LINEAR MODELS FOR CLASSIFICATION

J. Elder

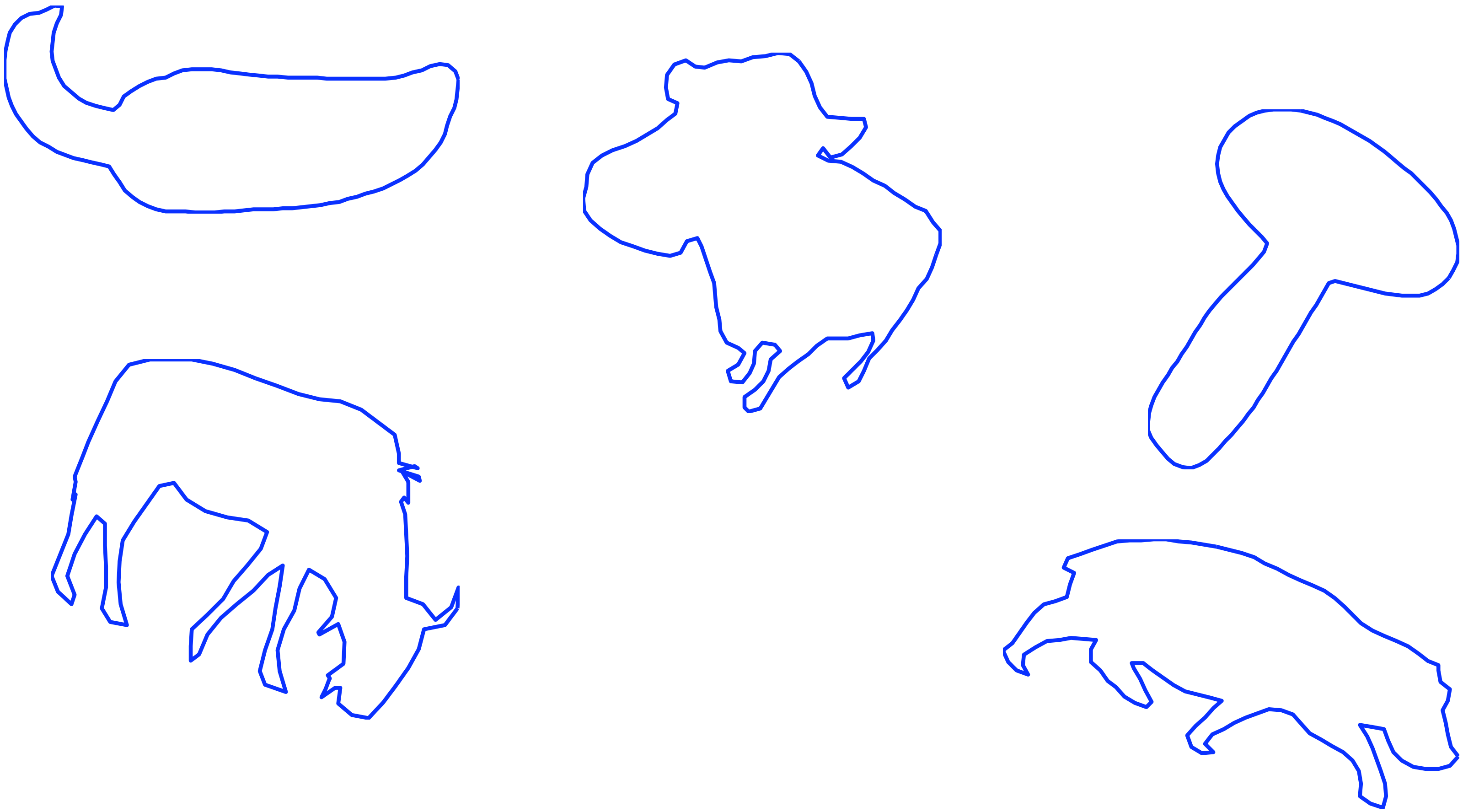
CSE 6390/PSYC 6225 Computational Modeling of Visual Perception

Classification: Problem Statement

- In regression, we are modeling the relationship between a continuous input variable *x* and a continuous target variable *t*.
- In classification, the input variable *x* is still continuous, but the target variable is discrete.
- In the simplest case, *t* can have only 2 values.

Example Problem

- Animal or Vegetable?

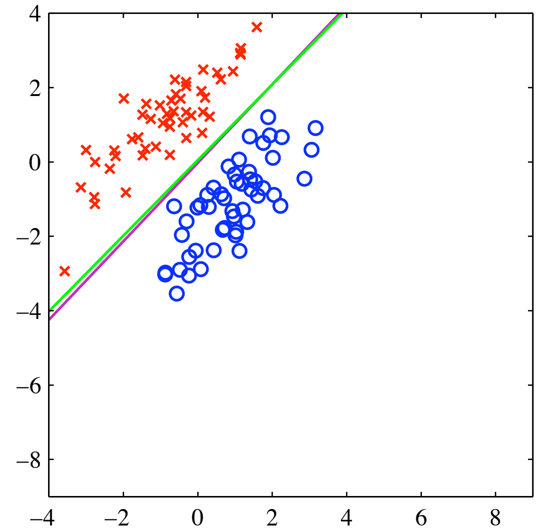


Linear Models for Classification

- Linear models for classification separate input vectors into classes using linear *decision boundaries.*
  - Example:

Input vector **x**

Two discrete classes C_1 and C_2



Discriminant Functions

A linear discriminant function $y(\mathbf{x}) = f\left(\mathbf{w}^t\mathbf{x} + w_0\right)$

maps a real input vector $\mathbf{x}$ to a scalar value $y(\mathbf{x})$.

$f(\cdot)$ is called an *activation function*.

Outline

- Linear activation functions
  - Least-squares formulation
  - Fisher's linear discriminant
- Nonlinear activation functions
  - Probabilistic generative models
  - Probabilistic discriminative models
    - Logistic regression
    - Bayesian logistic regression

Two Class Discriminant Function

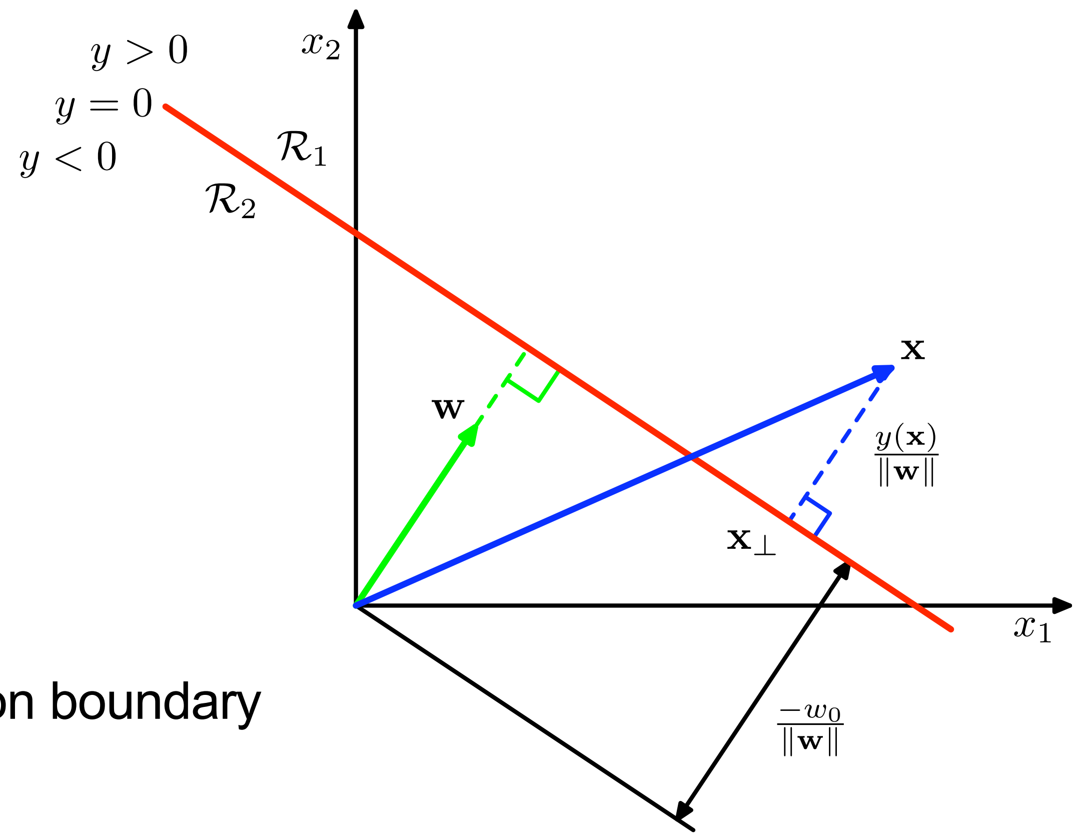
Let $f(\cdot)$ be the identity:

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

$y(\mathbf{x}) \geq 0 \rightarrow \mathbf{x}$ assigned to C_1

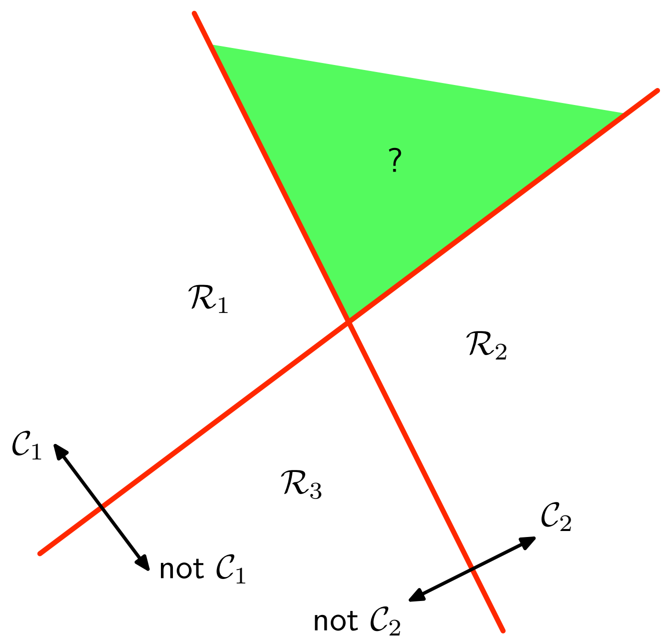
$y(\mathbf{x}) < 0 \rightarrow \mathbf{x}$ assigned to C_2

Thus $y(\mathbf{x}) = 0$ defines the decision boundary



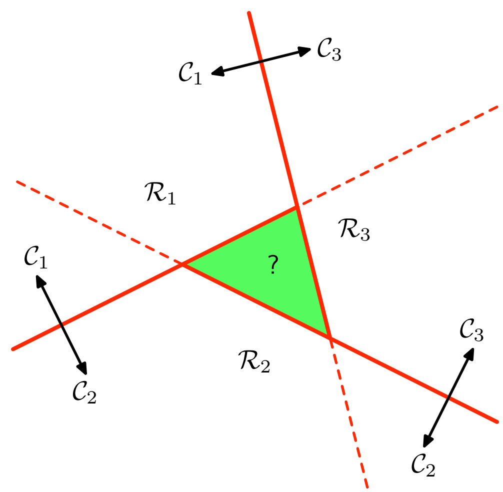
K>2 Classes

- Idea #1: Just use *K-1* discriminant functions, each of which separates one class C_k from the rest. (One-versus-the-rest classifier.)
- Problem: Ambiguous regions



$K>2$ Classes

- Idea #2: Use $K(K-1)/2$ discriminant functions, each of which separates two classes C_j, C_k from each other. (One-versus-one classifier.)
- Each point classified by majority vote.
- Problem: Ambiguous regions



$K>2$ Classes

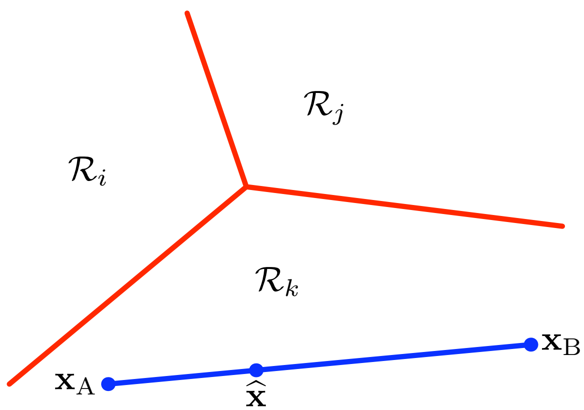
- Idea #3: Use K discriminant functions $y_k(x)$
- Use the **magnitude** of $y_k(x)$, not just the sign.

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

$\mathbf{x}$ assigned to C_k if $y_k(\mathbf{x}) > y_j(\mathbf{x}) \forall j \neq k$

Decision boundary $y_k(\mathbf{x}) = y_j(\mathbf{x}) \rightarrow \left(w_k - w_j\right)^t x + \left(w_{k0} - w_{j0}\right) = 0$

Results in decision regions that are simply-connected and convex.



Learning the Parameters

- Method #1: Least Squares

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

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

where

$$\tilde{\mathbf{x}} = (1, \mathbf{x}^t)^t$$

$\tilde{\mathbf{W}}$ is a $(D+1)\times K$ matrix whose kth column is $\tilde{\mathbf{w}}_k = \left(w_0, \mathbf{w}_k^t\right)^t$

Learning the Parameters

- Method #1: Least Squares

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

Training dataset $(\mathbf{x}_n, \mathbf{t}_n), \quad n = 1, \ldots, N$

where we use the 1-of-K coding scheme for $\mathbf{t}_n$

Let $\mathbf{T}$ be the $N \times K$ matrix whose n^{th} row is $\mathbf{t}_n^t$

Let $\tilde{\mathbf{X}}$ be the $N \times (D+1)$ matrix whose n^{th} row is $\tilde{\mathbf{x}}_n^t$

We define the error as $E_D(\tilde{\mathbf{W}}) = \frac{1}{2} \mathrm{Tr}\left\{ (\tilde{\mathbf{X}}\tilde{\mathbf{W}} - \mathbf{T})^t (\tilde{\mathbf{X}}\tilde{\mathbf{W}} - \mathbf{T}) \right\}$

Setting derivative wrt $\tilde{\mathbf{W}}$ yields:

$$\tilde{\mathbf{W}} = \left(\tilde{\mathbf{X}}^t \tilde{\mathbf{X}}\right)^{-1} \tilde{\mathbf{X}}^t \mathbf{T} = \tilde{\mathbf{X}}^\dagger \mathbf{T}$$

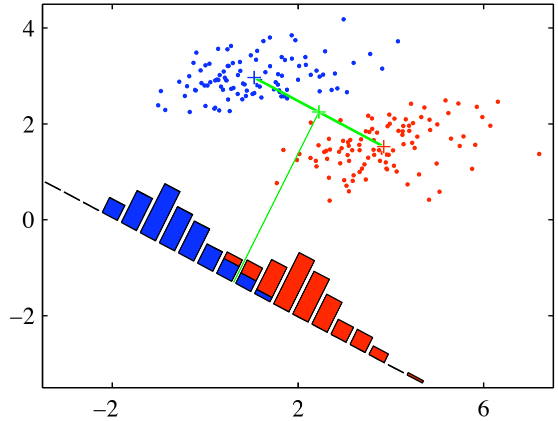
Fisher's Linear Discriminant

- Another way to view linear discriminants: find the 1D subspace that maximizes the separation between the two classes.

Let $\mathbf{m}_1 = \frac{1}{N_1} \sum_{n \in C_1} \mathbf{x}_n, \quad m_2 = \frac{1}{N_2} \sum_{n \in C_2} \mathbf{x}_n$

For example, might choose $\mathbf{w}$ to maximize $\mathbf{w}^t \left(\mathbf{m}_2 - \mathbf{m}_1 \right)$, subject to $\|\mathbf{w}\| = 1$

This leads to $\mathbf{w} \propto \mathbf{m}_2 - \mathbf{m}_1$



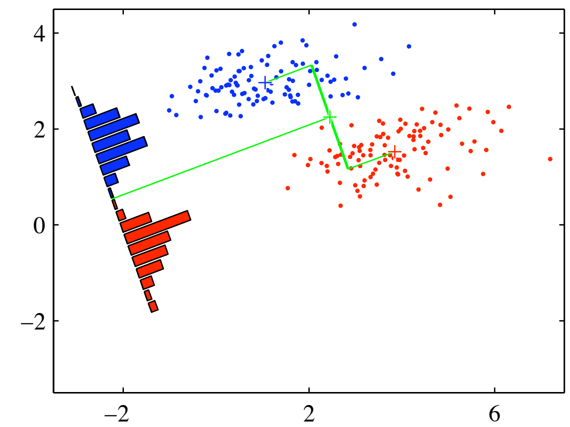
However, if conditional distributions are not isotropic, this is typically not optimal.

Fisher's Linear Discriminant

Let $m_1 = \mathbf{w}^t\mathbf{m}_1, \quad m_2 = \mathbf{w}^t\mathbf{m}_2$ be the conditional means on the 1D subspace.

Let $s_k^2 = \sum_{n \in C_k} (y_n - m_k)^2$ be the within-class variance on the subspace for class C_k

The Fisher criterion is then $J(\mathbf{w}) = \dfrac{(m_2 - m_1)^2}{s_1^2 + s_2^2}$



This can be rewritten as

$$J(\mathbf{w}) = \frac{\mathbf{w}^t\mathbf{S}_B\mathbf{w}}{\mathbf{w}^t\mathbf{S}_W\mathbf{w}}$$

where

$\mathbf{S}_B = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^t$ is the between-class variance

and

$\mathbf{S}_W = \sum_{n \in C_1} (x_n - \mathbf{m}_1)(x_n - \mathbf{m}_1)^t + \sum_{n \in C_2} (x_n - \mathbf{m}_2)(x_n - \mathbf{m}_2)^t$ is the within-class variance

$J(\mathbf{w})$ is maximized for $\mathbf{w} \propto \mathbf{S}_W^{-1}(\mathbf{m}_2 - \mathbf{m}_1)$

Connection between Least-Squares and FLD

Change coding scheme to

$$t_n = \frac{N}{N_1} \text{ for } C_1$$

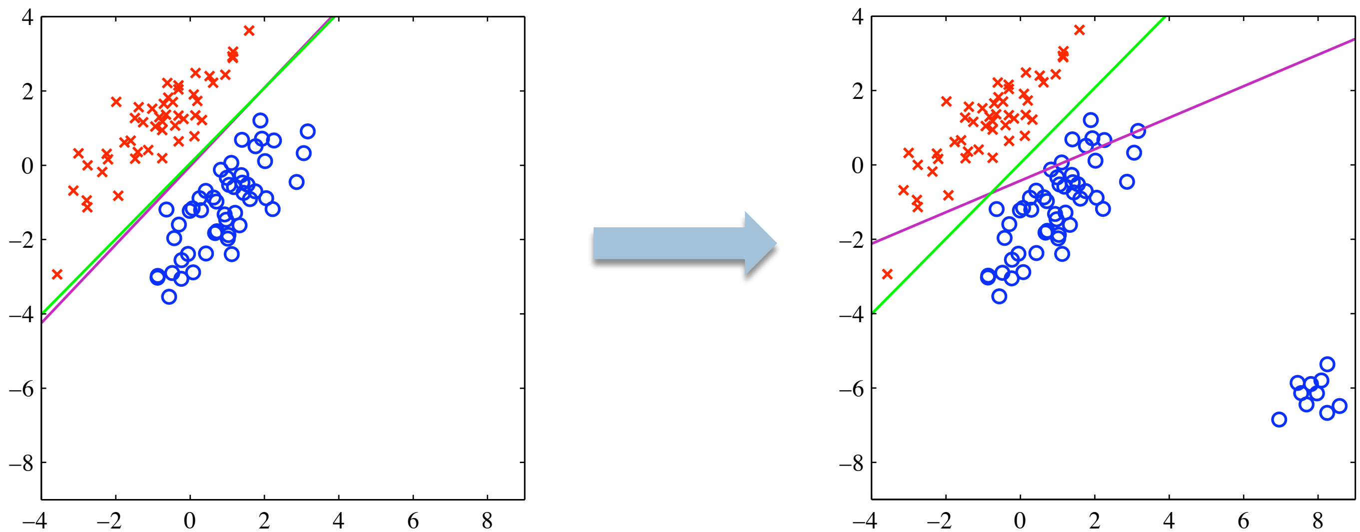
$$t_n = -\frac{N}{N_2} \text{ for } C_2$$

Then one can show that the ML $\mathbf{w}$ satisfies

$$\mathbf{w} \propto \mathbf{S}_W^{-1}\left(\mathbf{m}_2 - \mathbf{m}_1\right)$$

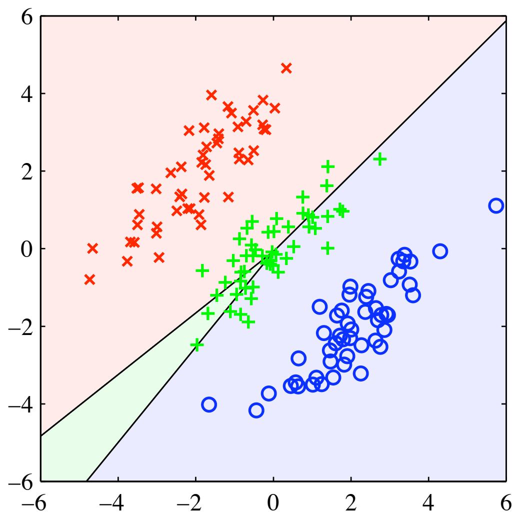
Least Squares Classifier

- Problem #1: Sensitivity to outliers



Least Squares Classifier

- Problem #2: Linear activation function is not a good fit to binary data. This can lead to problems.



Outline

- Linear activation functions
  - Least-squares formulation
  - Fisher's linear discriminant
- **Nonlinear activation functions**
  - Probabilistic generative models
  - Probabilistic discriminative models
    - Logistic regression
    - Bayesian logistic regression

Probabilistic Generative Models

- Consider first $K=2$:

By Bayes' equation, the posterior for class C_1 can be written:

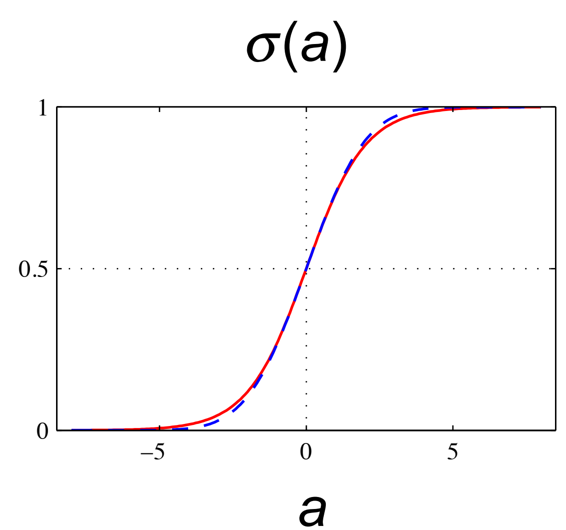
$$p(C_1 \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid C_1)p(C_1)}{p(\mathbf{x} \mid C_1)p(C_1) + p(\mathbf{x} \mid C_2)p(C_2)}$$

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

where

$$a = \log \frac{p(\mathbf{x} \mid C_1)p(C_1)}{p(\mathbf{x} \mid C_2)p(C_2)}$$

and $\sigma(a)$ is the logistic sigmoid function



Probabilistic Generative Models

Let's assume that the input vector $\mathbf{x}$ is multivariate normal, when conditioned upon the class C_k, and that the covariance is the same for all classes:

$$p(\mathbf{x} \mid C_k) = \frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} \exp\left\{ -\frac{1}{2} (\mathbf{x} - \mu_k)^t \Sigma^{-1} (\mathbf{x} - \mu_k) \right\}$$

Then we have that $p(C_1 \mid \mathbf{x}) = \sigma(\mathbf{w}^t \mathbf{x} + w_0)$

where

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

$$w_0 = -\frac{1}{2}\mu_1^t \Sigma^{-1} \mu_1 + \frac{1}{2}\mu_2^t \Sigma^{-1} \mu_2 + \log \frac{p(C_1)}{p(C_2)}$$

Thus we have a generalized linear model,
and the decision surfaces will be hyperplanes in the input space.

Probabilistic Generative Models

This result generalizes to $K > 2$ classes:

$$p(C_k \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid C_k)\, p(C_k)}{\sum_j p(\mathbf{x} \mid C_j)\, p(C_j)}$$

$$= \frac{\exp(a_k)}{\sum_j \exp(a_j)}$$ "softmax"

where

$$a_k = \log\left(p(\mathbf{x} \mid C_k)\, p(C_k)\right)$$

Then we have that $a_k(x) = \mathbf{w}_k^t \mathbf{x} + w_{k0}$

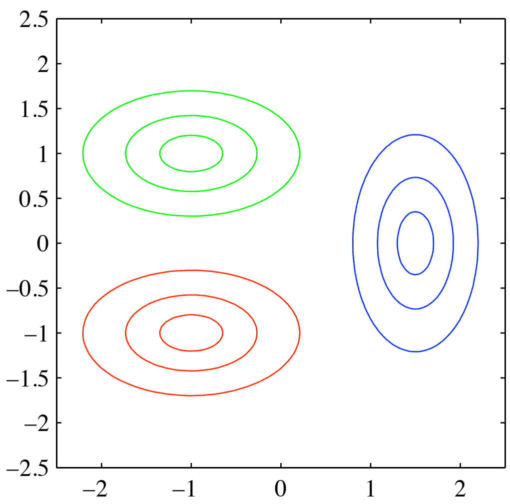
where

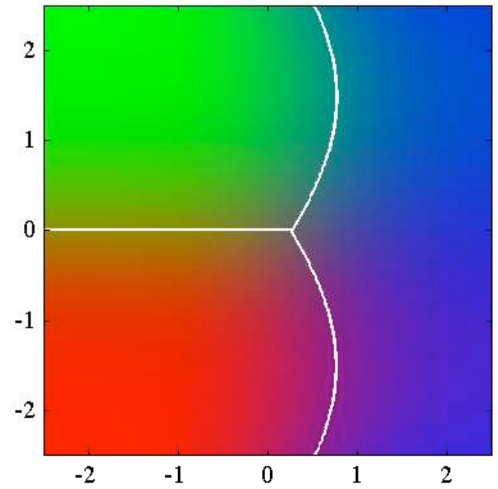
$$\mathbf{w}_k = \Sigma^{-1} \mu_k$$

$$w_{k0} = -\frac{1}{2} \mu_k^t \Sigma^{-1} \mu_k + \log p(C_k)$$

Non-Constant Covariance

- If the class-conditional covariances are different, the generative decision boundaries are in general quadratic.





ML for Probabilistic Generative Model

Let $t_n = 1$ denote Class 1, $t_n = 0$ denote Class 2.

Let $\pi = p(C_1)$ so that $1 - \pi = p(C_2)$

Then the ML estimates for the parameters are:

$$\pi = \frac{N_1}{N_1 + N_2}$$

$$\mu_1 = \frac{1}{N_1} \sum_{n=1}^{N} t_n \mathbf{x}_n$$

$$\mu_2 = \frac{1}{N_2} \sum_{n=1}^{N} (1 - t_n) \mathbf{x}_n$$

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

where

$$\mathbf{S}_1 = \frac{1}{N_1} \sum_{n \in C_1} (\mathbf{x}_n - \mu_1)(\mathbf{x}_n - \mu_1)^t$$

and

$$\mathbf{S}_2 = \frac{1}{N_2} \sum_{n \in C_2} (\mathbf{x}_n - \mu_2)(\mathbf{x}_n - \mu_2)^t$$

Probabilistic Discriminative Models

- An alternative to the generative approach is to model the dependence of the target variable t on the input vector x directly, using the activation function f.
- One big advantage is that there will typically be fewer parameters to determine.

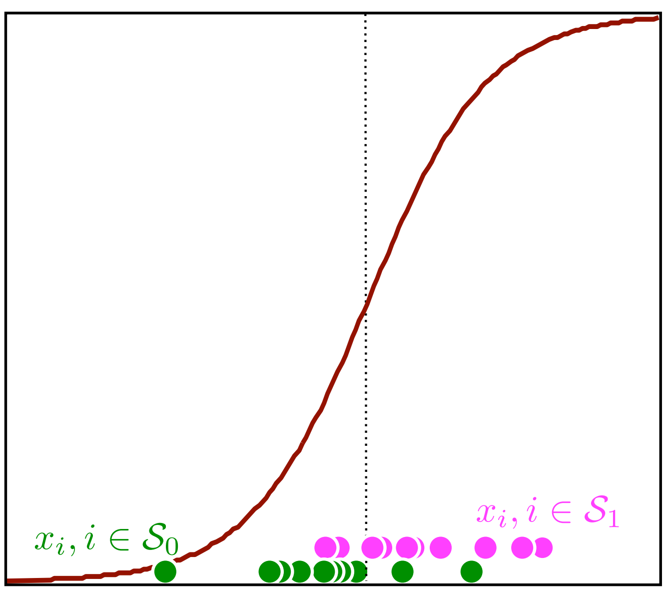
Logistic Regression ($K = 2$)

$$p(C_1 | \phi) = y(\phi) = \sigma(\mathbf{w}^t \phi)$$

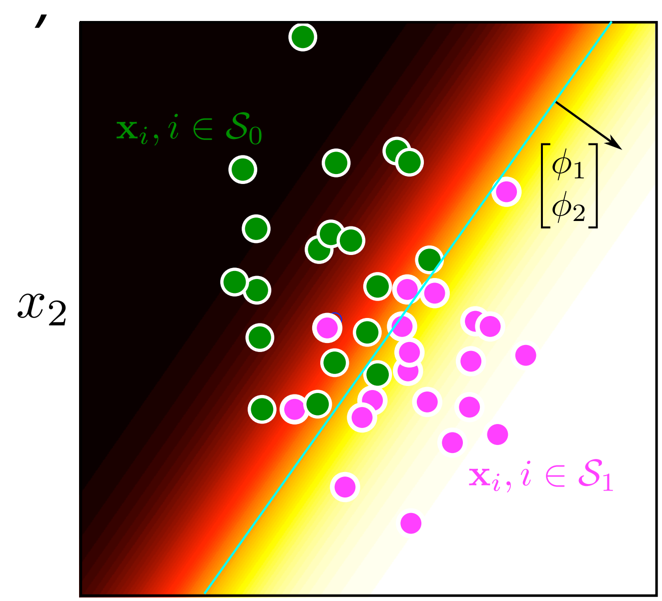
$$p(C_2 | \phi) = 1 - p(C_1 | \phi)$$

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

$$p(C_1 | \phi) = y(\phi) = \sigma(\mathbf{w}^t \phi)$$



$\mathbf{w}^t \phi$



x_1

Logistic Regression

$$p(C_1 | \phi) = y(\phi) = \sigma(\mathbf{w}^t \phi)$$
$$p(C_2 | \phi) = 1 - p(C_1 | \phi)$$

where

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

- Number of parameters
  - Logistic regression: M
  - Generative model: $2M + M(M+1)/2 + 1 = M(M+5)/2+1$

ML for Logistic Regression

$$p(\mathbf{t} \mid w) = \prod_{n=1}^{N} y_n^{t_n} \left\{1 - y_n\right\}^{1-t_n} \quad \text{where } \mathbf{t} = \left(t_1, \ldots, t_N\right)^t \text{ and } y_n = p\left(C_1 \mid \phi_n\right)$$

We define the error function to be $E(\mathbf{w}) = -\log p\left(\mathbf{t} \mid \mathbf{w}\right)$

Given $y_n = \sigma\left(a_n\right)$ and $a_n = \mathbf{w}^t \phi_n$, one can show that

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

Unfortunately, there is no closed form solution for **w**.

ML for Logistic Regression:

- Iterative Reweighted Least Squares
  - Although there is no closed form solution for the ML estimate of **w**, fortunately, the error function is convex.
  - Thus an appropriate iterative method is guaranteed to find the exact solution.
  - A good method is to use a local quadratic approximation to the log likelihood function (Newton-Raphson update):

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

where $\mathbf{H}$ is the Hessian matrix of $E(\mathbf{w})$

ML for Logistic Regression

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

where $\mathbf{H}$ is the Hessian matrix of $E(\mathbf{w})$:

$$\mathbf{H} = \Phi^t \mathbf{R} \Phi$$

where $\mathbf{R}$ is the $N \times N$ diagonal weight matrix with $R_{nn} = y_n\left(1 - y_n\right)$

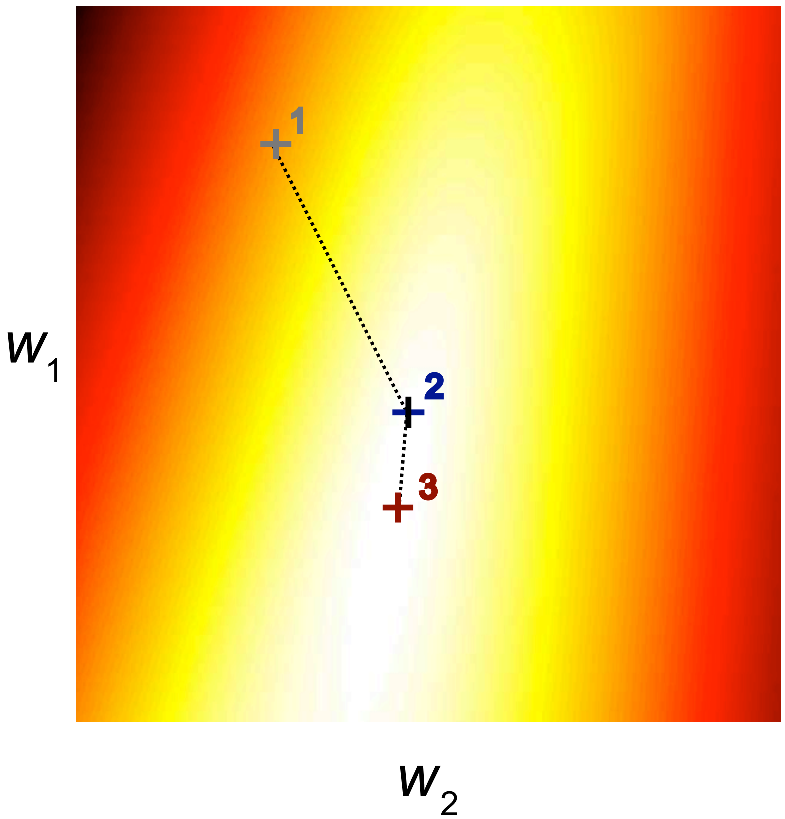
(Note that, since $\mathbf{R}_{nn} \geq 0$, $\mathbf{R}$ is positive semi-definite, and hence $\mathbf{H}$ is positive semi-definite Thus $E(\mathbf{w})$ is convex.)

Thus

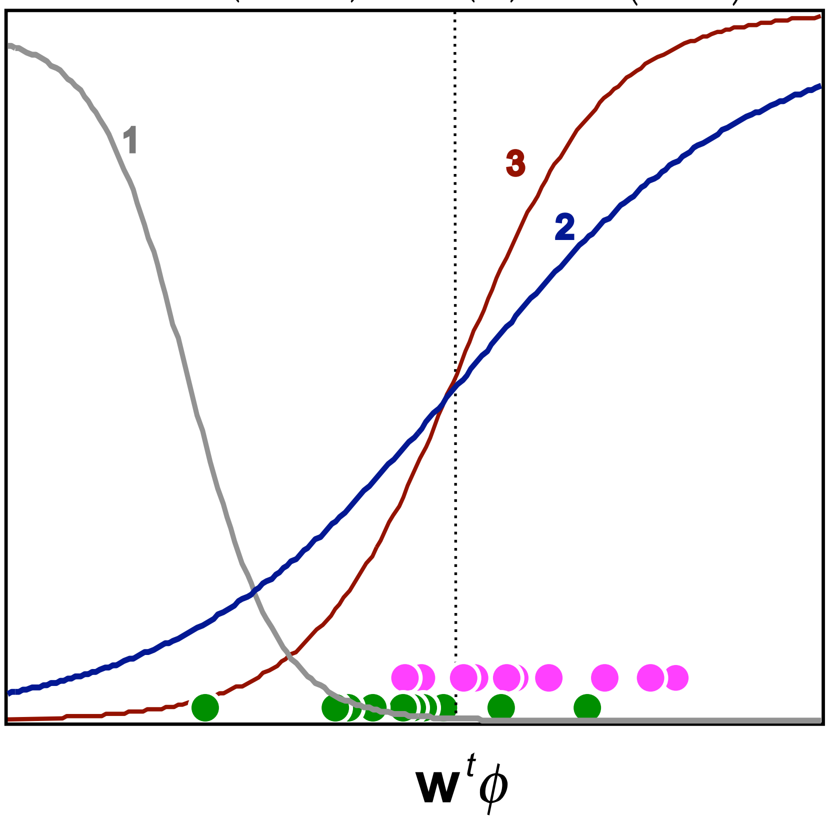
$$\mathbf{w}^{new} = \mathbf{w}^{(old)} - \left(\Phi^t \mathbf{R} \Phi\right)^{-1} \Phi^t \left(\mathbf{y} - \mathbf{t}\right)$$

ML for Logistic Regression

- Iterative Reweighted Least Squares



$$p(C_1 | \phi) = y(\phi) = \sigma(\mathbf{w}^t \phi)$$



Bayesian Logistic Regression

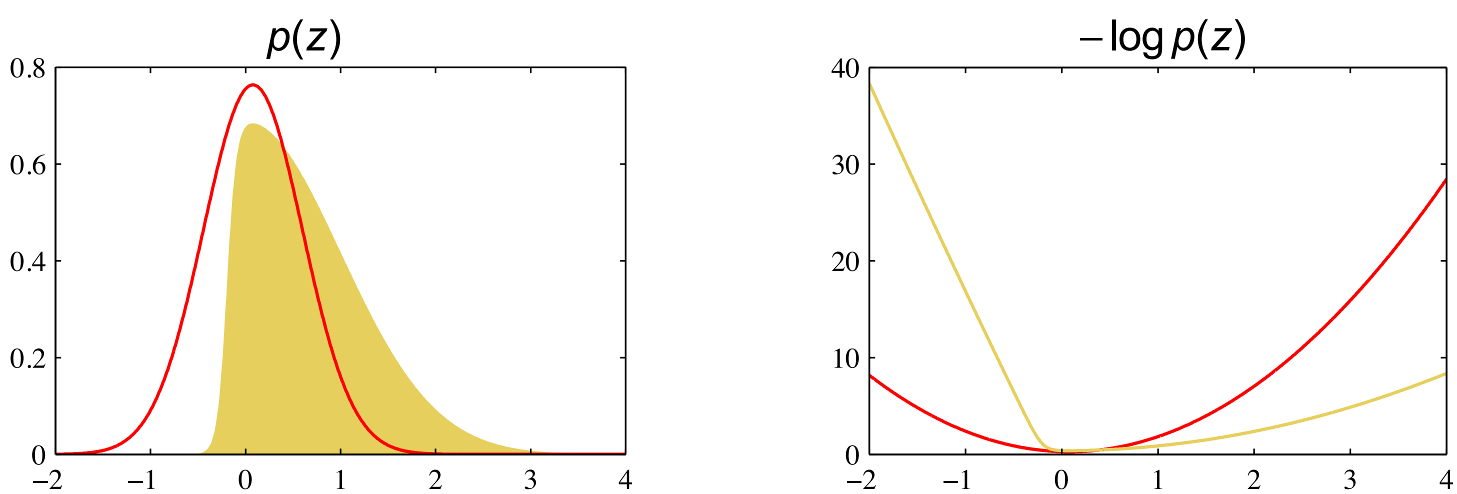
We can make logistic regression Bayesian by applying a prior over **w**:

$$p(\mathbf{w}) = N(\mathbf{w} \mid \mathbf{m}_0, \mathbf{S}_0)$$

- Unfortunately, the posterior over w will not be normal for logistic regression, and hence we cannot integrate over it analytically.
- This means that we cannot do Bayesian prediction analytically.
- However, there are methods for approximating the posterior that allow us to do approximate Bayesian prediction.

The Laplace Approximation

- In the Laplace approximation, we approximate the log of a distribution by a local, second order (quadratic) form, centred at the mode.
- This corresponds to a normal approximation to the distribution, with
  - mean given by the mode of the original distribution
  - precision matrix given by the Hessian of the negative log of the distribution



Bayesian Logistic Regression

- When applied to the posterior over **w** in logistic regression, this yields

$$p(\mathbf{w}) \simeq q(\mathbf{w}) = N\left(\mathbf{w} \mid \mathbf{w}_{MAP}, \mathbf{S}_N\right)$$

where

$$\mathbf{S}_N^{-1} = \mathbf{S}_0^{-1} + \sum_{n=1}^{N} y_n \left(1 - y_n\right) \phi_n \phi_n^t$$

Prediction

- Bayesian prediction requires that we integrate out this posterior over **w**:

$$p\left(C_1 \mid \phi, \mathbf{t}\right) = \int p\left(C_1 \mid \phi, \mathbf{w}\right) p(\mathbf{w} \mid \mathbf{t}) d\mathbf{w} \simeq \int \sigma\left(\mathbf{w}^t \phi\right) q(\mathbf{w}) d\mathbf{w}$$

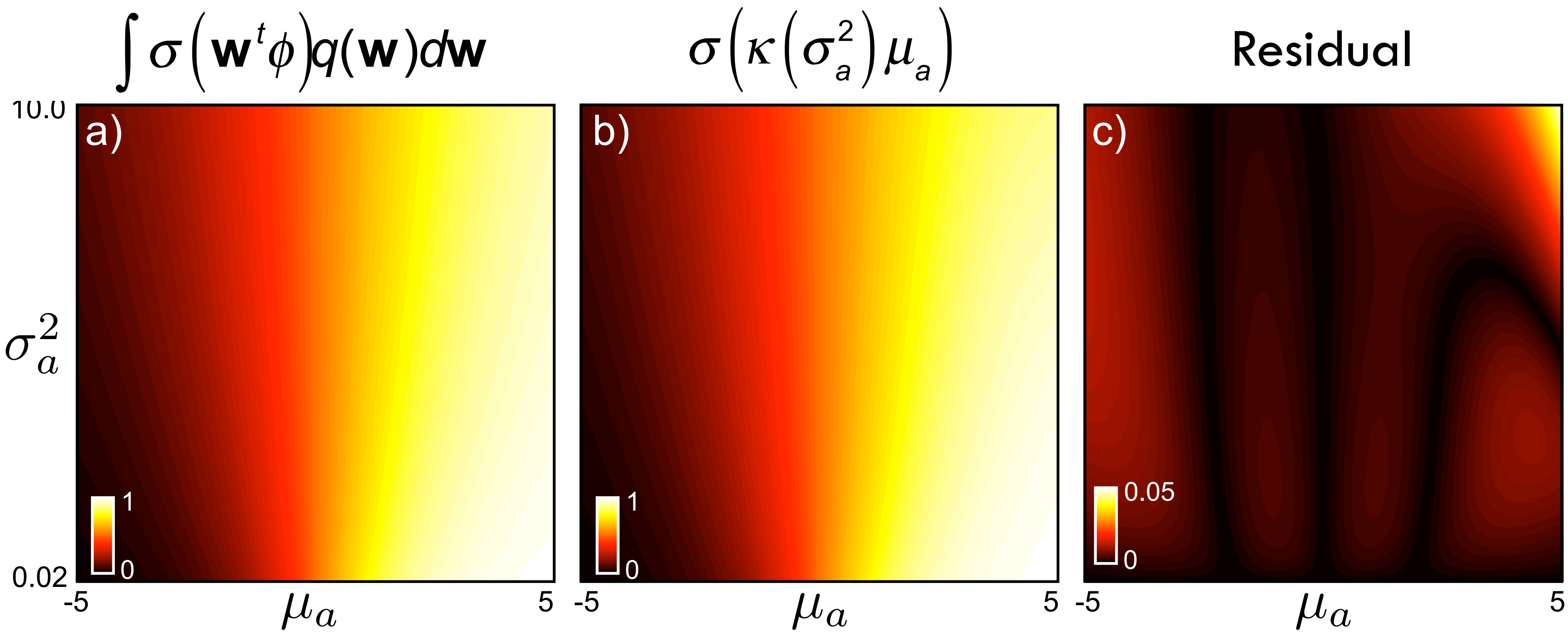
This integral is not tractable analytically.

However, approximation of the sigmoid function $\sigma(\cdot)$ by the inverse probit (cumulative normal) function yields an analytical solution:

$$p\left(C_1 \mid \phi, \mathbf{t}\right) \simeq \sigma\left(\kappa\left(\sigma_a^2\right)\mu_a\right),$$

where $\mu_a = \mathbf{w}_{MAP}^t \phi$, $\sigma_a^2 = \phi^t \mathbf{S}_N \phi$ and $\kappa\left(\sigma_a^2\right) = \left(1 + \pi\sigma_a^2 / 8\right)^{-1/2}$

Bayesian Logistic Regression



- This last approximation is excellent!