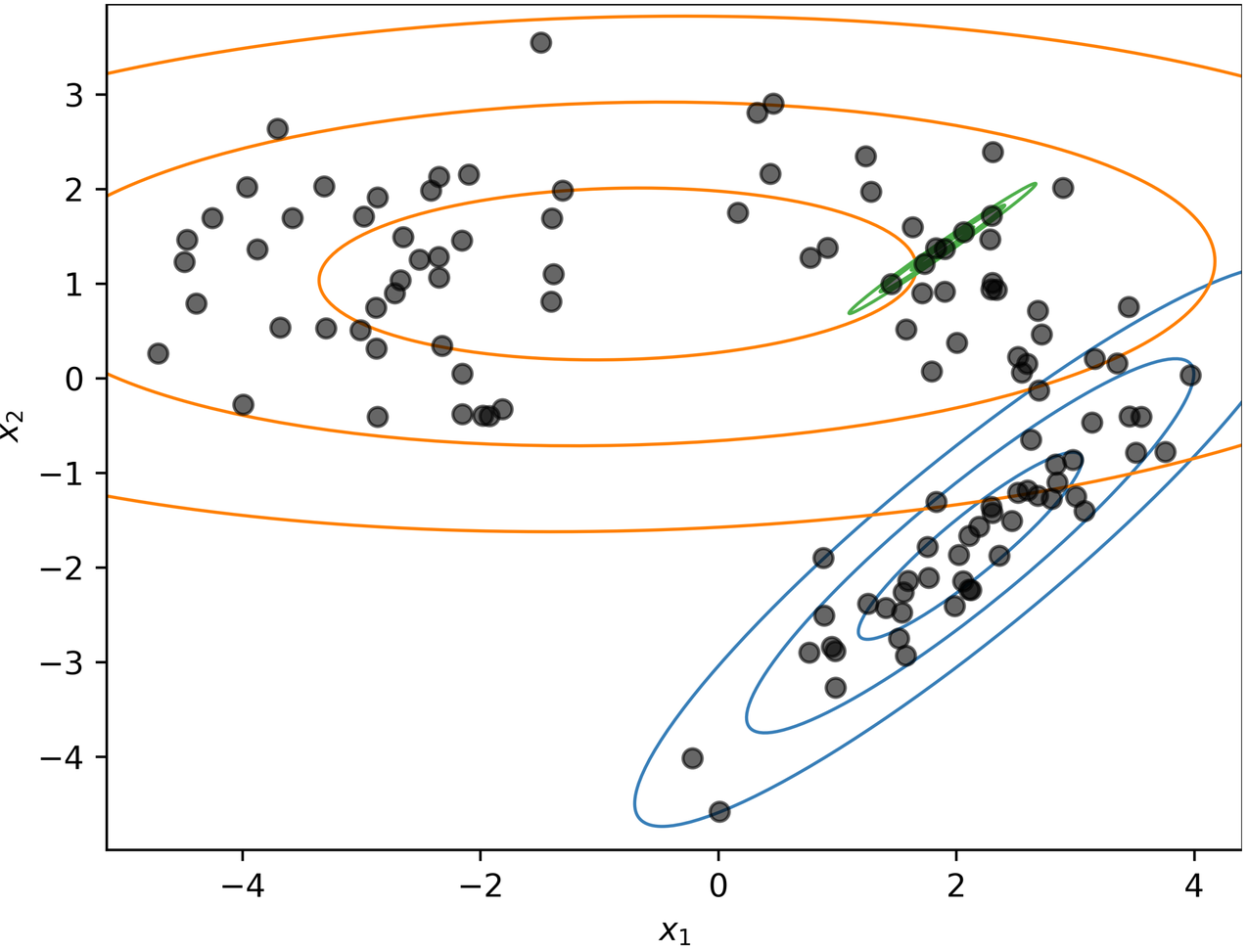


Hot-fix: reset the mean to a randomly chosen value, and the covariance to some large value and hope we don't have to reset again ¯_(ツ)_/¯

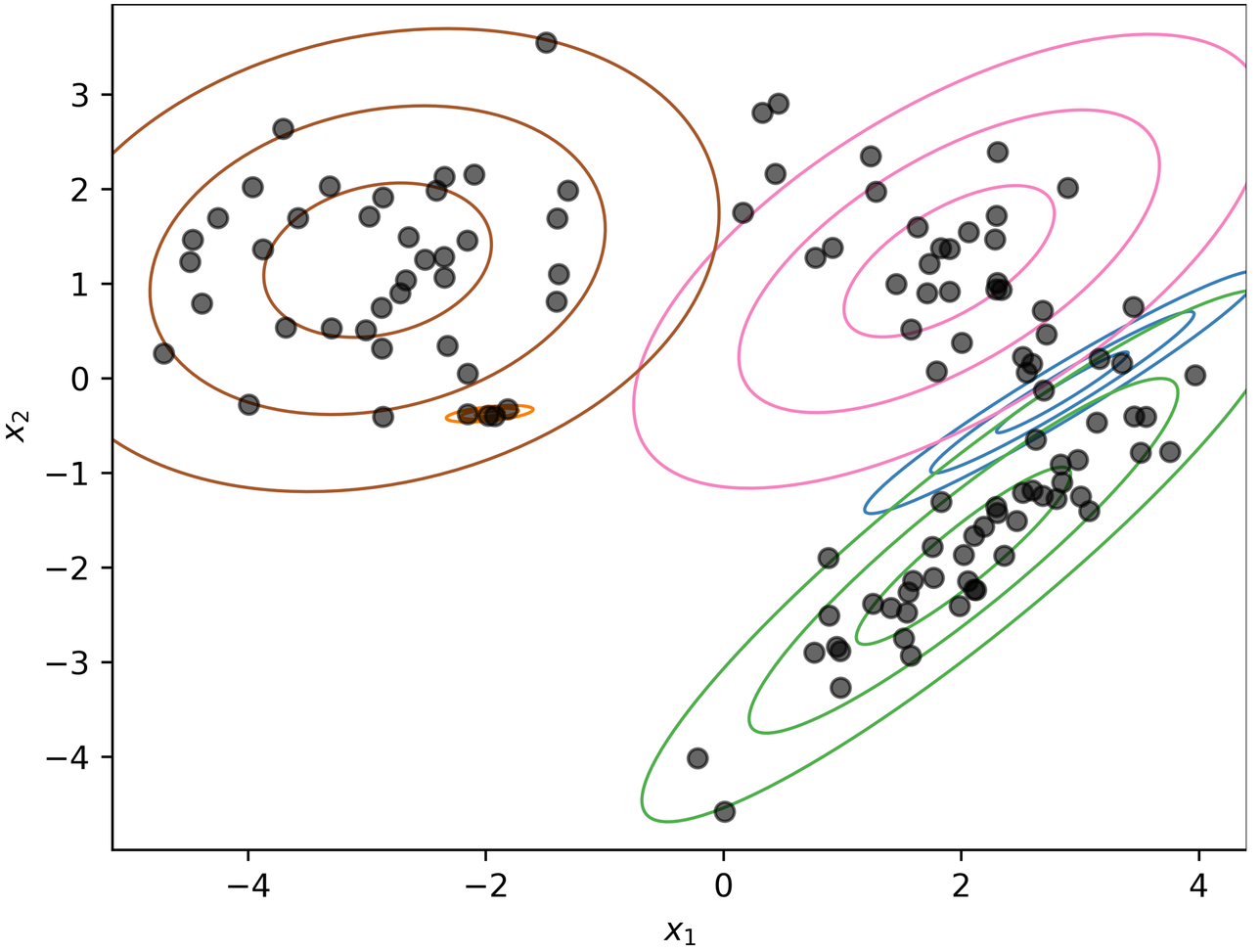
$$p_2(x_7 \,|\, \mu_2, \sigma_2^2) = \mathcal{N}(x_7; x_7, \sigma_2^2) = \frac{1}{\sqrt{2\pi}} \frac{1}{\sigma_2}$$

One component collapses to a data point!

Local minima

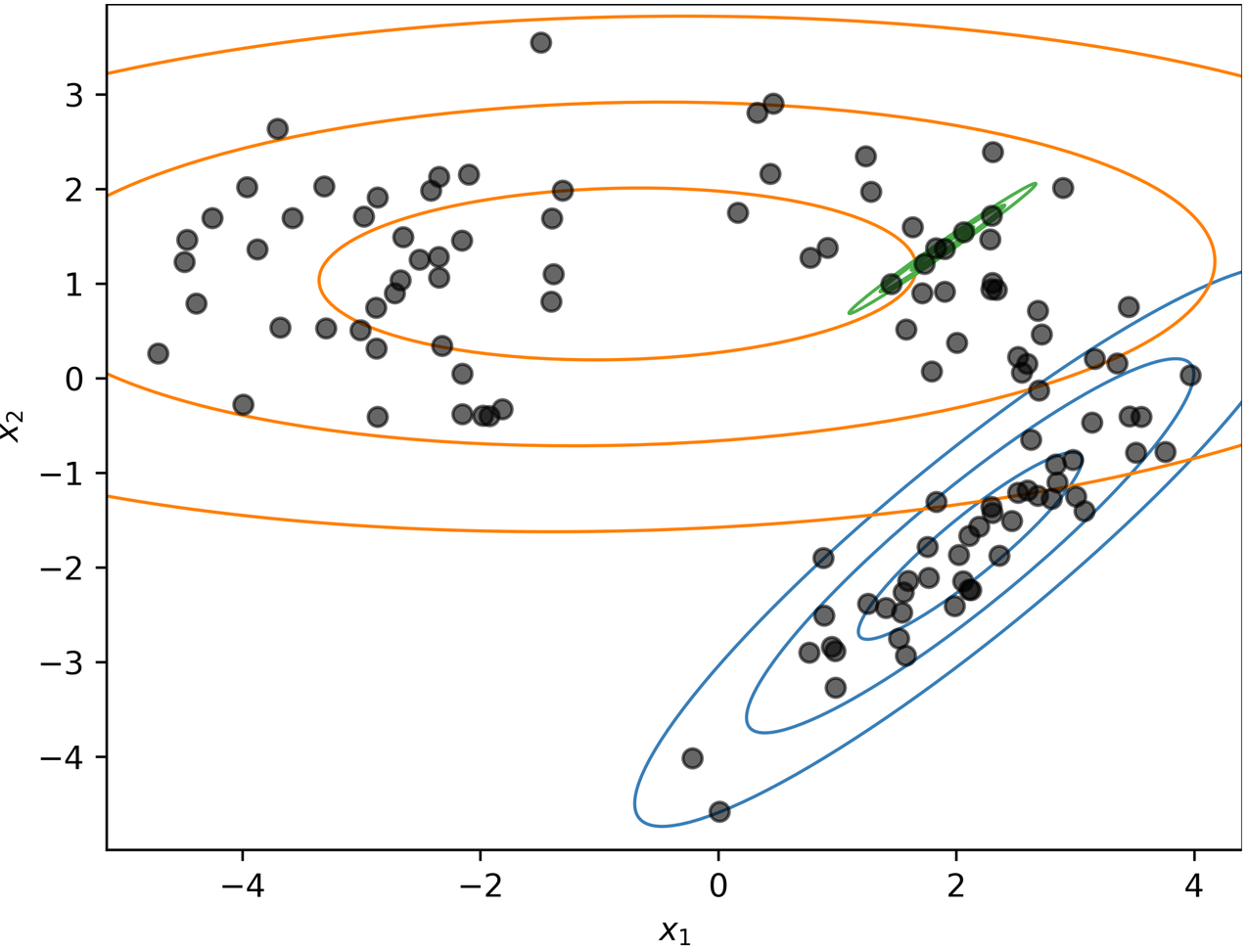


K = 3

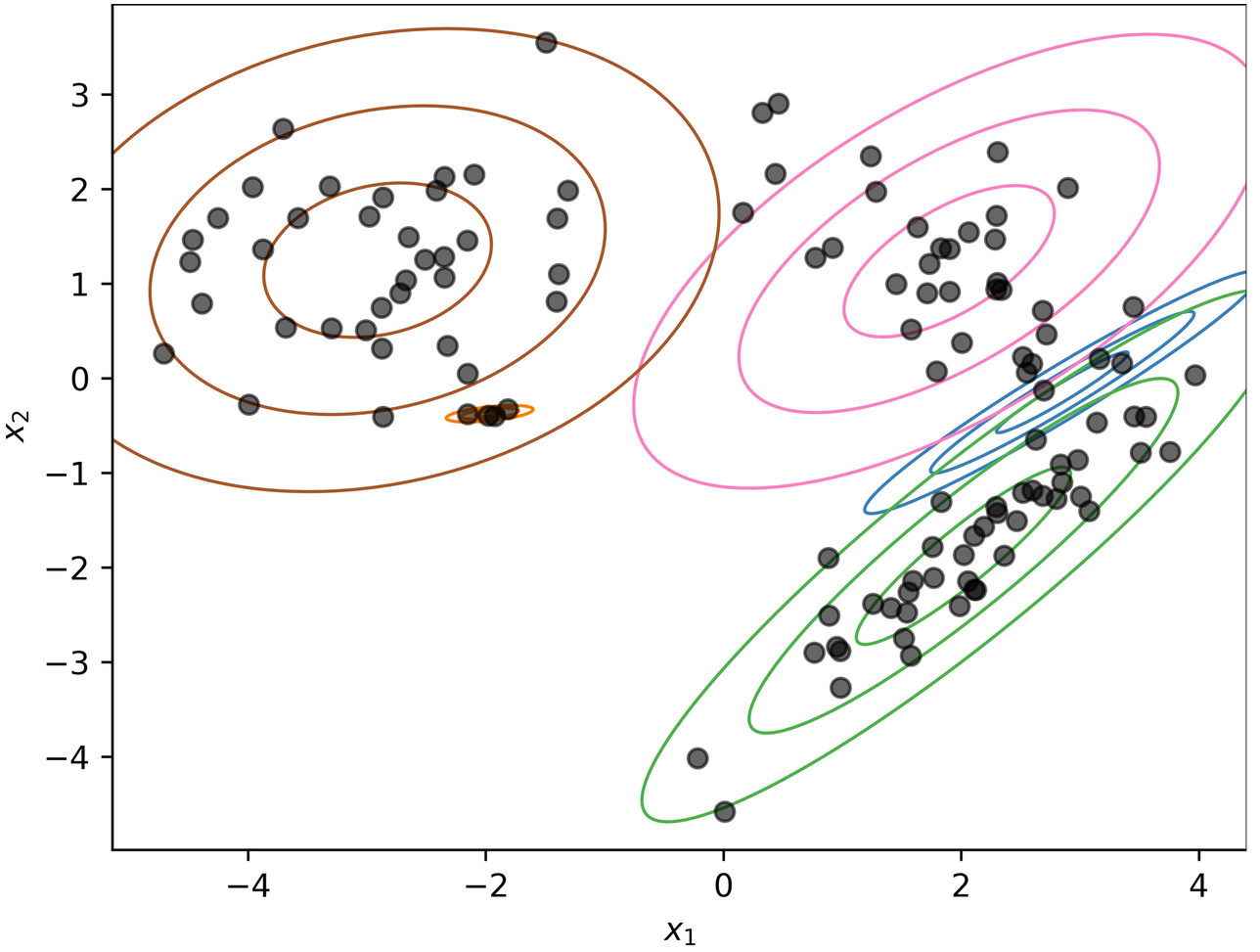


K = 5

Local minima



K = 3



K = 5

Practical initialisation: k-means, mean/cov = sample mean/cov, mixture weights = proportions

Overview

Last lecture:

1. **Model**: Gaussian mixture models
2. **Learning**: parameter learning using *maximum likelihood*
3. **Iterative** procedure, derive mean update

Initialise $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$

Repeat until convergence:

1. **Evaluate responsibilities** r_{nk} for every data point x_n using current parameters:

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}$$

2. **Re-estimate parameters** $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$ using the current responsibilities r_{nk}:

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^N r_{nk} x_n, \quad N_k = \sum_{n=1}^N r_{nk}$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^N r_{nk} (x_n - \mu_k)(x_n - \mu_k)^\intercal$$

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

This lecture:

1. Derive covariance and mixture weight updates
2. Issues with maximum likelihood and workarounds
3. K-means as a special case
4. Theoretical foundation*: EM algorithm (1 is E step, 2 is M step)

* not in assignment/exam

Relation to k-means

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}$$

Assume $\Sigma_1 = \Sigma_2 = \ldots = \Sigma_K = \epsilon \mathrm{I}$

Numerical example: $x_n = 5$, K = 3

Relation to k-means

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}$$

Assume $\Sigma_1 = \Sigma_2 = \ldots = \Sigma_K = \epsilon \mathrm{I}$

Numerical example: $x_n = 5$, K = 3

μ_k	π_k	$\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$	r_{nk}
8	0.3	0.02	0.27
-2	0.4	0.00016	0.01
4	0.3	0.0528	0.72

Soft assignment

$\epsilon = 4$

Relation to k-means

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}$$

Assume $\Sigma_1 = \Sigma_2 = \ldots = \Sigma_K = \epsilon \mathrm{I}$

Numerical example: $x_n = 5$, K = 3

μ_k	π_k	$\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$	r_{nk}
8	0.3	0.02	0.27
-2	0.4	0.00016	0.01
4	0.3	0.0528	0.72

Soft assignment

$\epsilon = 4$

μ_k	π_k	$\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$	r_{nk}
8	0.3	0.44×10^{-195}	0
-2	0.4	1.5×10^{-1064}	0
4	0.3	0.23×10^{-22}	1

Hard assignment

$\epsilon = 0.01$

Overview

Last lecture:

1. **Model**: Gaussian mixture models
2. **Learning**: parameter learning using *maximum likelihood*
3. **Iterative** procedure, derive mean update

Initialise $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$

Repeat until convergence:

1. **Evaluate responsibilities** r_{nk} for every data point x_n using current parameters:

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}$$

2. **Re-estimate parameters** $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$ using the current responsibilities r_{nk}:

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^N r_{nk} x_n, \quad N_k = \sum_{n=1}^N r_{nk}$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^N r_{nk} (x_n - \mu_k)(x_n - \mu_k)^\intercal$$

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

This lecture:

1. Derive covariance and mixture weight updates
2. Issues with maximum likelihood and workarounds
3. K-means as a special case
4. Theoretical foundation*: EM algorithm (1 is E step, 2 is M step)

* not in assignment/exam

GMMs - iterative procedure = *EM* algorithm

Initialise $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$

Repeat until convergence:

1. **Evaluate responsibilities** r_{nk} for every data point x_n using current parameters:

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}$$

E step

2. **Re-estimate parameters** $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$ using the current responsibilities r_{nk}:

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^N r_{nk} x_n, \quad N_k = \sum_{n=1}^N r_{nk}$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^N r_{nk} (x_n - \mu_k)(x_n - \mu_k)^\intercal$$

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

M step

Expectation maximisation (EM) algorithm*

Log-likelihood: $L(\theta) = \log p(X \mid \theta) = \sum_{n=1}^{N} \log p(x_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$

Goal: maximum likelihood, $\operatorname{argmax}_\theta \; p(X \mid \theta)$, which is equivalent to $\operatorname{argmax}_\theta \; \log p(X \mid \theta)$

Assume: **latent/unknown/hidden variables** $\{z_n\}_{n=1}^{N}$ representing *component assignment*

$$L(\theta) = \log p(X \mid \theta) = \log \int p(X, z \mid \theta) \, \mathrm{d}z = \log \int q(z) \frac{p(X, z \mid \theta)}{q(z)} \, \mathrm{d}z \geq \int q(z) \log \frac{p(X, z \mid \theta)}{q(z)} \, \mathrm{d}z := \mathscr{F}(q(z), \theta)$$

An arbitrary $q(z)$ — Jensen's inequality — Lower bound on $L(\theta)$

Instead of maximising $L(\theta)$ directly, we will maximise the lower bound $\mathscr{F}(q(z), \theta)$, alternating between $q(z)$ and θ while keeping the other fixed.

EM for GMMs

1. **Evaluate responsibilities** r_{nk} for every data point x_n using current parameters:

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}$$

E step

2. **Re-estimate parameters** $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ using the current responsibilities r_{nk}:

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} x_n, \quad N_k = \sum_{n=1}^{N} r_{nk}$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} (x_n - \mu_k)(x_n - \mu_k)^\intercal$$

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

M step

EM for GMMs

Maximising lower bound wrt $q(z)$
whilst keeping θ fixed

$$q(z)^{(t+1)} \leftarrow \text{argmax}_{q(z)}\, \mathcal{F}(q(z), \theta^{(t)}) = p(z \mid X, \theta^{(t)})$$

fill in values for the hidden variables according to their posterior probabilities

1. **Evaluate responsibilities** r_{nk} for every data point x_n using current parameters:

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}$$

E step

2. **Re-estimate parameters** $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ using the current responsibilities r_{nk}:

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} x_n, \quad N_k = \sum_{n=1}^{N} r_{nk}$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} (x_n - \mu_k)(x_n - \mu_k)^\intercal$$

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

M step

EM for GMMs

Maximising lower bound wrt $q(z)$ whilst keeping θ fixed

$$q(z)^{(t+1)} \leftarrow \text{argmax}_{q(z)}\, \mathcal{F}(q(z), \theta^{(t)}) = p(z \mid X, \theta^{(t)})$$

fill in values for the hidden variables according to their posterior probabilities

1. **Evaluate responsibilities** r_{nk} for every data point x_n using current parameters:

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}$$

E step

Maximising lower bound wrt θ whilst keeping $q(z)$ fixed

$$\theta^{(t+1)} \leftarrow \text{argmax}_{\theta}\, \mathcal{F}(q(z)^{t+1}, \theta)$$

learn model as if hidden variables were not hidden

2. **Re-estimate parameters** $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ using the current responsibilities r_{nk}:

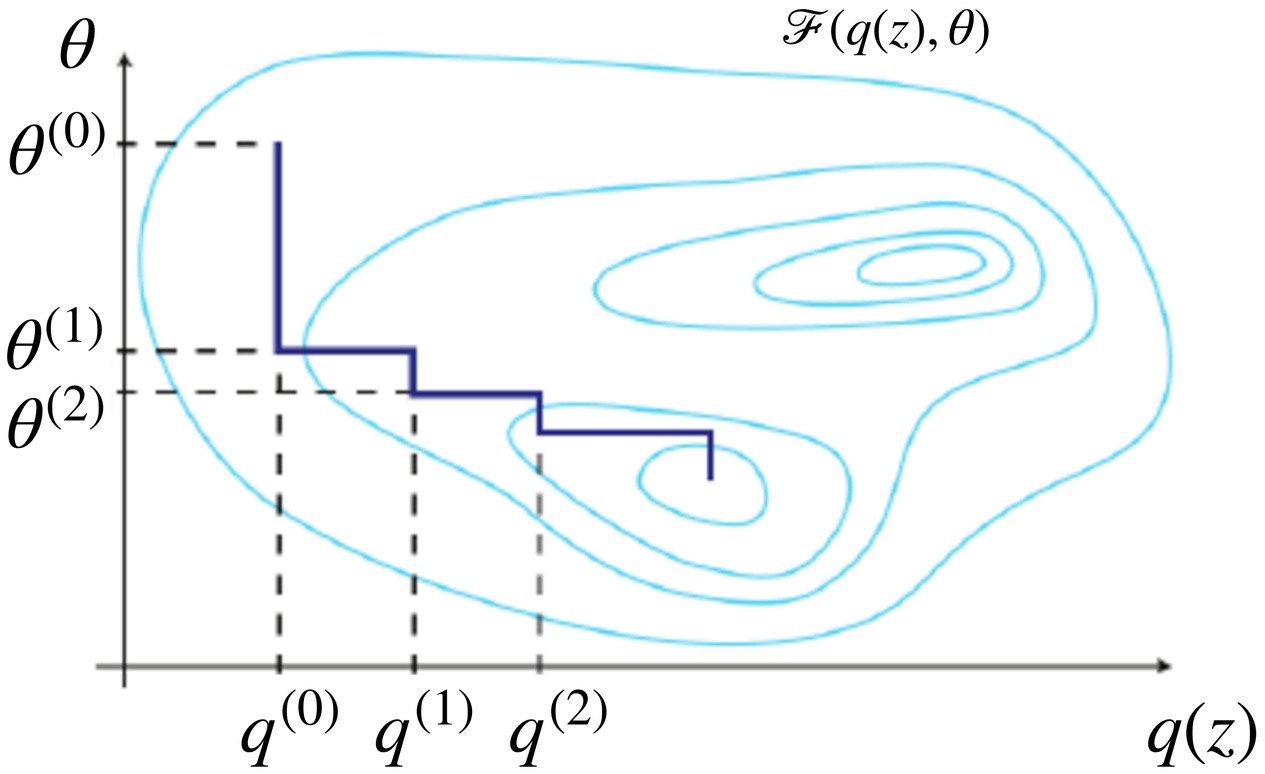
$$\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} x_n, \quad N_k = \sum_{n=1}^{N} r_{nk}$$

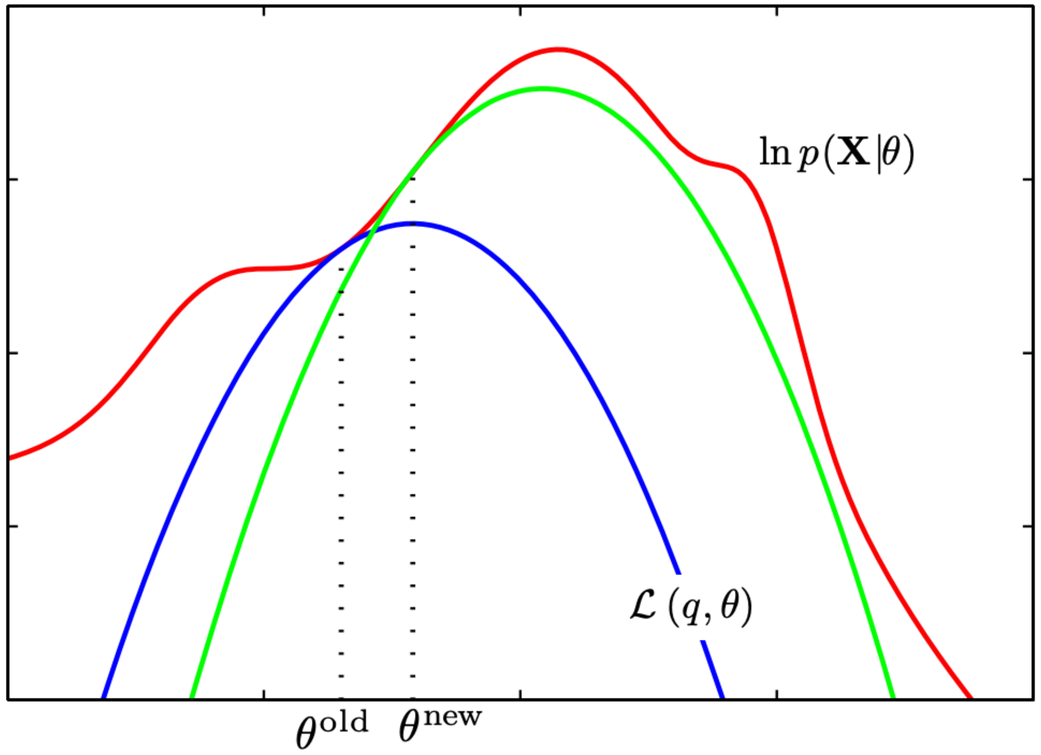
$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} (x_n - \mu_k)(x_n - \mu_k)^\intercal$$

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

M step

EM algorithm - visualisation





EM algorithm - more visualisation

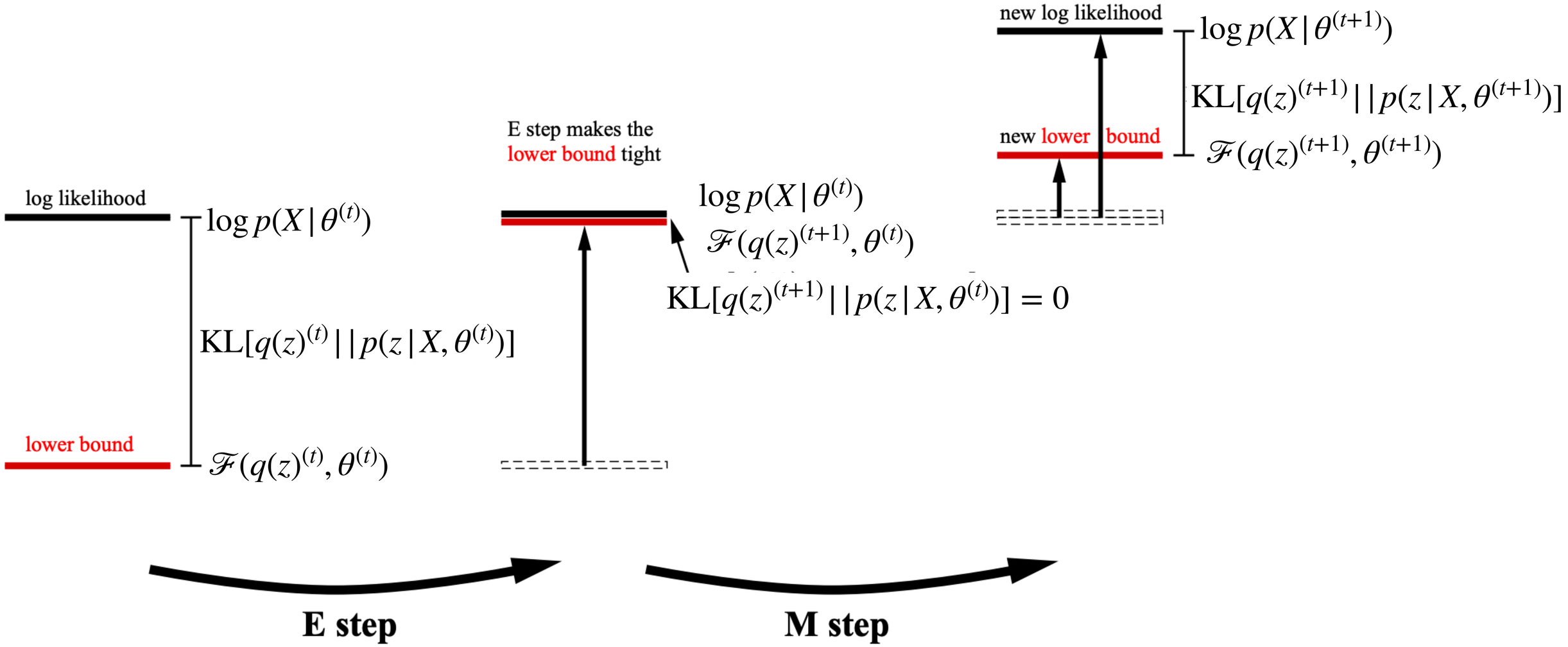


Figure credit: Beal (2003), Variational Algorithms for Approximate Bayesian Inference

Gaussian mixture models: check your understanding

Model: $$p_\theta(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k) \qquad 0 \leq \pi_k \leq 1, \sum_{k=1}^{K} \pi_k = 1$$

where we defined $\theta = \{\pi_k, \mu_k, \Sigma_k \text{ for } k = 1,2,\ldots,K\}$ as the parameters of the GMM.

- If K takes a greater value, the likelihood becomes greater after convergence.
- Assume we have N data points. The maximum likelihood will be achieved if we set $K = N$.
- In GMM, the EM algorithm gives us *global* minimum, because we can update π_k, μ_k, and Σ_k through closed-form solutions.
- GMM has a higher computational complexity than k-means.

Gaussian mixture models: check your understanding

Model: $$p_\theta(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k) \qquad 0 \leq \pi_k \leq 1, \sum_{k=1}^{K} \pi_k = 1$$

where we defined $\theta = \{\pi_k, \mu_k, \Sigma_k \text{ for } k = 1,2,\ldots,K\}$ as the parameters of the GMM.

- If K takes a greater value, the likelihood becomes greater after convergence. Yes
- Assume we have N data points. The maximum likelihood will be achieved if we set $K = N$.
- In GMM, the EM algorithm gives us *global* minimum, because we can update π_k, μ_k, and Σ_k through closed-form solutions.
- GMM has a higher computational complexity than k-means.

Gaussian mixture models: check your understanding

Model: $$p_\theta(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k) \qquad 0 \leq \pi_k \leq 1, \sum_{k=1}^{K} \pi_k = 1$$

where we defined $\theta = \{\pi_k, \mu_k, \Sigma_k \text{ for } k = 1,2,\ldots,K\}$ as the parameters of the GMM.

- If K takes a greater value, the likelihood becomes greater after convergence. Yes
- Assume we have N data points. The maximum likelihood will be achieved if we set $K = N$. Yes
- In GMM, the EM algorithm gives us *global* minimum, because we can update π_k, μ_k, and Σ_k through closed-form solutions.
- GMM has a higher computational complexity than k-means.

Gaussian mixture models: check your understanding

Model: $$p_\theta(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k) \qquad 0 \leq \pi_k \leq 1, \sum_{k=1}^{K} \pi_k = 1$$

where we defined $\theta = \{\pi_k, \mu_k, \Sigma_k \text{ for } k = 1,2,\ldots,K\}$ as the parameters of the GMM.

- If K takes a greater value, the likelihood becomes greater after convergence. Yes
- Assume we have N data points. The maximum likelihood will be achieved if we set $K = N$. Yes
- In GMM, the EM algorithm gives us *global* minimum, because we can update π_k, μ_k, and Σ_k through closed-form solutions. No
- GMM has a higher computational complexity than k-means.

Gaussian mixture models: check your understanding

Model: $$p_\theta(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k) \qquad 0 \leq \pi_k \leq 1, \sum_{k=1}^{K} \pi_k = 1$$

where we defined $\theta = \{\pi_k, \mu_k, \Sigma_k \text{ for } k = 1,2,\ldots,K\}$ as the parameters of the GMM.

- If K takes a greater value, the likelihood becomes greater after convergence. Yes
- Assume we have N data points. The maximum likelihood will be achieved if we set $K = N$. Yes
- In GMM, the EM algorithm gives us *global* minimum, because we can update π_k, μ_k, and Σ_k through closed-form solutions. No
- GMM has a higher computational complexity than k-means. Yes

Overview

Last lecture:

1. **Model**: Gaussian mixture models
2. **Learning**: parameter learning using *maximum likelihood*
3. **Iterative** procedure, derive mean update

Initialise $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$

Repeat until convergence:

1. **Evaluate responsibilities** r_{nk} for every data point x_n using current parameters:

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}$$

2. **Re-estimate parameters** $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$ using the current responsibilities r_{nk}:

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^N r_{nk} x_n, \quad N_k = \sum_{n=1}^N r_{nk}$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^N r_{nk} (x_n - \mu_k)(x_n - \mu_k)^\intercal$$

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

This lecture:

1. Derive covariance and mixture weight updates
2. Issues with maximum likelihood and workarounds
3. K-means as a special case
4. Theoretical foundation*: EM algorithm (1 is E step, 2 is M step)

* not in assignment/exam