

STA 4273H: Statistical Machine Learning

Russ Salakhutdinov

Department of Statistics

rsalakhu@utstat.toronto.edu

<http://www.utstat.utoronto.ca/~rsalakhu/>

Sidney Smith Hall, Room 6002

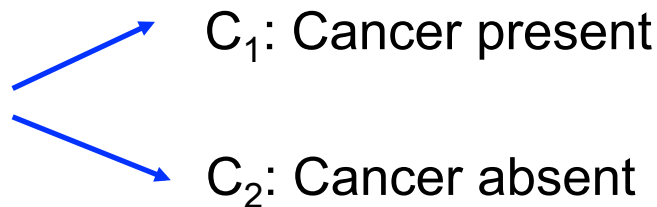
Lecture 3

Linear Models for Classification

- So far, we have looked at the linear models for regression that have particularly simple analytical and computational properties.
- We will now look at analogous class of models for solving classification problems.
- We will also look at the Bayesian treatment of linear models for classification.

Classification

- The goal of classification is to assign an input \mathbf{x} into one of K discrete classes C_k , where $k=1,\dots,K$.
- Typically, each input is assigned only to one class.
- **Example:** The input vector \mathbf{x} is the set of pixel intensities, and the output variable t will represent the presence of cancer, class C_1 , or absence of cancer, class C_2 .




\mathbf{x} -- set of pixel intensities


Linear Classification

- The goal of classification is to assign an input \mathbf{x} into one of K discrete classes C_k , where $k=1,\dots,K$.
- The input space is divided into decision regions whose boundaries are called **decision boundaries** or **decision surfaces**.
- We will consider linear models for classification. Remember, in the simplest linear regression case, **the model is linear in parameters**:

$$y(\mathbf{x}, \mathbf{w}) = \mathbf{x}^T \mathbf{w} + w_0.$$


adaptive parameters

$$y(\mathbf{x}, \mathbf{w}) = f(\mathbf{x}^T \mathbf{w} + w_0).$$


fixed nonlinear function:
activation function

- For classification, we need to predict discrete class labels, or posterior probabilities that lie in the range of $(0, 1)$, so we use a nonlinear function.

Linear Classification

$$y(\mathbf{x}, \mathbf{w}) = f(\mathbf{x}^T \mathbf{w} + w_0).$$

- The **decision surfaces** correspond to $y(\mathbf{x}, \mathbf{w}) = \text{const}$, so that $\mathbf{x}^T \mathbf{w} + w_0 = \text{const}$, and hence **the decision surfaces are linear functions of \mathbf{x} , even if the activation function is nonlinear.**
- These class of models are called **generalized linear models.**
- Note that these models are no longer linear in parameters, due to the presence of nonlinear activation function.
- This leads to more complex analytical and computational properties, compared to linear regression.
- Note that we can make **a fixed nonlinear transformation of the input variables** using a vector of basis functions $\phi(\mathbf{x})$, as we did for regression models.

Notation

- In the case of two-class problems, we can use the binary representation for the target value $t \in \{0, 1\}$, such that $t=1$ represents the **positive class** and $t=0$ represents the **negative class**.
 - We can interpret the value of t as the probability of the positive class, and the output of the model can be represented as the probability that the model assigns to the positive class.

- If there are K classes, we use a **1-of- K encoding scheme**, in which \mathbf{t} is a vector of length K containing a single 1 for the correct class and 0 elsewhere.

- For example, if we have $K=5$ classes, then an input that belongs to class 2 would be given a target vector:

$$t = (0, 1, 0, 0, 0)^T.$$

- We can interpret a vector \mathbf{t} as a vector of class probabilities.

Three Approaches to Classification

- Construct a **discriminant function** that directly maps each input vector to a specific class.
- Model the conditional probability distribution $p(\mathcal{C}_k|\mathbf{x})$, and then use this distribution to make optimal decisions.
- There are two alternative approaches:
 - **Discriminative Approach**: Model $p(\mathcal{C}_k|\mathbf{x})$, directly, for example by representing them as parametric models, and optimize for parameters using the training set (e.g. logistic regression).
 - **Generative Approach**: Model class conditional densities $p(\mathbf{x}|\mathcal{C}_k)$ together with the prior probabilities $p(\mathcal{C}_k)$ for the classes. Infer posterior probability using Bayes' rule:

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\mathbf{x})}.$$

We will consider next.

Probabilistic Generative Models

- Model class conditional densities $p(\mathbf{x}|\mathcal{C}_k)$ **separately for each class**, as well as the **class priors** $p(\mathcal{C}_k)$.
- Consider the case of two classes. The posterior probability of class \mathcal{C}_1 is given by:

$$\begin{aligned} p(\mathcal{C}_1|\mathbf{x}) &= \frac{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1) + p(\mathbf{x}|\mathcal{C}_2)p(\mathcal{C}_2)} \\ &= \frac{1}{1 + \exp(-a)} = \sigma(a), \end{aligned}$$

where we defined:

$$a = \ln \frac{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\mathbf{x}|\mathcal{C}_2)p(\mathcal{C}_2)} = \ln \frac{p(\mathcal{C}_1|\mathbf{x})}{1 - p(\mathcal{C}_1|\mathbf{x})},$$

Logistic sigmoid
function



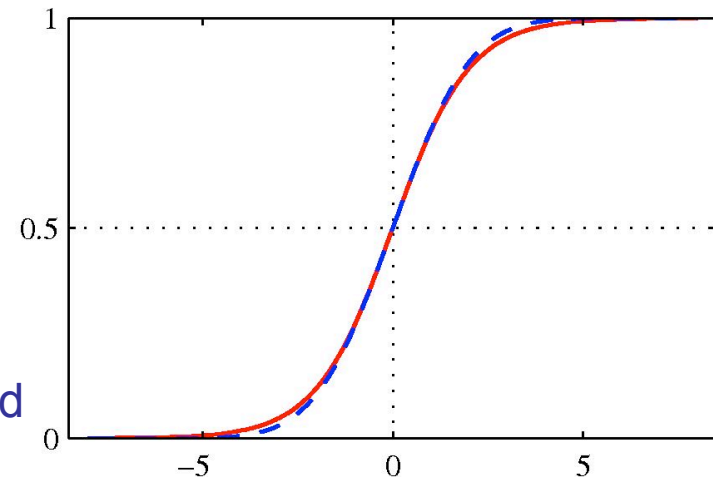
which is known as the **logit function**. It represents the log of the ration of probabilities of two classes, also known as the **log-odds**.

Sigmoid Function

- The posterior probability of class C_1 is given by:

$$p(C_1|\mathbf{x}) = \frac{p(\mathbf{x}|C_1)p(C_1)}{p(\mathbf{x}|C_1)p(C_1) + p(\mathbf{x}|C_2)p(C_2)}$$
$$= \frac{1}{1 + \exp(-a)} = \sigma(a),$$

Logistic sigmoid
function



- The term sigmoid means S-shaped: it maps the whole real axis into (0 1).
- It satisfies:

$$\sigma(-a) = 1 - \sigma(a), \quad \frac{d}{da}\sigma(a) = \sigma(a)(1 - \sigma(a)).$$

Softmax Function

- For case of $K > 2$ classes, we have the following **multi-class generalization**:

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{\sum_j p(\mathbf{x}|\mathcal{C}_j)p(\mathcal{C}_j)} = \frac{\exp(a_k)}{\sum_j \exp(a_j)}, \quad a_k = \ln[p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)].$$

- This **normalized exponential** is also known as the **softmax function**, as it represents a **smoothed version of the max function**:

$$\text{if } a_k \gg a_j, \forall j \neq k, \text{ then } p(\mathcal{C}_k|\mathbf{x}) \approx 1, p(\mathcal{C}_j|\mathbf{x}) \approx 0.$$

- We now look at some specific forms of class conditional distributions.

Example of Continuous Inputs

- Assume that the input vectors for each class are from a Gaussian distribution, and all classes share the same covariance matrix:

$$p(\mathbf{x}|\mathcal{C}_k) = \frac{1}{(2\pi)^{D/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu}_k)\right).$$

- For the case of two classes, the posterior is logistic function:

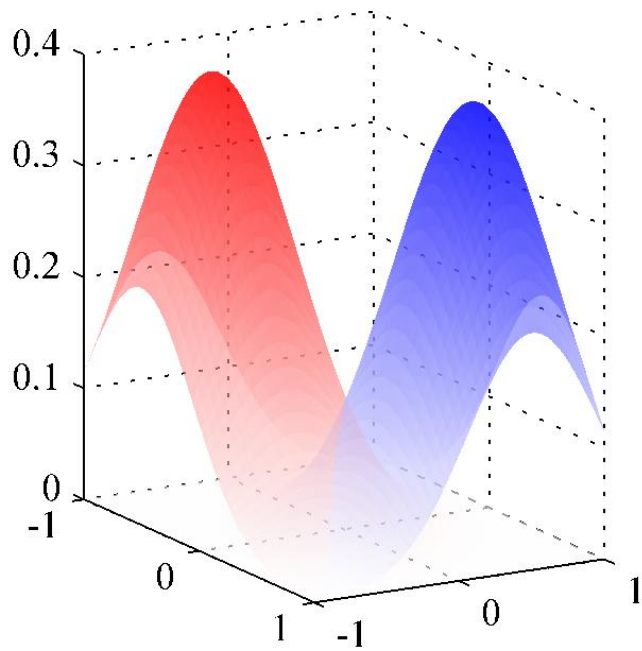
$$p(\mathcal{C}_k|\mathbf{x}) = \sigma(\mathbf{w}^T \mathbf{x} + w_0),$$

where we have defined:

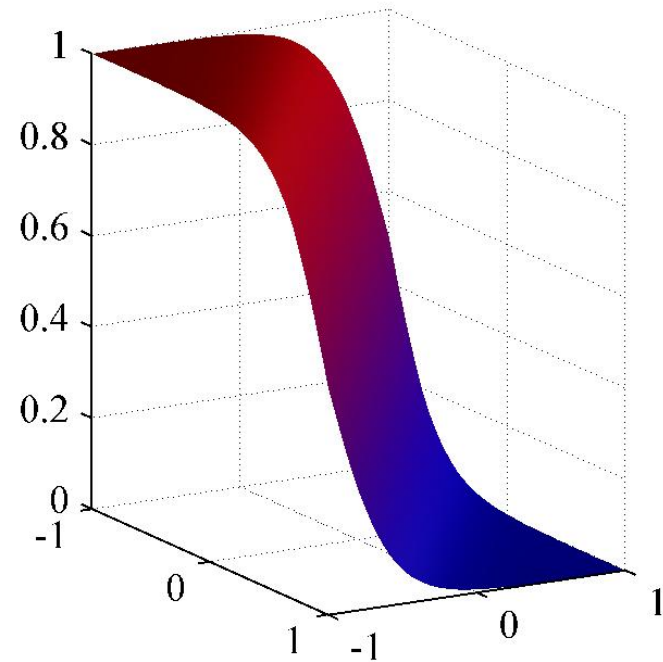
$$\begin{aligned} \mathbf{w} &= \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2), \\ w_0 &= -\frac{1}{2}\boldsymbol{\mu}_1^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_1 + \frac{1}{2}\boldsymbol{\mu}_2^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_2 + \ln \frac{p(\mathcal{C}_1)}{p(\mathcal{C}_2)}. \end{aligned}$$

- The quadratic terms in \mathbf{x} cancel (due to the assumption of common covariance matrices).
- This leads to a linear function of \mathbf{x} in the argument of logistic sigmoid. Hence the decision boundaries are linear in input space.

Example of Two Gaussian Models



Class-conditional densities for two classes



The corresponding posterior probability $p(C_1|\mathbf{x})$, given by the sigmoid function of a linear function of \mathbf{x} .

Case of K Classes

- For the case of K classes, the posterior is a softmax function:

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{\sum_j p(\mathbf{x}|\mathcal{C}_j)p(\mathcal{C}_j)} = \frac{\exp(a_k)}{\sum_j \exp(a_j)},$$

$$a_k = \mathbf{w}_k^T \mathbf{x} + w_{k0},$$

where, similar to the 2-class case, we have defined:

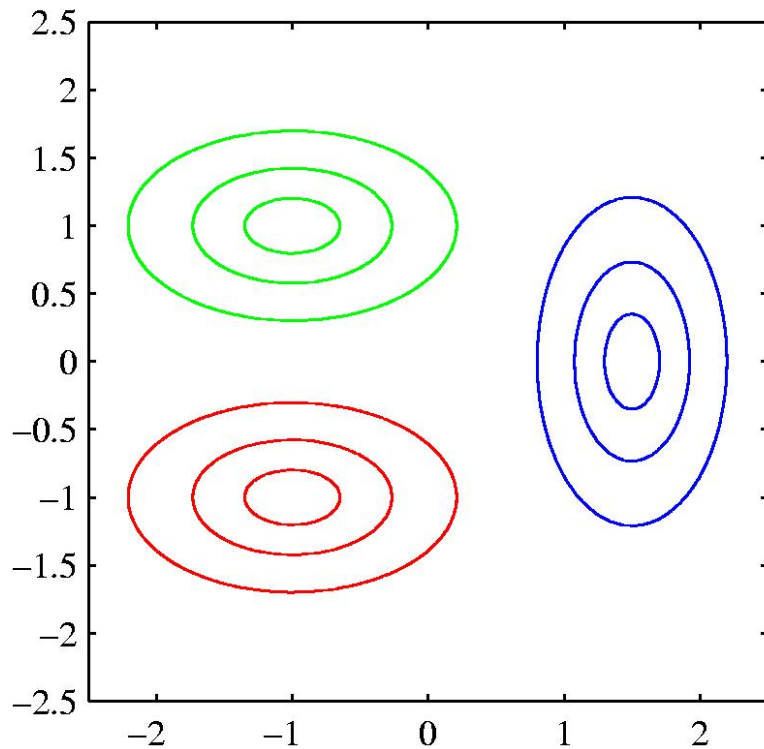
$$\mathbf{w}_k = \Sigma^{-1} \boldsymbol{\mu}_k,$$

$$w_{k0} = -\frac{1}{2} \boldsymbol{\mu}_k^T \Sigma^{-1} \boldsymbol{\mu}_k + \ln p(\mathcal{C}_k).$$

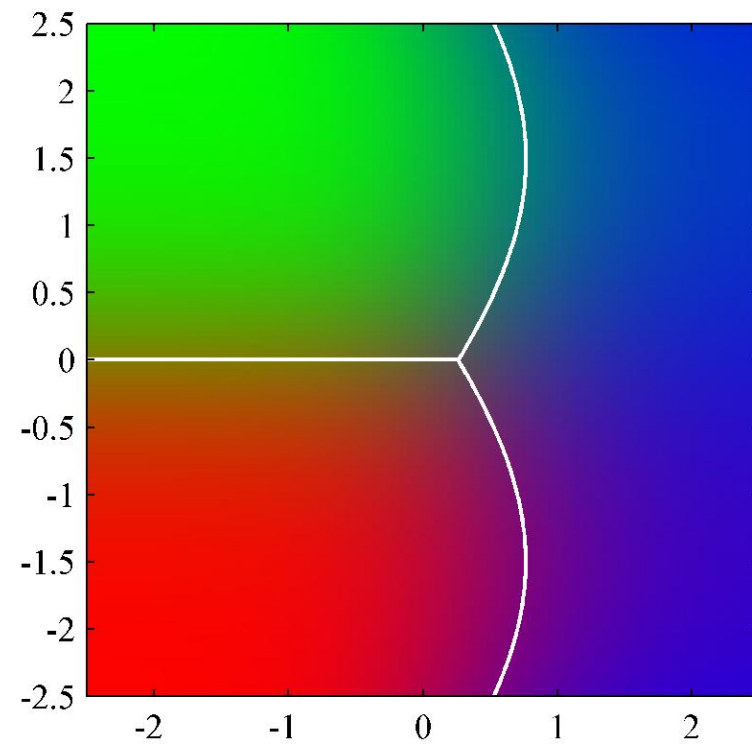
- Again, the decision boundaries are linear in input space.
- If we allow each class-conditional density to have its own covariance, we will obtain quadratic functions of \mathbf{x} .
- This leads to a quadratic discriminant.

Quadratic Discriminant

The decision boundary is linear when the covariance matrices are the same and quadratic when they are not.



Class-conditional densities for three classes



The corresponding posterior probabilities for three classes.

Maximum Likelihood Solution

- Consider the case of two classes, each having a Gaussian class-conditional density with shared covariance matrix.
- We observe a dataset $\{\mathbf{x}_n, t_n\}$, $n = 1, \dots, N$.
 - Here $t_n=1$ denotes class C_1 , and $t_n=0$ denotes class C_2 .
 - Also denote $p(C_1) = \pi$, $p(C_2) = 1 - \pi$.
- The **likelihood function** takes form:

$$p(\mathbf{t}, \mathbf{X} | \pi, \boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \boldsymbol{\Sigma}) = \prod_{n=1}^N \left[\pi \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_1, \boldsymbol{\Sigma}) \right]^{t_n} \left[(1 - \pi) \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_2, \boldsymbol{\Sigma}) \right]^{1-t_n} .$$

Data points
from class C_1 .

Data points
from class C_2 .

- As usual, we will maximize the log of the likelihood function.

Maximum Likelihood Solution

$$p(\mathbf{t}, \mathbf{X} | \pi, \boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \boldsymbol{\Sigma}) = \prod_{n=1}^N \left[\pi \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_1, \boldsymbol{\Sigma}) \right]^{t_n} \left[(1 - \pi) \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_2, \boldsymbol{\Sigma}) \right]^{1-t_n}.$$

- Maximizing the respect to π , we look at the terms of the log-likelihood functions that depend on π :

$$\sum_n [t_n \ln \pi + (1 - t_n) \ln(1 - \pi)] + \text{const.}$$

Differentiating, we get:

$$\pi = \frac{1}{N} \sum_{n=1}^N t_n = \frac{N_1}{N_1 + N_2}.$$

- Maximizing the respect to $\boldsymbol{\mu}_1$, we look at the terms of the log-likelihood functions that depend on $\boldsymbol{\mu}_1$:

$$\sum_n t_n \ln \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_1, \boldsymbol{\Sigma}) = -\frac{1}{2} \sum_n t_n (\mathbf{x}_n - \boldsymbol{\mu}_1)^T \boldsymbol{\Sigma}^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_1) + \text{const.}$$

Differentiating, we get:

$$\boldsymbol{\mu}_1 = \frac{1}{N_1} \sum_{n=1}^N t_n \mathbf{x}_n.$$

And similarly:

$$\boldsymbol{\mu}_2 = \frac{1}{N_2} \sum_{n=1}^N (1 - t_n) \mathbf{x}_n.$$

Maximum Likelihood Solution

$$p(\mathbf{t}, \mathbf{X} | \pi, \boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \boldsymbol{\Sigma}) = \prod_{n=1}^N \left[\pi \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_1, \boldsymbol{\Sigma}) \right]^{t_n} \left[(1 - \pi) \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_2, \boldsymbol{\Sigma}) \right]^{1-t_n}.$$

- Maximizing the respect to $\boldsymbol{\Sigma}$:

$$\begin{aligned} & -\frac{1}{2} \sum_n t_n \ln |\boldsymbol{\Sigma}| - \frac{1}{2} \sum_n t_n (\mathbf{x}_n - \boldsymbol{\mu}_1)^T \boldsymbol{\Sigma}^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_1) \\ & -\frac{1}{2} \sum_n (1 - t_n) \ln |\boldsymbol{\Sigma}| - \frac{1}{2} \sum_n (1 - t_n) (\mathbf{x}_n - \boldsymbol{\mu}_2)^T \boldsymbol{\Sigma}^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_2) \\ & = -\frac{N}{2} \ln |\boldsymbol{\Sigma}| - \frac{N}{2} \text{Tr}(\boldsymbol{\Sigma}^{-1} \mathbf{S}). \end{aligned}$$

- Here we defined:

$$\mathbf{S} = \frac{N_1}{N} \mathbf{S}_1 + \frac{N_2}{N} \mathbf{S}_2,$$

$$\mathbf{S}_1 = \frac{1}{N_1} \sum_{n \in \mathcal{C}_1} (\mathbf{x}_n - \boldsymbol{\mu}_1)(\mathbf{x}_n - \boldsymbol{\mu}_1)^T,$$

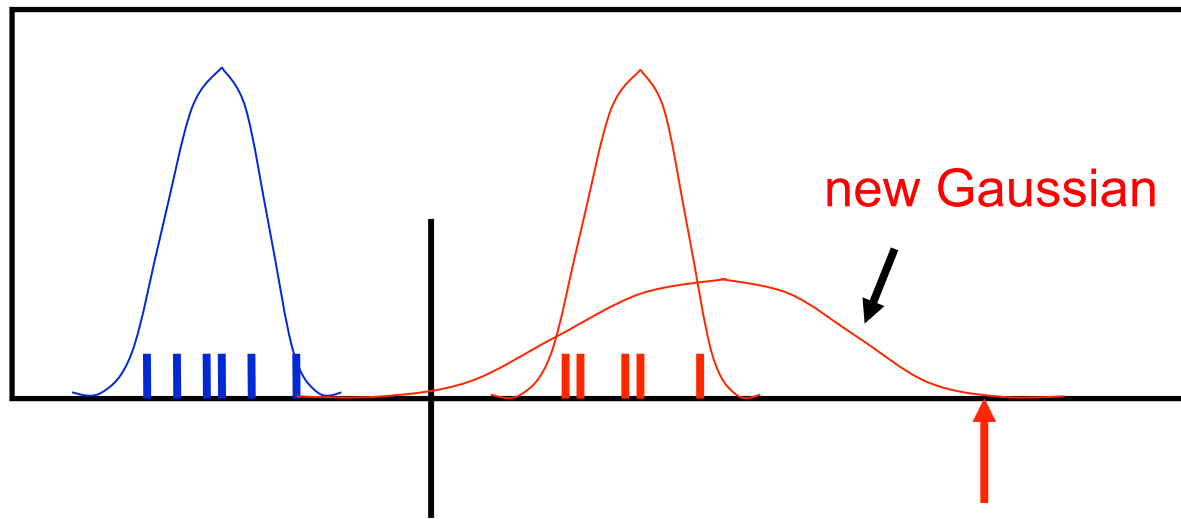
$$\mathbf{S}_2 = \frac{1}{N_2} \sum_{n \in \mathcal{C}_2} (\mathbf{x}_n - \boldsymbol{\mu}_2)(\mathbf{x}_n - \boldsymbol{\mu}_2)^T.$$

- Using standard results for a Gaussian distribution we have:

$$\boldsymbol{\Sigma} = \mathbf{S}.$$

- Maximum likelihood solution represents a **weighted average of the covariance matrices associated with each of the two classes.**

Example



decision
boundary

What happens to the
decision boundary if we
add a new red point here?

- For generative fitting, the red mean moves rightwards but the decision boundary moves leftwards! If you believe the data is Gaussian, this is reasonable.
- How can we fix this?

Three Approaches to Classification

- Construct a **discriminant function** that directly maps each input vector to a specific class.
- Model the conditional probability distribution $p(\mathcal{C}_k|\mathbf{x})$, and then use this distribution to make optimal decisions.
- There are two approaches:

- **Discriminative Approach**: Model $p(\mathcal{C}_k|\mathbf{x})$, directly, for example by representing them as parametric models, and optimize for parameters using the training set (e.g. logistic regression).

- **Generative Approach**: Model class conditional densities $p(\mathbf{x}|\mathcal{C}_k)$ together with the prior probabilities $p(\mathcal{C}_k)$ for the classes. Infer posterior probability using Bayes' rule:

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\mathbf{x})}.$$

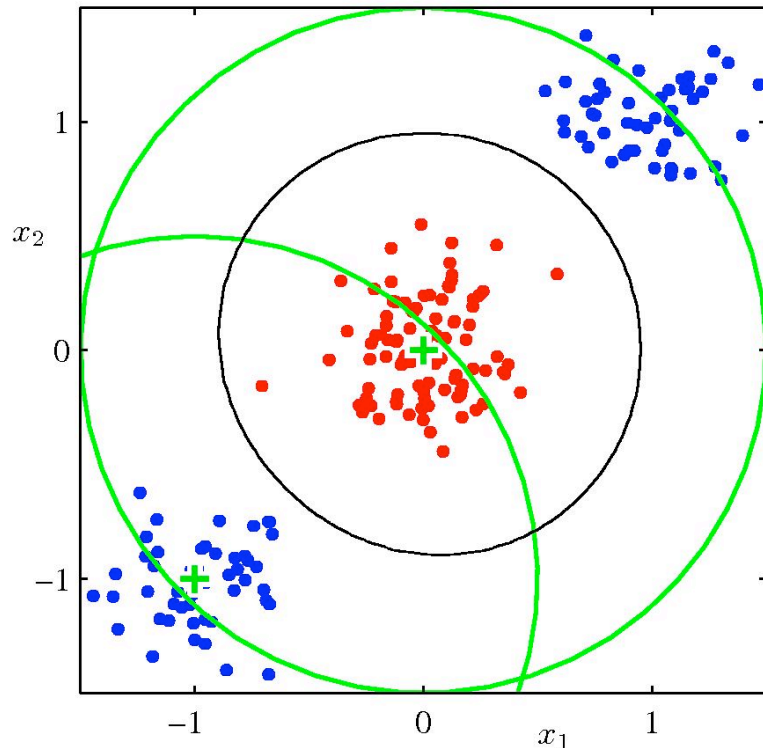
We will consider next.

Fixed Basis Functions

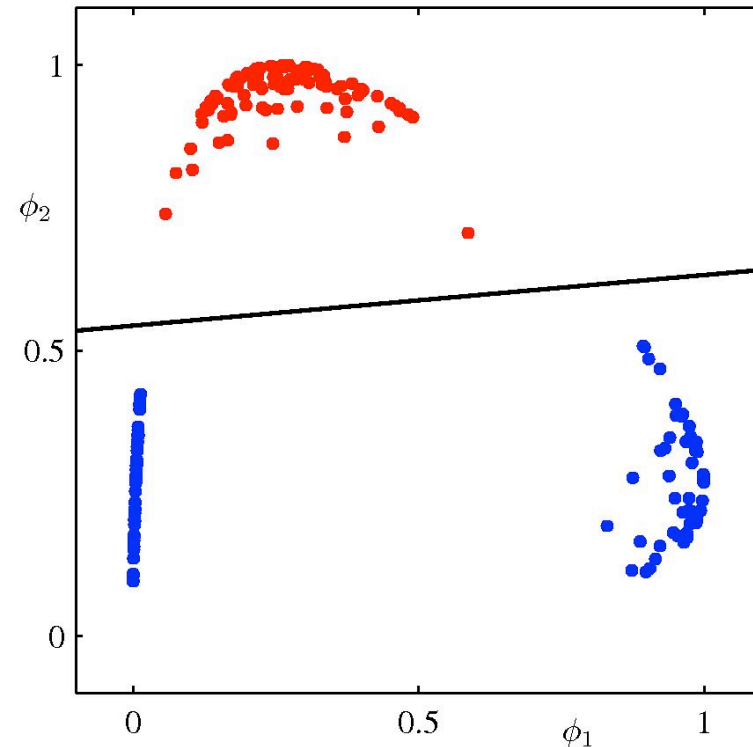
- So far, we have considered classification models that work directly in the input space.
- All considered algorithms are equally applicable if we first make a fixed nonlinear transformation of the input space using vector of basis functions $\phi(\mathbf{x})$.
- Decision boundaries will be linear in the feature space ϕ , but would correspond to nonlinear boundaries in the original input space \mathbf{x} .
- Classes that are linearly separable in the feature space $\phi(\mathbf{x})$ need not be linearly separable in the original input space.

Linear Basis Function Models

Original input space



Corresponding feature space using two Gaussian basis functions



- We define two Gaussian basis functions with centers shown by green the crosses, and with contours shown by the green circles.
- Linear decision boundary (right) is obtained using logistic regression, and corresponds to nonlinear decision boundary in the input space (left, black curve).

Logistic Regression

- Consider the problem of two-class classification.
- We have seen that the posterior probability of class C_1 can be written as a **logistic sigmoid function**:

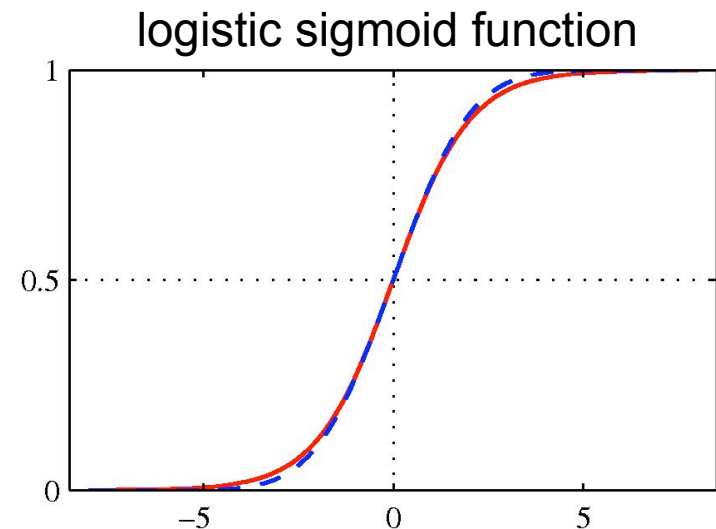
$$p(C_1|\mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{w}^T \mathbf{x})} = \sigma(\mathbf{w}^T \mathbf{x}),$$

where $p(C_2|\mathbf{x}) = 1 - p(C_1|\mathbf{x})$, and we omit the bias term for clarity.

- This model is known **as logistic regression** (although this is a model for classification rather than regression).

Note that for generative models, we would first determine the class conditional densities and class-specific priors, and then use Bayes' rule to obtain the posterior probabilities.

Here we model $p(C_k|\mathbf{x})$ directly.



ML for Logistic Regression

- We observed a training dataset $\{\mathbf{x}_n, t_n\}$, $n = 1, \dots, N$; $t_n \in \{0, 1\}$.
- Maximize the probability of getting the label right, so the likelihood function takes form:

$$p(\mathbf{t}|\mathbf{X}, \mathbf{w}) = \prod_{n=1}^N \left[y_n^{t_n} (1 - y_n)^{1-t_n} \right], \quad y_n = \sigma(\mathbf{w}^T \mathbf{x}_n).$$

- Taking the negative log of the likelihood, we can define **cross-entropy error function** (that we want to minimize):

$$E(\mathbf{w}) = -\ln p(\mathbf{t}|\mathbf{X}, \mathbf{w}) = -\sum_{n=1}^N \left[t_n \ln y_n + (1 - t_n) \ln(1 - y_n) \right] = \sum_{n=1}^N E_n.$$

- Differentiating and using the chain rule:

$$\frac{d}{dy_n} E_n = \frac{y_n - t_n}{y_n(1 - y_n)}, \quad \frac{d}{d\mathbf{w}} y_n = y_n(1 - y_n)\mathbf{x}_n, \quad \frac{d}{da} \sigma(a) = \sigma(a)(1 - \sigma(a)).$$

$$\frac{d}{d\mathbf{w}} E_n = \frac{dE_n}{dy_n} \frac{dy_n}{d\mathbf{w}} = (y_n - t_n)\mathbf{x}_n.$$

- Note that the factor involving the derivative of the logistic function cancelled.

ML for Logistic Regression

- We therefore obtain:

$$\nabla E(\mathbf{w}) = \sum_{n=1}^N (y_n - t_n) \mathbf{x}_n.$$

prediction target

- This takes exactly the same form as **the gradient of the sum-of-squares error function** for the linear regression model.
- Unlike in linear regression, there is **no closed form solution**, due to nonlinearity of the logistic sigmoid function.
- **The error function is convex** and can be optimized using standard gradient-based (or more advanced) optimization techniques.
- Easy to adapt to the **online learning setting**.

Multiclass Logistic Regression

- For the multiclass case, we represent posterior probabilities by a **softmax transformation** of linear functions of input variables :

$$p(\mathcal{C}_k|\mathbf{x}) = y_k(\mathbf{x}) = \frac{\exp(\mathbf{w}_k^T \mathbf{x})}{\sum_j \exp(\mathbf{w}_j^T \mathbf{x})}.$$

- Unlike in generative models, here we will use maximum likelihood to **determine parameters of this discriminative model directly**.
- As usual, we observed a dataset $\{\mathbf{x}_n, t_n\}$, $n = 1, \dots, N$, where we use 1-of-K encoding for the target vector \mathbf{t}_n .
- So if \mathbf{x}_n belongs to class \mathcal{C}_k , then \mathbf{t} is a binary vector of length K containing a single 1 for element k (the correct class) and 0 elsewhere.
- For example, if we have K=5 classes, then an input that belongs to class 2 would be given a target vector:

$$t = (0, 1, 0, 0, 0)^T.$$

Multiclass Logistic Regression

- We can write down the likelihood function:

$$p(\mathbf{T}|\mathbf{X}, \mathbf{w}_1, \dots, \mathbf{w}_K) = \prod_{n=1}^N \left[\prod_{k=1}^K p(\mathcal{C}_k|\mathbf{x}_n)^{t_{nk}} \right] = \prod_{n=1}^N \left[\prod_{k=1}^K y_{nk}^{t_{nk}} \right]$$

 $N \times K$ binary matrix of target variables.

Only one term corresponding to correct class contributes.

where $y_{nk} = p(\mathcal{C}_k|\mathbf{x}_n) = \frac{\exp(\mathbf{w}_k^T \mathbf{x}_n)}{\sum_j \exp(\mathbf{w}_j^T \mathbf{x}_n)}$.

- Taking the negative logarithm gives the **cross-entropy entropy function** for multi-class classification problem:

$$E(\mathbf{w}_1, \dots, \mathbf{w}_K) = -\ln p(\mathbf{T}|\mathbf{X}, \mathbf{w}_1, \dots, \mathbf{w}_K) = -\sum_{n=1}^N \left[\sum_{k=1}^K t_{nk} \ln y_{nk} \right].$$

- Taking the gradient:

$$\nabla E_{\mathbf{w}_j}(\mathbf{w}_1, \dots, \mathbf{w}_K) = \sum_{n=1}^N (y_{nj} - t_{nj}) \mathbf{x}_n.$$

Special Case of Softmax

- If we consider a softmax function for two classes:

$$p(\mathcal{C}_1|\mathbf{x}) = \frac{\exp(a_1)}{\exp(a_1) + \exp(a_2)} = \frac{1}{1 + \exp(-(a_1 - a_2))} = \sigma(a_1 - a_2).$$

- So the **logistic sigmoid is just a special case of the softmax function** that avoids using redundant parameters:
 - Adding the same constant to both a_1 and a_2 has no effect.
 - The over-parameterization of the softmax is because probabilities must add up to one.

Recap

- **Generative approach:** Determine the class conditional densities and class-specific priors, and then use Bayes' rule to obtain the posterior probabilities.
 - Different models can be trained separately on different machines.
 - It is easy to add a new class without retraining all the other classes.
- **Discriminative approach:** Train all of the model parameters to maximize the probability of getting the labels right.
Model $p(\mathcal{C}_k|\mathbf{x})$ directly.

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\mathbf{x})}.$$

Bayesian Logistic Regression

- We next look at the Bayesian treatment of logistic regression.
- For the two-class problem, the likelihood takes form:

$$p(\mathbf{t}|\mathbf{X}, \mathbf{w}) = \prod_{n=1}^N \left[y_n^{t_n} (1 - y_n)^{1-t_n} \right], \quad y_n = \frac{1}{1 + \exp(-\mathbf{w}^T \mathbf{x}_n)} = \sigma(\mathbf{w}^T \mathbf{x}_n).$$

- Similar to Bayesian linear regression, we could start with a Gaussian prior:

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_0, \mathbf{S}_0).$$

- However, the posterior distribution

$$p(\mathbf{w}|\mathbf{X}, \mathbf{t}) \propto p(\mathbf{t}|\mathbf{X}, \mathbf{w})p(\mathbf{w}).$$

is no longer Gaussian, and we cannot analytically integrate over model parameters \mathbf{w} .

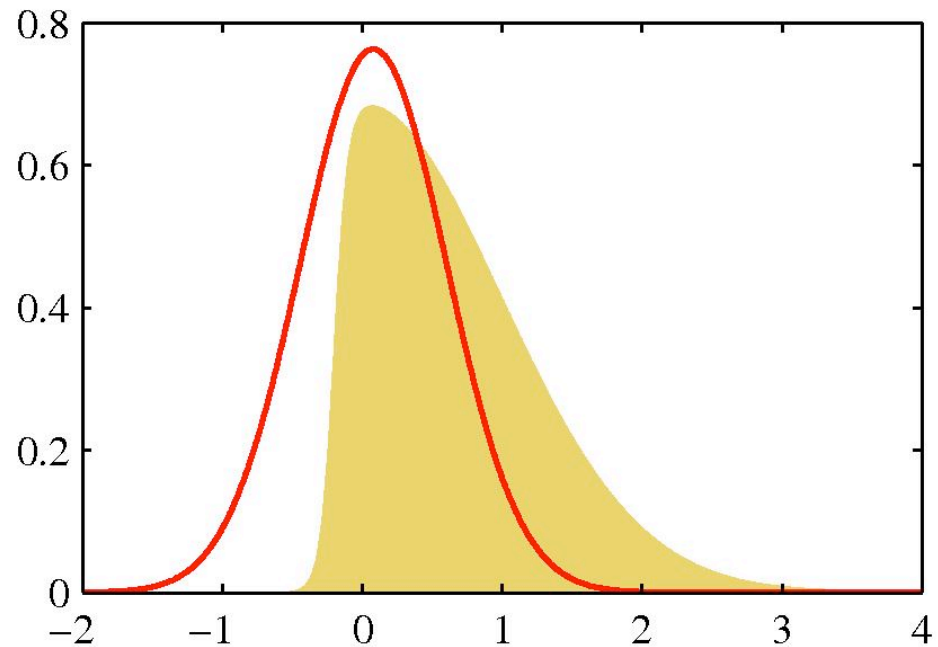
- We need to introduce some approximations.

Pictorial illustration

- Consider a simple distribution:

$$p(w) \propto \exp(-w^2)\sigma(20w + 4).$$

- The plot shows the normalized distribution (in yellow), which is not Gaussian.
- The red curve displays the corresponding Gaussian approximation.



Recap: Computational Challenge of Bayesian Framework

Remember: the big challenge is computing the posterior distribution. There are several main approaches:

- **Analytical integration**: If we use “conjugate” priors, the posterior distribution can be computed analytically (we saw this for Bayesian linear regression).

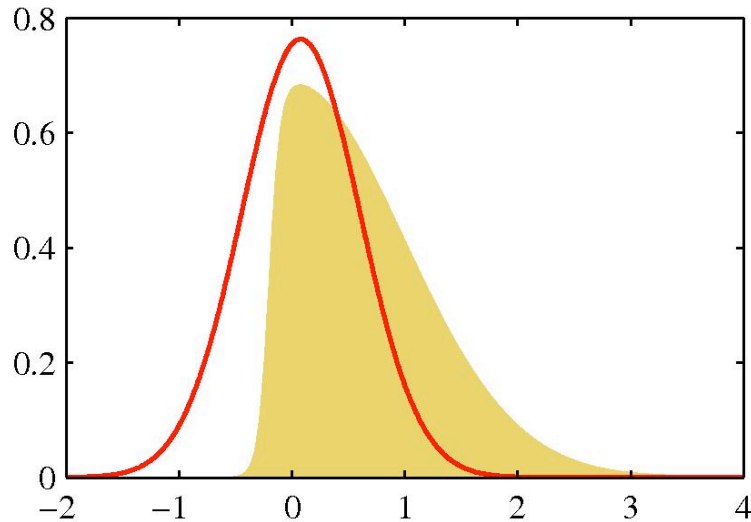
We will consider Laplace approximation next.

- **Gaussian (Laplace) approximation**: Approximate the posterior distribution with a Gaussian. Works well when there is a lot of data compared to the model complexity (as posterior is close to Gaussian).

- **Monte Carlo integration**: The dominant current approach is Markov Chain Monte Carlo (MCMC) -- simulate a Markov chain that converges to the posterior distribution. It can be applied to a wide variety of problems.

- **Variational approximation**: A cleverer way to approximate the posterior. It often works much faster, but not as general as MCMC.

Laplace Approximation



- We will use the following notation:

$$p(\mathbf{z}) = \frac{\tilde{p}(\mathbf{z})}{\mathcal{Z}}, \quad \mathcal{Z} = \int \tilde{p}(\mathbf{z}) d\mathbf{z}.$$

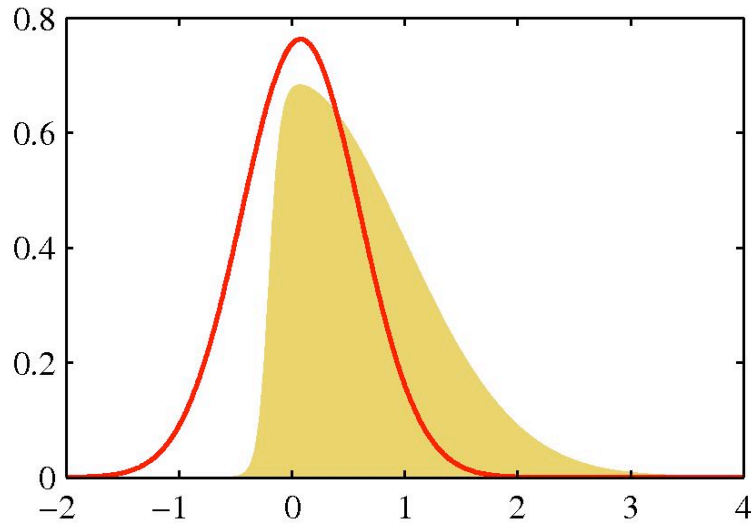
- We can evaluate $\tilde{p}(\mathbf{z})$ point-wise but cannot evaluate \mathcal{Z} .

- For example

$$p(\mathbf{w}|\mathcal{D}) = \frac{p(\mathcal{D}|\mathbf{w})p(\mathbf{w})}{p(\mathcal{D})}.$$

- **Goal:** Find a Gaussian approximation $q(\mathbf{z})$ which is centered on a mode of the distribution $p(\mathbf{z})$.

Laplace Approximation



- We will use the following notation:

$$p(\mathbf{z}) = \frac{\tilde{p}(\mathbf{z})}{\mathcal{Z}}, \quad \mathcal{Z} = \int \tilde{p}(\mathbf{z}) d\mathbf{z}.$$

- At the stationary point \mathbf{z}_0 , the gradient $\nabla \tilde{p}(\mathbf{z}_0)$ vanishes.
- Consider a **Taylor approximation** $\ln \tilde{p}(\mathbf{z})$ around \mathbf{z}_0 .

$$\ln \tilde{p}(\mathbf{z}) \approx \ln \tilde{p}(\mathbf{z}_0) - \frac{1}{2}(\mathbf{z} - \mathbf{z}_0)^T A(\mathbf{z} - \mathbf{z}_0),$$

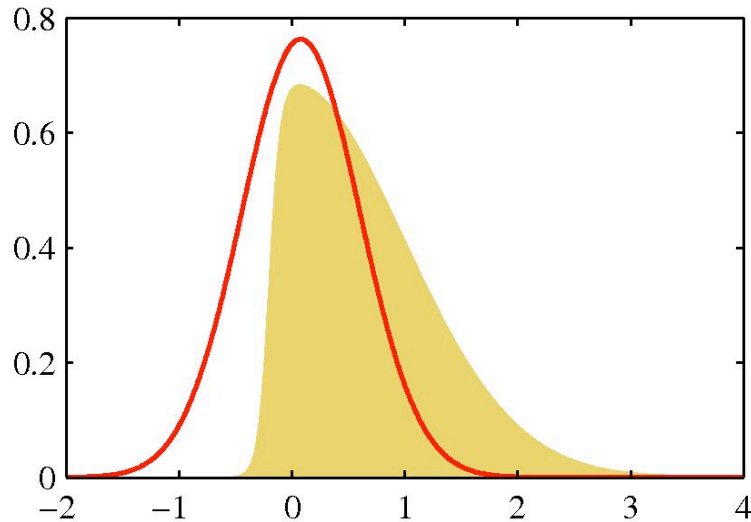
where A is a Hessian matrix:

$$A = - \nabla \nabla \ln \tilde{p}(\mathbf{z})|_{\mathbf{z}=\mathbf{z}_0}.$$

- Exponentiating both sides:

$$\tilde{p}(\mathbf{z}) \approx \tilde{p}(\mathbf{z}_0) \exp \left(-\frac{1}{2}(\mathbf{z} - \mathbf{z}_0)^T A(\mathbf{z} - \mathbf{z}_0) \right).$$

Laplace Approximation



- We will use the following notation:

$$p(\mathbf{z}) = \frac{\tilde{p}(\mathbf{z})}{\mathcal{Z}}, \quad \mathcal{Z} = \int \tilde{p}(\mathbf{z}) d\mathbf{z}.$$

- Using Taylor approximation, we get:

$$\tilde{p}(\mathbf{z}) \approx \tilde{p}(\mathbf{z}_0) \exp \left(-\frac{1}{2} (\mathbf{z} - \mathbf{z}_0)^T A (\mathbf{z} - \mathbf{z}_0) \right).$$

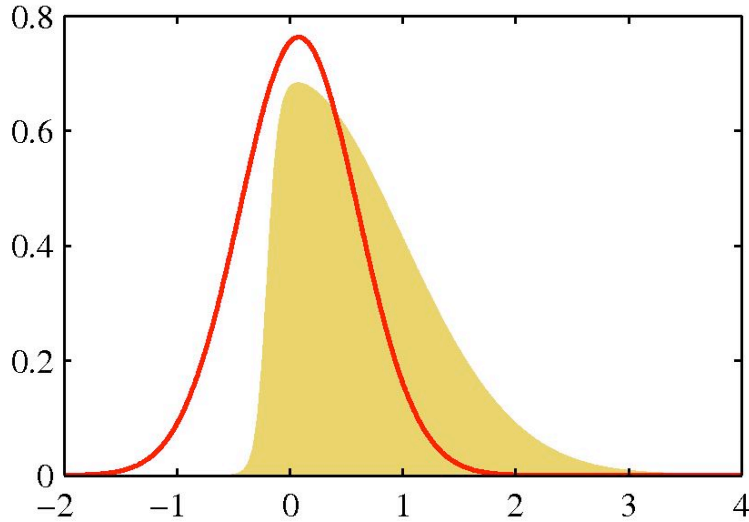
- Hence a **Gaussian approximation** for $p(\mathbf{z})$ is:

$$q(\mathbf{z}) = \frac{|A|^{1/2}}{(2\pi)^{D/2}} \exp \left(-\frac{1}{2} (\mathbf{z} - \mathbf{z}_0)^T A (\mathbf{z} - \mathbf{z}_0) \right),$$

where \mathbf{z}_0 is the mode of $p(\mathbf{z})$, and A is the Hessian:

$$A = -\nabla \nabla \ln \tilde{p}(\mathbf{z})|_{\mathbf{z}=\mathbf{z}_0}.$$

Laplace Approximation



- We will use the following notation:

$$p(\mathbf{z}) = \frac{\tilde{p}(\mathbf{z})}{\mathcal{Z}}, \quad \mathcal{Z} = \int \tilde{p}(\mathbf{z}) d\mathbf{z}.$$

- Using Taylor approximation, we get:

$$\tilde{p}(\mathbf{z}) \approx \tilde{p}(\mathbf{z}_0) \exp\left(-\frac{1}{2}(\mathbf{z} - \mathbf{z}_0)^T A(\mathbf{z} - \mathbf{z}_0)\right).$$

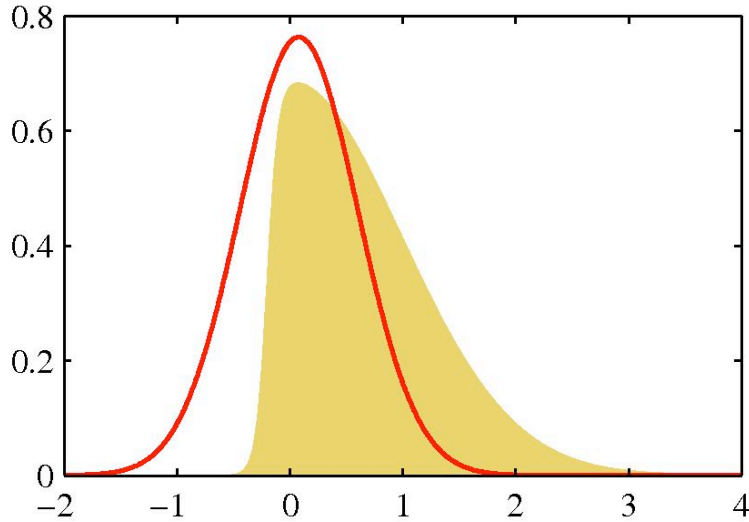
- **Bayesian inference:** $p(\mathbf{w}|\mathcal{D}) = \frac{p(\mathcal{D}|\mathbf{w})p(\mathbf{w})}{p(\mathcal{D})}$.

- **Identify:** $\tilde{p}(\theta|\mathcal{D}) = p(\mathcal{D}|\theta)p(\theta)$, $\mathcal{Z} = \int p(\mathcal{D}|\theta)p(\theta)d\theta$.

- The **posterior is approximately Gaussian** around the MAP estimate:

$$p(\theta|\mathcal{D}) \approx \frac{|A|^{1/2}}{(2\pi)^{D/2}} \exp\left(-\frac{1}{2}(\theta - \theta_{\text{MAP}})^T A(\theta - \theta_{\text{MAP}})\right).$$

Laplace Approximation



- We will use the following notation:

$$p(\mathbf{z}) = \frac{\tilde{p}(\mathbf{z})}{\mathcal{Z}}, \quad \mathcal{Z} = \int \tilde{p}(\mathbf{z}) d\mathbf{z}.$$

- Using Taylor approximation, we get:

$$\tilde{p}(\mathbf{z}) \approx \tilde{p}(\mathbf{z}_0) \exp\left(-\frac{1}{2}(\mathbf{z} - \mathbf{z}_0)^T A(\mathbf{z} - \mathbf{z}_0)\right).$$

$$\mathcal{Z} = \int \tilde{p}(\mathbf{z}) d\mathbf{z} \approx \tilde{p}(\mathbf{z}_0) \int \exp\left(-\frac{1}{2}(\mathbf{z} - \mathbf{z}_0)^T A(\mathbf{z} - \mathbf{z}_0)\right) = \tilde{p}(\mathbf{z}_0) \frac{(2\pi)^{D/2}}{|A|^{1/2}}.$$

- We can approximate Model Evidence: $p(\mathcal{D}) = \int p(\mathcal{D}|\theta)P(\theta)d\theta$, using Laplace approximation:

$$\ln p(\mathcal{D}) \approx \underbrace{\ln p(\mathcal{D}|\theta_{\text{MAP}})}_{\text{Data fit}} + \underbrace{\ln P(\theta_{\text{MAP}}) + \frac{D}{2} \ln 2\pi - \frac{1}{2} \ln |A|}_{\text{Occam factor: penalize model complexity}}.$$

Data fit

Occam factor: penalize model complexity

Bayesian Information Criterion

- BIC can be obtained from the Laplace approximation:

$$\ln p(\mathcal{D}) \approx \ln p(\mathcal{D}|\theta_{\text{MAP}}) + \ln P(\theta_{\text{MAP}}) + \frac{D}{2} \ln 2\pi - \frac{1}{2} \ln |A|,$$

by taking the large sample limit ($N \rightarrow \infty$) where N is the number of data points.

$$\ln p(\mathcal{D}) \approx \ln p(\mathcal{D}|\theta_{\text{MAP}}) - \frac{1}{2} D \ln N.$$

- **Quick and easy**, does not depend on the prior.
- Can use **maximum likelihood estimate** instead of the MAP estimate.
- D denotes the number of **well-determined** parameters.
- **Danger**: Counting parameters can be tricky (e.g. infinite models).

Bayesian Logistic Regression

- Remember the likelihood:

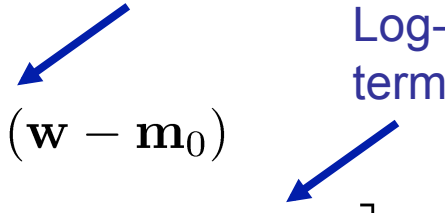
$$p(\mathbf{t}|\mathbf{X}, \mathbf{w}) = \prod_{n=1}^N \left[y_n^{t_n} (1 - y_n)^{1-t_n} \right], \quad y_n = \frac{1}{1 + \exp(-\mathbf{w}^T \mathbf{x}_n)} = \sigma(\mathbf{w}^T \mathbf{x}_n).$$

- And the prior: $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_0, \mathbf{S}_0)$.

- The log of the posterior takes form:

$$\begin{aligned} \ln p(\mathbf{w}|\mathbf{X}, \mathbf{t}) = & -\frac{1}{2}(\mathbf{w} - \mathbf{m}_0)^T \mathbf{S}_0^{-1}(\mathbf{w} - \mathbf{m}_0) \\ & + \sum_{n=1}^N \left[t_n \ln y_n + (1 - t_n) \ln(1 - y_n) \right] + \text{const.} \end{aligned}$$

Log-prior term Log-likelihood term



- We first maximize the log-posterior to get the MAP estimate: \mathbf{w}_{MAP} .
- The inverse of covariance is given by the matrix of second derivatives:

$$\mathbf{S}_N^{-1} = -\nabla \nabla \ln p(\mathbf{w}|\mathbf{X}, \mathbf{t}) = \mathbf{S}_0^{-1} + \sum_n y_n(1 - y_n) \mathbf{x}_n \mathbf{x}_n^T.$$

- The Gaussian approximation to the posterior distribution is given by:

$$q(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{w}_{\text{MAP}}, \mathbf{S}_N).$$

Predictive Distribution

- The **predictive distribution** for class C_1 , given a new input \mathbf{x}^* is given by **marginalizing with respect to posterior distribution** $p(\mathbf{w}|\mathbf{X}, \mathbf{t})$, which is itself approximated by a Gaussian distribution:

$$p(C_1|\mathbf{x}^*, \mathbf{t}, \mathbf{X}) = \int p(C_1|\mathbf{x}^*, \mathbf{w})p(\mathbf{w}|\mathbf{t}, \mathbf{X})d\mathbf{w}$$
$$\approx \int \sigma(\mathbf{w}^T \mathbf{x}^*)q(\mathbf{w})d\mathbf{w},$$

Still not tractable.

with the corresponding probability for class C_2 given by:

$$p(C_2|\mathbf{x}^*, \mathbf{t}, \mathbf{X}) = 1 - p(C_1|\mathbf{x}^*, \mathbf{t}, \mathbf{X}).$$

- The convolution of Gaussian with logistic sigmoid cannot be evaluated analytically.

Predictive Distribution

$$p(\mathcal{C}_1 | \mathbf{x}^*, \mathbf{X}, \mathbf{t}) \approx \int \sigma(\mathbf{w}^T \mathbf{x}^*) q(\mathbf{w}) d\mathbf{w}.$$

- Note that the logistic function depends on \mathbf{w} only through its projection onto \mathbf{x}^* . Denoting $a = \mathbf{w}^T \mathbf{x}^*$, we have:

$$\sigma(\mathbf{w}^T \mathbf{x}^*) = \int \delta(a - \mathbf{w}^T \mathbf{x}^*) \sigma(a) da,$$

where δ is the Dirac delta function. Hence

$$\int \sigma(\mathbf{w}^T \mathbf{x}^*) q(\mathbf{w}) d\mathbf{w} = \int \sigma(a) p(a) da, \quad \text{where } p(a) = \int \delta(a - \mathbf{w}^T \mathbf{x}^*) q(\mathbf{w}) d\mathbf{w}.$$

1-dimensional
integral.

- Let us characterize $p(a)$.
- The delta function imposes a linear constraint on \mathbf{w} . It forms a marginal distribution from the joint $q(\mathbf{w})$ by marginalizing out all directions orthogonal to \mathbf{x}^* .
- Since $q(\mathbf{w})$ is Gaussian, the marginal is also Gaussian.

Predictive Distribution

$$\int \sigma(\mathbf{w}^T \mathbf{x}^*) q(\mathbf{w}) d\mathbf{w} = \int \sigma(a) p(a) da, \quad \text{where } p(a) = \int \delta(a - \mathbf{w}^T \mathbf{x}^*) q(\mathbf{w}) d\mathbf{w}.$$

- We can evaluate the mean and variance of the marginal $p(a)$.

$$\mu_a = \mathbb{E}[a] = \int a p(a) da = \int \mathbf{w}^T \mathbf{x}^* q(\mathbf{w}) d\mathbf{w} = \mathbf{w}_{\text{MAP}}^T \mathbf{x}^*.$$

$$\sigma_a^2 = \text{var}[a] = \int p(a) [a^2 - \mathbb{E}[a]^2] =$$

$$= \int [(\mathbf{w}^T \mathbf{x}^*)^2 - (\mathbf{w}_{\text{MAP}}^T \mathbf{x}^*)^2] q(\mathbf{w}) d\mathbf{w} = \mathbf{x}^{*T} \mathbf{S}_N \mathbf{x}^*.$$

Same form as the predictive distribution for the Bayesian linear regression model.

- Hence we obtain approximate predictive:

$$p(\mathcal{C}_1 | \mathbf{x}^*, \mathbf{X}, \mathbf{t}) \approx \int \sigma(\mathbf{w}^T \mathbf{x}^*) q(\mathbf{w}) d\mathbf{w} = \int \sigma(a) \mathcal{N}(a | \mu_a, \sigma_a^2).$$

- The integral is 1-dimensional and can further be approximated via:

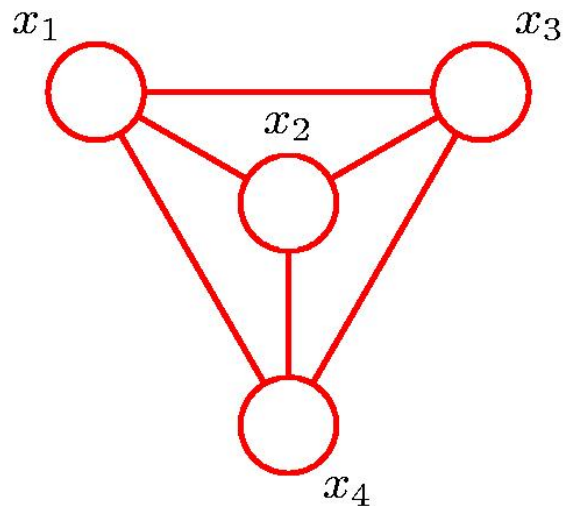
$$\int \sigma(a) \mathcal{N}(a | \mu_a, \sigma_a^2) \approx \sigma(k \mu_a), \quad \text{where } k = (1 + \pi \sigma_a^2 / 8)^{-1/2}.$$

Graphical Models

- Probabilistic graphical models provide a powerful framework for representing **dependency structure between random variables**.
- Graphical models offer several useful properties:
 - They provide **a simple way to visualize the structure of a probabilistic model** and can be used to motivate new models.
 - They provide **various insights into the properties of the model**, including conditional independence.
 - Complex computations (e.g. inference and learning in sophisticated models) can be expressed in terms of **graphical manipulations**.

Graphical Models

- A graph contains a set of nodes (vertices) connected by links (edges or arcs)



- In a probabilistic graphical model, each **node** represents a **random variable**, and **links** represent **probabilistic dependencies** between random variables.
- The graph specifies the way in which the joint distribution over all random variables decomposes into a **product of factors**, where each factor depends on a subset of the variables.
- Two types of graphical models:
 - **Bayesian networks**, also known as Directed Graphical Models (the links have a particular directionality indicated by the arrows)
 - **Markov Random Fields**, also known as Undirected Graphical Models (the links do not carry arrows and have no directional significance).
- **Hybrid graphical models** that combine directed and undirected graphical models, such as Deep Belief Networks.

Bayesian Networks

- Directed Graphs are useful for expressing **causal relationships** between random variables.
- Let us consider an arbitrary joint distribution $p(a, b, c)$ over three random variables a, b , and c .
- Note that at this point, we do not need to specify anything else about these variables (e.g. whether they are discrete or continuous).
- By application of the **product rule of probability** (twice), we get

$$p(a, b, c) = p(c|a, b)p(a, b) = p(c|a, b)p(b|a)p(a)$$

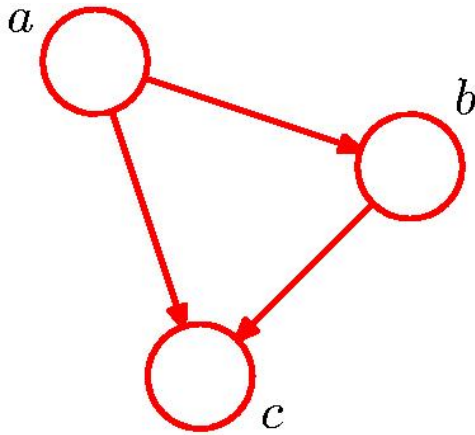
- This decomposition holds for any choice of the joint distribution.

Bayesian Networks

- By application of the product rule of probability (twice), we get

$$p(a, b, c) = p(c|a, b)p(a, b) = p(c|a, b)p(b|a)p(a)$$

- Represent the joint distribution in terms of a simple graphical model:



- Introduce a node for each of the random variables.
- Associate each node with the corresponding conditional distribution in above equation.
- For each conditional distribution we add directed links to the graphs from the nodes corresponding to the variables on which the distribution is conditioned.

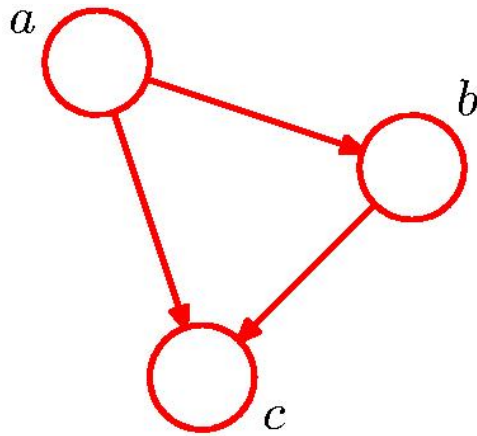
- Hence for the factor $p(c|a, b)$, there will be links from nodes a and b to node c.
- For the factor $p(a)$, there will be no incoming links.

Bayesian Networks

- By application of the product rule of probability (twice), we get

$$p(a, b, c) = p(c|a, b)p(a, b) = p(c|a, b)p(b|a)p(a)$$

- If there is a link going from node a to node b, then we say that:



- node a is a **parent** of node b.
- node b is a **child** of node a.

- For the decomposition, we choose **a specific ordering** of the random variables: a,b,c.

- If we chose a **different ordering**, we would get a **different graphical representation** (we will come back to that point later).

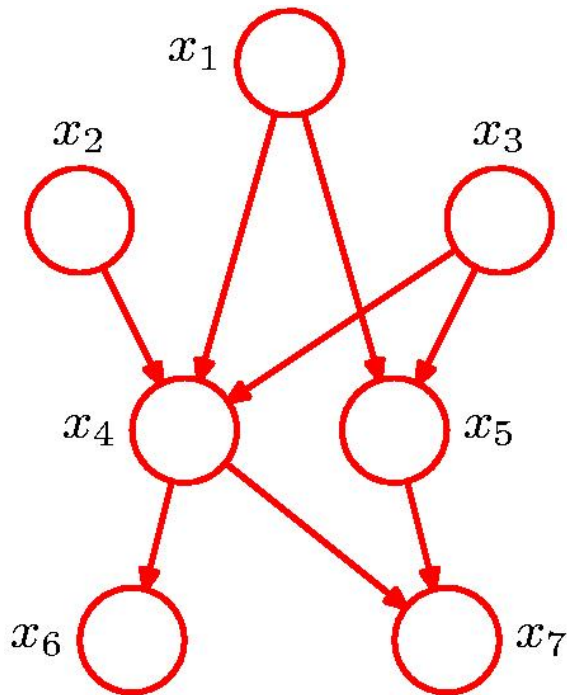
- The joint distribution over K variables factorizes:

$$p(x_1, \dots, x_K) = p(x_K|x_1, \dots, x_{K-1}) \dots p(x_2|x_1)p(x_1)$$

- If each node has incoming links from all lower numbered nodes, then the graph is **fully connected**; there is a link between all pairs of nodes.

Bayesian Networks

- **Absence of links** conveys certain information about the properties of the class of distributions that the graph conveys.



- Note that this graph is not fully connected (e.g. there is no link from x_1 to x_2).

- The joint distribution over x_1, \dots, x_7 can be written as a **product of a set of conditional distributions**.

$$p(x_1, \dots, x_7) = p(x_1)p(x_2)p(x_3)p(x_4|x_1, x_2, x_3) \\ p(x_5|x_1, x_3)p(x_6|x_4)p(x_7|x_4, x_5)$$

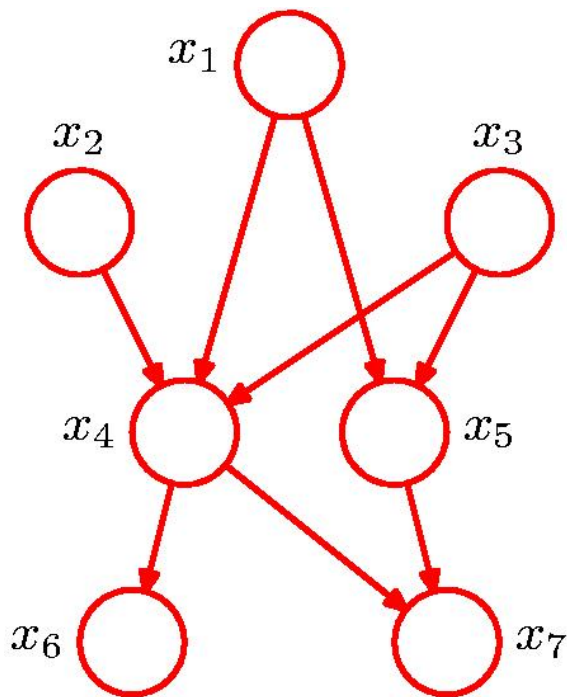
- Note that according to the graph, x_5 will be conditioned only on x_1 and x_3 .

Factorization Property

- The joint distribution defined by the graph is given by **the product of a conditional distribution** for each node conditioned on its parents:

$$p(\mathbf{x}) = \prod_{k=1}^K p(x_k | \text{pa}_k)$$

where pa_k denotes a set of parents for the node x_k .



- This equation expresses a **key factorization property of the joint distribution** for a directed graphical model.

- Important restriction: There must be **no directed cycles!**

- Such graphs are also called **directed acyclic graphs (DAGs)**.

Discrete Variables

- General joint distribution: K^2-1 parameters.



$$p(\mathbf{x}_1, \mathbf{x}_2 | \boldsymbol{\mu}) = \prod_{k=1}^K \prod_{l=1}^K \mu_{kl}^{x_{1k} x_{2l}}$$

- Independent joint distribution: $2(K-1)$ parameters.



$$\hat{p}(\mathbf{x}_1, \mathbf{x}_2 | \boldsymbol{\mu}) = \prod_{k=1}^K \mu_{1k}^{x_{1k}} \prod_{l=1}^K \mu_{2l}^{x_{2l}}$$

- We dropped the link between the nodes, so each variables is described by a separate multinomial distribution.

Discrete Variables

- In general:
 - Fully connected graphs have completely general distributions and have exponential $K^M - 1$ number of parameters (**too complex**).
 - If there are no links, the joint distribution fully factorizes into the product of the marginals, and have $M(K-1)$ parameters (**too simple**).
 - Graphs that have an **intermediate level of connectivity** allow for more general distributions compared to the fully factorized one, while requiring fewer parameters than the general joint distribution.

- Let us look at the example of the chain graph.

Chain Graph

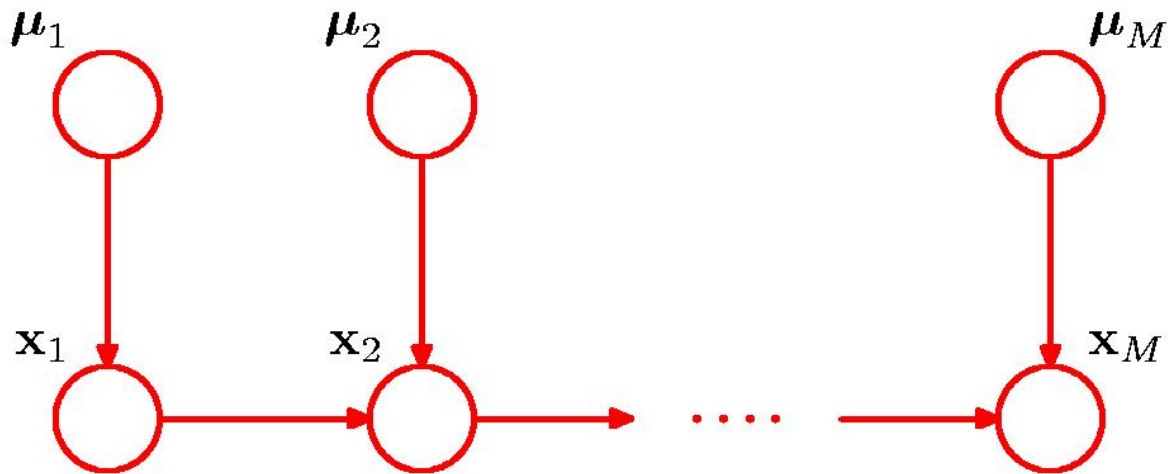
- Consider an M-node Markov chain:



- The marginal distribution $p(\mathbf{x}_1)$ requires $K-1$ parameters.
- The remaining conditional distributions $p(\mathbf{x}_i | \mathbf{x}_{i-1}), i = 2, \dots, M$ require $K(K-1)$ parameters.
- Total number of parameters: $K-1 + (M-1)(K-1)K$, which is quadratic in K and linear in the length M of the chain.
- This graphical model forms the basis of a simple **Hidden Markov Model**.

Adding Priors

- We can turn a graph over discrete random variables into a Bayesian model by introducing Dirichlet priors for the parameters
- From a graphical point of view, each node acquires an additional parent representing the Dirichlet distribution over parameters.

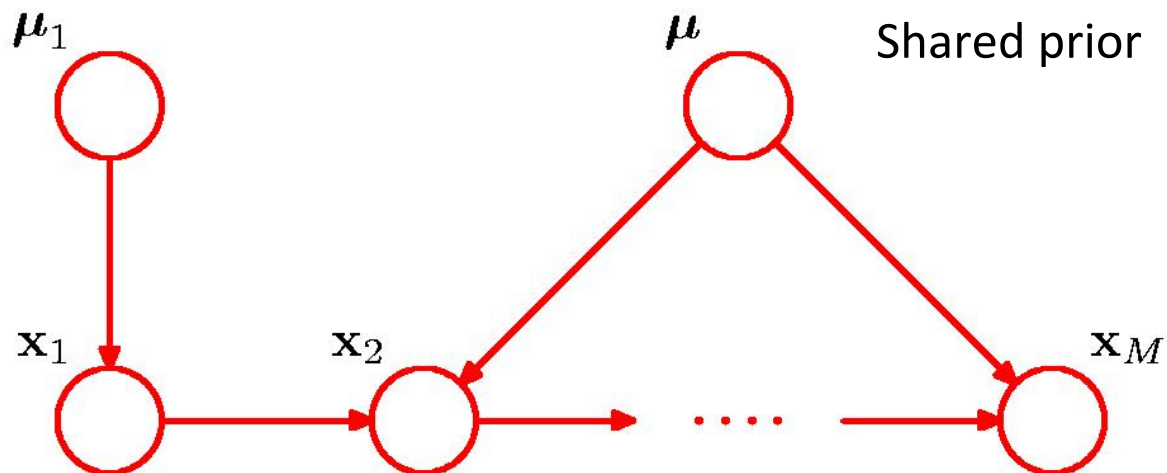


$$p(\{\mathbf{x}_m, \boldsymbol{\mu}_m\}) = p(\mathbf{x}_1 | \boldsymbol{\mu}_1) p(\boldsymbol{\mu}_1) \prod_{m=2}^M p(\mathbf{x}_m | \mathbf{x}_{m-1}, \boldsymbol{\mu}_m) p(\boldsymbol{\mu}_m)$$

$$p(\boldsymbol{\mu}_m) = \text{Dir}(\boldsymbol{\mu}_m | \boldsymbol{\alpha}_m)$$

Shared Prior

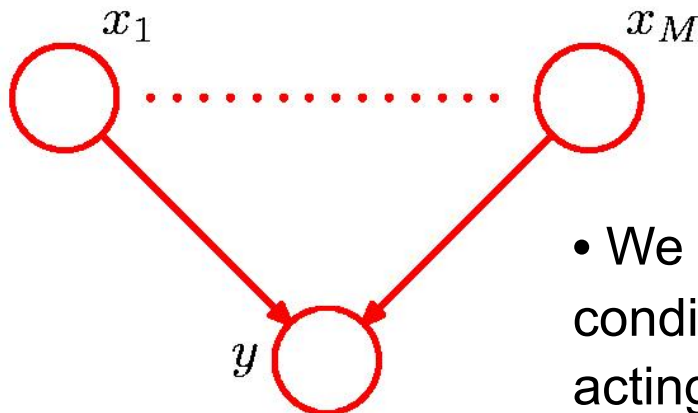
- We can further share the common prior over the parameters governing the conditional distributions.



$$p(\{\mathbf{x}_m\}, \mu_1, \mu) = p(\mathbf{x}_1 | \mu_1) p(\mu_1) \prod_{m=2}^M p(\mathbf{x}_m | \mathbf{x}_{m-1}, \mu) p(\mu)$$

Parameterized Models

- We can use parameterized models to control exponential growth in the number of parameters.



If x_1, \dots, x_M are discrete, K -state variables, $p(y = 1 | x_1, \dots, x_M)$ in general has $O(K^M)$ parameters.

- We can obtain a more parsimonious form of the conditional distribution by using a logistic function acting on a **linear combination of the parent variables**:

$$p(y = 1 | x_1, \dots, x_M) = \sigma \left(w_0 + \sum_{i=1}^M w_i x_i \right) = \sigma(\mathbf{w}^T \mathbf{x})$$

- This is a more restricted form of conditional distribution, but it requires only $M+1$ parameters (linear growth in the number of parameters).

Linear Gaussian Models

- So far we worked with joint probability distributions over a set of discrete random variables (expressed as nodes in directed acyclic graphs).
- We now show how a **multivariate Gaussian distribution** can be expressed as a **directed graph** corresponding to a **linear Gaussian model**.
- Consider an arbitrary acyclic graph over D random variables, in which each node represent a single continuous Gaussian distribution with its mean given by the linear function of the parents:

$$p(x_i | \text{pa}_i) = \mathcal{N} \left(x_i \mid \sum_{j \in \text{pa}_i} w_{ij} x_j + b_i, v_i \right)$$

where w_{ij} and b_i are parameters governing the mean, and v_i is the variance.

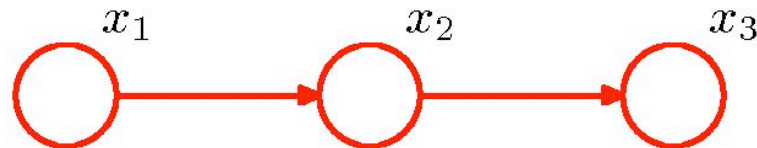
Linear Gaussian Models

- The log of the joint distribution takes form:

$$\ln p(\mathbf{x}) = \sum_{i=1}^D \ln p(x_i | \text{pa}_i) = - \sum_{i=1}^D \frac{1}{2v_i} \left(x_i - \sum_{j \in \text{pa}_i} w_{ij} x_j - b_i \right)^2 + \text{const},$$

where 'const' denotes terms independent of \mathbf{x} .

- This is a quadratic function of \mathbf{x} , and hence the joint distribution $p(\mathbf{x})$ is a **multivariate Gaussian**.
- For example, consider a directed graph over three Gaussian variables with one missing link:



Computing the Mean

- We can determine the mean and covariance of the joint distribution.

Remember:

$$p(x_i | \text{pa}_i) = \mathcal{N} \left(x_i \mid \sum_{j \in \text{pa}_i} w_{ij} x_j + b_i, v_i \right)$$

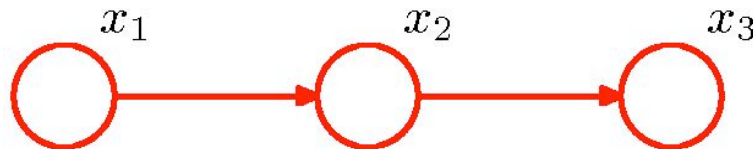
hence

$$x_i = \sum_{j \in \text{pa}_i} w_{ij} x_j + b_i + \sqrt{v_i} \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, 1),$$

so its expected value:

$$\mathbb{E}[x_i] = \sum_{j \in \text{pa}_i} w_{ij} \mathbb{E}[x_j] + b_i.$$

- Hence we can find components: $\mathbb{E}[\mathbf{x}] = [\mathbb{E}[x_1], \dots, \mathbb{E}[x_D]]$ by doing **ancestral pass**: start at the top and proceed in order (see example):



Computing the Covariance

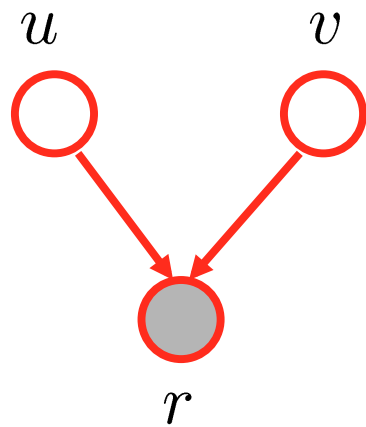
- We can obtain the i,j element of the covariance matrix in the form of a recursion relation:

$$\begin{aligned}\text{cov}[x_i, x_j] &= \mathbb{E} [(x_i - \mathbb{E}[x_i])(x_j - \mathbb{E}[x_j])] \\ &= \mathbb{E} \left[(x_i - \mathbb{E}[x_i]) \left(\sum_{k \in \text{pa}_j} w_{jk} (x_k - \mathbb{E}[x_k]) + \sqrt{v_j} \epsilon_j \right) \right] \\ &= \sum_{k \in \text{pa}_j} w_{jk} \text{cov}[x_i, x_k] + I_{ij} v_j.\end{aligned}$$

- Consider two cases:
 - There are no links in the graph (**graph is fully factorized**), so that w_{ij} 's are zero. In this case: $\mathbb{E}[\mathbf{x}] = [b_1, \dots, b_D]^T$, and the covariance is diagonal $\text{diag}(v_1, \dots, v_D)$. The joint distribution represents D independent univariate Gaussian distributions.
 - The graph is **fully connected**. The total number of parameters is $D + D(D-1)/2$. The covariance corresponds to a general symmetric covariance matrix.

Bilinear Gaussian Model

- Consider the following model:

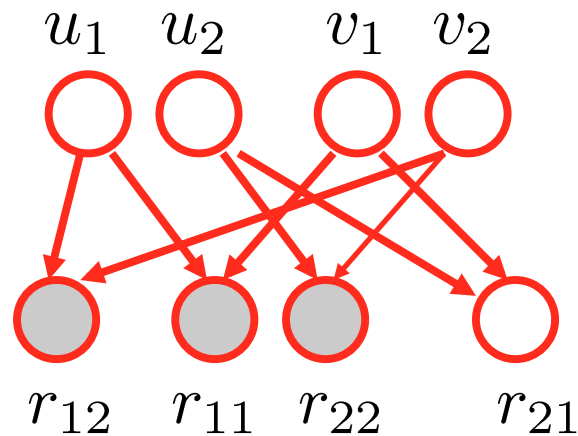


$$u \sim \mathcal{N}(0, 1),$$

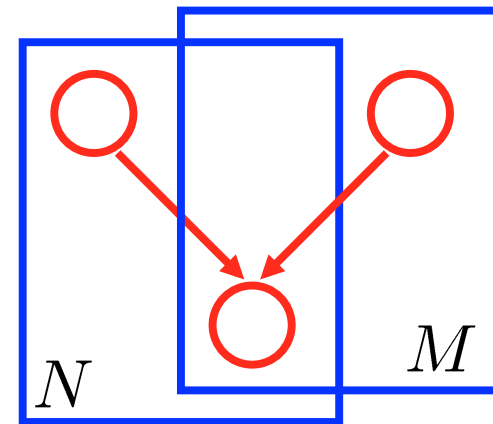
$$v \sim \mathcal{N}(0, 1),$$

$$r \sim \mathcal{N}(uv, 1).$$


 Gaussian terms



	★★☆	?	?	★★☆	★★☆
	?	★★☆	★★★★	?	★★★★
	★★★★	?	★★☆	★★★★	?



$$u_i \sim \mathcal{N}(0, 1), \quad i = 1, \dots, N$$

$$v_j \sim \mathcal{N}(0, 1), \quad j = 1, \dots, M$$

$$r_{ij} \sim \mathcal{N}(u_i v_j, 1).$$

- The mean is given by the product of two Gaussians.

Hierarchical Models

