Summary Neuroinformatics

Matrices

- Addition
  $\mathbf{A}+\mathbf{B}=\mathbf{C}$ with $\mathbf{A},\mathbf{B},\mathbf{C}\in\mathbb{R}^{n\times p}$
  $c_{i,j}=a_{i,j}+b_{i,j}$
- Subtraction
  $\mathbf{A}-\mathbf{B}=\mathbf{C}$ with $\mathbf{A},\mathbf{B},\mathbf{C}\in\mathbb{R}^{n\times p}$ $c_{i,j}=a_{i,j}-b_{i,j}$
- Scaling
  $\alpha\mathbf{A}=\mathbf{C}$ with $\mathbf{A},\mathbf{C}\in\mathbb{R}^{n\times p}$ and $\alpha\in\mathbb{R}$
  $c_{i,j}=\alpha a_{i,j}$
- Transposed Matrix
  $\mathbf{A}^T:(\mathbf{A}_{i,j})^T=\mathbf{A}_{j,i}$ with $(\mathbf{AB})^T=(\mathbf{B})^T(\mathbf{A})^T$
- Inverse Matrix
  $\mathbf{A}^{-1}\mathbf{A}=\mathbf{I}$
  - A -1 do exist iff det(A)≠0
    (for det(A)=0, the matrix A is singular and cannot be inverted)
  - only defined for NxN matrices

- Pseudo Inverse
  - has the properties:

$$AA^{\dagger}A=A$$
$$A^{\dagger}AA^{\dagger}=A^{\dagger}$$

  - For an invertible matrix, the pseudo inverse become equivalent to the inverse.
  - Can be computed using Single Value Dicomposition (SVD)

  - Moore-Penrose Pseudoinverse $\boldsymbol{\Phi}^{\dagger}\equiv\left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^{\mathrm{T}}$
    - also possible for NxM matrices

- Multiplication

$$\mathbf{A}\in\mathbb{R}^{n\times p}\text{ with }\mathbf{A}=\begin{bmatrix}\boxed{a_{1,1}\ \dots a_{1,p}}\\ \ddots & \\ a_{n,1}\ \dots a_{n,p}\end{bmatrix}_{n\times p}$$

$$\mathbf{B}\in\mathbb{R}^{p\times q}\text{ with }\mathbf{B}=\begin{bmatrix}\boxed{\begin{matrix}b_{1,1}\\ \ddots\\ b_{p,1}\end{matrix}} & \dots & \boxed{\begin{matrix}b_{1,q}\\ \\ b_{p,q}\end{matrix}}\end{bmatrix}_{p\times q}\qquad\begin{matrix}c_{11}\\ c_{1q}\end{matrix}$$

$$\mathbf{C}=\mathbf{A}\cdot\mathbf{B}:\ \text{with }c_{i,j}=\sum_{k=1}^{p}a_{i,k}b_{k,j}\text{ with }\mathbf{C}\in\mathbb{R}^{n\times q}$$

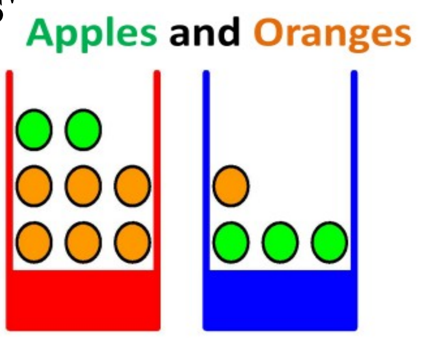
row times column vectors

- positive definite   If A is symmetric, and its quadratic forms are strictly positive, for non-zero x :
  $x^T Ax>0$ (Geometry: bowl-shaped minima at x = 0)

- linear combinations
  - linear independent: The vectors $a_1, a_2, \ldots, a_n \in R^n$ are linearly Independent if we cannot find a set of scalars c (other than all c=0) for that holds: $c_1\mathbf{a}_1 + c_2\mathbf{a}_2 +, \ldots, + c_m\mathbf{a}_m = 0$ with $c_i \in R$
  - linearly dependent: If there exist at least one set of scalars c then at least one vectors of the set can be expressed by the other.
  - Rank: maximal number of linearly independent columns/rows
    - full rank: matrix has a rank as large as possible
    - rank deficient: matrix that has not a full rank
- linear system of equations: $Ax=b$
- least squares solutions:
  - $A^T Ax = A^T b$ (only if column rank is full)
  - Now $(A^T A)$ is square, symmetric and invertible.
  - $x=(A^T A)-1 A^T b \ldots$ now solves the system of equations!
  - project b onto column space, then solve the equation.
- Pseudo inverse: $(A^T A)^{-1} A^T$ or $\Phi^\dagger \equiv (\Phi^T \Phi)^{-1} \Phi^T$

Probabilities and Bayes'

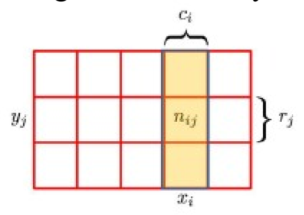
Classes x: Color: Red or Blue
Classes y: Fruit: Apple or Orange



2 x 2 Classes (color x fruit)

	Blue	red
Appels	3	2
Oranges	1	6

- Marginal Probability: Probability of X=x irrespective of class y



$$N = \sum_{i,j} n_{i,j} \quad c_i = \sum_j n_{i,j}$$

$$p(X = x_i) = \frac{c_i}{N}.$$

- Conditional Probability: Probability of Y=y given class X=x

$$p(Y = y_j | X = x_i) = \frac{n_{ij}}{c_i}$$

- Joint Probability: Probability of X=x and Y=y

$$p(X = x_i, Y = y_j) = \frac{n_{ij}}{N}$$

- Sum Rule:

Sum Rule

$$p(X = x_i) = \frac{c_i}{N} = \frac{1}{N}\sum_{j=1}^{L} n_{ij}$$

- Product Rule:

Product Rule

$$p(X = x_i, Y = y_j) = \frac{n_{ij}}{N} = \frac{n_{ij}}{c_i} \cdot \frac{c_i}{N}$$
$$= p(Y = y_j | X = x_i) p(X = x_i)$$

Joint probability

- Bayes' Theorem:

$$p(Y|X) = \frac{p(X|Y)p(Y)}{p(X)}$$

$$p(X) = \sum_{Y} p(X|Y)p(Y)$$

posterior ∝ likelihood × prior

$y = D$ (observed data)
$x = \vec{w}$ (set model parameter)

$$p(\vec{w}|D) = \frac{p(D|\vec{w})p(\vec{w})}{p(D)}$$

$$p(D) = \int P(D|\vec{w})p(\vec{w})d\vec{w}$$

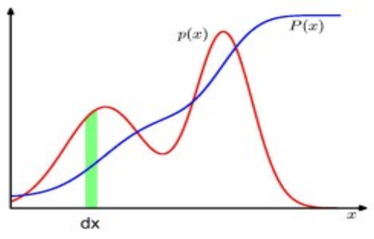
Normalisation

Posterior:
$P(\vec{\omega}|D)$ = probability of a model with set of parameters given the data (= D)

Likelihood:
$P(D|\vec{\omega})$ probability of the data given with set of parameters of a model =

$$p(\mathbf{x}|\mu, \sigma^2) = \prod_{n=1}^{N} \mathcal{N}(x_n|\mu, \sigma^2)$$

- Probability Densities



p(x) is a density with

  - Probability Density function (PDF): $p(x) \geqslant 0$

$$\int_{-\infty}^{\infty} p(x)\, \mathrm{d}x = 1$$

→ wahrscheinlichkeit für ein event zu einem bestimmten Zeitpunkt p(x)
→ summe aller dieser wahrscheinlichkeiten=1

  - Probability: $p(x \in (a, b)) = \int_a^b p(x)\, \mathrm{d}x$

→ wahrscheinlichkeit das ein event einsetzt zwischen zeipunkt a und b

  - Cumulative Density function (CDF): $P(z) = \int_{-\infty}^{z} p(x)\, \mathrm{d}x$

→ fläche bis zu einem zeitpunkt z

(wahrscheinlichkeit das bis zu diesem zeitpunkt ein Ereignis eingesetzt hat)

- Expectations

$$\mathbb{E}[f] = \sum_x p(x)f(x) \qquad \mathbb{E}[f] = \int p(x)f(x)\,\mathrm{d}x$$

  - Conditional Expectation (discrete)

$$\mathbb{E}_x[f|y] = \sum_x p(x|y)f(x)$$

  - Alternative notation:

$$E_x[f(x)] = \langle f(x)\rangle_x$$

- Variance

Variance describes the variability of a single variable

$$var(f) = \sigma^2(f) = \langle [f(x) - \langle f(x)\rangle]^2\rangle = \langle f(x)^2\rangle - \langle f(x)\rangle^2$$

- Covariance

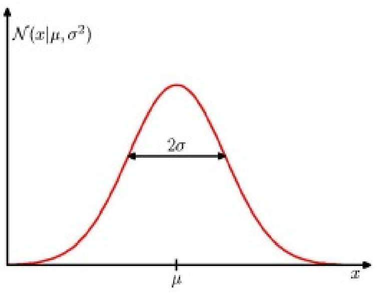
Covariance describes the variability across pairs of a set of variables.

$$\begin{aligned} cov(x,y) &= \langle [(x - \langle x\rangle)(y - \langle y\rangle)]\rangle_{x,y} \\ &= \langle xy\rangle_{x,y} - \langle x\rangle_x \langle y\rangle_y \end{aligned}$$

$$\begin{aligned} \mathrm{cov}[x,y] &= \mathbb{E}_{x,y}\left[\{x - \mathbb{E}[x]\}\{y - \mathbb{E}[y]\}\right] \\ &= \mathbb{E}_{x,y}[xy] - \mathbb{E}[x]\mathbb{E}[y] \end{aligned}$$

$$\begin{aligned} \mathrm{cov}[\mathbf{x},\mathbf{y}] &= \mathbb{E}_{\mathbf{x},\mathbf{y}}\left[\{\mathbf{x} - \mathbb{E}[\mathbf{x}]\}\{\mathbf{y}^{\mathrm{T}} - \mathbb{E}[\mathbf{y}^{\mathrm{T}}]\}\right] \\ &= \mathbb{E}_{\mathbf{x},\mathbf{y}}[\mathbf{x}\mathbf{y}^{\mathrm{T}}] - \mathbb{E}[\mathbf{x}]\mathbb{E}[\mathbf{y}^{\mathrm{T}}] \end{aligned}$$

- Gaussian Distribution



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

$$\mathcal{N}(x|\mu,\sigma^2) > 0 \qquad \int_{-\infty}^{\infty} \mathcal{N}\left(x|\mu,\sigma^2\right)\,\mathrm{d}x = 1$$

  - Gaussian Mean

$$\mathbb{E}[x] = \int_{-\infty}^{\infty} \mathcal{N}\left(x|\mu,\sigma^2\right) x\,\mathrm{d}x = \mu$$

$$\mathbb{E}[x^2] = \int_{-\infty}^{\infty} \mathcal{N}\left(x|\mu,\sigma^2\right) x^2\,\mathrm{d}x = \mu^2 + \sigma^2$$

  - Gaussian Variance

$$\mathrm{var}[x] = \mathbb{E}[x^2] - \mathbb{E}[x]^2 = \sigma^2$$

Model fitting

- Maximum Likelihood Approach
  - Select a model with parameter set w (i.e. gaussian ,model) $p(D|\vec{\omega})$
  - maximize the likelihood after w $(i.e.\ y \text{ and } \sigma)$ $\quad p(x|\mu,\sigma) \sim \prod_i \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$

- Maximum <u>Log</u> Likelihood Approach
  - maximize likelihood after w: $\quad p(D|\vec{w}) = \prod N(x_i|\mu,\sigma)$

    $\rightarrow$

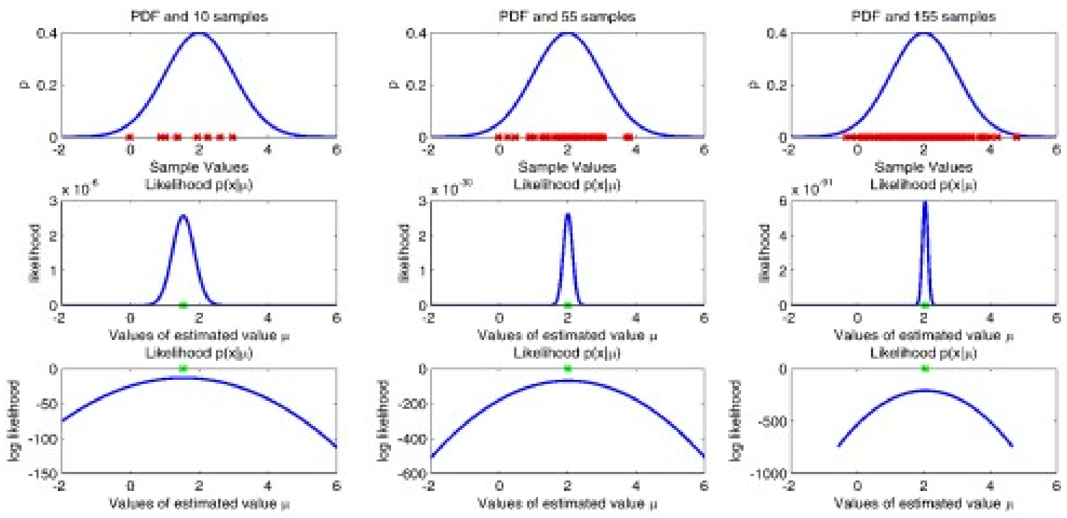
$$\frac{\partial}{\partial\mu}\log(p(D|\vec{w})) = \frac{\partial}{\partial\mu} K \sum_{i=1}^{N}\left[-\frac{(x_i-\mu)^2}{\sigma^2}\right] = 0$$

  - General for a Gaussian

$$\ln p\left(\mathbf{x}|\mu,\sigma^2\right) = -\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2 - \frac{N}{2}\ln\sigma^2 - \frac{N}{2}\ln(2\pi)$$

$$\mu_{\mathrm{ML}} = \frac{1}{N}\sum_{n=1}^{N} x_n \qquad\qquad \sigma^2_{\mathrm{ML}} = \frac{1}{N}\sum_{n=1}^{N}(x_n-\mu_{\mathrm{ML}})^2$$

$\rightarrow$ Maximum likelihood estimator of μ minimizes squared error between model and dataset



With increasing number of samples the likelihood function becomes tighter, and the curvature of the log likelihood increases

Bias

- assume that we have a certain Gaussian PDF with a given set of real parameters μ and σ
  → what happens with μ_{ML} when the number of independent samples increases, each containing K values: $$E(\mu_{ML})(K) = \mu$$
  → what happens with σ_{ML} when the number of independent samples increases, each containing K values: $$E(\sigma_{ML})(K) = \frac{K-1}{K} \cdot \sigma$$

- Bias:
  systematic difference between the expected value of an estimate and the real value
  $$Bias(\hat{\theta}) = E[\hat{\theta}] - \theta$$
  - the value of μ_{ML} of the real expected value of %my of an Gaussian distribution is unbiased: $Bias(\hat{\mu} = \mu_{ML}) = E[\mu_{ML}] - \mu$ $\quad \mathbb{E}[\mu_{\mathrm{ML}}] = \mu$
  - the value of σ_{ML} of the σ of an Gaussian distribution is biased:

$$\sigma^2_{\mathrm{ML}} = \frac{1}{N}\sum_{n=1}^{N}(x_n - \mu_{\mathrm{ML}})^2 \qquad Bias(\sigma^2_{ML}) = \frac{-\sigma^2}{K}$$

$$\mathbb{E}[\sigma^2_{\mathrm{ML}}] = \left(\frac{N-1}{N}\right)\sigma^2 \qquad \begin{aligned}\widetilde{\sigma}^2 &= \frac{N}{N-1}\sigma^2_{\mathrm{ML}} \\ &= \frac{1}{N-1}\sum_{n=1}^{N}(x_n - \mu_{\mathrm{ML}})^2\end{aligned}$$

Combining Likelihoods with Fischer Information

- identical distributed independent data sets $\vec{x}_i$:

Likelihoods: $p(\vec{x}_i|\vec{w}) = L_i(\vec{w})$

Likelihood: $L(\vec{w}) = \prod_i p(\vec{x}_i|\vec{w}) = \prod_i L_i(\vec{w})$

Log Likelihood: $\log L(\vec{w}) = \sum_i \log p(\vec{x}_i|\vec{w}) = \sum_i \log L_i(\vec{w})$

(important for successive updates of ML estimates

Fisher information

- measures the importance of a parameter for the model

Score function: $S(w) = \frac{\partial}{\partial w} logL(\vec{x}, w)$

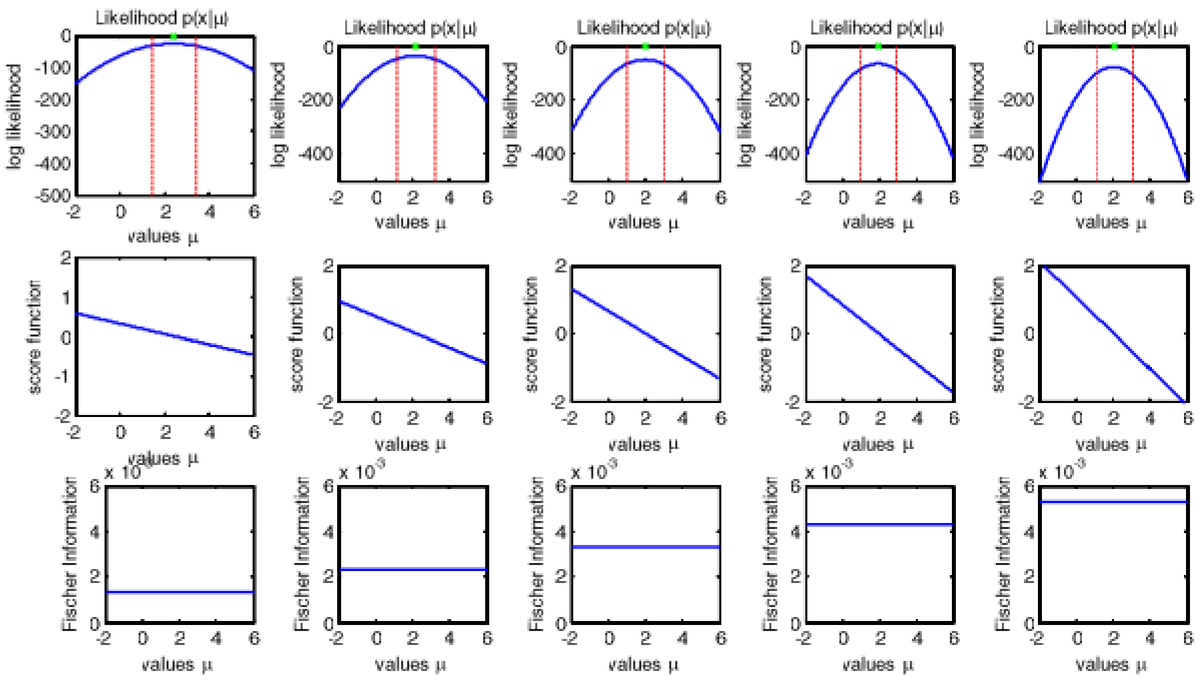
ML: $S(w_{ML}) = 0$

Fischer Information: $I(w) = -\frac{\partial^2}{\partial w^2} logL(\vec{x}, w)$

- Measures the curvature. For $w = w_{ML}$ it measures the curvature at the Maximum likelihood estimators of w . A tight and high peak indicates high sensitivity toward that parameter.

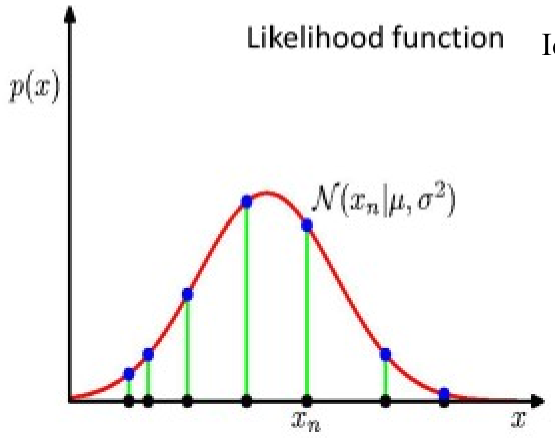
Therefore the Fischer information is large.

Increasing Fischer Information



- way of measuring the amount of information that an observable random variable X carries about an unknown parameter θ upon which the probability of X depends
- is high if the likelihood changes very strongly with small changes of the parameter θ
  → large Fischer information indicates a rather precise estimate of the parameter θ

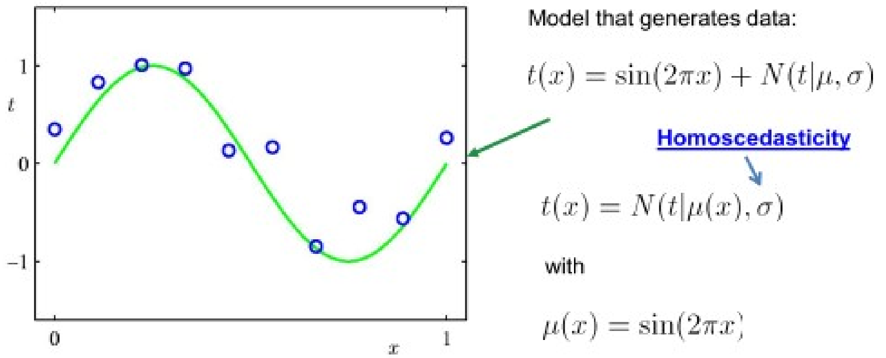
Regression



Idea: use maximum likelihood to estimate μ and σ from a data set x

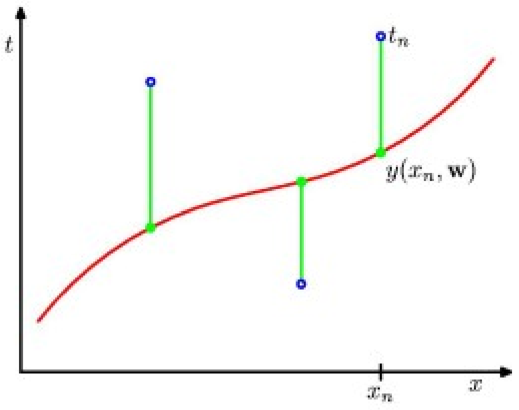
- Example: Polynomial Curve Fitting

  → use maximum likelihood to estimate μ(*x*) and σ from a data set ***x*** (dataset), we thereby assume an explicit dependence of μ on x



Goal of curve fitting:

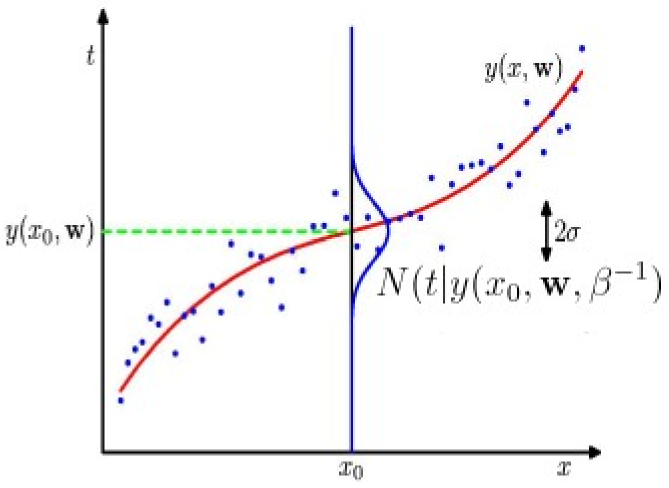
- to be able to make predictions for the target variable t given some new value of the input variable x on the basis of a set of training data comprising N input values $x=(x_1,\ldots,x_N)^T$ and their corresponding target values $t=(t_1,\ldots,t_N)^T$



to estimate $\mu_{ML}(x)=y(x,\vec{w})$
we define a flexible function $y(x,\vec{w})$
And adjust the parameters such that the likelihood for the data $t=(t_1,\ldots,t_N)^T$ sampled at the points $x=(x_1,\ldots,x_N)^T$ is maximized.

→ means minimizing the squared error (for gaussian error)

$$E(\mathbf{w})=\frac{1}{2}\sum_{n=1}^{N}\{y(x_n,\mathbf{w})-t_n\}^2$$



β is inverse variance of the Gaussian

with $y(x,w)$ being parameterized as a polynomial. This is a linear model of the parameters w

$$y(x,\mathbf{w})=w_0+w_1x+w_2x^2+\ldots+w_Mx^M=\sum_{j=0}^{M}w_jx^j$$

- Linear Regression: simplest linear model for regression is one that involves a linear combination of the input variables $y(x,w)=w_0+w_1x_1+\ldots+w_Dx_D$ where $x=(w_1,\ldots,x_D)^T$
  - Key property: is a linear function of the parameters $w_0,\ldots,w_D$
  - has much limitations (because its only fitted to the sample data x_i)

→ extend class of models by considering combinations of fixed nonlinear functions of the input variables, of the form

$$y(\mathbf{x},\mathbf{w})=w_0+\sum_{j=1}^{M-1}w_j\phi_j(\mathbf{x})$$

- $\Phi_0(x)$ are basis functions
- w_0 allows for any fixed offset in the data (sometimes called a bias parameter → not bias in the statistical sense)
- often convenient to define an additional dummy 'basis function' $\Phi_0(x)=1$ so that $y(x,w)=\sum_{j=0}^{M-1}w_j\Phi_j(x)=w^T\Phi(x)$

→ linear model (linear in w)
polynomials as linear model:

$$y(x,\mathbf{w})=w_0+w_1x+w_2x^2+\ldots+w_Mx^M=\sum_{j=0}^{M}w_jx^j$$

Log Likelihood function for Gaussian distribution

$$\ln p(\mathbf{t}|\mathbf{x},\mathbf{w},\beta) = \underbrace{-\frac{\beta}{2}\sum_{n=1}^{N}\{y(x_n,\mathbf{w})-t_n\}^2}_{\beta E(\mathbf{w})} + \frac{N}{2}\ln\beta - \frac{N}{2}\ln(2\pi)$$

Determine $\mathbf{w}_{\mathrm{ML}}$ by minimizing sum-of-squares error, $E(\mathbf{w})$.

$$\Downarrow$$

Score function

$$\nabla LL = \nabla ln(p(\mathbf{t}|\mathbf{x},\mathbf{w},\beta)) = -\nabla\frac{\beta}{2}\sum_{n=1}^{N}(t_n - y(x_n,\mathbf{w}))^2$$

- determine ML estimate by setting score function to zero

$$0 = \sum_{n=1}^{N} t_n \boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}} - \mathbf{w}^{\mathrm{T}}\left(\sum_{n=1}^{N}\boldsymbol{\phi}(\mathbf{x}_n)\boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}}\right)$$

→ gives us $\mathbf{w}_{\mathrm{ML}} = \left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{t}$

→ design Matrix

$$\boldsymbol{\Phi} = \begin{pmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \cdots & \phi_{M-1}(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \cdots & \phi_{M-1}(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \cdots & \phi_{M-1}(\mathbf{x}_N) \end{pmatrix}$$

- M basis functions, N Data points x_i

Results in: $\mathbf{w}_{\mathrm{ML}} = \left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{t}$

determine β_{ML} by minimizing the remaining terms and using w_{ML}

$$\frac{1}{\beta_{\mathrm{ML}}} = \frac{1}{N}\sum_{n=1}^{N}\{y(x_n,\mathbf{w}_{\mathrm{ML}})-t_n\}^2$$

How we determined w_{ML} in the homework:

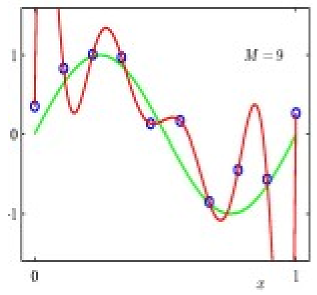
- determine the design matrix with the x values filled in every row (if we had n x_i values, we had a $m \times n$ matrix where m was the order of the polynomial function we used.
- In every row of the design matrix the x values were potentiated to the power of m
- then we got w_{ML} by fitting the design matrix with the y sample. Glmfit() gave us a potentially right model for w_{ML} with which the sample matrix(design matrix) could be multiplied to get a good guess for our model

Regression in short:

- We assume we have a Gaussian distribution with an expected value being a function of x
$$t(x)=N(t|\mu(x),\sigma^2)$$
- We assume Homoscedasticity (Varians is identical for all x)
- We model the dependence of μ(x) by using a linear combination of fixed nonlinear basis functions φj(x) , of the form
$$y(x,w)=\sum_{j=0}^{M-1} w_j \Phi_j(x)=w^T \Phi(x)$$
- We find the parameters w_j by using ML estimate that is the minimum squared error. To find this ML we use the Penrose pseudo inverse.

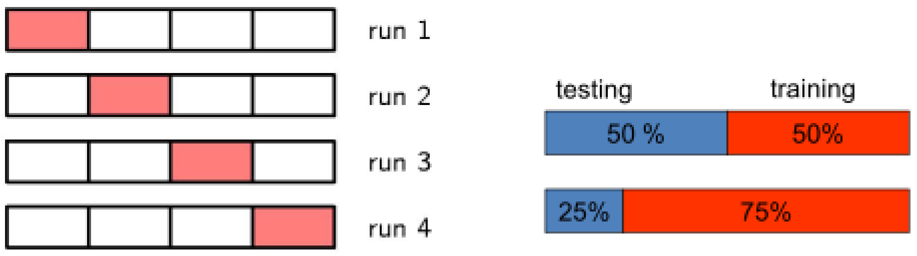
Cross Validation

- to avoid Overfitting: the model we approximated fits every sample value of the training data but at the same time the estimation for the real values of the curve are not appropriate



$\rightarrow$ reduced by. Increasing number of samples used to generate the approximation

- Cross validation: partition your data into k chunks of testing and training data $\rightarrow$ K-fold Cross Validation



- Goal: achieve good generalization by making accurate predictions for new data
- Idea:
  - 1. Use the training data to estimate Parameters use ML, i.e. Maximize Likelihood:
$$p(\vec{t}^{train}|\vec{x}^{train},\vec{w},\beta)=\prod_{n=1}^{N} N(t_n^{train}|y(x_n^{train},\vec{w}),\beta^{-1})$$
  -
  - 2. Use the ML Parameters to derive likelihood on test data Test data Likelihood:
$$p(\vec{t}^{test}|\vec{x}^{test},\vec{w}_{ML},\beta_{ML})=\prod_{n=1}^{N} N(t_n^{test}|y(x_n^{test},\vec{w}_{ML}),\beta_{ML}^{-1})$$

Root-Mean-Square Error (RMS): $$E_{RMS}=\sqrt{\frac{2E(w^*)}{N}}$$

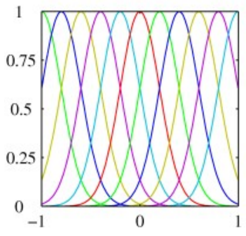
- measures the error, that is the difference of the real curve and our approximated model

Squared Error : $E(w)=\frac{1}{2}\sum_{n=1}^{N}\{y(x_n,w)-t_n\}^2$

Splines

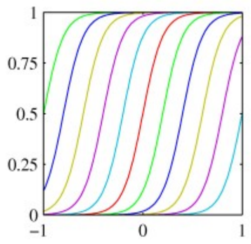
→ Regression with Basis functions

- Why? : Because before, when we defined the model as a combination of basis functions, all basis functions were global → a (small) change of x affects all basis functions
- gobal basis functions: non zero for the entire range of $x \neq 0$
- Gaussian basis functions: $\phi_j(x) = \exp\left\{-\frac{(x-\mu_j)^2}{2s^2}\right\}$



    - are relatively local (small change in x only effects nearby basis functions)
    - Easy to use since μ_j and s control location and scale (width)

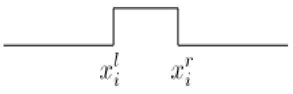
- Sigmoidal basis functions $\phi_j(x) = \sigma\left(\frac{x-\mu_j}{s}\right)$ with $\sigma(a) = \frac{1}{1+\exp(-a)}$



    - relatively local
    - Easy to use since μ_j and s control location and scale (width)

- Bin based basis functions

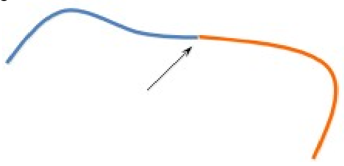
$$\phi_i = 1 \text{ for } x > x_i^l \;\&\; x \leq x_i^r \text{ otherwise } \phi_i = 0$$



    - strictly functions that are local step functions (non zero for the range between two bounds)
    - each function is not smooth
    - linear combination cannot be smooth

Splines

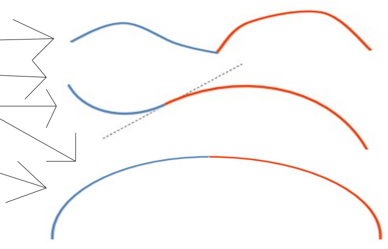
- piecewise defined polynomial functions ← Splines
- we need:
    - Order k (highest order of the polynomial pieces),
    - Vector of breakpoints (= **knots**)
    - Vector with # continuity conditions that have to be met at each breakpoint
- Idea:
    - Divide the interval of approximation (of some function) into subintervals with breakpoints $\xi_0 \leq \xi_1 \leq \ldots \leq \xi_{n+k}$ .On each subinterval i, a polynomial P_i of order k is defined.
    - Use different polynomial functions for different paths of the curve



- Advantage:
  - Flexibility
  - Locality

- Issue:
  - Smoothness at “joints” (continuity)
  - Continuity C^k indicates adjacent curves have the same k-th derivative at their joints

- Adjacent curves: Same endpoints $P_i(1)=P_{i+1}(0)$
  AND Same first derivatives $P_i'(1)=P_{i+1}'(0)$
  AND Same second derivatives $P_i''(1)=P_{i+1}''(0)$

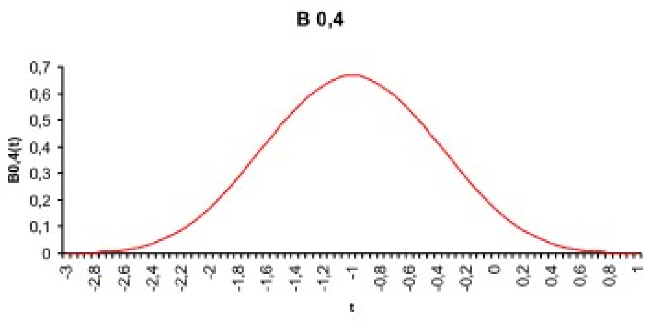


- Discontinuous curves sometimes called C^{-1}

- Cubic Spline S(x):
  - most popular choice with k=4

$$\textbf{for } \xi_i \le x \le \xi_{i+1}: \qquad S(x) = P_i(x) = a_0 + a_1x + a_2x^2 + a_3x^3$$

  - all polynomial pieces have to agree at the inner breakpoints up to the second derivative → C^2 Continuity
  - Problems:
    - assume a knot vector with n+1 breakpoints
    - for a cubic spline with p-pieces of order 4: 4n parameters need defining → solve a system with 4n equations
    - equations arise as a result of the continuity conditions, but in general, the problem is underspecified
      → small tricks and algorithms to construct cubic splines
- Breakpoints (Knots)
  - Introducing a breakpoint means we allow p_{i+1} to differ from p_i
    → needs degree of freedom
    → causes loss of continuity
    → p_{i+1} and p_i agree in one less derivative
  - Introducing a breakpoint creates on additional B-Spline
    → additional model coefficient (degree of freedom)
  - Introducing the same breakpoint m times into the knot sequence causes m losses of continuity



Exemplary Cubic Spline
with breakpoints/knots (-3,-2,-1,0,1)

At the inner breakpoints (-2,-1,0), all polynomial pieces agree up to their second derivative.

$$B_{0,4}(t) = \frac{1}{6}\begin{cases} (t+3)^3 & -3 \le t < -2 \\ -3t^3 - 15t^2 - 21t - 5 & -2 \le t < -1 \\ 3t^3 + 3t^2 - 3t + 1 & -1 \le t < 0 \\ (1-t)^3 & 0 \le t < 1 \end{cases}$$

Splines + Regression as we knew it:

- Spline functions S(x) of order k, with knot vector t, and $0 \leq v_i \leq k$ continuity conditions at breakpoint t_i can be expressed in a form S(x) = y(x,w)

$$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) = \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x})$$

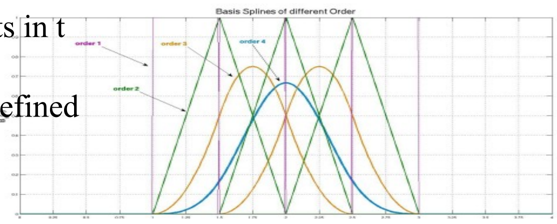
$\rightarrow \Phi_j(x) = B_j(x)$ = Basis Splines (B-Splines)

B-Splines

- basis for a vector space of functions $\$_{k,t,v}$
- Each element of this vector space is a spline S(x) as defined above
- All splines can therefore be expressed as linear combinations in terms of the B-Splines $B_j(x)$
- B-Spline Construction → Cox-de Boor algorithm:
    - recurrence relation starts with the 1st order B-splines, just boxes, and builds up successively higher orders

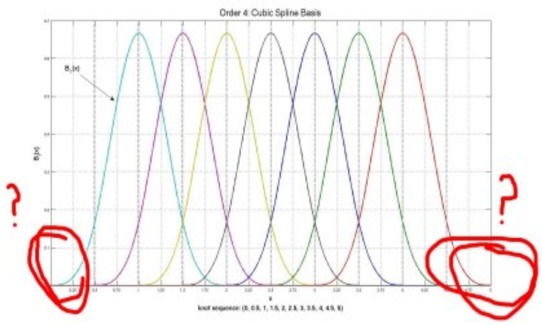
$$B_{k,1}(t) = \begin{cases} 1 & t_k \leq t \leq t_{k+1} \\ 0 & \text{otherwise} \end{cases} \qquad B_{k,d}(t) = \left(\frac{t - t_k}{t_{k+d-1} - t_k} \right) B_{k,d-1}(t) + \left(\frac{t_{k+d} - t}{t_{k+d} - t_{k+1}} \right) B_{k+1,d-1}(t)$$

- are defined solely in terms of lower order B-Splines and knots in t
    - Let n be the order of your spline.
    - At each knot in t (except the last n knots), a B-Spline is defined
    - knot sequence does not need to be uniform

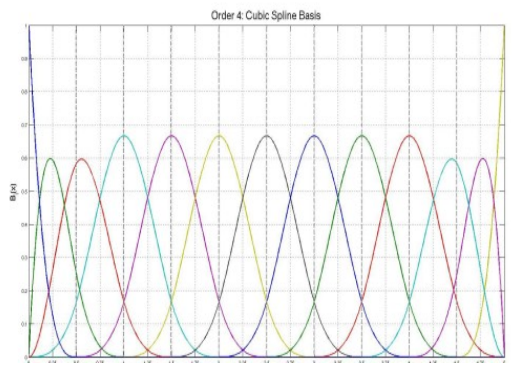


- Extended knot sequence:
    - so we don't get 0 at the ends of our interval
      → You need to extend the knot sequence for the spline to be determined on the whole interval
    - Spline of order n: The two endpoints have to occur n times in the knot sequence.
    - **k = knot multiplicity t + # continuity conditions at t** (k= length of knot-vector)

$t = (0,0,0,0,\ 0.5,\ 1,\ 1.5,\ 2,\ 2.5,\ 3,\ 3.5,\ 4,\ 4.5,\ 5,5,5,5)$

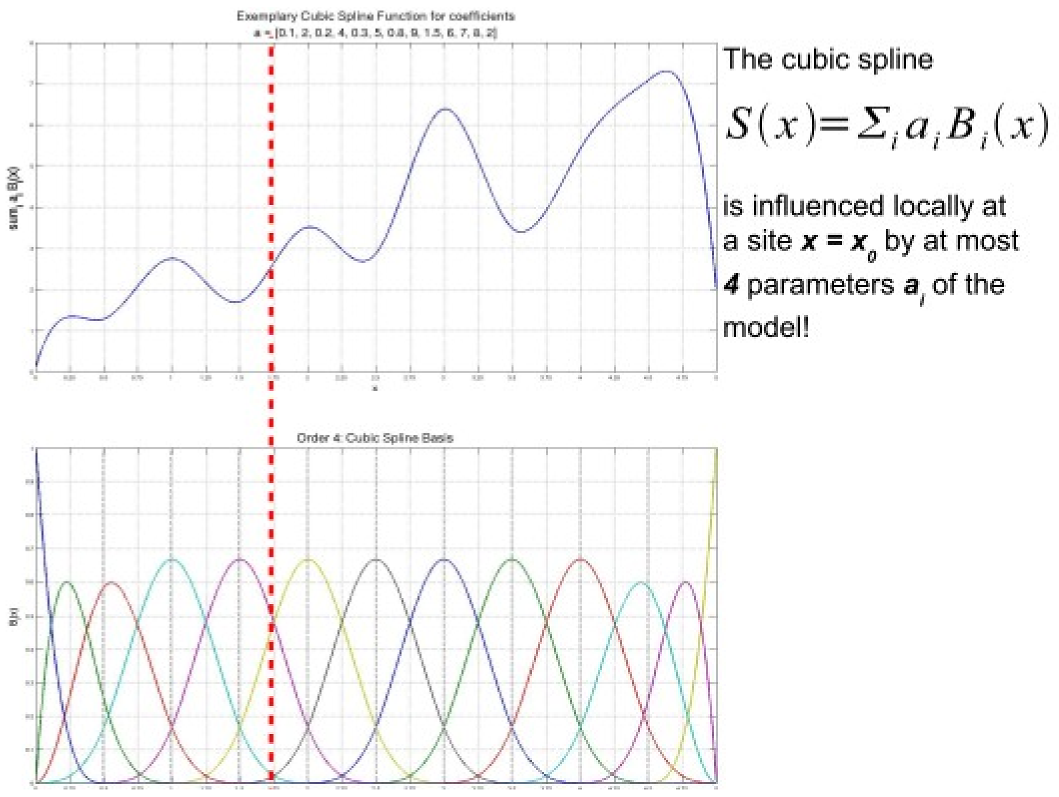


→ → →



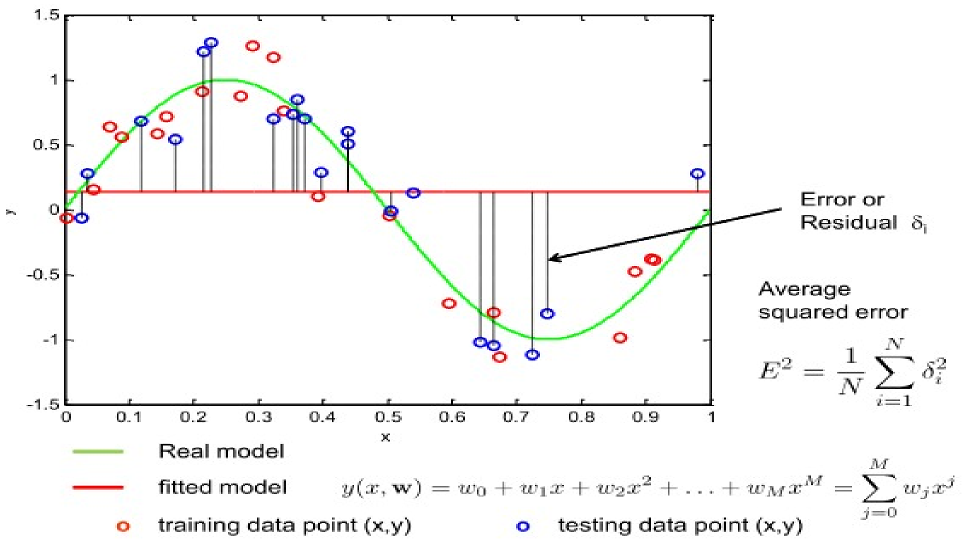
Example for a spline of order k with knot vector t:

- Each B-Spline is defined on an interval of k+1 successive knots
- The i-th B-Spline of order k, $B_{i,k}(x)$, is zero outside of $t_i \leq x \leq t_{i+k}$



Facts about polynomial interpolation

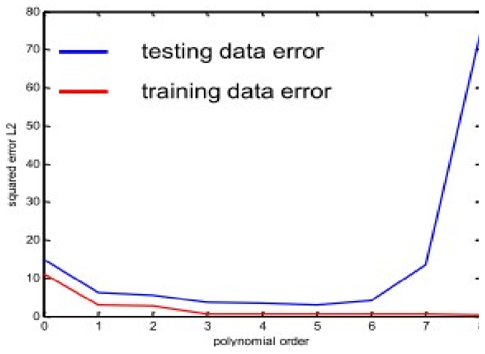
- A polynomial that interpolates a function g at k points $t_{1,\ldots}, t_k$ is uniquely determined and of order k
- If the polynomial agrees k-fold (in k derivatives) with g at some t_i , then its parameters (leading coefficients in Newton Form) can be written in terms of the derivatives



Model Selection

Squared Error

- The squared error between the fitted model and the training data decreases with increasing order of the polynomial
- The shape of the curves as well as the optimal complexity of the model is stochastic and depends on the actual training data set and testing data set



→ Model Selection approach

- best with N-Fold cross validation
  → n- chunks each used as test and training data with n runs
  → every run compute the error rate
  → FINALLY: take the model with the best performance (e.g. with the lowest average error rate over all runs)

Likelihood Ratio

- to compare the likelihood of different parameter values within the same model and one dataset we use likelihood ratio :

  Likelihoods: $L(\vec{x}, w_1)$ and $L(\vec{x}, w_2)$

  Use Likelihood ratio c: $\frac{L(\vec{x}, w_1)}{L(\vec{x}, w_2)} = c$

- since likelihood is not a probability (→ it is not normalized), it is defined it is only meaningful up to a multiplicative factor

  $$p(\vec{x}|w) = c \cdot L(\vec{x}, w) \text{ with } c > 0$$

  → therefore we use the ratio of two likelihoods → which is invariant of a multiplicative factor: $L_1' = c \cdot L_1$ and $L_2' = c \cdot L_2$ we get $\frac{L_1}{L_2} = \frac{L_1'}{L_2'}$

- if we scaled the Likelihood to a certain value (i.e. the maximum of the likelihood scaled to 1) we can compare the likelihoods from different parameters using this scale

Nested Models

- A model is nested within another nesting model if it is a special case of that nesting model

  The nesting model is the most complex model:

  $$y_i = \alpha x_i + \beta x_i + \gamma x_i + \epsilon$$

  Nested models are:

  $$y_i = \alpha x_i + \epsilon \ , \ y_i = \beta x_i + \epsilon \ , \ y_i = \gamma x_i + \epsilon$$

  $$y_i = \alpha x_i + \beta x_i + \epsilon \ , \ y_i = \beta x_i + \gamma x_i + \epsilon \ , \ y_i = \alpha x_i + \gamma x_i + \epsilon$$

- Nested models are all models with k < N: $$y_i = \sum_{j=0}^{\kappa} \alpha_j x_i^j + \epsilon$$

- Likelihood ration test (for nested models)
  Normally:

  Likelihoods: $L(\vec{x}, w)$ and $L(\vec{x}, w_n)$

  $$w = [w_1, w_2, \dots w_k] \text{ and e.g. } w_n = [w_1' = 0, w_2, \dots w_k]$$

  Likelihood ratio c: $$\frac{L(\vec{x}, w)}{L(\vec{x}, w_n)} = c \text{ with } \log\left(\frac{L(\vec{x}, w)}{L(\vec{x}, w_0)}\right) = \log(\vec{x}, w) - \log(\vec{x}, w_n)$$

  $\rightarrow$ $c \geq 0$ (always) for nested models on the same data

nested models:

- compare likelihood of different parameter values within the same model and one dataset:

  Likelihoods: $L(\vec{x}, w)$ and $L(\vec{x}, w_0)$

  Use Likelihood ratio c: $$\frac{L(\vec{x}, w_0)}{L(\vec{x}, w)} = c \text{ with } \log\frac{L(\vec{x}, w_0)}{L(\vec{x}, w)} \sim \chi_1^2$$

- A X^2 variable with k degrees of freedom is defined as the sum of the squares of k independent standard normal random variables

  $$\chi^2 = \sum_{i=i}^{k} y_i^2 \text{ with } p(y_i) = N(\mu = 0, \sigma = 1)$$

- Properties of nested models:
  - a nested model can be derived from the most complex nesting model by zeroing the coefficients of a subset of model parameters.
  - We can compare Likelihoods between the nesting and nested models, given the same data set
  - One way to compare likelihood is the likelihood ratio
  - The likelihood ratio, is X^2 distributed with number of degrees of freedom equal to the difference in parameters between models A and B
  - For comparison of nested models is approximately X^2 regardless whether the individual deviances for each the nested and nesting model are X^2 distributed or not
  - The degree of freedom of the X^2 distribution is equal to the number of zeroed coefficients

Deviance

- deviance of generalized linear models (GLM)
- The analysis of deviance is a generalization of the of the classical analysis of variance (R^2)
- Deviance and analysis of variance can be used as a measure of fit or lack of fit
- measure of distance between a particular model and the saturated model
- **deviance**
  - for theoretical Model $D(x,w)$
- **observed deviance**
  - for measured data and estimated parameters $D(x,\hat{w})$

Likelihood $L(\vec{x},w)$

- Likelihood of the observed data x given the set of parameters w

Likelihood of a saturated model $L(\vec{x},w_{sat})=L(\vec{x},\vec{x})$

- Likelihood of the observed data x given a model that used x a parameters

Deviance:

$$D(x,w)=2\log\left(\frac{L(\vec{x},x)}{L(\vec{x},w)}\right)\sim X^2$$

- use Likelihood ratio

EXAMPLE: How to use Deviance with nested models:

Likelihood:

$$L(\vec{x},w) \text{ and } L(\vec{x},w_n)$$

$$H1: w=[w_1,w_2,w_3] \quad H0: w_n=[w_1,w_2=0,w_3=0]$$

$$y_i^{H1}=w_1f_1(x_i)+w_2f_2(x_i)+w_3f_3(x_i)+\epsilon \quad y_i^{H0}=w_1f_1(x_i)+\epsilon$$

$$\text{Deviance: } D(w,w_n)=2\log\left(\frac{L(\vec{x},w)}{L(\vec{x},w_n)}\right)\sim\chi^2_{d=2}$$

For D> critical value based on chi^2 indicates H1 is the better model !

Single Parameter or Univariate testing

Suppose we have a univariate model.

- When we want to test the single parameter w against a null model w_0, we can use the Log likelihood ratio test, and the Fischer Information.
- The importance of one parameter for the model is described by the Fischer Information

Suppose we have a multivariate model.

- When we want to test the single parameter of the model, say w against a null model w_0, we can use the Log likelihood ratio test, and the Fischer Information.
- The importance of each individual parameter for the model is described by the individual Fischer Information
- In case one want to compare to entire models one need to use the Log Likelihood ratio test

Bias Variance decomposition

- we decompose the Error into three terms: bias (mismatch of model and real data), variance and noise term
- Error: $$L2 = \frac{1}{N}\sum_{n=1}^{N}(t_n - y_n)^2$$
- Computation of the decomposition terms:

We have a real data model (function): $f(x) \in \mathbb{R}$

Samples (t_n,x_n): $t_n = f(x_n) + \epsilon$ Linear model: $y(x, \vec{w})$

Squared error between samples and the model (Loss function)

$$L2 = \frac{1}{N}\sum_{n=1}^{N}(t_n - y_n)^2 \qquad E[L2] = \frac{1}{N}\sum_{n=1}^{N}E[(t_n - y_n)^2]$$

→ we are interested in the relation of the error to the real function f(x)
→ therefore we add a zero and expand the squared term

Result:

$$E[L2] = \frac{1}{N}\sum_{n=1}^{N}(E[\epsilon^2] + E[(f_n - E[y_n])^2] + E[(E[y_n] - y_n)^2]$$

expected error = noise + bias^2 + variance of the model

Noise:	That is the noise of the data. Note that the noise in the data equals a lower bound of the Expected value of the L2. With other words, a perfect model has a minimal finite Error.
Bias:	Systematic mismatch between the expected (average model for large number of data sets) and the real function f.
Variance:	Variability of the model across different data sets.

- Properties of
  - Noise term:
    - is independent of the model and is there a lower bound of the expected overall error if the model (????? HÄ ?????)
  - Bias^2:
    - gets smaller for more complex models
      → reflects the flexibility of a model to adapt to a certain real function
  - Variance:
    - increases if we increase the complexity of the model
      → fitting the same model to different independent data sets gives a larger variance across the fits if the model is complex

→ illustrates the trade off for a good model
  - A optimal model uses a optimal complexity and thereby keeping both the bias or the variance small.
  - A model that over fits, is very complex and has a very small bias. At the same time the variance is large since the model adapts to the noise structure as well.
  - A model that is to simple, has very small variance, but the model has a large bias as well.

Akaike Information Criterion (AIC)

- relates likelihood to K-L distance and includes an explicit term for model complexity

$$AIC = -2 \log L(Data|\hat{w}) + 2K$$

→ is an estimate of the expected, relative distance between the fitted model and the unknown true function that generated the observed data .
→ K = number of estimated parameters

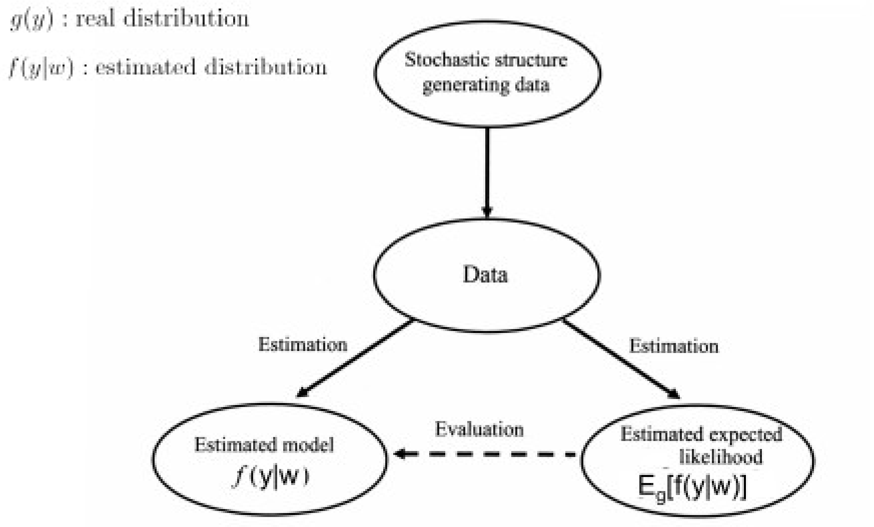

$g(y)$: real distribution
$f(y|w)$: estimated distribution
Stochastic structure generating data
Data
Estimation
Estimation
Estimated model $f(y|w)$
Evaluation
Estimated expected likelihood $E_g[f(y|w)]$

Fig. 3.5. Use of data in the estimations of the parameter of a model and of the expected log-likelihood.

Kullback-Leibler Distance

- given real distribution f(y) and model g with parameters w

$$\begin{aligned} KL : I(f,g) &= \int f(y) \log\left(\frac{f(y)}{g(y|w)}\right) dy \\ &= \int f(y) \log(f(y))\, dy - \int f(y) \log(g(y|w))\, dy \\ &= E_f[\log f(y)] - E_f[\log g(y|w)] \\ &= const. - E_f[\log g(y|w)] \end{aligned}$$

→ impossible to derive for a general case (f is mostly unknown)
→ possible with AIC: the expected value of KL can be derived if w is estimated by a maximum likelihood estimator hat (w)

Aikaike showed: we can derive an asymptotically unbiased estimator of the mean value of KL, given the number of parameters K

$$\hat{I}(f, g(\hat{w})) = \log L(Data|\hat{w}) - K$$

AIC

- we select the model with smallest value of AIC
- AIC will identify the best model in the set, even if all models are poor
- Unless the sample size (n) is large with respect to the number of estimated parameters (K), use of AICc is recommended

$$\begin{aligned} AIC_c &= -2\ln L(\hat{w}, x) + 2K\left(\frac{n}{n-K-1}\right) \\ &= -2\ln L(\hat{w}, x) + 2K + \frac{2K(K+1)}{n-K-1} \end{aligned}$$

→ Generally, you should use AICc when the ratio of n/K is small (less than ~ 40), based on K from the global (most complicated) model.
→ Use AIC or AICc consistently in an analysis rather than mix the two criteria

- Rules for using the AIC
  - Differences in AIC (Δi's) can be used to interpret strength of evidence for one model vs. another.
  - AIC can be used on other models than nested models
  - AIC can be used to compare different assumption about the underlying distribution, but using the same model
  - LRTs and AIC can be used as one basis for selecting the "best" PDF for a given dataset and model
  - But more generally, an examination of the distribution of the residuals should guide the choice of the appropriate PDF
  - There will be cases where different PDFs are appropriate for different models applied to the same dataset

Model Selection

- All model selection methods are based on the assumption that the likelihood function is correct. This may well not be the case.
- Check the residuals about the fits to the data for all models – it may be that none of the models are fitting the data. Model selection makes little sense if none of the models fit the data.
- Always plot the fits of the different models. Even if one model is significantly better than another, the improved fit may be qualitatively "insubstantial".

Model selection with AIC

Multinomial Distribution

- generalization of the binomial distribution
- binomial distribution is the probability distribution of the number of "successes" in n independent Bernoulli trials, with the same probability of "success" on each trial
- In a multinomial distribution, the analog of the Bernoulli distribution is the categorical distribution, where each trial results in exactly one of some fixed finite number k of possible outcomes, with probabilities

$$p_1, \ldots, p_k \text{ (with } p_i \geq 0 \text{ for } i = 1,\ldots, k \text{ and)}, \quad \sum_{i=1}^{k} p_i = 1$$

- The random variables X i indicates the number of times outcome number i was observed over the n trials.
- The vector $X=(X_1, \ldots, X_k)$ follows a multinomial distribution with parameters n and p, where $p=(p_1, \ldots, p_k)$.

Regularization

- Regularized Least Squares $E_D(\mathbf{w}) + \lambda E_W(\mathbf{w})$

  Data term + Regularization term

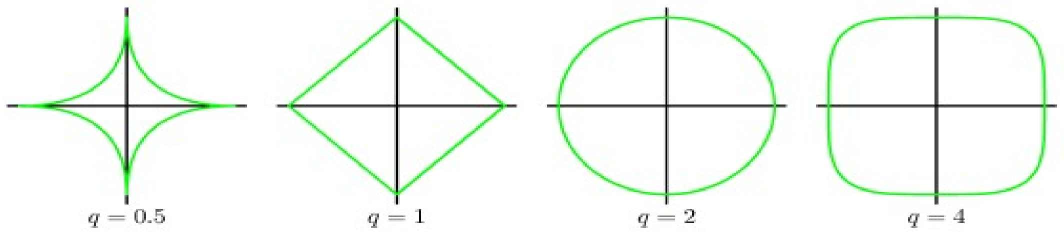
  → with the sum-of-squares error function and a quadratic regularizer, we get

$$\frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\phi(\mathbf{x}_n)\}^2 + \frac{\lambda}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}$$

, is called the regularization coefficient.

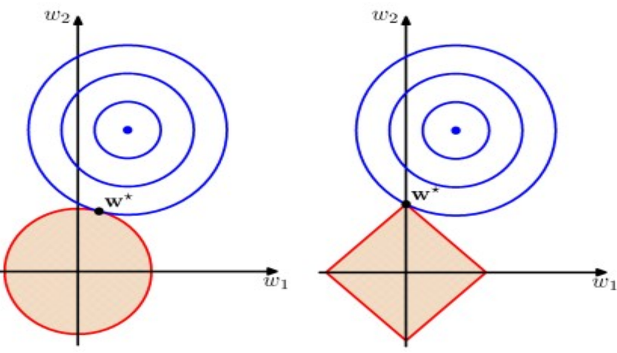
  → with varying degree of regularization the average of our model gets nearer to the real curve

- more general regularizer: $\frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\phi(\mathbf{x}_n)\}^2 + \frac{\lambda}{2}\sum_{j=1}^{M}|w_j|^q$



Lasso Quadratic

$\rightarrow \quad w_1 = w_2 + c \qquad c = (w_1^2 + w_2^2)$



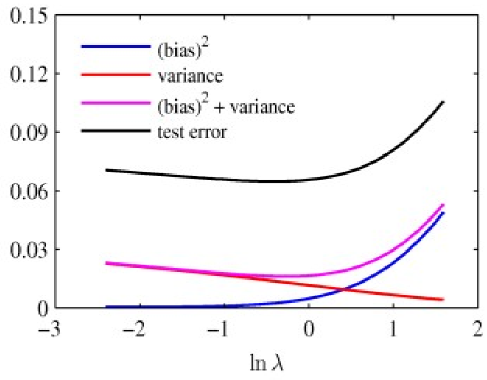
- Regularized Least Squares
  → Lasso tends to generate sparser solutions than a quadratic regularizer
  - overall cost function is the sum of two parabolic bowls
  - sum is also a parabolic bowl
  - the combined minimum lies on the line between the minimum of the squared error and the origin
  - the regularizer just shrinks the weights
  - penalizing the absolute values of the weights

$$\widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(\mathbf{x}_n, \mathbf{w}) - t_n\}^2 + \lambda \sum_i |\mathbf{w}_i|$$

  - minimum is unique because the cost function is convex (a bowl plus an inverted pyramid)
  - As lambda is increased, many of the weights go to exactly zero

Bias-Variance Trade-off

- an over-regularized model (large λ) will have a high bias
- an under-regularized model (small λ) will have a high variance



Maximum Likelihood for nongaussian distributions

- Binomial Distribution $\quad \mathrm{Bin}(m|N,\mu) = \binom{N}{m}\mu^m(1-\mu)^{N-m}$

→ $p(x=1|\mu)=\mu$
→ one coin is flipping, outcome can be heads or tails, we want the probability of this one flip and its outcome (1 or 0)

- Bernoulli Distribution

$$\begin{aligned}\mathrm{Bern}(x|\mu) &= \mu^x(1-\mu)^{1-x}\\ \mathbb{E}[x] &= \mu\\ \mathrm{var}[x] &= \mu(1-\mu)\end{aligned}$$

→ $p(m \ \ heads|N,\mu)$
→ N coin flips, we want the probability that at flip number m the outcome is heads
→ can also be written as $\quad p(x|\eta) = \sigma(-\eta)\exp(\eta x)$
where

- is a member of the exponential family

$$\begin{aligned}u(x) &= x\\ h(x) &= 1\\ g(\eta) &= 1-\sigma(\eta) = \sigma(-\eta)\end{aligned}$$

ML for Bernoulli

Given: $\mathcal{D} = \{x_1, \ldots, x_N\}$, m heads (1), $N-m$ tails (0)

likelihood:
$$p(\mathcal{D}|\mu) = \prod_{n=1}^{N} p(x_n|\mu) = \prod_{n=1}^{N} \mu^{x_n}(1-\mu)^{1-x_n}$$

$$\ln p(\mathcal{D}|\mu) = \sum_{n=1}^{N} \ln p(x_n|\mu) = \sum_{n=1}^{N} \{x_n \ln \mu + (1-x_n)\ln(1-\mu)\}$$

Maximizing likelihood after μ
$$\mu_{\mathrm{ML}} = \frac{1}{N}\sum_{n=1}^{N} x_n = \frac{m}{N}$$

→ Bernoulli can be written as:
$$\begin{aligned} p(x|\mu) &= \mathrm{Bern}(x|\mu) = \mu^x(1-\mu)^{1-x} \\ &= \exp\{x\ln\mu + (1-x)\ln(1-\mu)\} \\ &= (1-\mu)\exp\left\{\ln\left(\frac{\mu}{1-\mu}\right)x\right\} \end{aligned}$$

→ $\eta = \ln\left(\frac{\mu}{1-\mu}\right)$ and so $\mu = \underbrace{\sigma(\eta) = \frac{1}{1+\exp(-\eta)}}_{\text{Logistic sigmoid}}$.

See further Bayes' rules For two classes

→ thus bernoulli can be written as: $p(x|\eta) = \sigma(-\eta)\exp(\eta x)$
where $u(x) = x$
$h(x) = 1$
$g(\eta) = 1 - \sigma(\eta) = \sigma(-\eta)$

→ we get
$$\nabla g(\boldsymbol{\eta}) \underbrace{\int h(\mathbf{x})\exp\{\boldsymbol{\eta}^{\mathrm{T}}\mathbf{u}(\mathbf{x})\}\,\mathrm{d}\mathbf{x}}_{1/g(\boldsymbol{\eta})} + g(\boldsymbol{\eta})\underbrace{\int h(\mathbf{x})\exp\{\boldsymbol{\eta}^{\mathrm{T}}\mathbf{u}(\mathbf{x})\}\,\mathbf{u}(\mathbf{x})\,\mathrm{d}\mathbf{x}}_{\mathbb{E}[\mathbf{u}(\mathbf{x})]} = 0$$

→ and thus
$$-\nabla \ln g(\boldsymbol{\eta}) = \mathbb{E}[\mathbf{u}(\mathbf{x})]$$

→ Now likelihood function is given:
$$p(\mathbf{X}|\boldsymbol{\eta}) = \left(\prod_{n=1}^{N} h(\mathbf{x}_n)\right) g(\boldsymbol{\eta})^N \exp\left\{\boldsymbol{\eta}^{\mathrm{T}}\sum_{n=1}^{N}\mathbf{u}(\mathbf{x}_n)\right\}.$$

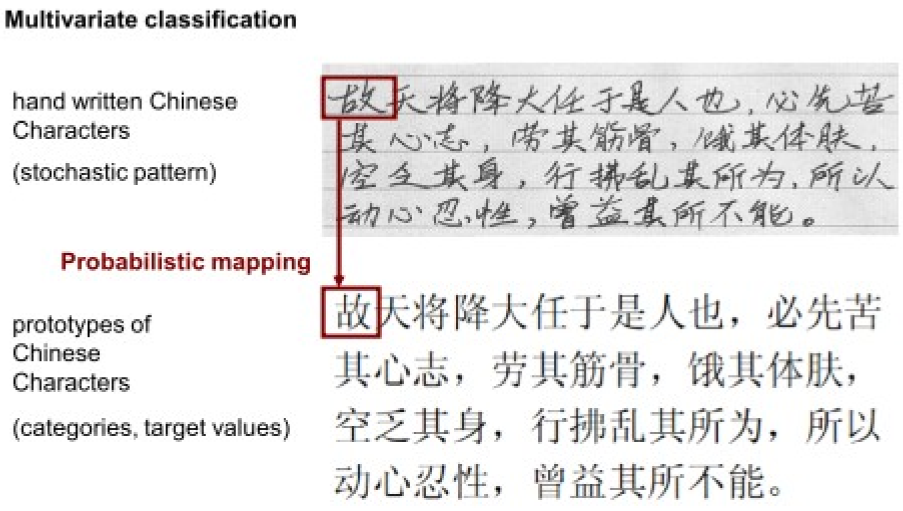
Thus we have
$$-\nabla \ln g(\boldsymbol{\eta}_{\mathrm{ML}}) = \frac{1}{N}\underbrace{\sum_{n=1}^{N}\mathbf{u}(\mathbf{x}_n)}_{\text{Sufficient statistic}}$$

Natural parameter η

$$p(\mathbf{x}|\boldsymbol{\eta}) = h(\mathbf{x})g(\boldsymbol{\eta})\exp\left\{\boldsymbol{\eta}^{\mathrm{T}}\mathbf{u}(\mathbf{x})\right\}$$

so $g(\eta)$ can be interpreted as a normalization coefficient $g(\boldsymbol{\eta})\int h(\mathbf{x})\exp\left\{\boldsymbol{\eta}^{\mathrm{T}}\mathbf{u}(\mathbf{x})\right\}\,\mathrm{d}\mathbf{x} = 1$

Multivariate Pattern Recognition


Multivariate classification

hand written Chinese Characters

(stochastic pattern)

故天将降大任于是人也，必先苦其心志，劳其筋骨，饿其体肤，空乏其身，行拂乱其所为，所以动心忍性，曾益其所不能。

Probabilistic mapping

prototypes of Chinese Characters

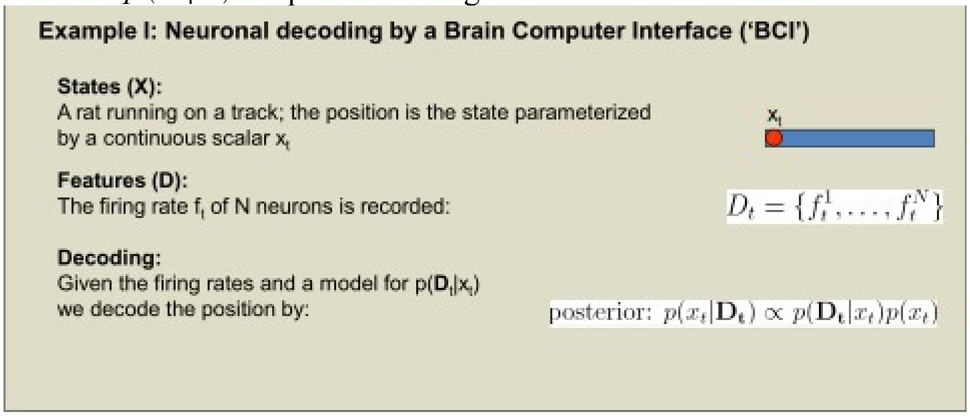
(categories, target values)

故天将降大任于是人也，必先苦其心志，劳其筋骨，饿其体肤，空乏其身，行拂乱其所为，所以动心忍性，曾益其所不能。

Supervised learning	Unsupervised learning
→ Target values are known	→ Target values are unknown
Making predictions from data based on a model • Regrassion: ○ target values are continuous (forecasts) • Classification: ○ target values are discrete categories (character recognition)	Discover structure in data • Clustering: ○ discover groups of similar examples within data • Dimensionality reduction ○ project data rom a high to a low dimensional space

Bayesian regression and classification

- Regression: Suppose we record data that composes a finite set of features (D) given a finite set of continuous states (X). We than use the likelihood $p(D|X)$ to derive the posterior $p(X|D)$ to predict states given the data


Example I: Neuronal decoding by a Brain Computer Interface ('BCI')

States (X):
A rat running on a track; the position is the state parameterized by a continuous scalar x_t

Features (D):
The firing rate f_t of N neurons is recorded: $D_t = \{f_t^1, \ldots, f_t^N\}$

Decoding:
Given the firing rates and a model for $p(\mathbf{D}_t|x_t)$ we decode the position by: posterior: $p(x_t|\mathbf{D}_t) \propto p(\mathbf{D}_t|x_t)p(x_t)$

- generative models: model the joint distribution of states (X) and features (D)
  - by sampling from them it is possible to generate synthetic data points in the input space

  - Generative models separate the inference stage form the decision stage:
    1. Inference stage:
       We model the likelihood p(D|X) and prior p(X)
    2. Decision stage:
       We use the likelihood and prior to derive the posterior to make a decision $p(X|D) \propto p(D|X)\,p(X)$
  - Classification with Generative Models
    - Inference and decision stage are separated
    - Requires the modeling of the likelihood p(D|C k) and prior p(C k)
      Modeling the likelihood and prior can be based on, e.g :
      - parametric modeling using maximum likelihood → requires assumption of the distribution of the class conditional densities $p(D|C_k)$
      - non-parametric density estimation
    - Can be used for any arbitrary likelihood $p(D|C_k)$ and prior $p(C_k)$
    - Insides regarding the underlying stochastic system

→ Alternative: Direct model of the posterior $p(C_k|D)$ (→ Logistic regression)

Bayes' rules for two classes (states): C_1, C_2

with $a = -\ln \dfrac{p(x|C_2)\,p(C_2)}{p(x ivides C_1)\,p(C_1)}$ we get: $p(C_1|x) = \dfrac{1}{1+\exp(-a)} = \sigma(a)$

Logistic sigmoid function
→ a equals the natural parame
Of a generalized linear mode

Modeling class conditional densities $p(x|C_k)$
- class conditional densities are Gaussian and share the same covariance matrix (Σ)

Given Gaussian class conditional densities and two classes

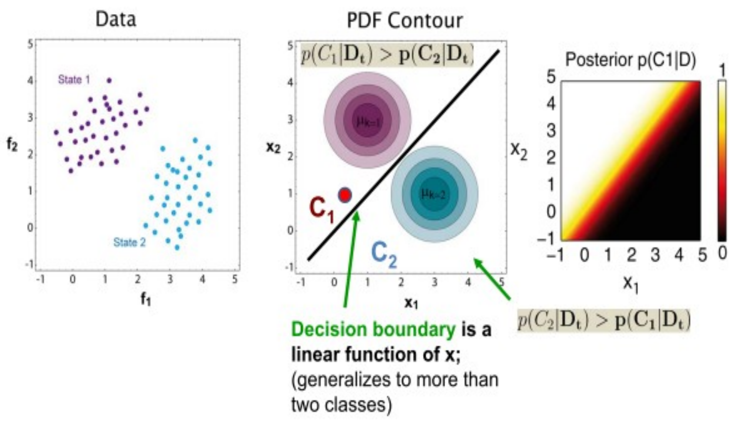
We get: $p(C_1|\mathbf{x}) = \sigma(a)$ with $a = \mathbf{w}^T\mathbf{x} + w_{(}$

with: $\mathbf{w}^T = \Sigma^{-1}(\mu_1 - \mu_2)$

$$w_0 = -\frac{1}{2}\left(\mu_1^T\Sigma^{-1}\mu_1 - \mu_2^T\Sigma^{-1}\mu_2 + \frac{\ln p(C_1)}{\ln p(C_2)}\right)$$

Smallest misclassification rate for $p(C_k|\mathbf{x}) = const.$
(not shown here, see module 2)

$p(C_1|\mathbf{x}) = \sigma(\mathbf{w}^T\mathbf{x} + w_0) = const. = 0.5$

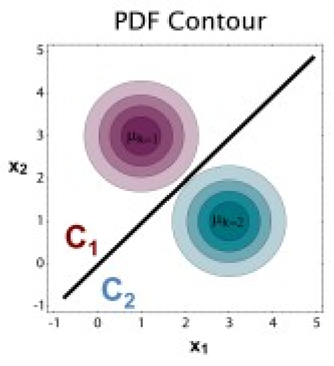


Classification of Boundaries:

Gaussian densities with identical covariance matrix:

$$x^T\Sigma_2^{-1}x - x^T\Sigma_1^{-1}x = 0$$

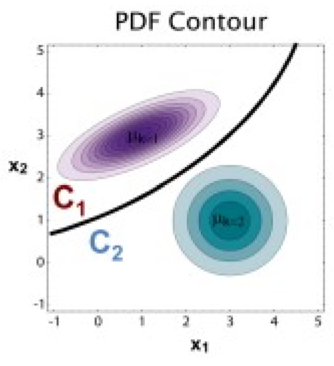
→ linear classification boundary

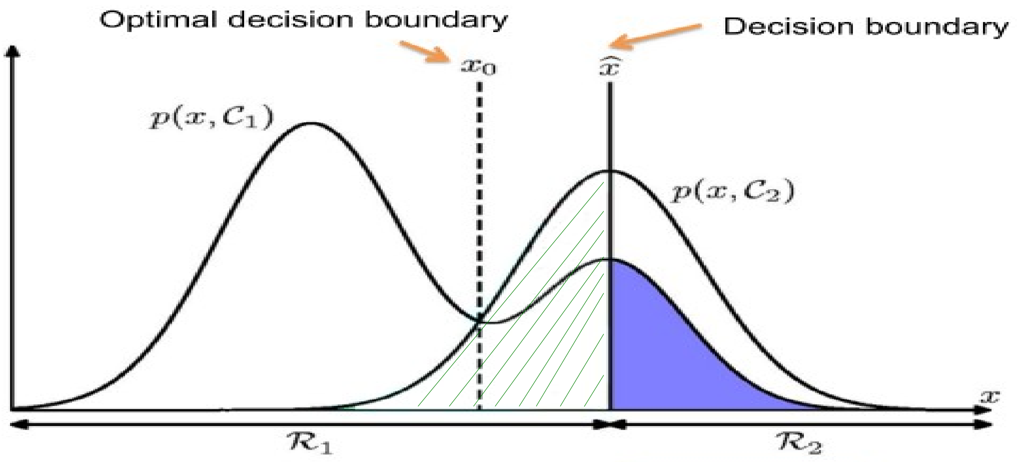


Gaussian densities with different covariance matrices:

$$x^T\Sigma_2^{-1}x - x^T\Sigma_1^{-1}x \neq 0$$

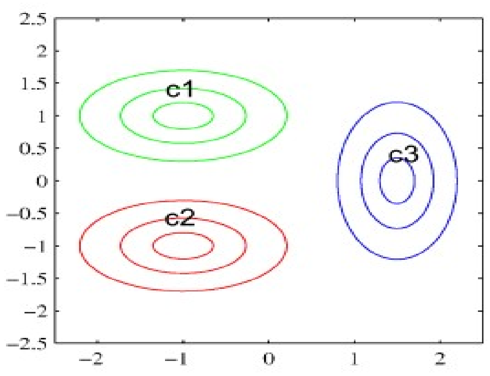
→ quadratic classification boundary

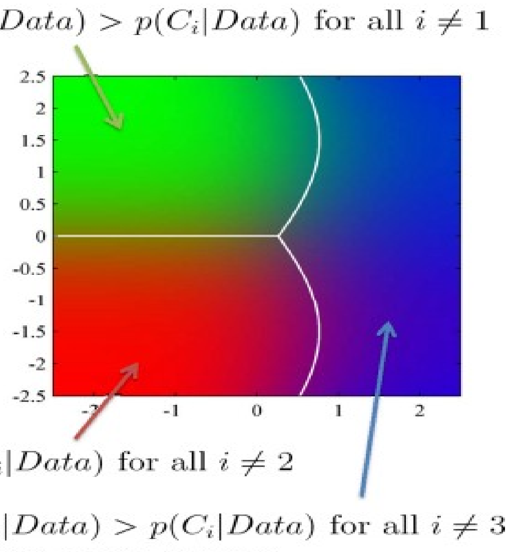




$$\begin{aligned} p(\text{mistake}) &= \boxed{p(\mathbf{x} \in \mathcal{R}_1, \mathcal{C}_2)} + \boxed{p(\mathbf{x} \in \mathcal{R}_2, \mathcal{C}_1)} \\ &= \int_{\mathcal{R}_1} p(\mathbf{x}, \mathcal{C}_2)\,\mathrm{d}\mathbf{x} + \int_{\mathcal{R}_2} p(\mathbf{x}, \mathcal{C}_1)\,\mathrm{d}\mathbf{x}. \end{aligned}$$

Classification for more than two classes:





Summary:

- Assuming a multinomial class conditional distribution of the features, and a sharing of the same covariance, we have seen that we get linear functions for the decision boundaries between classes.

$$p(C_1|\mathbf{x}) = \sigma(\mathbf{w}^T\mathbf{x} + w_0) = const. \qquad \text{with: } \sigma(a) = \frac{1}{1+\exp(-a)}$$

  We identify this as a logistic regression that is a special case of the Generalized Linear Model (GLM), allowing for an efficient modeling of the decision boundaries because:

  1. Iterative reweighed least squares is an efficient algorithm to solve the GLM
  2. Direct modeling of the posterior often requires less model parameters

  Suppose we have an M-dimensional feature space and 2 classes:

  - Number of parameters of the Generative Gaussian model :
    2M for means + M(M+1)/2 for shared covariance + prior = M(M+5)/2 +1 parameters

  - Number of parameters of the logistic regression :
    M+1 for posterior density = M +1 parameters

Support Vector Machines

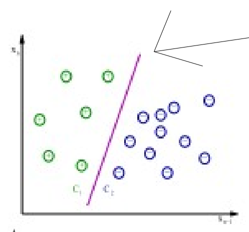
- Motivation for Support Vector Machines:
  - The risk of misclassification is minimized for maximal margins
  - support vectors: training patterns defining the margin
- Definition: Support Vector Machine
  → Construction of separating hyperplane with maximal margin
- Notation: **Training Set** $\mathcal{D} = \{(\boldsymbol{x}_1, d_1), (\boldsymbol{x}_2, d_2), ..., (\boldsymbol{x}_N, d_N)\}$
  with training patterns $\boldsymbol{x}_i \in \mathbb{R}^n$ and
  class labels (targets) $d_i \in \{+1, -1\}$

  Hyperplane $H : \boldsymbol{w}^T\boldsymbol{x} + w_0 = 0$
  with weight vector $\boldsymbol{w}^T$ and bias weight w_0

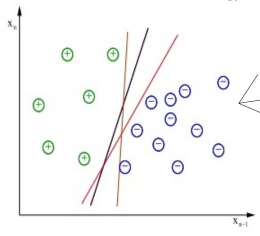
  Margin The margin is denoted by $\rho \in \mathbb{R}^+$

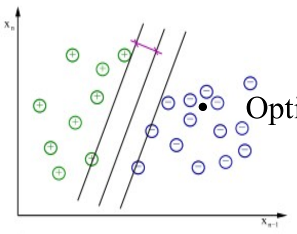
Maximum Margin Hyperplanes

- Linear Classification with Hyperplanes
  - Hyperplanes: Linear decision boundaries within $\mathbb{R}^n$
    A hyperplane $H \subset R^n$ is defined by weights $w = (w_1, \ldots, w_n)^T$ and a bias weight w_0 such that $x \in H \Leftrightarrow w^T x + w_0 = 0$



  - are equivalent to linear functions $H : \mathbb{R}^{n-1} \to \mathbb{R}$
  - Classification Task: Find separating hyperplane H with weights w and bias w_0 such that positive/negative patterns are correctly classified
  - if more than one hyperplane separates the training set, we find the one that has the best performance on a test set distinct from the training set





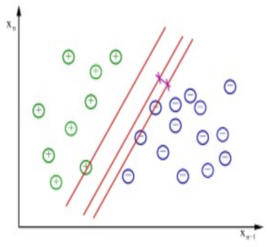


Figure: An intermediate margin

Optimal Seperating Hyperplanes

- the Margin: The margin $\rho \in \mathbb{R}$ with respect to a hyperplane H and a set of labeled training patterns D is given by the minimal distance of H to a training pattern $x \in D$
- Pattern Distance
  - Since $H: w_0 + w^T x = 0$, the vector w is orthogonal to H
  - The distance from x^+ to H is given by $\left|\frac{w_0 + w^T x^+}{\|w\|}\right|$
  - This measure holds both for positive and negative patterns

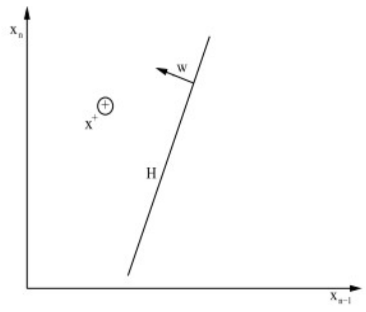


Figure: Hyperplane Parametrization

- Maximal Margin Hyperplanes

$$\max_{\boldsymbol{w}, w_0} \quad \rho^2$$
$$\text{subject to} \quad \rho > 0$$
$$d_i \cdot \frac{\boldsymbol{w}^T \boldsymbol{x} + w_o}{||\boldsymbol{w}||} \geq \rho \quad \forall (\boldsymbol{x}_i, d_i) \in \mathcal{D}$$

Completing The Problem Specification

The optimization contains a degree of freedom. The weights w and the bias w_0 can be scaled by a positive number without changing the hyperplane H. We remove this degree of freedom by an additional constraint: $\|w\| = \frac{1}{\rho}$

Replacing $\rho = \frac{1}{||\boldsymbol{w}||}$ yields

$$\max_{\boldsymbol{w}, w_0} \quad \frac{1}{||\boldsymbol{w}||^2}$$
$$\text{subject to} \quad \frac{1}{||\boldsymbol{w}||} > 0$$
$$d_i \cdot (\boldsymbol{w}^T \boldsymbol{x} + w_o) \geq 1 \quad \forall (\boldsymbol{x}_i, d_i) \in \mathcal{D}$$

Changing into an minimization problem and removing unnecessary constraints yields

$$\min_{\boldsymbol{w}, w_0} \quad \frac{1}{2}||\boldsymbol{w}||^2$$
$$\text{subject to} \quad d_i \cdot (\boldsymbol{w}^T \boldsymbol{x} + w_o) \geq 1 \quad \forall (\boldsymbol{x}_i, d_i) \in \mathcal{D}$$

$\Uparrow$

Final Mathematical Form of Optimization Problem

$\Uparrow$

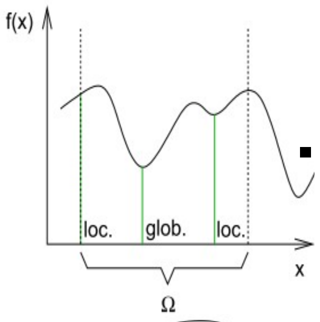
The solution of this optimization problem (if exists) constitutes the optimal separating hyperplane with maximal margin known as the Support Vector Machine (SVM)

Optimization Theory

- Convex Sets and Functions
  Optimization under Constraints
  - Definition → Optimization Problem
    Let $f:\mathbb{R}^n \to \mathbb{R}$ be a target function and let $\Omega \subseteq \mathbb{R}^n$ be a set of feasible points. A general optimization problem under constraints takes the form

    $$\min_{\boldsymbol{x}} \quad f(\boldsymbol{x})$$
    $$\text{subject to} \quad \boldsymbol{x} \in \Omega$$

    - The set of feasible points Ω is mostly given by inequality (equality) constraints $f(x) \leq g_i(x)$ with $g_i:\mathbb{R}^n \to \mathbb{R}$
    - Maximization problems can be described in the same way, since it holds that $max\, f(x) \Leftrightarrow min - f(x)$
    - For computing SVMs, we will concentrate on convex, differentiable target functions and linear constraints



- Global Minimum
  A feasible point x ∈ Ω is called a global minimum, if $\forall\ y \in\ \Omega: f(x) \leq f(y)$

- Local Minimum
  A feasible point x ∈ Ω is called a local minimum, if it exists $\epsilon \in 0\, (\epsilon \in \mathbb{R}^+)$ such that $\forall\ y \in \Omega: \|x - y\| < \epsilon \Rightarrow f(x) \leq f(y)$

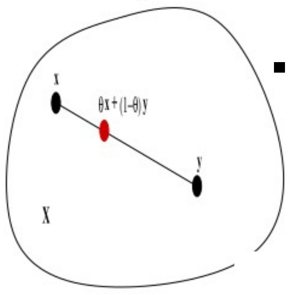


Figure: Convex Set

- Convex Set
  A set $X \subseteq \mathbb{R}^n$ is called convex, if for all points x ∈ X, y ∈ X and 0 ≤ θ ≤ 1, it holds that $\theta x + (1-\theta) y \in X$
  → A set is convex, if a straight line connecting any two points from the set is also part of the set

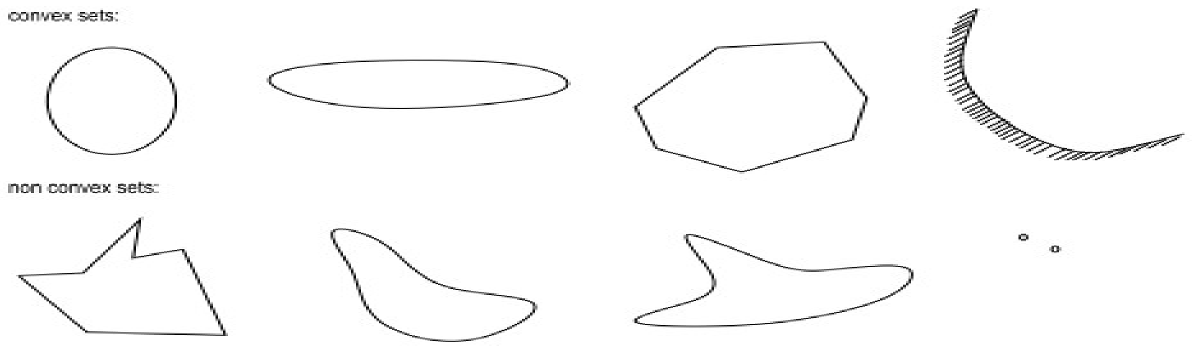


Figure: Examples for convex/non-convex sets

- Properties of Convex sets

Lemma (Intersections of Convex Sets)

Let $X \subseteq \mathbb{R}^n$ and $Y \subseteq \mathbb{R}^n$ be convex sets. Then, $X \cap Y$ is also a convex set.

Proof.

Let $\boldsymbol{x}, \boldsymbol{y} \in X \cap Y$. Since $\boldsymbol{x} \in X$, $\boldsymbol{y} \in X$ and X is a convex set, it holds that $\theta\boldsymbol{x} + (1-\theta)\boldsymbol{y} \in X$. A similar argument shows $\theta\boldsymbol{x} + (1-\theta)\boldsymbol{y} \in Y$. Thus, its holds that $\theta\boldsymbol{x} + (1-\theta)\boldsymbol{y} \in X \cap Y$. □

Unions of Convex Sets

Unions of convex sets are not necessarily convex!

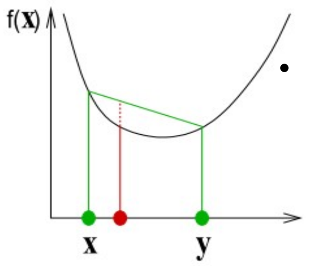


Figure: Convex Function

- Convex Functions
  A function $f:\mathbb{R}^n \to \mathbb{R}$ is called convex, if for all points $x \in \mathbb{R}^n, y \in \mathbb{R}^n$ and $0 \leq \theta \leq 1$, it holds that
  $$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta) f(y)$$
  - Properties of Convex functions
    - Linear Combinations of Convex Functions
      Let $f:\mathbb{R}^n \to R$ and $g:\mathbb{R}^n \to \mathbb{R}$ be convex functions and let a, b $\in$ R .
      $\to$ Then, (af + bg) is also a convex function.
    - Let $f:\mathbb{R}^n \to \mathbb{R}$ be a convex function. Then, each local minimum of f is also a global minimum.
    - Let $f:\mathbb{R}^n \to \mathbb{R}$ be a convex function. Then, each point $x \in \mathbb{R}^n$ satisfying $\forall\, i(1 \leq i \leq n): \partial \frac{f}{\partial} x_i(x) = 0$ is a local minimum of f .
    - The function f : $x \to x^2$ is convex.
    - The function f : x $\to$ $\|x\|^2$ is convex.

  - Constraints
    are often given by a set of inequality constraints $g_{1,} \dots, g_m$ such that the set of feasible points Ω is represented as $\Omega = \bigcap_{i=1}^{k} \{x | g_i(x) \leq 0\}$

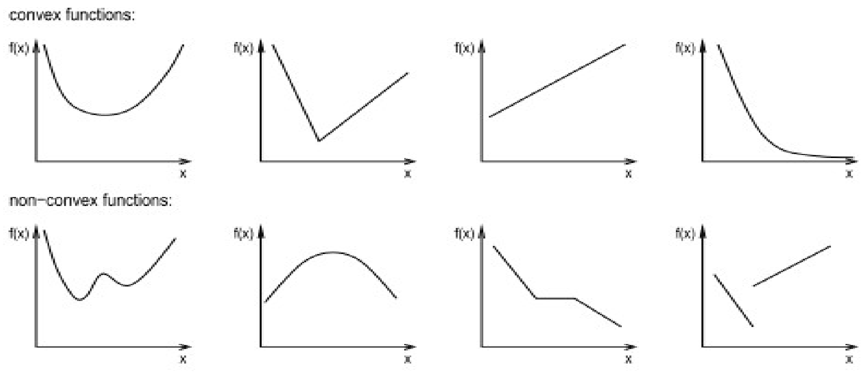
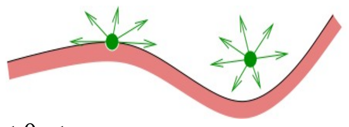


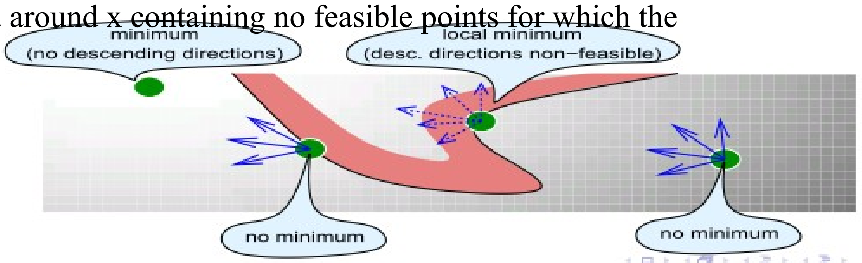
Figure: Examples for convex/non-convex functions

- Graphical Solution of Convex Optimization Problems
  Characterizing Local Minima:



  - For a feasible point x ∈ Ω we call a direction $v \in \mathbb{R}^n$ a feasible direction if exists r > 0 such that x + θv is a feasible point for all 0 ≤ θ ≤ r
  - For a feasible point x ∈ Ω we call a direction $v \in \mathbb{R}^n$ a descending direction if exists r > 0 such that f (x + θv) < f (x) for all 0 < θ ≤ r

  - A local minimum of f with respect to Ω is a feasible point for which the intersection of feasible and descending directions is empty.
    There is no feasible and descending direction in x
    ⇔ There is a small area around x containing no feasible points for which the value of f is smaller
    ⇔ Local minimum in x

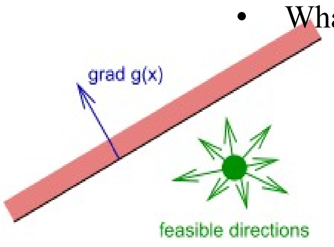


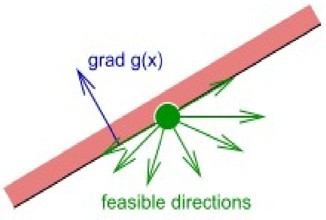
- What are the descending directions at x?
  - Gradient: Let $f : \mathbb{R}^n \rightarrow \mathbb{R}$ be a target function. The gradient of f is given by the vector grad $f := (\nabla f) i := \frac{\partial f}{\partial x_i}$. The gradient points to the direction of steepest ascent of f .

    Since $\langle \boldsymbol{v}, \nabla f(\boldsymbol{x}) \rangle = ||\boldsymbol{v}|| \, ||\nabla f(\boldsymbol{x})|| \cos(\alpha)$, it holds that

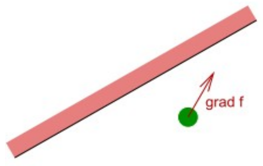
    1. $\langle \boldsymbol{v}, \nabla f(\boldsymbol{x}) \rangle < 0$: $\boldsymbol{v}$ is descending direction
    2. $\langle \boldsymbol{v}, \nabla f(\boldsymbol{x}) \rangle > 0$: $\boldsymbol{v}$ is ascending direction

- What are the feasible directions at x?
  - An inequality constraint g is called active at point x if g(x) = 0. Otherwise it is called inactive.
  - Inactive constraints: Inactive constraints do not restrict feasible directions
  - Active constraints: Directions v with $\langle v, \nabla g(x) \rangle > 0$ are infeasible

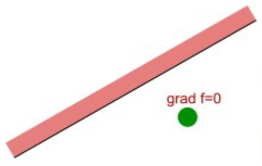




- Examples for inactive constraints

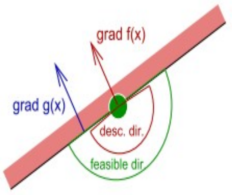


$\nabla f(\boldsymbol{x}) \neq \boldsymbol{0} \Rightarrow$ no minimum

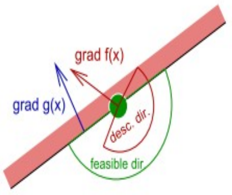


$\nabla f(\boldsymbol{x}) = \boldsymbol{0} \Rightarrow$ minimum

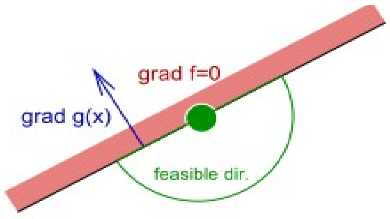
Examples for one active constraint:



$-\nabla f = \lambda \nabla g$ with $\lambda < 0$:
All descending directions feasible.
No minimum



$-\nabla f = \lambda \nabla g + \boldsymbol{u}$ with $\lambda < 0$ and $\boldsymbol{u} \perp \nabla g$, $\boldsymbol{u} \neq \boldsymbol{0}$:
Some descending directions feasible.
No minimum



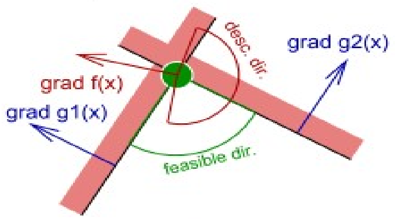
$-\nabla f = \boldsymbol{0} = 0 \nabla g$:
No descending directions feasible.
Minimum

For one active constraint we found:
Decomposing $-\nabla f(\boldsymbol{x}) = \lambda \nabla g(\boldsymbol{x}) + \boldsymbol{u}$ with $\boldsymbol{u} \perp \nabla g(\boldsymbol{x})$ we find feasible **and** descending directions only if $\lambda < 0$ or $\boldsymbol{u} \neq \boldsymbol{0}$

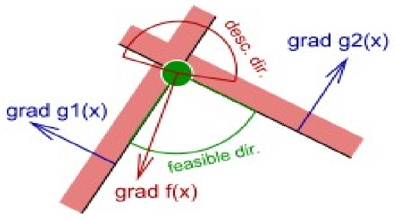
$$\boldsymbol{x} \text{ is minimum if we find } \lambda \geq 0 \text{ with } \nabla f(\boldsymbol{x}) + \lambda \nabla g(\boldsymbol{x}) = \boldsymbol{0}$$

Remark: this principle can be generalized to more active constraints

Examples with two constraints:



$-\nabla f = \lambda_1 \nabla g_1 + \lambda_2 \nabla g_2$ with
$\lambda_1 < 0, \lambda_2 \geq 0$:
Some descending directions feasible.
No minimum



$-\nabla f = \lambda_1 \nabla g_1 + \lambda_2 \nabla g_2$ with
$\lambda_1 \geq 0, \lambda_2 \geq 0$:
All descending directions non-feasible.
Minimum

Lemma

A feasible point $\boldsymbol{x}$ is a global minimum if exist $\alpha_j \geq 0$ such that

$$\nabla f(\boldsymbol{x}) + \sum_{\text{active constraints } j} \alpha_j \nabla g_j(\boldsymbol{x}) = 0$$

Since the inactive constraints are irrelevant, it holds that

Lemma

A feasible point $\boldsymbol{x}$ is a global minimum, if exist $\alpha_j \geq 0$ with

$$\alpha_j := \begin{cases} 0 & : \quad g_j(\boldsymbol{x}) < 0 \\ \geq 0 & : \quad g_j(\boldsymbol{x}) = 0 \end{cases} \quad \textit{such that}$$

$$\nabla f(\boldsymbol{x}) + \sum_{j=1}^{m} \alpha_j \nabla g_j(\boldsymbol{x}) = 0$$

- Lagrange Function $\quad L(\boldsymbol{x}, \boldsymbol{\alpha}) := f(\boldsymbol{x}) + \sum_{j=1}^{m} \alpha_j g_j(\boldsymbol{x})$
  - The α_i are called Lagrange Multipliers
    → optimization problem can be solved by inspecting the gradient of the lagrange function

- Analytical Solution of Convex Optimization Problems
  - Saddle Point Condition

    Let $f : \mathbb{R}^n \to \mathbb{R}$ be a target function and let $g_i : \mathbb{R}^n \to \mathbb{R}$, $(1 \leq i \leq m)$ be a set of inequality constraints defining an optimization problem

$$\begin{array}{rl} \min\limits_{\boldsymbol{x}} & f(\boldsymbol{x}) \\ \textit{subject to} & g_i(\boldsymbol{x}) \leq 0 \quad (1 \leq i \leq m) \end{array}$$

    Consider the Lagrange function $L(\boldsymbol{x}, \boldsymbol{\alpha}) := f(\boldsymbol{x}) + \sum_{i=1}^{m} \alpha_i g_i(\boldsymbol{x})$

    If a pair of vectors $(\boldsymbol{x}^*, \boldsymbol{\alpha}^*)$ exists with $\boldsymbol{\alpha}^* \in (\mathbb{R}^+)^m$, such that

$$\forall \boldsymbol{x} \in \mathbb{R}^n, \forall \boldsymbol{\alpha} \in (\mathbb{R}^+)^m : L(\boldsymbol{x}^*, \boldsymbol{\alpha}) \leq L(\boldsymbol{x}^*, \boldsymbol{\alpha}^*) \leq L(\boldsymbol{x}, \boldsymbol{\alpha}^*)$$

    then $\boldsymbol{x}^*$ is a valid solution of the optimization problem

  - Convex Optimization Problems

    Let $f : \mathbb{R}^n \to \mathbb{R}$, $g_i : \mathbb{R}^n \to \mathbb{R}$ $(1 \leq i \leq m)$ be convex and differentiable functions.

    How can we find (x^*, α^*) satisfying the saddle point condition?

$$\forall \boldsymbol{x} \in \mathbb{R}^n, \forall \boldsymbol{\alpha} \in (\mathbb{R}^+)^m : L(\boldsymbol{x}^*, \boldsymbol{\alpha}) \leq L(\boldsymbol{x}^*, \boldsymbol{\alpha}^*) \leq L(\boldsymbol{x}, \boldsymbol{\alpha}^*)$$

    Idea

    Decompose saddle point condition into two optimization problems.

Maximizing the Lagrange Function

$$\begin{aligned} \alpha^* \in \arg\max_{\alpha} L(\boldsymbol{x}^*, \alpha) \\ \Rightarrow \quad \frac{\partial L(\boldsymbol{x}^*, \alpha)}{\partial \alpha_i}(\alpha^*) &= 0 \quad (1 \leq i \leq m) \\ \Leftrightarrow \quad \frac{\partial (f(\boldsymbol{x}^*) + \sum_{i=1}^m \alpha_i g_i(\boldsymbol{x}^*))}{\partial \alpha_i}(\alpha^*) &= 0 \quad (1 \leq i \leq m) \\ \Leftrightarrow \quad g_i(\boldsymbol{x}^*) &= 0 \quad (1 \leq i \leq m) \end{aligned}$$

Minimizing the Lagrange Function

$$\begin{aligned} \boldsymbol{x}^* \in \arg\min_{\boldsymbol{x}} L(\boldsymbol{x}, \alpha^*) \\ \Leftrightarrow \quad \frac{\partial L(\boldsymbol{x}, \alpha^*)}{\partial x_i}(\boldsymbol{x}^*) &= 0 \quad (1 \leq i \leq m) \\ \Leftrightarrow \quad \frac{\partial (f(\boldsymbol{x}) + \sum_{i=1}^m \alpha_i^* g_i(\boldsymbol{x}))}{\partial x_i}(\boldsymbol{x}^*) &= 0 \quad (1 \leq i \leq m) \end{aligned}$$

Remember: The Lagrangre function is convex in $\boldsymbol{x}$!

Saddle Point Condition vs. System of Equations

$$\frac{\partial L(\boldsymbol{x}, \alpha^*)}{\partial x_i}(\boldsymbol{x}^*) = 0, \; g_i(\boldsymbol{x}^*) = 0 \quad (1 \leq i \leq m)$$

Is it necessary (and possible) to solve this system of equations exactly?

Lemma (Active Constraints)

If it exists a set $A \subseteq \{1, ..., m\}$, $\alpha^* \in (\mathbb{R}^+)^m$ and a feasible point $\boldsymbol{x}^* \in \mathbb{R}^n$ such that $g_i(\boldsymbol{x}^*) := \begin{cases} 0 & : \; i \in A \\ < 0 & : \; i \notin A \end{cases}$ and $\frac{\partial L(\boldsymbol{x}, \alpha^*)}{\partial x_i}(\boldsymbol{x}^*) = 0$ and $\forall i \notin A : \alpha_i^* = 0$, then $\boldsymbol{x}^*$ is a solution of the optimization problem.

The set A is called the set of active constraints!

Karush-Kuhn Tucker Conditions
Given a vector $x^* \in \mathbb{R}^n$ and a vector $\alpha^* \in (\mathbb{R}^+)^m$, the following conditions must be satisfied for $(1 \leq i \leq m)$ in order to qualify x* for a solution to the convex optimization problem:
→ α_j^* are Lagrange Multiplier

1.) $g_i(x^*) \leq 0$ 1.Case: g_i is active constraint $\Rightarrow g_i(x^*) = 0$
2.Case: g_i is inactive constraint $\Rightarrow g_i(x^*) < 0$

2.) $\frac{\partial L}{\partial x_i}(x^*, \alpha^*) = 0$ Part of system of equations

3.) $\alpha_i^* g_i(x^*) = 0$ 1. Case: g_i is active constraint $\Rightarrow$ $g_i(x^*) = 0$
2. Case: g_i is inactive constraint $\Rightarrow$ $\alpha_i^* = 0$

4.) $\alpha_i^* \geq 0$ Part of the saddle point conditions (KKT)

Theorem: Let $f: \mathbb{R}^n \to \mathbb{R}$ be a convex and differentiable target function and let $g_i: \mathbb{R}^n \to \mathbb{R}\ (1 \leq i \leq m)$ be a set of convex and differentiable inequality constraints. Then, a solution of the corresponding optimization problem is given by x^* iff. it exists Lagrange multiplier $\alpha^* \in (\mathbb{R}^+)^m$ such that the Karush-Kuhn Tucker conditions 1-4 are satisfied.

Brute Force Approach: Algorithm for Solving Convex Optimization Problems

```
1:  **for all** active sets $A \subseteq \{1, ..., m\}$ **do**
2:      Compute variables $x_i$ through $\frac{\partial L}{\partial x_i}(\boldsymbol{x}^*, \alpha^*) = 0$
3:      Compute nonzero lagrange multiplier $\alpha_i^* \in A$ through $g_i(\boldsymbol{x}^*) = 0$
4:      **if** $\exists i\ (1 \leq i \leq m) : \alpha_i^* < 0$ **then**
5:          Discard active set A
6:      **end if**
7:      **if** $\exists i\ (1 \leq i \leq m) : g_i(\boldsymbol{x}^*) > 0$ **then**
8:          Discard active set A
9:      **end if**
10: **end for**
11: Output variables $\boldsymbol{x}^*$
```

- Dual Problems
  Duality: Transforming the original (primal) optimization problem into an equivalent (dual) optimization problem possibly easier to solve

  Primal vs. Dual

$$\min_{\boldsymbol{x}} \ f(\boldsymbol{x}) \quad \text{subject to} \ g_i(\boldsymbol{x}) \leq 0 \qquad \Leftrightarrow \qquad \max_{\boldsymbol{y}} \ Q(\boldsymbol{y}) \quad \text{subject to} \ h_j(\boldsymbol{y}) \leq 0$$

  Lower Bound of Lagrangian
  $\rightarrow$ Infimum is the largest lower bound of the Lagrangian. If no bound exists, we say the infimum is $-\infty$

$$Q(\alpha) := \inf_{\boldsymbol{x} \in \mathbb{R}^n} L(\boldsymbol{x}, \alpha)$$

$$L(\boldsymbol{x}, \alpha) := f(\boldsymbol{x}) + \sum_{i=1}^{m} \alpha_i g_i(\boldsymbol{x})$$

Dual Problem
$\rightarrow \rightarrow \rightarrow \rightarrow \rightarrow$

$$\max_{\alpha} \ Q(\alpha) \quad \text{subject to} \ \alpha_i \geq 0 \quad (1 \leq i \leq m)$$

Equivalence of Primal and Dual

1.) If x is a feasible point of the primal problem and α is a feasible point of the dual problem, then it holds that $Q(\alpha) \leq f(x)$

2.) If x^* is a feasible point of the primal problem and α^* is a feasible point of the dual problem and $f(x^*) = Q(\alpha^*)$, then x^* is a solution of the primal problem and α^* is a solution of the dual problem

3.) Let f be a convex and differentiable target function and let the constraints g_i be linear. If a solution x^* of the primal problem exists, then it exists a solution α^* for the dual problem and it holds that $f(x^*) = Q(\alpha^*)$

Linear SVMs

- Primal Problem for Linear SVMs:

$$\min_{\boldsymbol{w}, w_0} \quad \frac{1}{2}||\boldsymbol{w}||^2$$
$$\text{subject to} \quad 1 - (d_i \cdot (\boldsymbol{w}^T \boldsymbol{x}_i + w_o)) \leq 0 \quad \forall (\boldsymbol{x}_i, d_i) \in \mathcal{D}$$

  - Properties:
    - Convex and differentiable target function
    - Linear constraints ($\Rightarrow$ convex and differentiable)
    - Number of constraints equals number of training samples
    - !°!°! targetfunction now depends on w (not on x)

- Dual Problem

$$\max_{\alpha} \quad Q(\alpha) = \inf_{\boldsymbol{w}, w_0} L(\boldsymbol{w}, w_0, \alpha)$$
$$\text{subject to} \quad \alpha_i \geq 0 \quad (1 \leq i \leq m)$$

Since $L(w, w_{0,} \alpha)$ is convex, we know that $\nabla L(w, w_{0,} \alpha) = 0$ is sufficient for a global minimum (infimum). Thus, inserting $w = \sum_{i=1}^{N} \alpha_i d_i x_i$ and $\sum_{i=1}^{N} \alpha_i d_i = 0$ yields $\rightarrow$

$\rightarrow$ Final Form of Dual Problem

$$\max_{\alpha} \quad -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} (\alpha_i \alpha_j d_i d_j \boldsymbol{x}_i^T \boldsymbol{x}) + \sum_{i=1}^{N} \alpha_i$$
$$\text{subject to} \quad \sum_{i=1}^{N} \alpha_i d_i = 0 \wedge \alpha_j \geq 0$$

- Primal vs. Dual
  - Dual problem can be also solved with active set methods (e.g.brute force)
  - After solving the dual, a solution of the primal problem is given by

$$\boldsymbol{w} = \sum_{i=1}^{N} \alpha_i d_i \boldsymbol{x}_i \qquad w_0 = d_i - \sum_{j=1}^{N} \alpha_j d_j \boldsymbol{x}_j^T \boldsymbol{x}_i \qquad \rho = \frac{1}{||\boldsymbol{w}||}$$

    - This follows immediately from the saddle point theorem!
  - For non-linear SVMs, we will see an advantage of solving the dual instead of the primal!

- KKT Conditions for Linear SVMs
  - Lagrange Function:

    $$L(w, w_0, \alpha) = \frac{1}{2}\|w\|^2 + \sum_{i=1}^{N} \alpha_i - \sum_{i=1}^{N} \alpha_i d_i w^T x_i - w_0 \sum_{i=1}^{N} \alpha_i d_i$$

  - Computing partial Derivatives

    $$\frac{\partial L}{\partial w_j} = w_j - \sum_{i=1}^{N} \alpha_i d_i x_i(j)$$

    $$\frac{\partial L}{\partial w_0} = - \sum_{i=1}^{N} \alpha_i d_i$$

  - Setting Partial Derivatives to Zero
    → according to the KKT-Condition $\nabla L(w, w_0, \alpha) = 0$ we get:

    $$w_j = \sum_{i=1}^{N} \alpha_i d_i x_i(j)$$

    $$\mathbf{w} = \sum_{i=1}^{N} \alpha_i d_i \mathbf{x}_i$$

    → still need the bias weight

  - Computing the Bias Weight
    - choose an active constraint → $g_i(w) = 0$

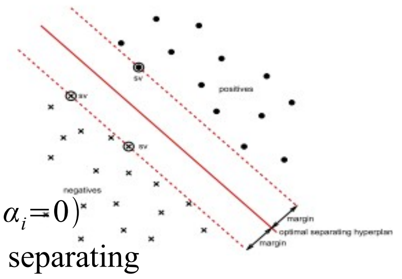
      $$0 = \sum_{i=1}^{N} \alpha_i d_i$$

    then we get: $d_i - \sum_{j=1}^{N} \alpha_j d_j x_j^T x_i = w_0$

- Modified Brute Force Approach

Algorithm for Solving Linear SVMs

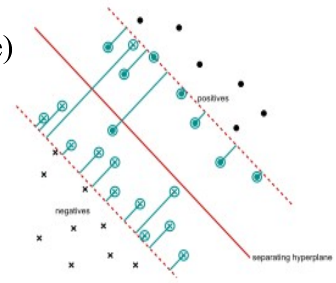
```
1:  for all active sets $A \subseteq \{1, ..., m\}$ do
2:      Compute weights $\mathbf{w}$ using the formula above
3:      Compute nonzero lagrange multiplier $\alpha_i \in A$ through $g_i(\mathbf{w}) = 0$
4:      if $\sum_{i=1}^{N} \alpha_i d_i \neq 0$ then
5:          Discard active set A
6:      end if
7:      if $\exists i \ (1 \leq i \leq m) : \alpha_i < 0$ then
8:          Discard active set A
9:      end if
10:     if $\exists i \ (1 \leq i \leq m) : g_i(\mathbf{w}) > 0$ then
11:         Discard active set A
12:     end if
13: end for
14: Compute bias $w_o$ using an active constraint
15: Output weight vector $\mathbf{w}$ and bias weight $w_o$
```



Summmary of Linear SVMs

- Each constraint corresponds to a training sample
-
- Some constraints become active $(\alpha_i \geq 0)$ and some become inactive $(\alpha_i = 0)$
- Training samples corresponding to active constraints are located next to separating hyperplane (on the margin)
- Training samples on the margin are called support vectors
- Solution of SVM only depends on support vectors

Soft Margin



- Idea:
  - SVMs provide perfect classification on training samples (hard margin case)
  - But what happens, if problem is not linearly separable?
  - Soft Margin SVMs tolerate classification errors
    → Maximizing Margin vs. Minimizing Error

- Extended Constraints

$$\begin{aligned} d_i \cdot (\boldsymbol{w}^T \boldsymbol{x}_i + w_o) &\geq 1 - \xi_i \\ \xi_i &\geq 0 \end{aligned}$$

The slack variables ξi allow the SVM to tolerate errors.

- Extended Target

$$\min_{\boldsymbol{w}, w_0, \xi,} \quad \frac{1}{2}||\boldsymbol{w}||^2 + C \sum_{i=1}^{N} \xi_i^2 \quad (C > 0 \text{ is a fixed constant})$$

Minimizing the error means to minimize the slack variables

- **Primal Problem for Soft Margin SVMs**

$$\min_{\boldsymbol{w}, w_0, \xi} \quad \frac{1}{2}||\boldsymbol{w}||^2 + C \sum_{i=1}^{N} \xi_i^2$$

$$\text{subject to} \quad d_i \cdot (\boldsymbol{w}^T \boldsymbol{x}_i + w_o) \geq 1 - \xi_i \wedge \xi_i \geq 0 \quad \forall (\boldsymbol{x}_i, d_i) \in \mathcal{D}$$

- Lagrange Function (with additional multipliers)

$$\begin{aligned} L(\boldsymbol{w}, w_0, \xi, \alpha) &= \frac{1}{2}||\boldsymbol{w}||^2 + C \sum_{i=1}^{N} \xi_i^2 \\ &+ \sum_{i=1}^{N} \alpha_i (1 - \xi_i - d_i(\boldsymbol{w}^T \boldsymbol{x}_i + w_o)) + \sum_{i=1}^{N} \alpha_{i+N}(-\xi_i) \end{aligned}$$

- Karush Kuhn Tucker Conditions for Soft Margin SVMs

$$\begin{aligned}
1 - \xi_i - d_i(\mathbf{w}^T\mathbf{x}_i + w_o)) &\leq 0 \quad \forall(\mathbf{x}_i, d_i)\\
-\xi_i &\leq 0 \quad \forall(\mathbf{x}_i, d_i)\\
\alpha_i(1 - \xi_i - d_i(\mathbf{w}^T\mathbf{x}_i + w_o)) &= 0 \quad \forall(\mathbf{x}_i, d_i)\\
\alpha_{i+N}(-\xi_i) &= 0 \quad \forall(\mathbf{x}_i, d_i)\\
\frac{\partial L}{\partial w_i} = w_j - \sum_{i=1}^{N} \alpha_i d_i x_i(j) &= 0 \quad \forall(\mathbf{x}_i, d_i)\\
\frac{\partial L}{\partial w_0} = -\sum_{i=1}^{N} \alpha_i d_i &= 0 \quad \forall(\mathbf{x}_i, d_i)\\
\frac{\partial L}{\partial \xi_i} 2C\xi_i - \alpha_i - \alpha_{i+N} &= 0 \quad \forall(\mathbf{x}_i, d_i)\\
\alpha_i \geq 0 \wedge \alpha_{i+N} \geq 0 & \qquad \forall(\mathbf{x}_i, d_i)
\end{aligned}$$

- Dual Problem for Soft Margin SVMs

$$\begin{aligned}
\max_{\boldsymbol{\alpha}} \quad & Q(\boldsymbol{\alpha}) := \inf_{\mathbf{w}, w_0, \xi} L(\mathbf{w}, w_0, \xi, \boldsymbol{\alpha})\\
\text{subject to} \quad & \alpha_i \geq 0 \;\; (1 \leq i \leq 2N)
\end{aligned}$$

  → Final Form:

$$\begin{aligned}
\max_{\boldsymbol{\alpha}} \quad & -\tfrac{1}{2}\textstyle\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i\alpha_j d_i d_j \mathbf{x}_i^T\mathbf{x}_j\\
& -\tfrac{1}{2}\textstyle\sum_{i=1}^{N} \frac{(\alpha_i+\alpha_{i+N})^2}{2C} + \sum_{i=1}^{N} \alpha_i\\
\text{subject to} \quad & \textstyle\sum_{i=1}^{N} \alpha_i d_i = 0 \wedge \alpha_j \geq 0 \;\; (1 \leq j \leq 2N)
\end{aligned}$$

- Summary of Soft Margin:
  - Fault Tolerant SVMs
  - Slack variables enable SVM to tolerate errors
  - Tolerance is controlled by constant C
    - Large C prefers small errors
    - Small C prefers large margin
  - Support vectors consists now of
  - Training samples on the margin bound
  - Misclassified training samples within the margin bound

Non Linear SVMs

two Ideas:

- Generalizing the target and the constraints to arbitrary (non-linear) functions
    - Losing convexity
    - KKT conditions are not sufficient for existence of optimum
    - Optimization process becomes inefficient
- Using non-linear feature mappings
    - Feature mapping $\varphi : \mathbb{R}^n \rightarrow \mathbb{R}^M$ maps into high dimensional space
    - Training samples $x \in \mathbb{R}^n$ are replaced by features $\varphi(x) \in \mathbb{R}^M$

Dual Problem with Features

$$\max_{\alpha} \quad -\tfrac{1}{2}\textstyle\sum_{i=1}^{N}\sum_{j=1}^{N}[\alpha_i\alpha_j d_i d_j \phi(\boldsymbol{x}_i)^T\phi(\boldsymbol{x})] + \sum_{i=1}^{N}\alpha_i$$
$$\text{subject to} \quad \textstyle\sum_{i=1}^{N}\alpha_i d_i = 0 \wedge \alpha_j \geq 0$$

Kernel Trick

Introduce kernel functions K_φ (x, y) = φ(x)^T φ(y)

- Kernel functions efficiently compute inner products within high dimensional feature spaces
- Training samples occur in the dual only in the form of inner products expressed by kernel functions.
- Simplifies solution of optimization problem for complex feature spaces

- Dual Problem with Kernels

$$\max_{\alpha} \quad -\tfrac{1}{2}\textstyle\sum_{i=1}^{N}\sum_{j=1}^{N}[\alpha_i\alpha_j d_i d_j K_\phi(\boldsymbol{x}_i,\boldsymbol{x}_j)] + \sum_{i=1}^{N}\alpha_i$$
$$\text{subject to} \quad \textstyle\sum_{i=1}^{N}\alpha_i d_i = 0 \wedge \alpha_j \geq 0$$

- Using Kernel Functions (Simple Way)
  - Design of feature space by creating feature mapping φ
  - Derivation of Kernel function Kφ
  - Solution of dual problem using Kernel function Kφ

  Using kernel functions this way gains not much compared to explicitly computing the features!

- Generic Kernels
  - Feature space may have infinite or very large dimension. It is not possible to compute features φ(x) explicitly
  - Explicit representation of feature space can be unknown
  - Can be used for many applications

- How can we classify an unknown pattern x?
  → Checking Hyperplane Condition

$$\begin{aligned}\boldsymbol{w}^T\phi(\boldsymbol{x}) + w_0 &= \sum_{i=1}^{N}(\alpha_i d_i \phi(\boldsymbol{x}_i)^T\phi(\boldsymbol{x})) + w_o \\ &= \sum_{i=1}^{N}(\alpha_i d_i K_\phi(\boldsymbol{x}_i,\boldsymbol{x})) + w_0\end{aligned}$$

  Kernel function can be used to check, if pattern is above/beneath the separating hyperplane

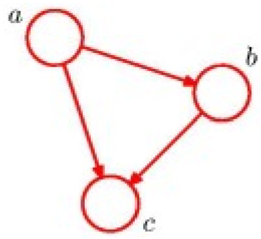
- Summary of Kernel Trick
  - Replacing the inner product by kernel function is possible
  - Explicit representation of features may be impossible
  - Nonetheless, predictions can be made
  - It is necessary to memorize support vectors and non-zero lagrange-multipliers

Summary of SVMs

- Linear classification with separating hyperplane
- Maximization of margin guarantees good generalization

- Convex optimization problem
- Solution is represented in terms of support vectors
- Soft Margin: Classification with errors
- Kernel-Trick: Non-linear decision boundary by using kernel functions to solve the classification problem in a high dimensional (possibly unknown) feature space
- Code for solving SVM optimization problems available for free (e.g. libSVM)

Graphical Models



- Probability theory :
  - ensures that the system as a whole is consistent
  - provides a way to interface models to data
- provide a way for dealing with uncertainty and complexity
- increasing role in design and analysis of machine learning algorithms
- notion of modularity –a complex system is built by combining simpler parts.
- Many of the classical multivariate probabilistic systems are special cases of the general graphical model formalism
- provide a way to view all systems as instances of a common underlying formalism
- provides a natural framework for the design of new systems

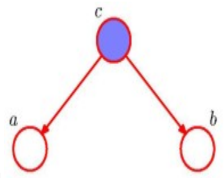
Probabilistic graphical models

- provide a simple way to visualize structure of probabilistic model
- can be used to design and motivate new models
- Insights into the properties of the model, including conditional independence properties, can be obtained by inspection of the graph
- complex computations can be expressed in terms of graphical manipulations, in which underlying mathematical expressions are carried along implicitly

Properties:

- nodes: = vertices
  - represent a random variable(or group of RV)

- links: = edges/arcs connect the nodes
  - express probabilistic relationships between the random variables
- the graph captures the way in which the joint distribution over all of the random variables can be decomposed into a product of factors each depending only on a subset of variables
- observed/fixed variables: are shaded nodes in the graph
- tail-to-tail:
  - always with respect to the path of that node
  - if a node is connected to the tails of two arrows (in our example tree it is c)



- head-to-tail
  - c is said to be head-to-tail with respect to the path from node a to node b. Such a path connects nodes a and b and renders them dependent

- head-to-head
  - connects to the heads of the two arrows
- conditional independence property : $a \perp\!\!\!\perp b \mid c$
-

directed graphical models

- useful for expressing causal relationships between RV
- e.g. Bayesian networks
- Links of the graphs have a particular directionality indicated by arrows

undirected graphical models

- express soft constraints between RV
- e.g. Markov random fields
- links do not carry arrows and have no directional significance

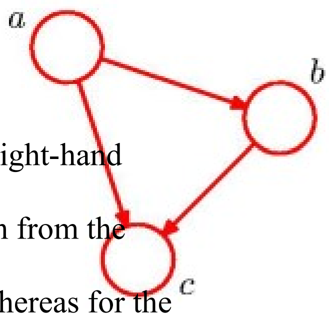
factor graph

- For the purposes of solving inference problems,it is often convenient to convert both directed and undirected graphs into a different representation called a factor graph

Conditional probabitlistic graphical models

→ we take the joint distribution of our nodes and use product rule to reformulate them

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



→ right hand side of this equation is a simple graphical model:

1. → introduce a node for each of the random variables a, b, and c
   → associate each node with the corresponding conditional distribution on the right-hand side of the eqn.
2. → for each conditional distribution we add directed links (arrows) to the graph from the nodes corresponding to the variables on which the distribution is conditioned.
   Thus for the factor p(c|a,b), there will be links from nodes a and b to node c, whereas for the factor p(a) there will b   e no incoming links.

- **Parent node:** a link is going from node a to node b   ↔ a is parent of b
- **child node :** a link is going from node a to node b ↔ b is child of a
- left-hand side of the equation above is Symmetrical with respect to the three variables a, b, and c, whereas the right-hand side is not
  → had we chosen a different ordering of a,b,c we would have obtained another decomposition and thereby a different graph:

$$\begin{aligned} p(a,b,c) &= p(c|a,b)p(b|a)p(a) \\ &= p(a|b,c)p(b|c)p(c) \\ &= p(b|a,c)p(c|a)p(b) \end{aligned}$$

- A directed graph for K variables in general:

$$p(x_1, x_2, \ldots, x_K) = p(x_K|x_1, x_2, \ldots, x_{K-1}) \cdot p(x_{K-1}|x_1, x_2, \ldots, x_{K-2}) \cdot \ldots \cdot p(x_2|x_1)p(x_1)$$

- **fully connected graph:** there is a link between every pair of nodes
- **directed cycles**
  → there are no closed paths within the graph such that we can move from node to node along links following the direction of the arrows and end up back at the starting node

- **directed acyclic graphs (DAGs)**
  - fully connected graph
  - no directed cycles

Conditional Probability Table (CPT)

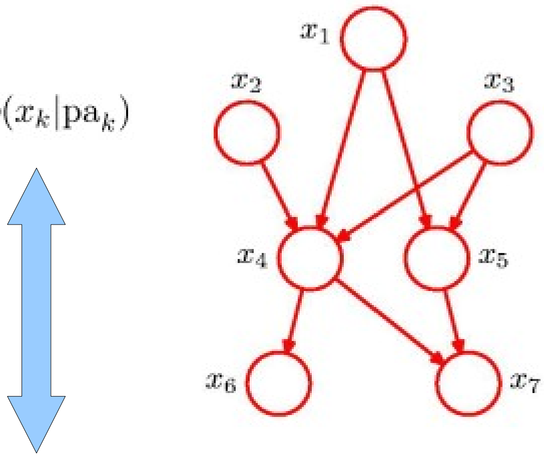
- topology of the network defines the conditional probability table (CPT) for each node.
- Each row in the table contains the conditional probability of each node value for a conditioning case.
- Each row must sum to 1, because the entries represent an exhaustive set of cases for the variable.
- A conditioning case is a possible combination of values for the p arent nodes.

General Factorization

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

→ joint distribution for a graph with K nodes.

→ pa_k = set of parents of x_k

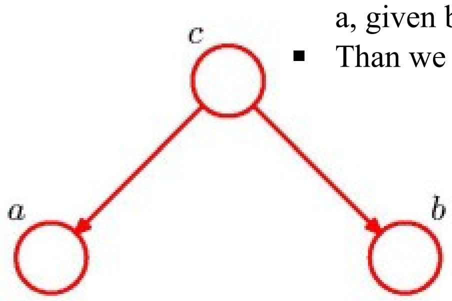


$$p(x_1, \ldots, x_7) = p(x_1)p(x_2)p(x_3)p(x_4|x_1, x_2, x_3) p(x_5|x_1, x_3)p(x_6|x_4)p(x_7|x_4, x_5)$$

Independence in graphical models

- Conditional Independence
  - most important property for graphical models is statistical independence
  - Consider three variables a, b, and c, and suppose that the conditional distribution of a, given b and c, is such that it does not depend on the value of b.
  - Than we say:

a is independent of b given c



a is independent of b given c: $p(a|b, c) = p(a|c)$

Alternatively:

$$p(a, b|c) = p(a|b, c)p(b|c) = p(a|c)p(b|c) \qquad a \perp\!\!\!\perp b \mid c$$

Conditioned on c, the joint distribution of a and b ***factorizes*** into the product of the marginal distribution of a and the marginal distribution of b (again both conditioned on c). This says that the variables a and b are statistically independent, given c.

Joint Distribution

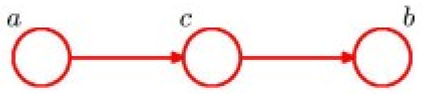
Same graph but now we suppose condition on the variable c

$$p(a,b|c) = \frac{p(a,b,c)}{p(c)} = p(a|c)p(b|c)$$

so we obtain the conditional independence property $a \perp\!\!\!\perp b \mid c$

→ path is the result from node a to node b via c
→ node c is then tail-to-tail
→ presence of such a path connecting nodes a and b causes these nodes to be dependent
→ However, when we condition on node c, the conditioned node 'blocks' the path from a to b and causes a and b to become **conditionally independent**

Causal Chains

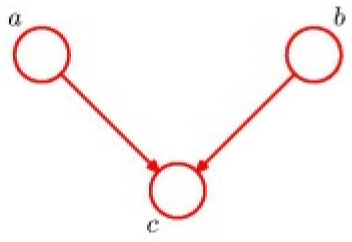


$$p(a,b,c) = p(a)p(c|a)p(b|c)$$

- suppose that none of the variables are observed
- test to see if a and b are independent by marginalizing c:

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

- which in general does not factorize into p(a)p(b), and so $a \not\perp\!\!\!\perp b \mid \emptyset$



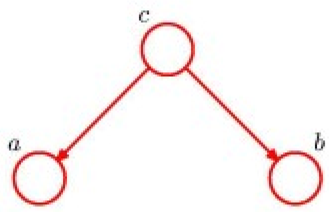
Joint Distribution

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

Marginalizing over c we obtain:

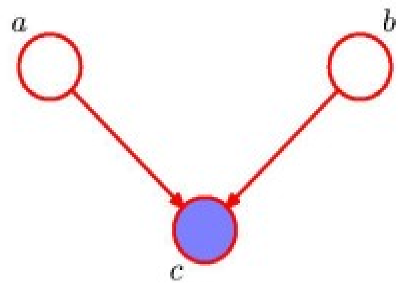
$$p(a,b) = p(a)p(b) \qquad a \perp\!\!\!\perp b \mid \emptyset$$

Note: this is the opposite of Example 1, with c unobserved.



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

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



Joint Distribution

$$p(a,b|c) = \frac{p(a,b,c)}{p(c)} = \frac{p(a)p(b)p(c|a,b)}{p(c)}$$

$$a \not\perp\!\!\!\perp b \mid c$$

When node c is unobserved, it 'blocks' the path, and the variables a and b are independent. However, conditioning on c 'unblocks' the path and renders a and b dependent.

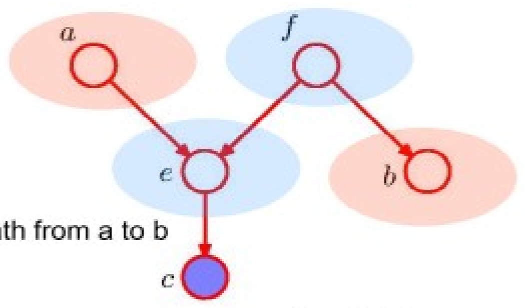
Summary:

- a tail-to-tail node or a head-to-tail node leaves a path unblocked unless it is observed in which case it blocks the path.
- a head-to-head node blocks a path if it is unobserved, but once the node, and/or at least one of its descendants, is observed the path becomes unblocked.

D-separation

We wish to ascertain whether a particular conditional independence statement $A \perp\!\!\!\perp B \mid C$ is implied by a given directed acyclic graph. To do so, we consider all possible paths from any node in A to any node in B.

- A, B, and C are non-intersecting subsets of nodes in a directed graph.
- A path from A to B is blocked if it contains a node such that either
  1. the arrows on the path meet either head-to-tail or tail-to-tail at the node, and the node is in the set C, or
  2. the arrows meet head-to-head at the node, and neither the node, nor any of its descendants, are in the set C.
  3. We say that node *y* is a ***descendant*** of node *x* if there is a path from ***x* to *y*** in which each step of the path **follows the directions of the arrows**.
- If all paths from A to B are blocked, A is said to be d-separated from B by C.
- If A is d-separated from B by C, the joint distribution over all variables in the graph satisfies $A \perp\!\!\!\perp B \mid C$, that is that A and B are conditionally independent



Path from a to b

$$a \not\perp\!\!\!\perp b \mid c$$

- not blocked by node *f* because it is a tail-to-tail node and not observed
- not blocked by node *e* because, because it is a head-to-head node, but it has a descendant *c* that is in the conditioning set.

We say that node *y* is a ***descendant*** of node *x* if there is a path from *x* to *y* in which each step of the path follows the directions of the arrows.

- A, B, and C are non-intersecting subsets of nodes in a directed graph. C is the set of observed nodes.
- A path from A to B is blocked if it contains a node such that either
  1. the arrows on the path meet either head-to-tail or tail-to-tail at the node, and the node is in the set C of observed nodes, or
  2. the arrows meet head-to-head at the node, and neither the node, nor any of its descendants, are in the set C.
- If all paths from A to B are blocked, A is said to be d-separated from B by C.