[Spring, 2017]

Linear Models for Classification

Pattern Recognition (BRI623)

Heung-Il Suk

hisuk@korea.ac.kr

http://www.ku-milab.org

Department of Brain and Cognitive Engineering,
Korea University

Contents

1. Introduction
2. Discriminant Functions
3. Probabilistic Generative Models
4. Probabilistic Discriminative Models
5. Laplace Approximation
6. Bayesian Logistic Regression

Introduction

Regression

- assign an input vector $\mathbf{x}$ to one or more continuous target variables t

Classification

- assign an input vector $\mathbf{x}$ to one of K discrete classes C_k, $k = 1, \ldots, K$

Linear Classification Models

- Disjoint classes (common)
- Input space: divided into decision regions
- Decision surfaces are linear functions of an input $\mathbf{x}$
  - $D - 1$ dimensional hyperplane within D dimensional input space
  - Straight line in $2D$
  - $2D$ plane in $3D$
  - Hyperplane in higher than $3D$
- Linearly separable
  - Data sets whose classes can be separated by linear decision surface

Class Label Representations

- Two class ($K = 2$): binary representation
  - $t \in \{1(C_1), 0(C_2)\}$: interpreting value of t as probability that class is C_1
  - $t \in \{1(C_1), -1(C_2)\}$ (also possible but not discussed here)
- For $K > 2$: 1-of-K coding scheme
  - $\mathbf{t} \in \{0,1\}^K$ is a vector of length K
  - *e.g.*, $\mathbf{t} = [0,1,0,0,0]^\top$: a pattern of class C_2 when $K = 5$
  - can interpret a value of t_k as probability of class C_k

Different Approaches to Classification

1. Discriminant function
   - Directly assign $\mathbf{x}$ to a specific class
   - *e.g.*, Fisher's linear discriminant, perceptron
2. Probabilistic models
   - Model $p\left(C_k | \mathbf{x}\right)$ in inference stage (directly or by a Bayes rule)
   - Use it to make optimal decisions

Probabilistic Models

- Generative
  - Model class conditional densities by $p\left(\mathbf{x}|C_k\right)$ together with prior probabilities $P\left(C_k\right)$
  - Then use a Bayes rule to compute posterior

$$p\left(C_k|\mathbf{x}\right) = \frac{p\left(\mathbf{x}|C_k\right)P\left(C_k\right)}{p\left(\mathbf{x}\right)}$$

- Discriminative
  - Directly model conditional probabilities $p\left(C_k|\mathbf{x}\right)$

- Separating inference from decision (to explicitly obtain posterior) is better
  - Minimize risk
  - Reject option (minimize expected loss)
  - Compensate for unbalanced data
    - Use modified balanced data & scale by class fractions
  - Combine models

From Linear Regression to Linear Classification

- Linear regression model $y\left(\mathbf{x},\mathbf{w}\right)$

$$y\left(\mathbf{x}\right)=\mathbf{w}^{\top}\mathbf{x}+w_0$$

- For classification, we wish to obtain a discrete output or posterior probabilities in a range $(0,1)$

$$y\left(\mathbf{x}\right)=f\left(\mathbf{w}^{\top}\mathbf{x}+w_0\right)$$

Generalized Linear Model

$$y\left(\mathbf{x}\right)=f\left(\mathbf{w}^{\top}\mathbf{x}+w_0\right)$$

- $f\left(\cdot\right)$: nonlinear, known as the activation function
- Decision surfaces
  - $y\left(\mathbf{x}\right)=$ constant or $\mathbf{w}^{\top}\mathbf{x}+w_0=$ constant
- Decision surfaces are linear functions of $\mathbf{x}$ even if $f\left(\cdot\right)$ is nonlinear (generalized linear model) [McCullagh and Nelder, 1989]
- Nonlinear in the parameter space $\mathbf{w}$ due to the nonlinear function $f\left(\cdot\right)$
  - Leads to more complex models for classification than regression

Discriminant Functions

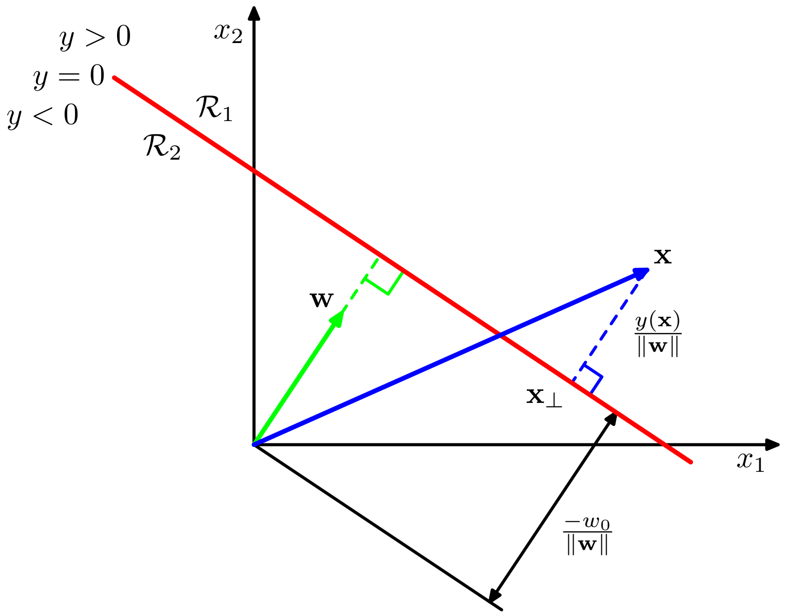
Geometry of Linear Discriminant Functions

- Two-class linear discriminant function

$$y(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + w_0$$

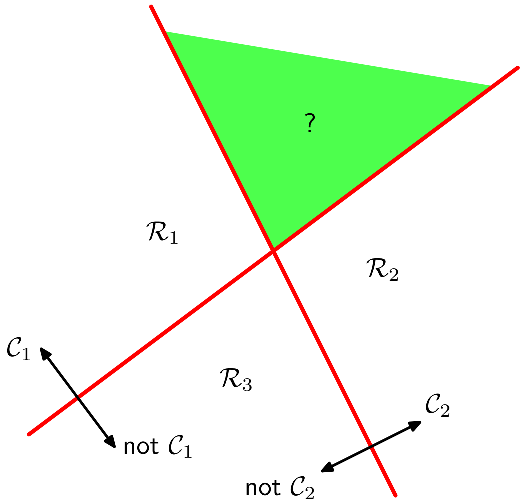
($\mathbf{w}$: weight vector, w_0: bias/threshold)

  - Assign $\mathbf{x}$ to class C_1 if $y(\mathbf{x}) \geq 0$, otherwise class C_2
  - Decision boundary: $y(\mathbf{x}) = 0$
    - Geometrically, $\mathbf{w}$ determines the orientation of the decision surface

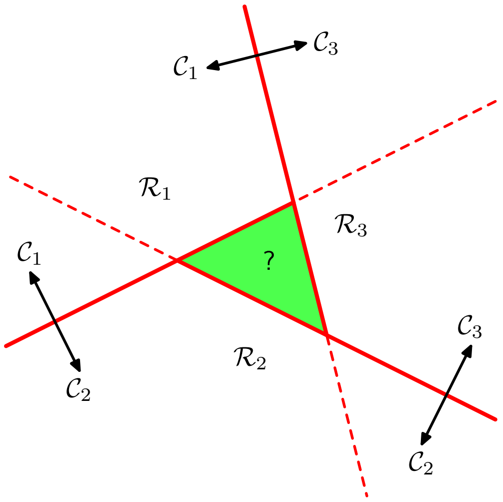


Multiple Classes with Binary Classifiers

One-versus-Rest ($K - 1$ classifiers)



One-versus-one ($K(K-1)/2$ classifiers)



Multiple Classes with K Discriminants

- Consider a single K class discriminant of the form

$$y_k\left(\mathbf{x}\right) = \mathbf{w}_k^\top \mathbf{x} + w_{k0}$$

- Assign a point $\mathbf{x}$ to class C_k if $y_k\left(\mathbf{x}\right) > y_j\left(\mathbf{x}\right)$ for all $j \neq k$
  - Decision boundary between class C_k and C_j: $y_k\left(\mathbf{x}\right) = y_j\left(\mathbf{x}\right)$
  - $D-1$ dimensional hyperplane defined by

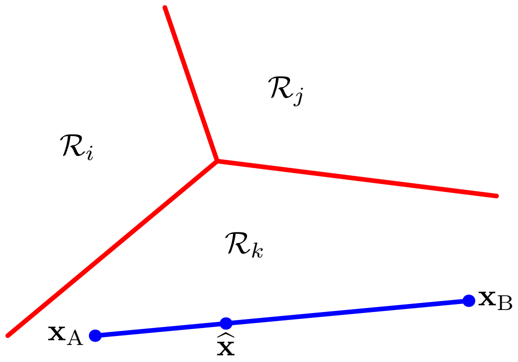
$$\left(\mathbf{w}_k - \mathbf{w}_j\right)^\top \mathbf{x} + \left(w_{k0} - w_{j0}\right) = 0$$

Decision regions of such a discriminant are always singly connected and convex

Convexity of Decision Regions

Proof!!!

$$\begin{aligned}\hat{\mathbf{x}} &= \lambda \mathbf{x}_A + (1-\lambda)\mathbf{x}_B \\ & \quad (0 \leq \lambda \leq 1) \\ y_k\left(\hat{\mathbf{x}}\right) &= \lambda y_k\left(\mathbf{x}_A\right) + (1-\lambda) y_k\left(\mathbf{x}_B\right) \\ y_k\left(\mathbf{x}_A\right) &> y_j\left(\mathbf{x}_A\right) \\ y_k\left(\mathbf{x}_B\right) &> y_j\left(\mathbf{x}_B\right) \\ y_k\left(\hat{\mathbf{x}}\right) &> y_j\left(\hat{\mathbf{x}}\right)\end{aligned}$$



Learning Parameters of Linear Discriminant Functions

- Least Squares
- Fisher's Linear Discriminant
- Perceptrons

Least Squares for Classification

- Analogous to regression, there exists a simple closed-form solution for parameters
- Each C_k, $k = 1, \ldots, K$, is described by

$$y_k(\mathbf{x}) = \mathbf{w}_k^\top \mathbf{x} + w_{k0}$$

- By grouping into vector notation

$$\mathbf{y}(\mathbf{x}) = \tilde{\mathbf{W}}^\top \tilde{\mathbf{x}}$$

  - Augmented vectors: $\tilde{\mathbf{w}}_k = \left[w_{k0}, \mathbf{w}_k^\top\right]^\top$, $\tilde{\mathbf{x}} = \left[1, \mathbf{x}^\top\right]^\top$
  - $\tilde{\mathbf{W}} = [\tilde{\mathbf{w}}_1, \cdots, \tilde{\mathbf{w}}_K]$
- A new input vector $\mathbf{x}$ is assigned to class for which the output $y_k = \tilde{\mathbf{w}}_k^\top \tilde{\mathbf{x}}$ is the largest.

Given a training set $\{\mathbf{x}_n, \mathbf{t}_n\}$, $n = 1, \ldots, N$,

- Sum of squares error function

$$E_D\left(\tilde{\mathbf{W}}\right) = \frac{1}{2}\mathrm{Tr}\left\{\left(\mathbf{X}\tilde{\mathbf{W}} - \mathbf{T}\right)^\top \left(\mathbf{X}\tilde{\mathbf{W}} - \mathbf{T}\right)\right\}$$

$$\text{where } \left\{ \begin{array}{l} \mathbf{T} \equiv \left[\mathbf{t}_1^\top; \cdots ; \mathbf{t}_N^\top\right] \in \mathbb{R}^{N \times K} \\ \mathbf{X} \equiv \left[\tilde{\mathbf{x}}_1^\top; \cdots ; \tilde{\mathbf{x}}_N^\top\right] \in \mathbb{R}^{N \times (D+1)} \end{array} \right.$$

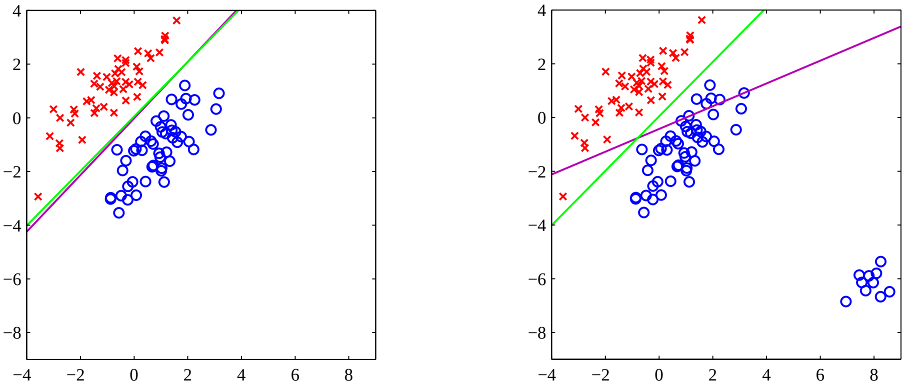
- Set derivative w.r.t. $\tilde{\mathbf{W}}$ to zero

$$\tilde{\mathbf{W}} = \underbrace{\left(\mathbf{X}^\top\mathbf{X}\right)^{-1}\mathbf{X}^\top}_{\mathbf{X}^\dagger:\text{ pseudo-inverse of }\mathbf{X}} \mathbf{T} = \mathbf{X}^\dagger\mathbf{T}$$

- After rearranging,

$$\mathbf{y}\left(\mathbf{x}\right) = \tilde{\mathbf{W}}^\top\mathbf{x} = \mathbf{T}^\top\left(\mathbf{X}^\dagger\right)^\top\mathbf{x}$$

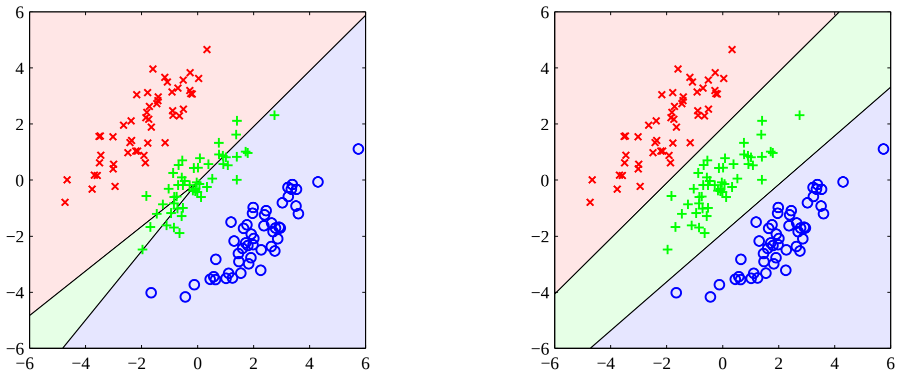
Least square is sensitive to outliers!!!



Magenta(least squares); Green(logistic regression)

- Sum of squared errors penalizes predictions that are “too correct” or long way from decision boundary
  - c.f.) Support Vector Machine: hinge loss function

Disadvantage of Least Squares!!!



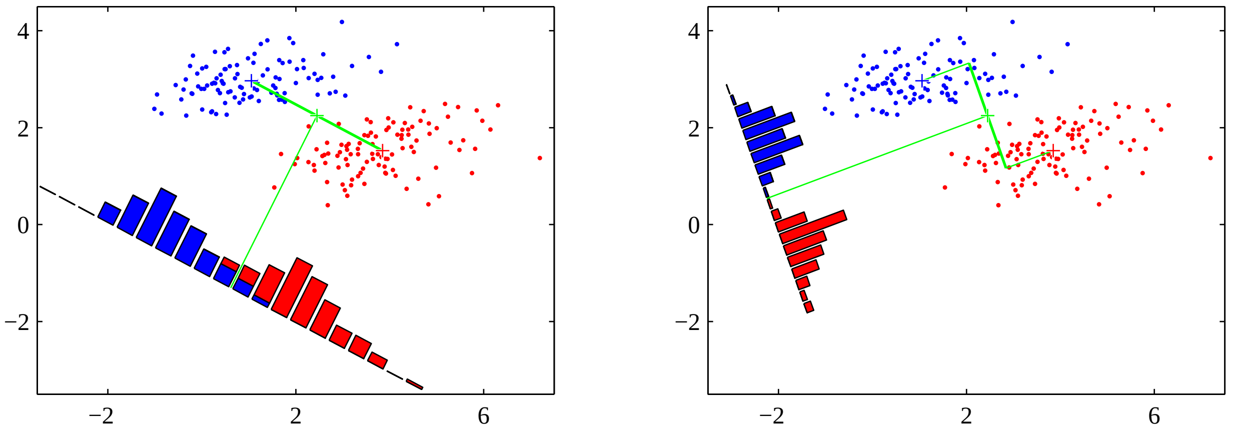
(Least squares) (Logistic regression)

The region of input space assigned to the green class is too small and so most of the points from this class are misclassified.

- (Recall) The decision boundary corresponds to maximum likelihood solution under a Gaussian conditional distribution.
- However, binary target vectors clearly have a distribution that is far from Gaussian.

Fisher's Linear Discriminant

- View classification in terms of dimensionality reduction
  - Relevant to 'curse of dimensionality'
  - Computational efficiency
- Project D-dimensional input vector $\mathbf{x}$ into a lower dimension
  - Classes well-separated in the original D-dimensional space may severely overlap in a lower dimension



Find a low-dimensional space such that when $\mathbf{x}$ is projected, classes are maximally separated.

For binary $(K = 2)$ classification,

- $z = \mathbf{w}^\top \mathbf{x}$: projection of $\mathbf{x} \in \mathbb{R}^D$ onto $\mathbf{w} \in \mathbb{R}^D$
- $\mathbf{m}_i \in \mathbb{R}^D$, $m_i \in \mathbb{R}$: means of samples from C_i before and after projection

$$\begin{aligned} m_1 &= \frac{\sum_{n \in C_1} \mathbf{w}^\top \mathbf{x}_n}{N_1} = \mathbf{w}^\top \mathbf{m}_1 \\ m_2 &= \frac{\sum_{n \in C_2} \mathbf{w}^\top \mathbf{x}_n}{N_2} = \mathbf{w}^\top \mathbf{m}_2 \end{aligned}$$

- s_i^2: *scatter* of samples from C_i

$$\begin{aligned} s_1^2 &= \sum_{n \in C_1} \left(\mathbf{w}^\top \mathbf{x}_n - m_1 \right)^2 \\ s_2^2 &= \sum_{n \in C_2} \left(\mathbf{w}^\top \mathbf{x}_n - m_2 \right)^2 \end{aligned}$$

For well-separation,

- means to be as far apart as possible: large $|m_1 - m_2|$
- scattered in as small a region as possible: small $s_1^2 + s_2^2$

Fisher's linear discriminant

$$\max_{\mathbf{w}} J(\mathbf{w}) = \max_{\mathbf{w}} \frac{(m_1 - m_2)^2}{s_1^2 + s_2^2}$$

a.k.a., Fisher's ratio or F-test

$$
\begin{aligned}
(m_1 - m_2)^2 &= \left(\mathbf{w}^\top \mathbf{m}_1 - \mathbf{w}^\top \mathbf{m}_2\right)^2 \\
&= \mathbf{w}^\top (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^\top \mathbf{w} \\
&= \mathbf{w}^\top \mathbf{S}_B \mathbf{w}
\end{aligned}
$$

$\mathbf{S}_B$: *between-class scatter matrix*

$$
\begin{aligned}
s_1^2 &= \sum_{n \in C_1} \left(\mathbf{w}^\top \mathbf{x}_n - m_1\right)^2 \\
&= \sum_{n \in C_1} \left(\mathbf{w}^\top \mathbf{x}_n - \mathbf{w}^\top \mathbf{m}_1\right)^2 \\
&= \mathbf{w}^\top \left[\sum_{n \in C_1} (\mathbf{x}_n - \mathbf{m}_1)(\mathbf{x}_n - \mathbf{m}_1)^\top\right] \mathbf{w} \\
&= \mathbf{w}^\top \mathbf{S}_1 \mathbf{w}
\end{aligned}
$$

($\mathbf{S}_1$: within-class scatter matrix for C_1)

$$
\begin{aligned}
s_1^2 + s_2^2 &= \mathbf{w}^\top \mathbf{S}_1 \mathbf{w} + \mathbf{w}^\top \mathbf{S}_2 \mathbf{w} \\
&= \mathbf{w}^\top \mathbf{S}_W \mathbf{w}
\end{aligned}
$$

$\mathbf{S}_W$: *total within-class scatter matrix*

$$\max_{\mathbf{w}} J(\mathbf{w}) = \max_{\mathbf{w}} \frac{\mathbf{w}^\top \mathbf{S}_B \mathbf{w}}{\mathbf{w}^\top \mathbf{S}_W \mathbf{w}} = \max_{\mathbf{w}} \frac{|\mathbf{w}^\top (\mathbf{m}_1 - \mathbf{m}_2)|^2}{\mathbf{w}^\top \mathbf{S}_W \mathbf{w}}$$

Taking the derivative of J w.r.t. $\mathbf{w}$ and setting it equal to 0

$$\frac{\mathbf{w}^\top (\mathbf{m}_1 - \mathbf{m}_2)}{\mathbf{w}^\top \mathbf{S}_W \mathbf{w}} \left(2(\mathbf{m}_1 - \mathbf{m}_2) - \frac{\mathbf{w}^\top (\mathbf{m}_1 - \mathbf{m}_2)}{\mathbf{w}^\top \mathbf{S}_W \mathbf{w}} \mathbf{S}_W \mathbf{w} \right) = 0$$

Given that $\frac{\mathbf{w}^\top(\mathbf{m}_1 - \mathbf{m}_2)}{\mathbf{w}^\top \mathbf{S}_W \mathbf{w}}$ is a constant

$$\mathbf{w} = c\mathbf{S}_W^{-1} (\mathbf{m}_1 - \mathbf{m}_2)$$

(c : some constant, simply set $c = 1$ and find $\mathbf{w}$)

- Strictly, not a discriminant but rather a specific choice of direction for projection of the data down to one dimension
- However, the projected data can subsequently be used to construct a discriminant.

- Decision by thresholding: C_1, if $y(\mathbf{x}) \geq y_0$; C_2, otherwise
- When $p(\mathbf{x}|C_i) \sim \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma})$,
  - Linear discriminant: $\mathbf{w} = \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$
  - Fisher's linear discriminant is optimal
- Optimal threshold
  - Having found Gaussian approximations to the projected classes, we can use decision theory to find the optimal threshold

Fisher's linear discriminant can be used even when the classes are not normal.

[Multiple Classes]

For multiclass $(K > 2)$ classification,

$$\mathbf{z} = \mathbf{W}^\top \mathbf{x}$$
$$(\mathbf{z} \in \mathbb{R}^k, \mathbf{W} \in \mathbb{R}^{D\times k})$$

- Within-class scatter matrix: $\mathbf{S}_W = \sum_{i=1}^{K} \mathbf{S}_i$
  - $\mathbf{S}_i = \sum_{n\in C_i} (\mathbf{x}_n - \mathbf{m}_i)(\mathbf{x}_n - \mathbf{m}_i)^\top$
- Between-class scatter matrix: $\mathbf{S}_B = \sum_{i=1}^{K} N_i (\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^\top$
  - $\mathbf{m} = \frac{1}{K}\sum_{i=1}^{K} \mathbf{m}_i$: mean of the total data set
  - N_i: number of training samples belonging to C_i

$$\max_{\mathbf{W}} J(\mathbf{W}) = \max_{\mathbf{W}} \frac{\mathbf{W}^\top \mathbf{S}_B \mathbf{W}}{\mathbf{W}^\top \mathbf{S}_W \mathbf{W}}$$

- Maximize $\mathbf{W}^\top \mathbf{S}_B \mathbf{W}$, while minimizing $\mathbf{W}^\top \mathbf{S}_W \mathbf{W}$
- But both $\mathbf{W}^\top \mathbf{S}_B \mathbf{W}$ and $\mathbf{W}^\top \mathbf{S}_W \mathbf{W}$ are $k \times k$ matrices.

(Tip)

- For a scatter(or covariance) matrix, a measure of spread is the determinant.
- Determinant is the product of eigenvalues
- Eigenvalue gives the variance along its eigenvector (component)

$$\max_{\mathbf{W}} J(\mathbf{W}) = \max_{\mathbf{W}} \frac{|\mathbf{W}^\top \mathbf{S}_B \mathbf{W}|}{|\mathbf{W}^\top \mathbf{S}_W \mathbf{W}|}$$

- Solution: the largest eigenvectors of $\mathbf{S}_W^{-1}\mathbf{S}_B$
  - $\mathbf{S}_W$ should be invertible
  - Otherwise, first use PCA to get rid of singularity
    - Make sure that PCA does not reduce dimensionality so much that LDA does not have anything left to work on

- $\mathbf{S}_B$: sum of K matrices of rank 1, i.e., $(\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^\top$
- Only $K-1$ of them are independent
- $\mathbf{S}_B$ has a maximum rank of $K-1 \Rightarrow k = K-1$

- LDA uses class separability as its goodness criterion
- After projection to the LDA space (i.e., dimension reduction), any classification method can be used.
  - LDA is a method for dimensionality reduction
  - NOT a classifier!!!

Perceptron

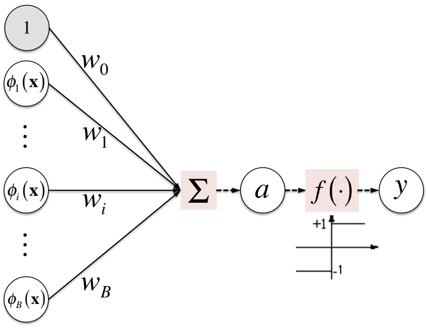
[Rosenblatt, 1958]

- Two-class model
  - An input vector $\mathbf{x}$ is first transformed using a fixed nonlinear transformation to give a feature vector $\phi(\mathbf{x})$
  - Then used to construct a generalized linear model

$$y(\mathbf{x}) = f\left(\mathbf{w}^\top \phi(\mathbf{x})\right)$$

$$\text{where } f(a) = \begin{cases} +1 & a \geq 0 \\ -1 & a < 0 \end{cases}$$

- Use a target coding scheme
  - $t = +1$ for class C_1, and $t = -1$ for C_2
  - Matching the choice of activation function



[Parameters Learning]

Error function minimization

- Error function: number of misclassifications
- This error function is a piecewise constant function of $\mathbf{w}$ with discontinuities (c.f., regression)
- No closed-form solution (no derivatives exist for non-smooth functions)
- Take an iterative approach

Perceptron Criterion

- Seeking $\mathbf{w}$ such that

$$\left\{ \begin{array}{l} \mathbf{x}_n \in C_1 \ (t_n = +1) \text{ will have } \mathbf{w}^\top \phi(\mathbf{x}_n) \geq 0 \\ \mathbf{x}_n \in C_2 \ (t_n = -1) \text{ will have } \mathbf{w}^\top \phi(\mathbf{x}_n) < 0 \end{array} \right\} \Rightarrow \mathbf{w}^\top \phi(\mathbf{x}_n)\, t_n \geq 0$$

  - Linearly bisecting the feature space

- For each misclassified sample, perceptron criterion tries to minimize

$$E_P(\mathbf{w}) = -\sum_{n \in M} \mathbf{w}^\top \phi(\mathbf{x}_n)\, t_n$$

M: a set of all misclassified samples

Perceptron Algorithm

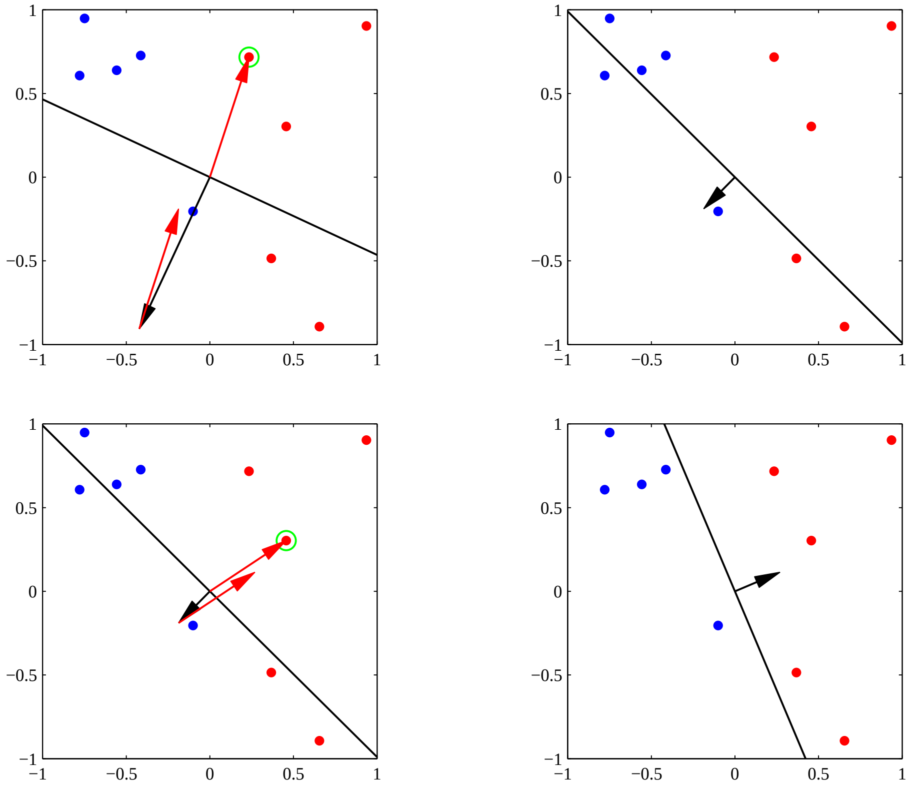
- Error function: $E_P(\mathbf{w}) = -\sum_{n \in M} \mathbf{w}^\top \phi(\mathbf{x}_n)\, t_n$
- Stochastic gradient descent

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E_P(\mathbf{w}) = \mathbf{w}^{(\tau)} + \eta \phi(\mathbf{x}_n)\, t_n$$

η: learning rate, τ: step index

  - Since $y(\mathbf{x}, \mathbf{w})$ is unchanged if we multiply $\mathbf{w}$ by a constant, we can set η equal to 1 without loss of generality.

- Interpretation: cycle through the training samples in turn
  - If misclassified, for class C_1 add $\phi(\mathbf{x}_n)$ to $\mathbf{w}$
  - If misclassified, for class C_2 subtract $\phi(\mathbf{x}_n)$ from $\mathbf{w}$



Black arrow: $\mathbf{w}$ (points towards the decision region of the red class), green point: misclassified

Effect of a single update: reduce the error from a misclassified sample

$$
\begin{aligned}
-\left(\mathbf{w}^{(\tau+1)}\right)^{\top} \phi\left(\mathbf{x}_n\right) t_n &= -\left(\mathbf{w}^{(\tau)}\right)^{\top} \phi\left(\mathbf{x}_n\right) t_n - \underbrace{\left(\phi\left(\mathbf{x}_n\right) t_n\right)^{\top} \phi\left(\mathbf{x}_n\right) t_n}_{\|\phi(\mathbf{x}_n) t_n\|^2 > 0} \\
&< -\left(\mathbf{w}^{(\tau)}\right)^{\top} \phi\left(\mathbf{x}_n\right) t_n
\end{aligned}
$$

- Not imply that the contribution to the error function from the other misclassified samples will have been reduced
- No guarantee to reduce the total error function at each stage

If there exists an exact solution,
it is guaranteed to find it in a finite number of steps.

Disadvantages of Perceptrons

- Not converge if classes are not linearly separable
- Not provide probabilistic output
- Not readily generalized to $K > 2$ classes
- Based on linear combinations of fixed basis functions

Probabilistic Generative Models

(Recap.) Generative vs. Discriminative

Generative models (2-step)

1. Infer class-conditional densities $p\left(\mathbf{x}|C_k\right)$ and priors $P\left(C_k\right)$
2. Use a Bayes rule to determine posterior probabilities

$$P\left(C_k|\mathbf{x}\right) = \frac{p\left(\mathbf{x}|C_k\right)P\left(C_k\right)}{p\left(\mathbf{x}\right)}$$

Discriminative models (1-step)

- Directly infer posterior probabilities $P\left(C_k|\mathbf{x}\right)$

In both cases, use a decision theory to assign each new $\mathbf{x}$ to a class

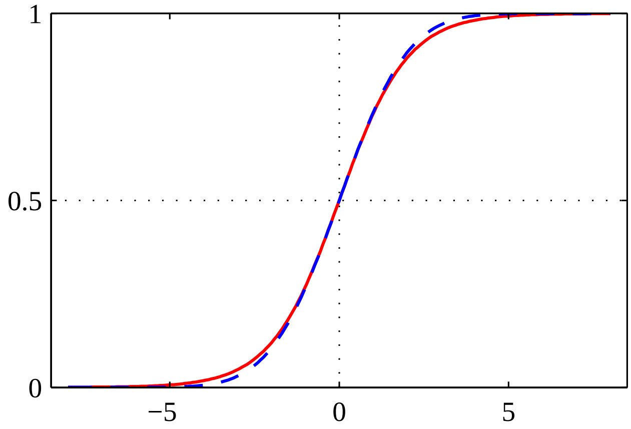
Posterior for class C_1 (in binary classification)

$$\begin{aligned} P\left(C_1|\mathbf{x}\right) &= \frac{p\left(\mathbf{x}|C_1\right)P\left(C_1\right)}{p\left(\mathbf{x}|C_1\right)P\left(C_1\right)+p\left(\mathbf{x}|C_2\right)P\left(C_2\right)} \\ &= \frac{1}{1+\exp\left(-a\right)} = \sigma\left(a\right) \end{aligned}$$

where $a = \ln\frac{p\left(\mathbf{x}|C_1\right)P\left(C_1\right)}{p\left(\mathbf{x}|C_2\right)P\left(C_2\right)}$

Logistic Sigmoid Function

- Sigmoid: "S"-shaped, squashing real axis into a finite interval
  - Maps real $a \in (-\infty, \infty)$ to a finite interval of $(0, 1)$



$\sigma(a) = \frac{1}{1+\exp(-a)}$

$\sigma(-a) = 1 - \sigma(a)$

$\frac{\partial\sigma}{\partial a} = \sigma(1-\sigma)$

$a = \ln\left(\frac{\sigma}{1-\sigma}\right)$ known as *logit* (inverse of the logistic sigmoid; log of the ratio of probabilities)

The dashed blue line is a scaled probit function (cdf of a zero-mean unit variance Gaussian).

Softmax Function

Generalization of a logistic sigmoid function for $K > 2$

$$P(C_k|\mathbf{x}) = \frac{p(\mathbf{x}|C_k)P(C_k)}{\sum_j p(\mathbf{x}|C_j)P(C_j)} = \frac{\exp(a_k)}{\sum_j \exp(a_j)}$$

where $a_k = \ln p(\mathbf{x}|C_k)P(C_k)$

- Softmax: smoothed version of the 'max' function
  - If $a_k \gg a_j$ for all $j \neq k$, then $p(C_k|\mathbf{x}) \simeq 1$ and $p(C_j|\mathbf{x}) \simeq 0$

Generative Model with Continuous Inputs

- Gaussian class-conditional densities with the same covariance

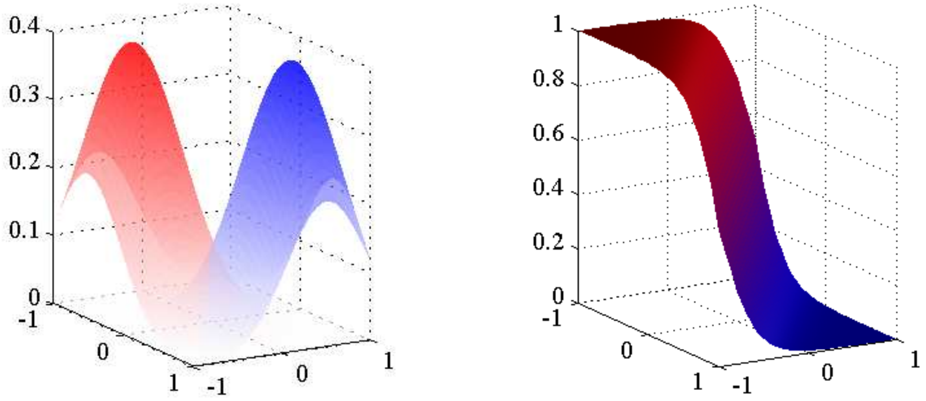
$$p(\mathbf{x}|C_k) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_k)^\top \Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu}_k)\right]$$

- Two-class case

$$P(C_1|\mathbf{x}) = \sigma\left(\ln \frac{p\left(\mathbf{x}|C_1\right)p\left(C_1\right)}{p\left(\mathbf{x}|C_2\right)p\left(C_2\right)}\right) = \sigma\left(\mathbf{w}^\top\mathbf{x} + w_0\right)$$

$$\text{[Quiz!!!]}\begin{cases}\mathbf{w} & =?\\ w_0 & =?\end{cases}$$

$p\left(\mathbf{x}|C_1\right); \quad p\left(\mathbf{x}|C_2\right)$ $\qquad$ $P\left(C_1|\mathbf{x}\right)$



The prior $P\left(C_k\right)$ enter only through the bias parameter:
$w_0 = -\frac{1}{2}\boldsymbol{\mu}_1^\top\Sigma^{-1}\boldsymbol{\mu}_1 + \frac{1}{2}\boldsymbol{\mu}_2^\top\Sigma^{-1}\boldsymbol{\mu}_2 + \ln\frac{P(C_1)}{P(C_2)}$

Changes in priors have the effect of

- making parallel shifts of the decision boundary
- more generally of the parallel contours of constant posterior probability

Continuous case with $K > 2$

$$P(C_k|\mathbf{x}) = \frac{p(\mathbf{x}|C_k)\,P(C_k)}{\sum_j p(\mathbf{x}|C_j)\,P(C_j)} = \frac{\exp(a_k)}{\sum_j \exp(a_j)}$$

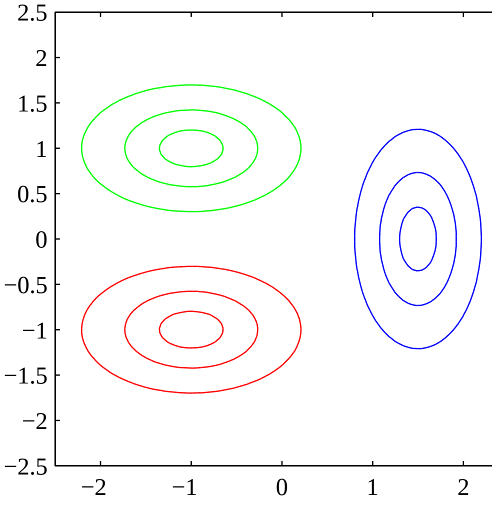
- Gaussian class-conditionals

$$a_k(\mathbf{x}) = \mathbf{w}_k^\top \mathbf{x} + w_{k0}$$

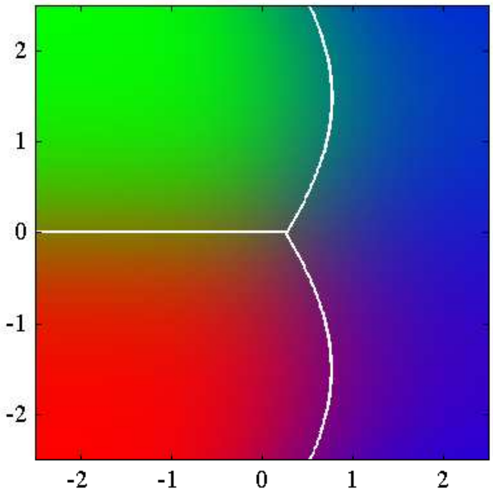
$$\begin{aligned}\mathbf{w}_k &= \Sigma^{-1}\boldsymbol{\mu}_k \\ w_{k0} &= -\tfrac{1}{2}\boldsymbol{\mu}_k^\top \Sigma^{-1}\boldsymbol{\mu}_k + \ln P(C_k)\end{aligned}$$

When relaxing the assumption of a shared covariance (*i.e.*, each class-conditional density has its own covariance), we get a quadratic discriminant.

Class-conditional densities



Posterior probabilities



Maximum Likelihood Solution

Given $\{\mathbf{x}_n, t_n\}_{n=1}^{N}$, where $t_n = \begin{cases} 1 & \text{, for } C_1 \\ 0 & \text{, for } C_2 \end{cases}$, let $P(C_1) = \pi$

$$
\begin{aligned}
p(\mathbf{x}, C_1) &= P(C_1)\, p(\mathbf{x}|C_1) = \pi \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_1, \Sigma) \\
p(\mathbf{x}, C_2) &= P(C_2)\, p(\mathbf{x}|C_2) = (1-\pi)\, \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_2, \Sigma)
\end{aligned}
$$

Likelihood:

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

where $\mathbf{t} = [t_1, \ldots, t_N]^\top$

[For convenience, maximize the log of the likelihood function]

$$
\sum_{n=1}^{N} t_n \ln [\pi \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_1, \Sigma)] + (1-t_n) \ln [(1-\pi)\, \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_2, \Sigma)]
$$

Estimates for π:

- Terms dependent on π

$$
\sum_{n=1}^{N} \{t_n \ln \pi + (1-t_n) \ln (1-\pi)\}
$$

- Taking derivatives w.r.t. π and setting equal to zero

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

  - MLE for π is the fraction of the sample points

Estimates for $\boldsymbol{\mu}_1/\boldsymbol{\mu}_2$:

- Terms dependent on $\boldsymbol{\mu}_1/\boldsymbol{\mu}_2$

$$\sum_{n=1}^{N} t_n \mathcal{N}\left(\mathbf{x}_n|\boldsymbol{\mu}_1, \Sigma\right) = \frac{1}{2}\sum_{n=1}^{N} t_n \left(\mathbf{x}_n - \boldsymbol{\mu}_1\right)^{\top} \Sigma^{-1} \left(\mathbf{x}_n - \boldsymbol{\mu}_1\right) + \text{const.}$$

$$\sum_{n=1}^{N} \left(1 - t_n\right) \mathcal{N}\left(\mathbf{x}_n|\boldsymbol{\mu}_2, \Sigma\right) = \frac{1}{2}\sum_{n=1}^{N} \left(1 - t_n\right) \left(\mathbf{x}_n - \boldsymbol{\mu}_2\right)^{\top} \Sigma^{-1} \left(\mathbf{x}_n - \boldsymbol{\mu}_2\right) + \text{const.}$$

- Taking derivatives w.r.t. $\boldsymbol{\mu}_1/\boldsymbol{\mu}_2$ and setting equal to zero

$$\boldsymbol{\mu}_1 = \frac{1}{N_1}\sum_{n=1}^{N} t_n \mathbf{x}_n \qquad \boldsymbol{\mu}_2 = \frac{1}{N_2}\sum_{n=1}^{N} \left(1 - t_n\right) \mathbf{x}_n$$

  - MLE for $\boldsymbol{\mu}_1/\boldsymbol{\mu}_2$ is the mean of the samples assigned to class C_1/C_2

Estimates for Σ:

- Terms dependent on Σ

$$-\frac{1}{2}\sum_{n=1}^{N} t_n \ln|\Sigma| - \frac{1}{2}\sum_{n=1}^{N} t_n \left(\mathbf{x}_n - \boldsymbol{\mu}_1\right)^{\top} \Sigma^{-1} \left(\mathbf{x}_n - \boldsymbol{\mu}_1\right)$$

$$-\frac{1}{2}\sum_{n=1}^{N} \left(1 - t_n\right) \ln|\Sigma| - \frac{1}{2}\sum_{n=1}^{N} \left(1 - t_n\right) \left(\mathbf{x}_n - \boldsymbol{\mu}_2\right)^{\top} \Sigma^{-1} \left(\mathbf{x}_n - \boldsymbol{\mu}_2\right)$$

$$= -\frac{N}{2}\ln|\Sigma| - \frac{N}{2}\text{Tr}\left\{\Sigma^{-1}\mathbf{S}\right\}$$

$$\text{where} \begin{cases} \mathbf{S} = \frac{N_1}{N}\mathbf{S}_1 + \frac{N_2}{N}\mathbf{S}_2 \\ \mathbf{S}_1 = \frac{1}{N_1}\sum_{N\in C_1} \left(\mathbf{x}_n - \boldsymbol{\mu}_1\right)\left(\mathbf{x}_n - \boldsymbol{\mu}_1\right)^{\top} \\ \mathbf{S}_2 = \frac{1}{N_2}\sum_{N\in C_2} \left(\mathbf{x}_n - \boldsymbol{\mu}_2\right)\left(\mathbf{x}_n - \boldsymbol{\mu}_2\right)^{\top} \end{cases}$$

- Taking derivatives w.r.t. Σ and setting equal to zero

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

  - MLE for Σ is the weighted average of the covariance matrices associated with each of the classes separately.

Discrete Features

- Assuming binary features $x_i \in \{0, 1\}$
- With D inputs, distribution is a table of 2^D numbers for each class
- Naïve Bayes assumption: independence among features
  - Class-conditional distributions

$$p(\mathbf{x}|C_k) = \prod_{i=1}^{D} \mu_{k_i}^{x_i} (1 - \mu_{k_i})^{1-x_i}$$

  - Substituting in the form needed for normalized exponential

$$\begin{aligned} a_k(\mathbf{x}) &= \ln [p(\mathbf{x}|C_k) p(C_k)] \\ &= \sum_{i=1}^{D} [x_i \ln \mu_{k_i} + (1 - x_i) \ln (1 - \mu_{k_i})] + \ln p(C_k) \end{aligned}$$

[Exercise 4.11]

- K classes for which the feature vector ϕ has M components each of which can take L discrete states
- Naïve Bayes assumption
  - Class-conditional distributions

$$\begin{aligned} p(\phi|C_k) &= \prod_{i=1}^{M} p(\phi_m|C_k) \\ p(\phi_m|C_k) &= \prod_{l=1}^{L} \mu_{kml}^{\phi_{ml}} \end{aligned}$$

  - Substituting in the form needed for normalized exponential

$$\begin{aligned} a_k(\mathbf{x}) &= \ln [p(\phi|C_k) p(C_k)] \\ &= \ln p(C_k) + \sum_{m=1}^{M} \ln p(\phi_m|C_k) \\ &= \ln p(C_k) + \sum_{m=1}^{M} \sum_{l=1}^{L} \phi_{ml} \ln \mu_{kml} \end{aligned}$$

Exponential Family

- Class-conditional that belong to the exponential family

$$p\left(\mathbf{x}|\boldsymbol{\lambda}_k\right) = h\left(\mathbf{x}\right) g\left(\boldsymbol{\lambda}_k\right) \exp\left[\boldsymbol{\lambda}_k^\top u\left(\mathbf{x}\right)\right]$$

- For $K \geq 2$ and $u\left(\mathbf{x}\right) = \mathbf{x}$

$$a_k\left(\mathbf{x}\right) = \boldsymbol{\lambda}_k^\top \mathbf{x} + \ln g\left(\boldsymbol{\lambda}_k\right) + \ln p\left(C_k\right)$$

Interim Summary

Estimating posterior probability

- Two-class: a *logistic sigmoid* acting on a linear function of $\mathbf{x}$
- Multiclass: a *softmax* transformation of a linear function of $\mathbf{x}$

- A wide choice of class-conditional distributions $p\left(\mathbf{x}|C_k\right)$
- For a specific choice of the class-conditional densities
  - Maximum likelihood to determine the parameters of the densities as well as the class priors $p\left(C_k\right)$ and then Bayes' theorem applied

Implicitly determining the parameters $\{\mathrm{w}_k\}$

Probabilistic Discriminative Models

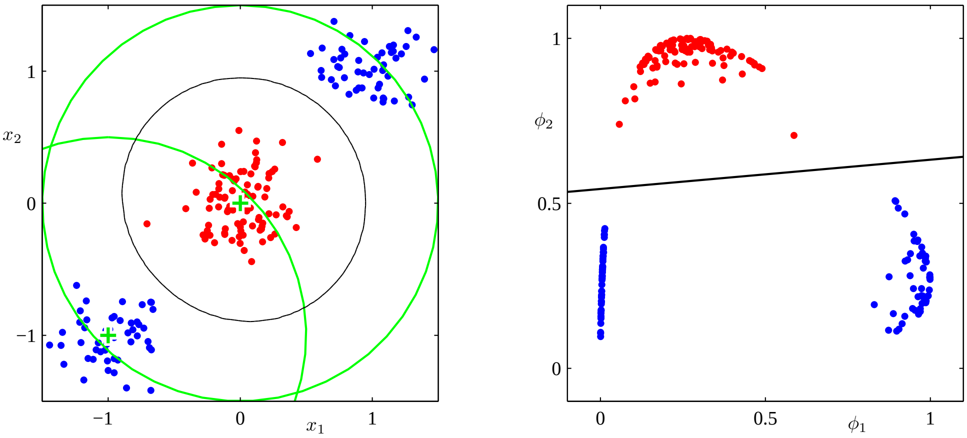
Alternative approach: '*discriminative*'

- To use the functional form of the generalized linear model explicitly
- To determine its parameters directly by using maximum likelihood
  - Efficient algorithm: *Iterative Reweighted Least Squares* (IRLS)
- Advantages
  - Typically fewer adaptive parameters to be determined
  - Improved performance when $p\left(\mathbf{x}|C_k\right)$ assumptions are poor approximations to the true distributions

Fixed Basis Functions

Replacement of the input vector $\mathbf{x}$ with a vector of basis functions $\phi(\mathbf{x})$

- Non-linear basis functions
  - Linearly separable in the feature space ϕ
  - Possibly non-linear decision boundaries in the original space



Basis functions: Gaussian with centers/contours shown by green crosses and contours

(Two-Class) Logistic Regression

- (From generative model) Posterior probability of class C_1: a logistic sigmoid acting on a linear function of the feature vector ϕ

$$P(C_1|\phi) = y(\phi) = \sigma\left(\mathbf{w}^\top \phi\right) \qquad P(C_2|\phi) = 1 - P(C_1|\phi)$$

  - $\sigma(\cdot)$: logistic sigmoid function

- In the terminology of statistics, known as '*logistic regression*'
  - This is a model for classification rather than regression.

For M-dimensional feature space ϕ

- M adjustable parameters
- *c.f.*, Generative with Gaussians: $M(M+5)/2+1$ growing quadratically
  - $2M$ parameters for means
  - $M(M+1)/2$ parameters for a shared covariance matrix
  - 2 parameters for class priors

[Determining Parameters of Logistic Regression]

For a dataset $\{\phi_n = \phi(\mathbf{x}_n), t_n\}$, where $t_n \in \{0,1\}$ with $n = 1, \ldots, N$

- Likelihood function

$$p(\mathbf{t}|\mathbf{w}) = \prod_{n=1}^{N} y_n^{t_n} \{1 - y_n\}^{1-t_n}$$

  - $y_n = P(C_1|\phi_n) = \sigma(a_n)$, where $a_n = \mathbf{w}^\top \phi_n$
  - $\mathbf{t} = (t_1, \ldots, t_N)^\top$

- Taking negative logarithm → *cross-entropy error function*

$$E(\mathbf{w}) = -\ln p(\mathbf{t}|\mathbf{w}) = -\sum_{n=1}^{N} \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\}$$

- Taking the gradient of the error function w.r.t. $\mathbf{w}$ [Quiz!!!]

$$\nabla E(\mathbf{w}) = ?$$

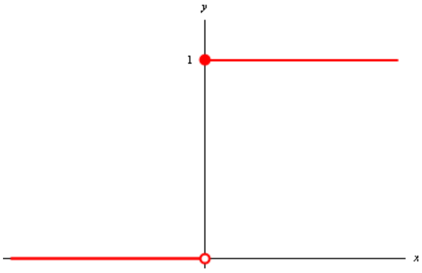
$$\text{(Tip)}\ \frac{\partial \sigma}{\partial a} = \sigma(1-\sigma)$$

- Taking the gradient of the error function w.r.t. $\mathbf{w}$

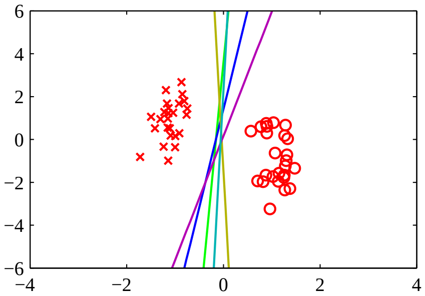
$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N} \underbrace{(y_n - t_n)}_{\text{error}} \phi_n$$

  - Contribution to gradient by data point n is given by an error between a target t_n and the prediction y_n times basis ϕ_n
  - The same form as the gradient of the sum-of-squares error function for the linear regression model

- However, the maximum likelihood solution has severe over-fitting problems for linearly separable data.
  - Logistic sigmoid function becomes a Heaviside step function
    - Every training point from each class k is assigned a posterior probability $P(C_k|\mathbf{x}) = 1$
  - There is typically a continuum of such solutions because any separating hyperplane will give rise to the same posterior probabilities at the training data points.



Heaviside step function



Continuum of solutions

- This singularity can be avoided by
  - inclusion of a prior and finding a MAP solution for $\mathbf{w}$
  - or equivalently by adding a regularization term to the error function

- In linear regression, the maximum likelihood solution, on the assumption of a *Gaussian noise model*, leads to a closed-form solution.
    - Due to a consequence of the quadratic dependence of the log likelihood on the parameter vector $\mathbf{w}$

$$\nabla_{\mathbf{w}}^{R} \ln p\left(\mathbf{t}|\mathbf{w},\beta\right) \Rightarrow \mathbf{w}_{ML}^{R} = \left(\Phi^{\top}\Phi + \lambda I\right)^{-1} \Phi^{\top}\mathbf{t}$$

- For logistic regression, there is no closed-form maximum likelihood solution.
    - Due to the *nonlinearity* of the logistic sigmoid function

$$\nabla E\left(\mathbf{w}\right) = \sum_{n=1}^{N}\left(y_n - t_n\right)\phi_n = \sum_{n=1}^{N}\nabla E_n$$

- Iterative approach

$$\mathbf{w}^{(\text{new})} = \mathbf{w}^{(\text{old})} - \eta\nabla E\left(\mathbf{w}\right)$$

- Alternative sequential iteration

$$\mathbf{w}^{(\text{new})} = \mathbf{w}^{(\text{old})} - \eta\nabla E_n\left(\mathbf{w}\right)$$

Iterative Reweighted Least Squares (IRLS)

- Efficient iterative technique based on the *Newton-Raphson* iterative optimization scheme
  - Using local quadratic approximation to the log likelihood function

$$\mathbf{w}^{(\text{new})} = \mathbf{w}^{(\text{old})} - \mathbf{H}^{-1}\nabla E(\mathbf{w})$$

  - $\mathbf{H}$: Hessian matrix whose elements are the second derivatives of $E(\mathbf{w})$ w.r.t. the components of $\mathbf{w}$

$$\nabla\nabla E(\mathbf{w})$$

▸ Newton-Raphson

e.g.) Newton-Raphson method to the linear regression with the sum-of-squares error function

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N}\left(\mathbf{w}^\top\phi_n - t_n\right)\phi_n = \Phi^\top\Phi\mathbf{w} - \Phi^\top\mathbf{t}$$

$$\nabla\nabla E(\mathbf{w}) = \sum_{n=1}^{N}\phi_n\phi_n^\top = \Phi^\top\Phi$$

  - Φ: $N \times M$ design matrix, whose n^{th} row is given by ϕ_n

- Newton-Raphson update for linear regression

$$\begin{aligned}\mathbf{w}^{(\text{new})} &= \mathbf{w}^{(\text{old})} - \left(\Phi^\top\Phi\right)^{-1}\left\{\Phi^\top\Phi\mathbf{w}^{(\text{old})} - \Phi^\top\mathbf{t}\right\}\\ &= \left(\Phi^\top\Phi\right)^{-1}\Phi^\top\mathbf{t}\end{aligned}$$

  - Same as the standard least-squares solution due to the quadratic form in the error function

- Newton-Raphson method to the cross-entropy error function for the logistic regression model

$$\nabla E\left(\mathbf{w}\right) = \sum_{n=1}^{N} \left(y_n - t_n\right) \phi_n = \Phi^\top \left(\mathbf{y} - \mathbf{t}\right)$$

$$\mathbf{H} = \nabla\nabla E\left(\mathbf{w}\right) = \sum_{n=1}^{N} y_n \left(1 - y_n\right) \phi_n \phi_n^\top = \Phi^\top \mathbf{R} \Phi$$

  - $\mathbf{R}$: $N \times N$ diagonal matrix with $R_{nn} = y_n \left(1 - y_n\right)$, $0 < y_n < 1$
  - $\mathbf{H}$ is no longer constant, but depends on $\mathbf{w}$ through the matrix $\mathbf{R}$, corresponding to the fact that the error function is not quadratic any longer.

- Newton-Raphson update for logistic regression

$$\begin{aligned}\mathbf{w}^{(\text{new})} &= \mathbf{w}^{(\text{old})} - \left(\Phi^\top \mathbf{R} \Phi\right)^{-1} \Phi^\top \left(\mathbf{y} - \mathbf{t}\right) \\ &= \left(\Phi^\top \mathbf{R} \Phi\right)^{-1} \left\{\Phi^\top \mathbf{R} \Phi \mathbf{w}^{(\text{old})} - \Phi^\top \left(\mathbf{y} - \mathbf{t}\right)\right\} \\ &= \left(\Phi^\top \mathbf{R} \Phi\right)^{-1} \Phi^\top \mathbf{R} \mathbf{z}\end{aligned}$$

where $\mathbf{z} = \Phi^\top \mathbf{w}^{(\text{old})} - \mathbf{R}^{-1} \left(\mathbf{y} - \mathbf{t}\right)$

- $\left(\Phi^\top \mathbf{R} \Phi\right)^{-1} \Phi^\top \mathbf{R} \mathbf{z}$: a form of a set of normal equations for a weighted least-squares problem
  - Because of $\mathbf{R}$ (not constant but depends on $\mathbf{w}$), we must apply the normal equations iteratively, each time using the new weight vector $\mathbf{w}$.
  - Since $\mathbf{H}$ is positive-definite (*i.e.*, for arbitrary $\mathbf{u}$, $\mathbf{u}^\top \mathbf{H} \mathbf{u} > 0$), the error function is a convex function of $\mathbf{w}$ and so has a unique solution.

Iterative Reweighted Least Squares (IRLS)

Multiclass Logistic Regression

Work with a softmax function instead of logistic sigmoid

$$P\left(C_k|\mathbf{x}\right) = y_k\left(\phi\right) = \frac{\exp\left(a_k\right)}{\sum_j \exp\left(a_j\right)} \quad \text{where } a_k = \mathbf{w}_k^\top \phi$$

- Likelihood function

$$p\left(\mathbf{T}|\mathbf{w}_1, \ldots, \mathbf{w}_K\right) = \prod_{n=1}^{N} \prod_{k=1}^{K} P\left(C_k|\mathbf{x}_n\right)^{t_{nk}} = \prod_{n=1}^{N} \prod_{k=1}^{K} y_{nk}^{t_{nk}}$$

  - $\mathbf{t}_n \in \{0,1\}^K$: 1-of-$K$ coding scheme
  - $y_{nk} = y_k\left(\phi_n\right)$, $\mathbf{T} = [t_{nk}]$: $N \times K$ matrix of target variables

- Taking negative logarithm → *cross-entropy error function*

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

Machine
Intelligence
Lab

[Determining Parameters of Multiclass Logistic Regression]

- Taking the gradient of the error function w.r.t. $\mathbf{w}_j$ [Quiz!!!]

$$\nabla_{\mathbf{w}_j} E\left(\mathbf{w}_1, \ldots, \mathbf{w}_K\right) = ?$$

$$\text{(Tip)} \quad \frac{\partial}{\partial x} \frac{f\left(x\right)}{g\left(x\right)} = \frac{f'\left(x\right) g\left(x\right) - f\left(x\right) g'\left(x\right)}{g\left(x\right)^2}$$

Machine
Intelligence
Lab

[Determining Parameters of Multiclass Logistic Regression]

- Taking the gradient of the error function w.r.t. $\mathbf{w}_j$

$$\nabla_{\mathbf{w}_j} E\left(\mathbf{w}_1, \ldots, \mathbf{w}_K\right) = \sum_{n=1}^{N} \underbrace{\left(y_{nj} - t_{nj}\right)}_{\text{error}} \phi_n$$

  - Contribution to gradient by data point n is given by an error between a target t_{nj} and prediction y_{nj} times basis ϕ_n

- Making use of it to give a sequential algorithm in which patterns are presented one at a time

$$\mathbf{w}_j^{(\text{new})} = \mathbf{w}_j^{(\text{old})} - \eta \nabla_j E_n \left(\mathbf{w}_1, \ldots, \mathbf{w}_K\right)$$

  - Update should be conducted for all parameters $\{\mathbf{w}_j\}$ simultaneously

IRLS for Multiclass Logistic Regression

- Newton-Raphson update for multiclass logistic regression

$$\mathbf{H} = \nabla_{\mathbf{w}_k} \nabla_{\mathbf{w}_j} E\left(\mathbf{w}_1, \ldots, \mathbf{w}_K\right) = \sum_{n=1}^{N} y_{nk} \left(I_{kj} - y_{nj}\right) \phi_n \phi_n^{\top}$$

  - Again, $\mathbf{H}$ is positive-definite and so the error function has a unique minimum.

Laplace Approximation

Bayesian Inference

- **Model fitting**: given a training data $\mathcal{D}$, inferring the probability distribution over parameters θ in model $\mathcal{M}$

$$p\left(\theta|\mathcal{D},\mathcal{M}\right)=\frac{p\left(\mathcal{D}|\theta,\mathcal{M}\right)p\left(\theta|\mathcal{M}\right)}{p\left(\mathcal{D}|\mathcal{M}\right)}$$

- **Prediction**: given an input $\mathbf{x}$, inferring the probability distribution over outputs y

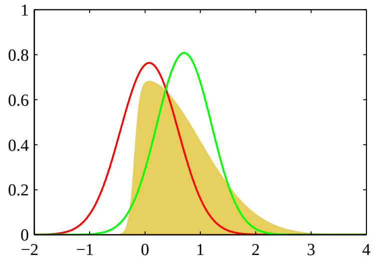
$$p\left(y|\mathbf{x},\mathcal{D},\mathcal{M}\right)=\int p\left(y|\mathbf{x},\theta,\mathcal{D},\mathcal{M}\right)p\left(\theta|\mathcal{D},\mathcal{M}\right)d\theta$$

- **Model comparison**

$$p\left(\mathcal{M}|\mathcal{D}\right)=\frac{p\left(\mathcal{D}|\mathcal{M}\right)p\left(\mathcal{M}\right)}{p\left(\mathcal{D}\right)}$$

Approximate Inference

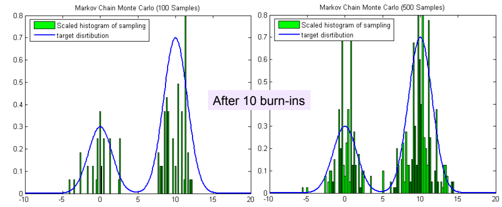
- **Analytical approximation**
  - Laplace approximation: local Gaussian approximation
  - Variational inference: global approximation



- **Numerical sampling**
  - Markov Chain Monte-Carlo (MCMC): *e.g.*, Metropolis-Hastings, Gibbs
  - Sequential importance sampling (*a.k.a.*, particle filtering)

target distribution: $p(x) \sim 0.3\exp(-0.2x^2) + 0.7\exp(-0.2(x-10)^2)$

proposal distribution: $q(x^c \mid x) \sim N(x^{(i)}, 10)$



After 10 burn-ins

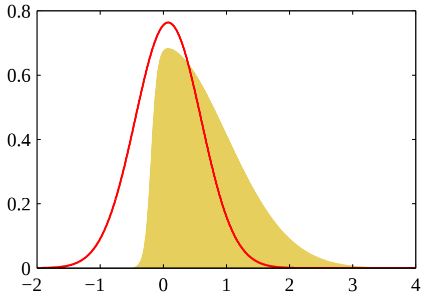
Laplace Approximation

Aims to find a Gaussian approximation $q(z)$ to a probability density defined over a set of continuous variables

One-dimensional case: a single continuous variable z with a distribution $p(z)$

$$p(z) = \frac{1}{Z} f(z)$$

$$\text{where } Z = \int f(x)dz\text{: normalization coefficient}$$



- Shaded yellow region: $p(z)$
- Red: $q(z)$

- First find a mode of $p(z)$, *i.e.*, a point z_0 such that

$$p'(z_0) = \left.\frac{df(z)}{dz}\right|_{z=z_0} = 0$$

- Taylor expansion of $\ln f(z)$ centered on the mode z_0 ▸ Taylor Series
  - Logarithm of a Gaussian distribution: a quadratic function of variables

$$\begin{aligned}\ln f(z) &\simeq \ln f(z_0) + \underbrace{\left(\frac{d}{dz}\ln f(z)\Big|_{z=z_0}\right)}_{=0}(z-z_0) + \frac{1}{2}\underbrace{\left(\frac{d^2}{dz^2}\ln f(z)\Big|_{z=z_0}\right)}_{\equiv -A}(z-z_0)^2 \\ &= \ln f(z_0) - \frac{1}{2}A(z-z_0)^2\end{aligned}$$

- Taking exponential

$$f(z) \simeq f(z_0)\exp\left\{-\frac{A}{2}(z-z_0)^2\right\}$$

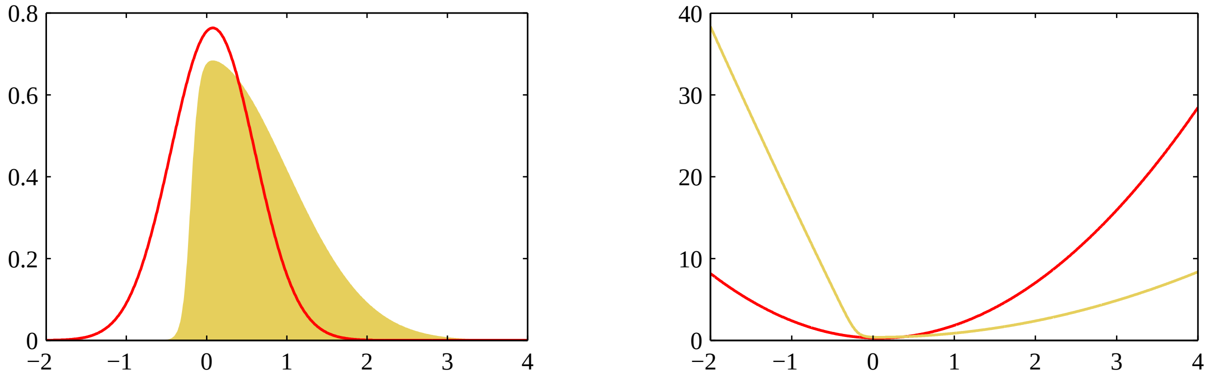
- Obtain a normalized distribution $q(z)$ by making use of the standard result for the normalization of a Gaussian

$$q(z) = \frac{1}{Z}f(z) = \left(\frac{A}{2\pi}\right)^{1/2}\exp\left\{-\frac{A}{2}(z-z_0)^2\right\} \sim \mathcal{N}\left(z|z_0, A^{-1}\right)$$

where $Z = \int f(z)dz = f(z_0)\int \exp\left\{-\frac{A}{2}(z-z_0)^2\right\}dz = f(z_0)\frac{(2\pi)^{1/2}}{A^{1/2}}$

  - Gaussian approximation will only be well defined if its precision $A > 0$
  - *i.e.*, the stationary point z_0 must be a local maximum, so that the second derivative of $f(z)$ at the point z_0 is negative.

Applied to the distribution of $p(z) \propto \exp(-z^2/2)\sigma(20z+4)$
where $\sigma(z)$: logistic sigmoid function



- (Left) normalized distribution $p(z)$ in yellow, together with the Laplace approximation centred on the mode z_0 of $p(z)$ in red
- (Right) the negative logarithms of the corresponding curves

Over M-dimensional space $\mathbf{z}$

$$p(\mathbf{z}) = \frac{1}{Z} f(\mathbf{z})$$

- First find a mode of $p(\mathbf{z})$

$$p'(\mathbf{z}_0) = \nabla f(\mathbf{z})|_{\mathbf{z}=\mathbf{z}_0} = 0$$

- Approximate $f(z)$ using second derivative

$$\begin{aligned} \ln f(\mathbf{z}) &\simeq \ln f(\mathbf{z}_0) + \underbrace{\left(\nabla \ln f(\mathbf{z})|_{\mathbf{z}=\mathbf{z}_0}\right)}_{=0} (\mathbf{z}-\mathbf{z}_0) + \frac{1}{2} \underbrace{\left(\nabla^2 \ln f(\mathbf{z})|_{\mathbf{z}=\mathbf{z}_0}\right)}_{\equiv -\mathbf{A}} (\mathbf{z}-\mathbf{z}_0)^2 \\ &= \ln f(\mathbf{z}_0) - \frac{1}{2} (\mathbf{z}-\mathbf{z}_0)^\top \mathbf{A} (\mathbf{z}-\mathbf{z}_0) \end{aligned}$$

- Taking exponential

$$f(\mathbf{z}) \simeq f(\mathbf{z}_0) \exp\left\{-\frac{1}{2}\left(\mathbf{z}-\mathbf{z}_0\right)^{\top} \mathbf{A}\left(\mathbf{z}-\mathbf{z}_0\right)\right\}$$

- Obtain a normalized distribution $q(z)$ by making use of the standard result for the normalization of a Gaussian

$$q(\mathbf{z}) = \frac{1}{Z} f(\mathbf{z}) = \frac{|A|^{1/2}}{(2\pi)^{M/2}} \exp\left\{-\frac{1}{2}\left(\mathbf{z}-\mathbf{z}_0\right)^{\top} A\left(\mathbf{z}-\mathbf{z}_0\right)\right\} \sim \mathcal{N}\left(\mathbf{z}|\mathbf{z}_0, \mathbf{A}^{-1}\right)$$

where $Z = \int f(\mathbf{z}) d\mathbf{z} = f(\mathbf{z}_0) \int \exp\left\{-\frac{1}{2}\left(z-z_0\right)^{\top} \mathbf{A}\left(z-z_0\right)\right\} d\mathbf{z} = f(\mathbf{z}_0) \frac{(2\pi)^{M/2}}{|A|^{1/2}}$

  - Gaussian approximation will only be well defined if its precision matrix $\mathbf{A}$ is positive definite
  - *i.e.*, the stationary point z_0 must be a local maximum, not a minimum or saddle point

Interim Summary

Applying Laplace approximation

- Find the mode $\mathbf{z}_0$ and then evaluate the Hessian matrix at that mode
  - Mode: typically found by running some form of numerical optimization algorithm
- Multimodal: different Laplace approximations according to which mode is being considered
- Most useful in situations where the number of data points is relatively large
  - Better approximated by a Gaussian as the number of observed data points is increased ($\because$ central limit theorem)

Major weakness of the Laplace approximation

- Since it is based on a Gaussian distribution, it is only directly applicable to real variables
- May be possible to apply the Laplace approximation to a transformation of the variable
  - *e.g.*) if 0$\leq \tau < \infty$ then we can consider a Laplace approximation of $\ln \tau$
- It is based purely on the aspects of the true distribution at a specific value of the variable, and so can fail to capture important global properties.

Alternative approach to adopt a more global perspective: variational methods (Chapter 10)

Model Comparison and BIC

- Obtaining an approximation to the normalization constant Z

$$\begin{aligned} Z &= \int f(\mathbf{z})\, d\mathbf{z} \\ &\simeq f(\mathbf{z}_0) \int \exp\left\{-\frac{1}{2}(\mathbf{z}-\mathbf{z}_0)^\top \mathbf{A}(\mathbf{z}-\mathbf{z}_0)\right\} d\mathbf{z} \\ &= f(\mathbf{z}_0)\frac{(2\pi)^{M/2}}{|\mathbf{A}|^{1/2}} \end{aligned}$$

Approximation of the model evidence

Approximation to the model evidence $p\left(\mathcal{D}|\mathcal{M}_i\right)$

$$p\left(\mathcal{D}|\mathcal{M}_i\right) = \int p\left(\mathcal{D}|\boldsymbol{\theta}_i\right) p\left(\boldsymbol{\theta}_i\right) d\boldsymbol{\theta}_i$$

- Dataset $\mathcal{D}$ and a set of models $\{\mathcal{M}_i\}$ having parameters $\{\boldsymbol{\theta}_i\}$
- Likelihood function $p\left(\mathcal{D}|\boldsymbol{\theta}_i, \mathcal{M}_i\right)$ per model
- Prior $p\left(\boldsymbol{\theta}_i|\mathcal{M}_i\right)$ over the parameters

$$p\left(\mathcal{D}\right) = \int p\left(\mathcal{D}|\boldsymbol{\theta}\right) p\left(\boldsymbol{\theta}\right) d\boldsymbol{\theta}$$

(Omitting the conditioning on $\mathcal{M}_i$ to keep the notation uncluttered)

- Identifying $f\left(\boldsymbol{\theta}\right) = p\left(\mathcal{D}|\boldsymbol{\theta}\right) p\left(\boldsymbol{\theta}\right)$ and applying

$$p\left(\mathcal{D}\right) = Z = f\left(\mathbf{z}_0\right) \frac{(2\pi)^{M/2}}{|\mathbf{A}|^{1/2}}$$

$$\ln p\left(\mathcal{D}\right) \simeq \ln p\left(\mathcal{D}|\boldsymbol{\theta}_{\mathsf{MAP}}\right) + \underbrace{\ln p\left(\boldsymbol{\theta}_{\mathsf{MAP}}\right) + \frac{M}{2}\ln\left(2\pi\right) - \frac{1}{2}\ln|\mathbf{A}|}_{\text{Occam factor}}$$

  - $\mathbf{A}$: *Hessian* matrix of second derivatives of the negative log posterior

$$\mathbf{A} = -\nabla\nabla \ln p\left(\mathcal{D}|\boldsymbol{\theta}_{\mathsf{MAP}}\right) p\left(\boldsymbol{\theta}_{\mathsf{MAP}}\right) = -\nabla\nabla \ln p\left(\boldsymbol{\theta}_{\mathsf{MAP}}|\mathcal{D}\right)$$

  - *Occam factor*: penalizing the model complexity

- When assume that the Gaussian prior distribution over parameters is broad, and that the Hessian has full rank

$$\ln p\left(\mathcal{D}\right) \simeq \ln p\left(\mathcal{D}|\boldsymbol{\theta}_{\mathsf{MAP}}\right) - \frac{M}{2}\ln\left(N\right)$$

  - N: the number of data points
  - M: the number of parameters in $\boldsymbol{\theta}$

known as *Bayesian Information Criterion (BIC)*

- Complexity measures such as AIC and BIC have the virtue of being easy to evaluate, but can also give misleading results.
- In particular, the assumption that the Hessian matrix has full rank is often not valid since many of the parameters are not ‘well-determined’.

Bayesian Logistic Regression

Introduction

- Bayesian inference for logistic regression: intractable
  - Normalization of the product of a prior distribution and a likelihood function
  - Likelihood function: a product of logistic sigmoid functions, one for every data point
- Evaluation of the predictive distribution: also intractable

Use of approximation methods
(e.g., Laplace or variational methods, sampling methods)

Bayesian Logistic Regression

- Given a Gaussian prior and likelihood function

$$p(\mathbf{w}) = \mathcal{N}\left(\mathbf{w}|\mathbf{m}_0, \mathbf{S}_0\right) \qquad p\left(\mathbf{t}|\mathbf{w}\right) = \prod_{n=1}^{N} y_n^{t_n} \left\{1 - y_n\right\}^{1-t_n}$$

- Posterior distribution over $\mathbf{w}$ with $\mathbf{t} = (t_1, \ldots, t_N)$ is given by

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

- Log of the posterior distribution

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

where $y_n = P\left(C_1|\phi_n\right) = \sigma\left(\mathbf{w}^\top \phi_n\right)$

Laplace Approximation

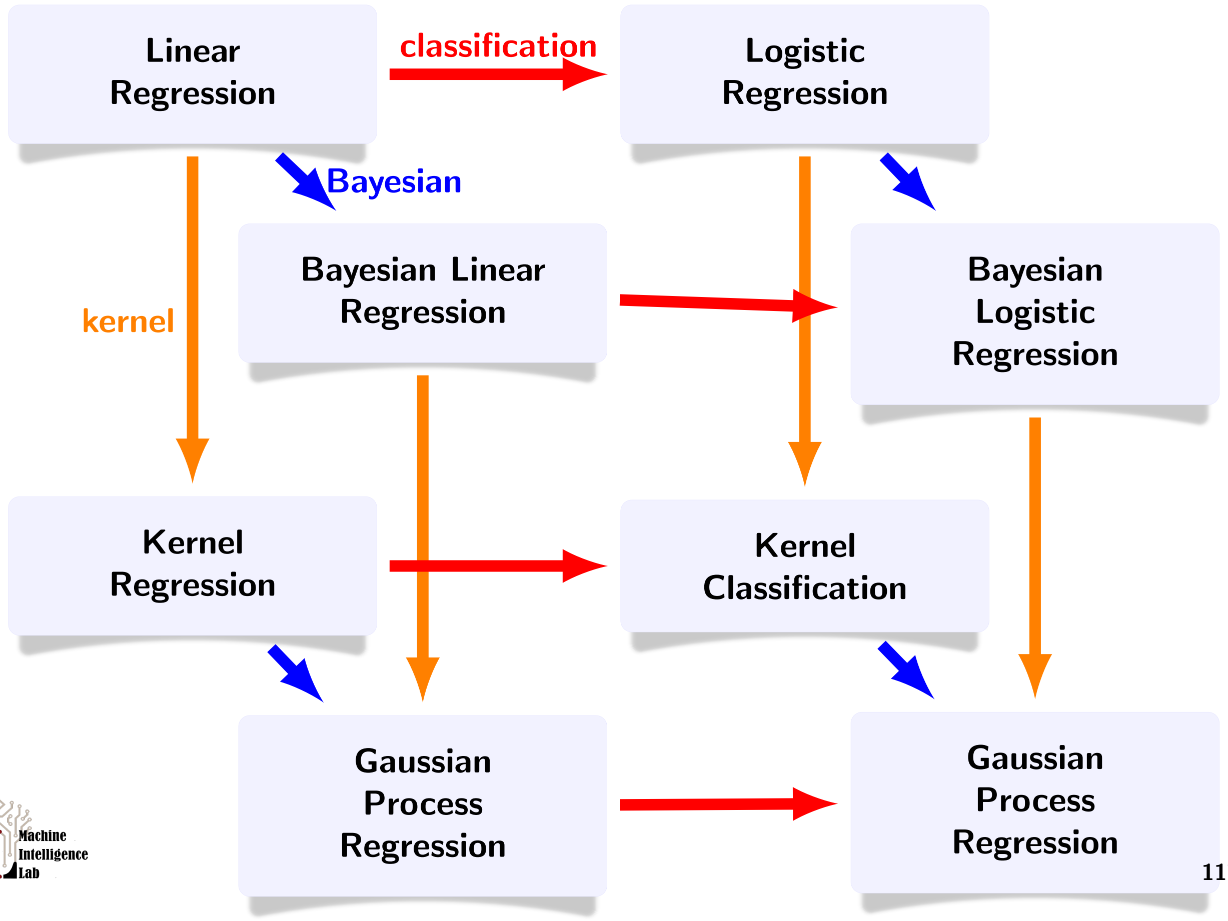
Gaussian approximation to the posterior distribution

- Maximize the posterior distribution to find the MAP solution $\mathbf{w}_{\mathsf{MAP}}$, which defines the mean of the Gaussian
  - Can be done by numerical optimization
- Find the covariance given by the inverse of the matrix of second derivatives of the negative log-likelihood

$$\mathbf{S}_N = -\nabla\nabla \ln p(\mathbf{w}|\mathbf{t}) = \mathbf{S}_0^{-1} + \sum_{n=1}^{N} y_n \left(1 - y_n\right) \phi_n \phi_n^{\top}$$

$$q(\mathbf{w}) = \mathcal{N}\left(\mathbf{w}|\mathbf{w}_{\mathsf{MAP}}, \mathbf{S}_N\right)$$

Still need to marginalize w.r.t the posterior distribution in order to make predictions



Linear Regression

classification

Logistic Regression

Bayesian

kernel

Bayesian Linear Regression

Bayesian Logistic Regression

Kernel Regression

Kernel Classification

Gaussian Process Regression

Gaussian Process Regression

Appendix

Taylor Expansion

An approximation to a function at a point $\mathbf{x}_o$

$$f(\mathbf{x}) = f\left(\mathbf{x}_o\right)+\nabla f\left(\mathbf{x}_o\right)^T\left(\mathbf{x}-\mathbf{x}_o\right)+\frac{1}{2}\left(\mathbf{x}-\mathbf{x}_o\right)^T\nabla^2 f\left(\mathbf{x}_o\right)\left(\mathbf{x}-\mathbf{x}_o\right)+\cdots$$

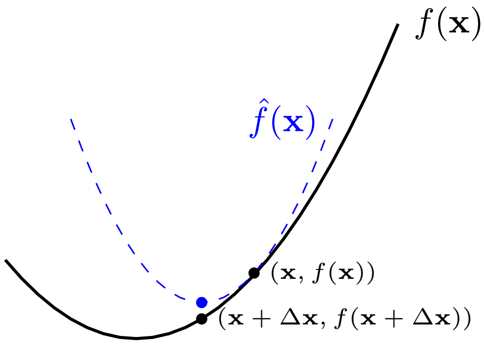
By denoting $\mathbf{x}-\mathbf{x}_o$ as $\Delta\mathbf{x}_o$

$$f(\mathbf{x}) = f\left(\mathbf{x}_o\right)+\nabla f\left(\mathbf{x}_o\right)^T\Delta\mathbf{x}+\frac{1}{2}\Delta\mathbf{x}^T\nabla^2 f\left(\mathbf{x}_o\right)\left(\Delta\mathbf{x}\right)+\cdots$$

Newton's Method

Second-order approximation

$$f(\mathbf{x}+\Delta\mathbf{x}) \approx f(\mathbf{x}) + \nabla f(\mathbf{x})^T \Delta\mathbf{x} + \frac{1}{2}\Delta\mathbf{x}^T \nabla^2 f(\mathbf{x}) \Delta\mathbf{x}$$


$f(\mathbf{x})$
$\hat{f}(\mathbf{x})$
$(\mathbf{x}, f(\mathbf{x}))$
$(\mathbf{x}+\Delta\mathbf{x}, f(\mathbf{x}+\Delta\mathbf{x}))$

Choose $\Delta\mathbf{x}$ to minimize the approximation

$$\Delta\mathbf{x} = -\left[\nabla^2 f(\mathbf{x})\right]^{-1} \nabla f(\mathbf{x})$$

▸ IRLS