STA 4273H: Statistical Machine Learning

Russ Salakhutdinov

Department of Computer Science
Department of Statistical Sciences
rsalakhu@cs.toronto.edu
http://www.cs.utoronto.ca/~rsalakhu/

Lecture 3

Linear Models for Classification

- So far, we have looked at the linear models for regression that have particularly simple analytical and computational properties.

- We will now look at the analogous class of models for solving classification problems.

- We will also look at the Bayesian treatment of linear models for classification.

Classification

• The goal of classification is to assign an input **x** into one of K discrete classes C_k, where k=1,..,K.

• Typically, each input is assigned only to one class.

• Example: The input vector **x** is the set of pixel intensities, and the output variable t will represent the presence of cancer, class C_1, or absence of cancer, class C_2.

C_1: Cancer present

C_2: Cancer absent

x -- set of pixel intensities

Linear Classification

• The goal of classification is to assign an input **x** into one of K discrete classes C_k, where k=1,..,K.

• The input space is divided into decision regions whose boundaries are called decision boundaries or decision surfaces.

• We will consider linear models for classification. Remember, in the simplest linear regression case, the model is linear in parameters:

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

adaptive parameters

fixed nonlinear function:
activation function

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

Linear Classification

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

- The decision surfaces correspond to $y(\mathbf{x}, \mathbf{w}) = \text{const},$ so that $\mathbf{x}^T \mathbf{w} + w_0 = \text{const},$ and hence the decision surfaces are linear functions of **x**, even if the activation function is nonlinear.

- This class of models is called generalized linear models.

- Note that these models are no longer linear in parameters, due to the presence of nonlinear activation function.

- This leads to more complex analytical and computational properties, compared to linear regression.

- Note that we can make a fixed nonlinear transformation of the input variables using a vector of basis functions $\phi(\mathbf{x}),$ as we did for regression models.

Notation

• In the case of two-class problems, we can use the binary representation for the target value $t \in \{0, 1\}$, such that t=1 represents the positive class and t=0 represents the negative class.

- We can interpret the value of t as the probability of the positive class, and the output of the model can be represented as the probability that the model assigns to the positive class.

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

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

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

- We can interpret a vector **t** as a vector of class probabilities.

Three Approaches to Classification

- First approach: Construct a discriminant function that directly maps each input vector to a specific class.

- Model the conditional probability distribution $p(\mathcal{C}_k|\mathbf{x})$, and then use this distribution to make optimal decisions.

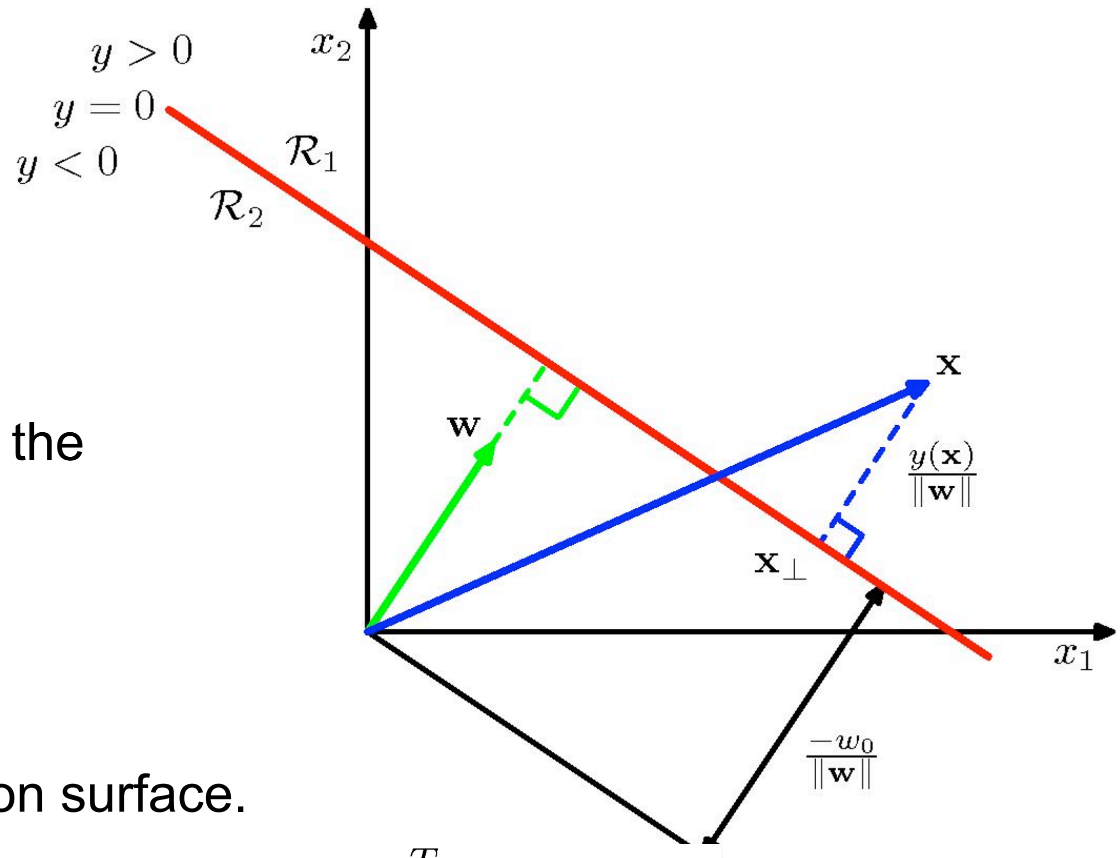
- There are two alternative approaches:
  - Discriminative Approach: Model $p(\mathcal{C}_k|\mathbf{x})$, directly, for example by representing them as parametric models, and optimize for parameters using the training set (e.g. logistic regression).
  - Generative Approach: Model class conditional densities $p(\mathbf{x}|\mathcal{C}_k)$ together with the prior probabilities $p(\mathcal{C}_k)$ for the classes. Infer posterior probability using Bayes' rule:

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

    - For example, we could fit multivariate Gaussians to the input vectors of each class. Given a test vector, we see under which Gaussian the test vector is most probable.

Discriminant Functions

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



- Assign **x** to C_1 if $y(\mathbf{x}) \geq 0,$ and class C_2 otherwise.
- Decision boundary:

$$y(\mathbf{x}) = 0.$$

- If two points $\mathbf{x_A}$ and $\mathbf{x_B}$ lie on the decision surface, then:

$$y(\mathbf{x}_A) = y(\mathbf{x}_B) = 0,$$
$$\mathbf{w}^T(\mathbf{x}_A - \mathbf{x}_B) = 0.$$

- **w** is orthogonal to the decision surface.

- If **x** is a point on the decision surface, then: $\frac{\mathbf{w}^T \mathbf{x}}{||\mathbf{w}||} = -\frac{w_0}{||\mathbf{w}||}.$

- Hence w_0 determines the location of the decision surface.

Multiple Classes

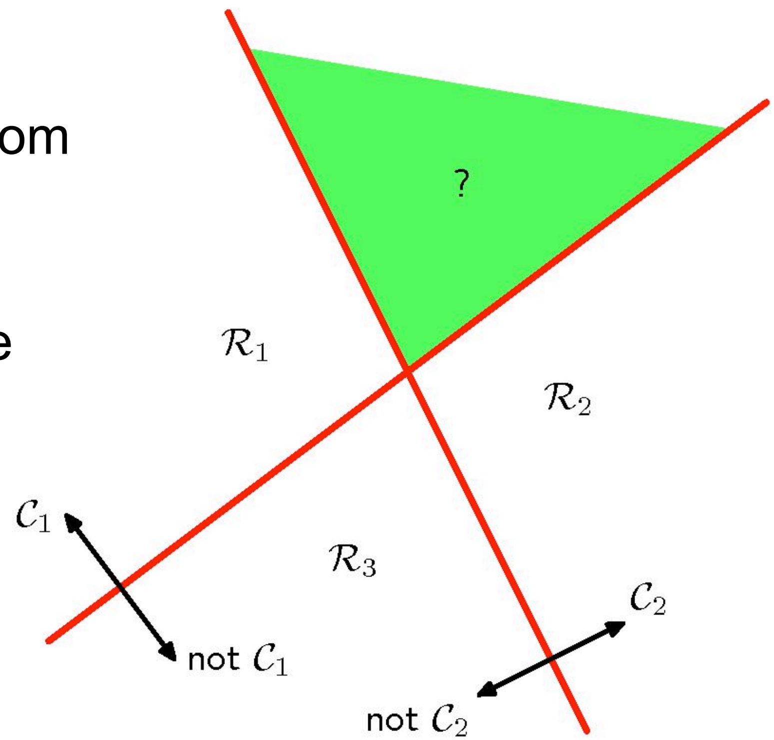
• Consider the extension of linear discriminants to K>2 classes.

• One option is to use K-1 classifiers, each of which solves a two class problem:

- Separate points in class C_k from points not in that class.

• There are regions in input space that are ambiguously classified.

One-versus-the-rest



Multiple Classes

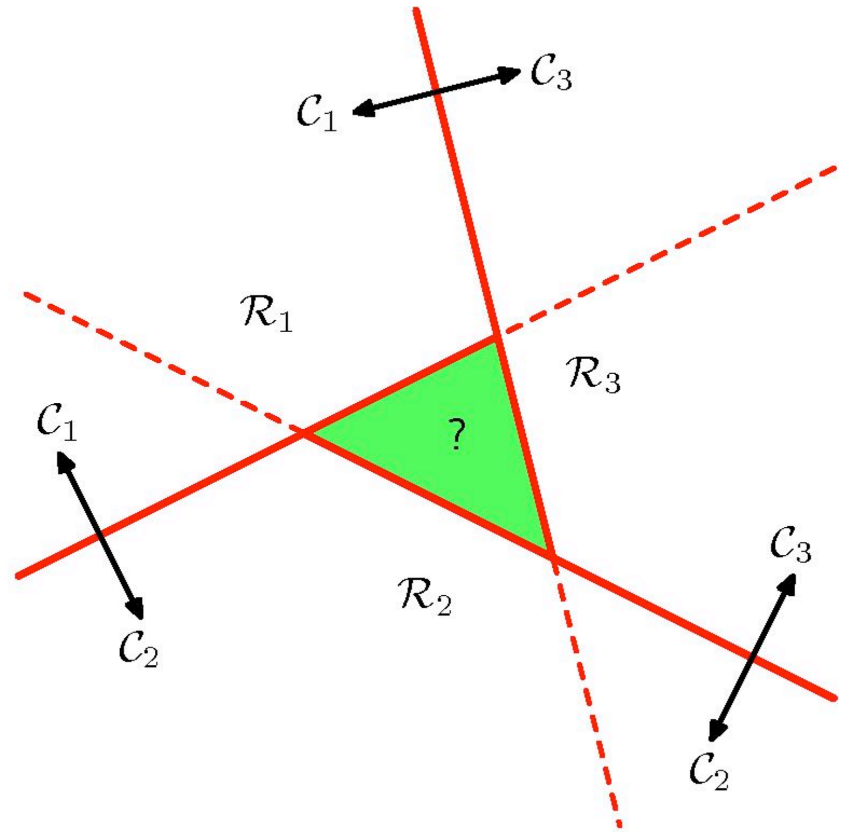
• Consider the extension of linear discriminants to K>2 classes.

• An alternative is to use K(K-1)/2 binary discriminant functions.

- Each function discriminates between two particular classes.

• Similar problem of ambiguous regions.

One-versus-one



Simple Solution

- Use K linear discriminant functions of the form:

$$y_k(\mathbf{x}) = \mathbf{x}^T \mathbf{w}_k + w_{k0}, \text{ where } k = 1, ..., K.$$

- Assign **x** to class C_k, if $y_k(\mathbf{x}) > y_j(\mathbf{x}) \; \forall j \neq k$ (pick the max).

- This is guaranteed to give decision boundaries that are singly connected and convex.

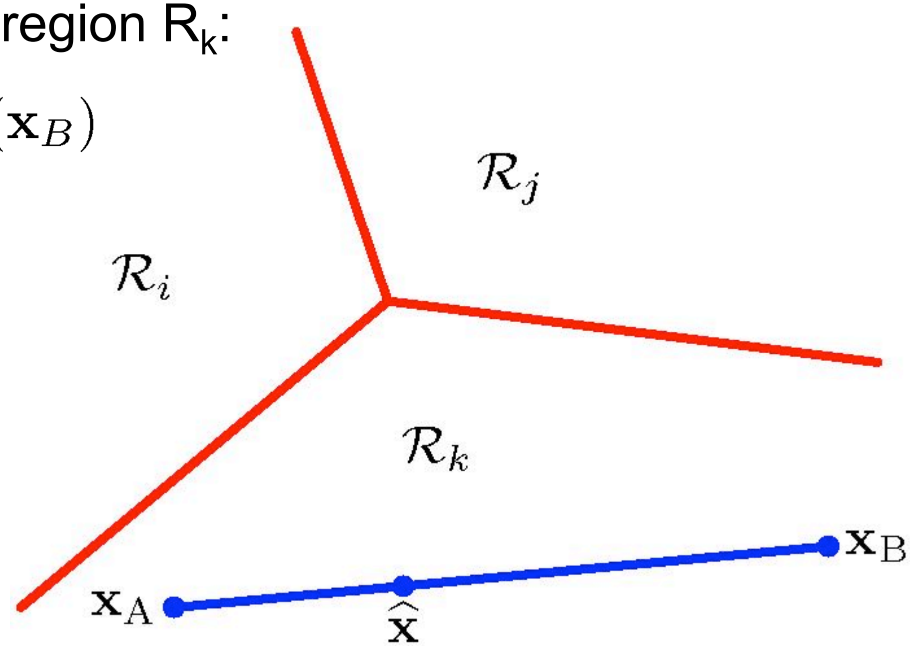
- For any two points that lie inside the region R_k:

$$y_k(\mathbf{x}_A) > y_j(\mathbf{x}_A) \text{ and } y_k(\mathbf{x}_B) > y_j(\mathbf{x}_B)$$

implies that for any positive α

$$y_k(\alpha \mathbf{x}_A + (1-\alpha)\mathbf{x}_B) > y_j(\alpha \mathbf{x}_A + (1-\alpha)\mathbf{x}_B)$$

due to linearity of the discriminant functions.



Least Squares for Classification

- Consider a general classification problem with K classes using 1-of-K encoding scheme for the target vector **t**.

- Remember: Least Squares approximates the conditional expectation $\mathbb{E}[\mathbf{t}|\mathbf{x}]$.

- Each class is described by its own linear model:

$$y_k(\mathbf{x}) = \mathbf{x}^T \mathbf{w}_k + w_{k0}, \text{ where } k = 1, ..., K.$$

- Using vector notation, we can write:

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

(D+1) $\times$ K matrix whose kth column comprises of D+1 dimensional vector:

$$\tilde{\mathbf{w}}_k = (w_{k0}, \mathbf{w}_k^T)^T.$$

corresponding augmented input vector:

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

Least Squares for Classification

- Consider observing a dataset $\{\mathbf{x_n}, t_n\}$, where n=1,…,N.

- We have already seen how to do least squares. Using some matrix algebra, we obtain the optimal weights:

$$\tilde{\mathbf{W}} = (\tilde{\mathbf{X}}^T \tilde{\mathbf{X}})^{-1} \tilde{\mathbf{X}}^T \mathbf{T}$$

Optimal weights

N $\times$ (D+1) input matrix whose n^{th} row is $\tilde{\mathbf{x}}_n^T$.

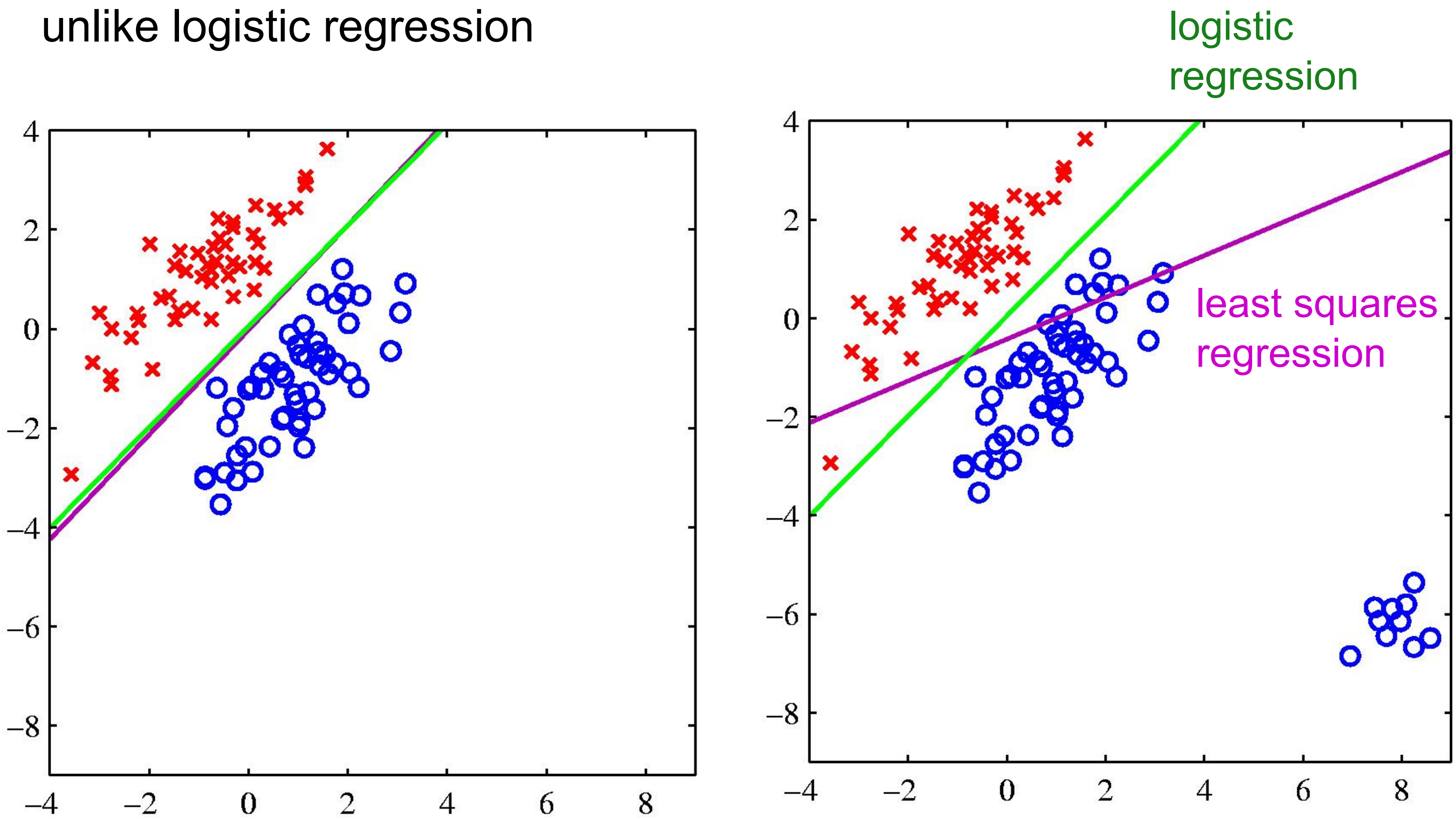
N $\times$ K target matrix whose n^{th} row is $\mathbf{t}_n^T$.

- A new input x is assigned to a class for which $y_k = \tilde{\mathbf{x}}^T \tilde{\mathbf{w}}_k$ is largest.

- There are however several problems when using least squares for classification.

Problems using Least Squares

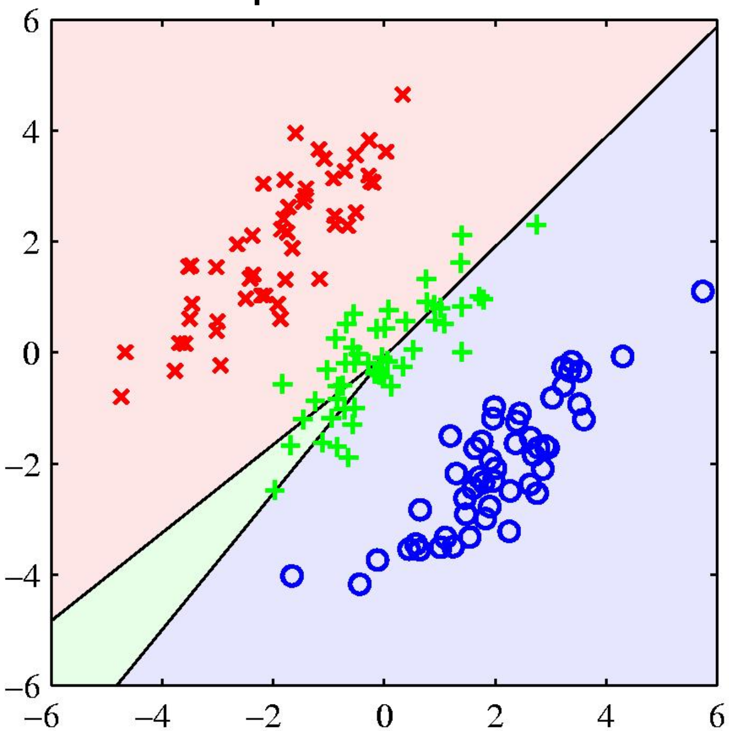
Least squares is highly sensitive to outliers,
unlike logistic regression



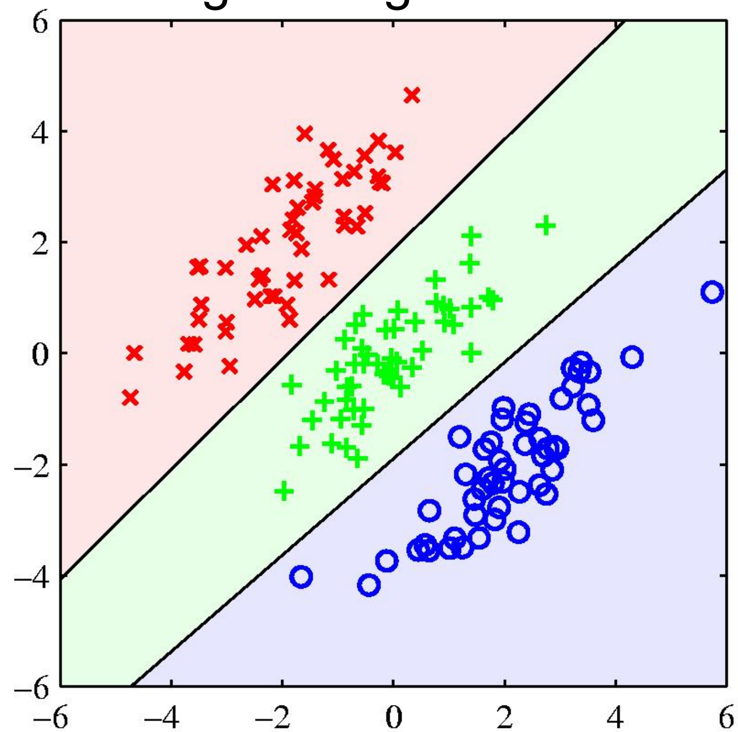
Problems using Least Squares

Example of a synthetic dataset containing 3 classes, where lines denote decision boundaries.

Least squares discriminant



Logistic regression



Many green points are misclassified.

Fisher's Linear Discriminant

• Dimensionality reduction: Suppose we take a D-dim input vector and project it down to one dimension using:

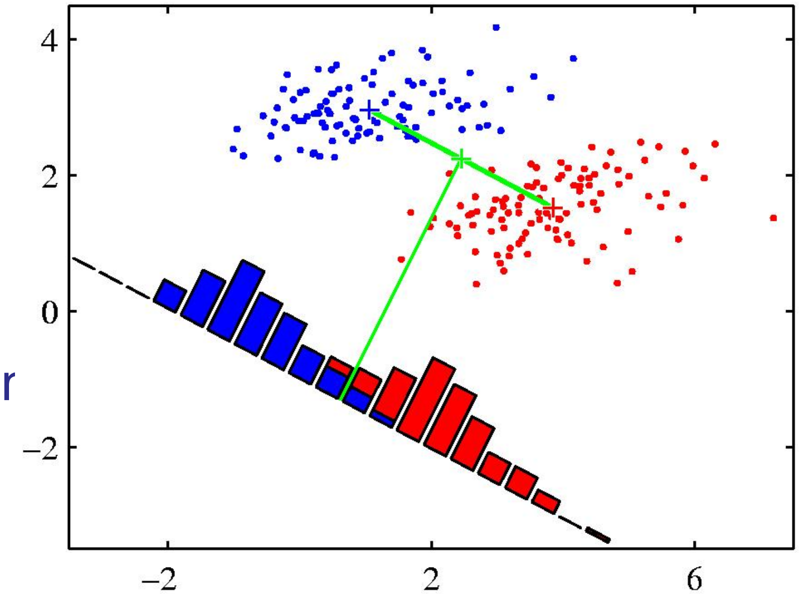
$$y = \mathbf{w}^T \mathbf{x}.$$

• Idea: Find the projection that maximizes the class separation.

• The simplest measure of separation is the separation of the projected class means. So we project onto the line joining the two means.

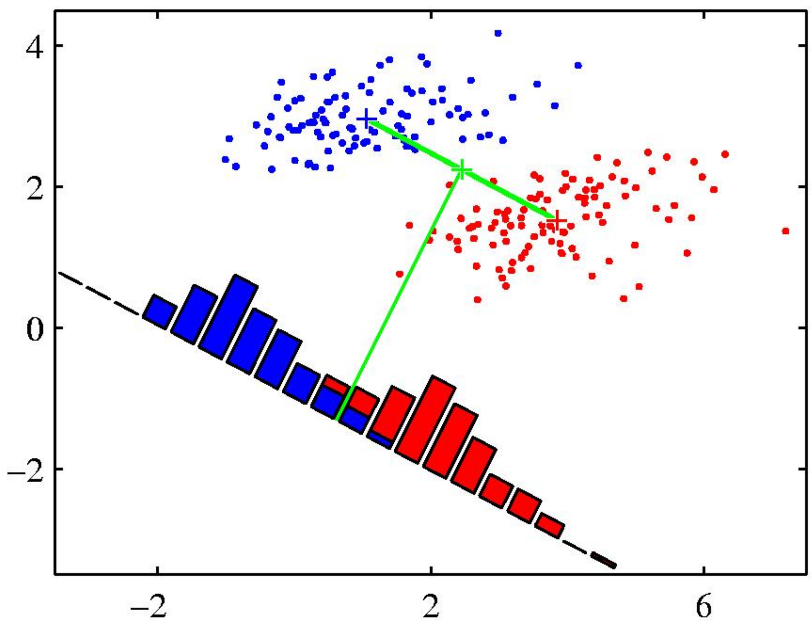
• The problem arises from a strongly non-diagonal covariance of the class distributions.

• Fisher's idea: Maximize a function that
  - gives the largest separation between the projected class means,
  - but also gives a small variance within each class, minimizing the class overlap.

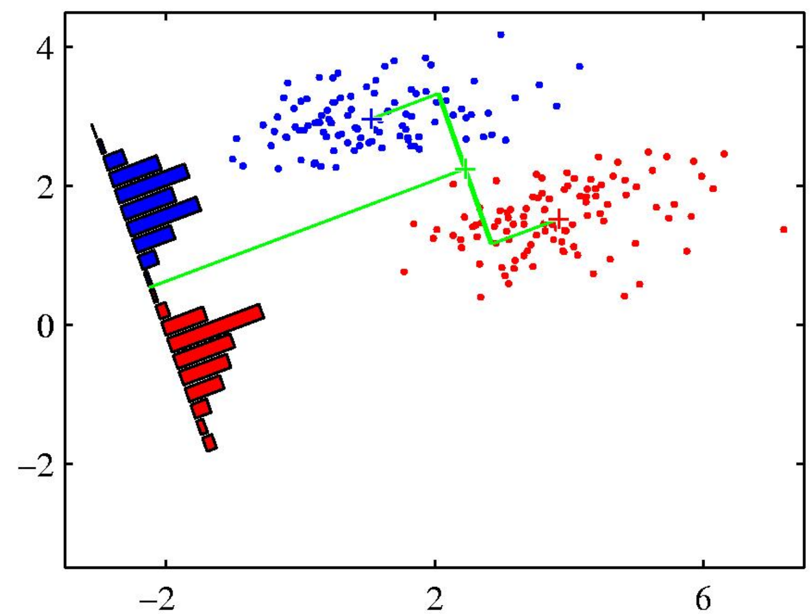


When projected onto the line joining the class means, the classes are not well separated.

Pictorial Illustration



When projected onto the line joining the class means, the classes are not well separated.



Corresponding projection based on the Fisher's linear discriminant.

Fisher's Linear Discriminant

- Let the mean of two classes be given by:

$$\mathbf{m}_1 = \frac{1}{N_1} \sum_{n \in \mathcal{C}_1} \mathbf{x}_n, \quad \mathbf{m}_2 = \frac{1}{N_2} \sum_{n \in \mathcal{C}_2} \mathbf{x}_n,$$

- Projecting onto the vector separating the two classes is reasonable:

$$\mathbf{w} \propto \mathbf{m}_1 - \mathbf{m}_2.$$

- But we also want to minimize within-class variance:

$$s_1^2 = \sum_{n \in \mathcal{C}_1} (y_n - m_1)^2, \quad s_2^2 = \sum_{n \in \mathcal{C}_2} (y_n - m_2)^2,$$

where $m_k = \mathbf{w}^T \mathbf{m}_k.$

$$y_n = \mathbf{w}^T \mathbf{x}_n.$$

- We can define the total within-class variance be $s_1^2 + s_2^2$.

- Fisher's criterion: maximize the ratio of the between-class variance to within-class variance:

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

between

within

Fisher's Linear Discriminant

- We can make dependence on **w** explicit:

$$J(\mathbf{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2} = \frac{\mathbf{w}^T S_b \mathbf{w}}{\mathbf{w}^T S_w \mathbf{w}},$$

where the between-class and within-class covariance matrices are given by:

$$S_b = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^T,$$

$$S_w = \sum_{n \in \mathcal{C}_1} (\mathbf{x}_n - \mathbf{m}_1)(\mathbf{x}_n - \mathbf{m}_1)^T + \sum_{n \in \mathcal{C}_2} (\mathbf{x}_n - \mathbf{m}_2)(\mathbf{x}_n - \mathbf{m}_2)^T.$$

- Intuition: differentiating with respect to **w**:

$$(\mathbf{w}^T S_b \mathbf{w}) S_w \mathbf{w} = (\mathbf{w}^T S_w \mathbf{w}) S_b \mathbf{w}.$$

scalar factors

is always in the direction of $(\mathbf{m}_2 - \mathbf{m}_1)$.

- Multiplying by S_w^{-1}, the optimal solution is:

$$\mathbf{w} \propto S_w^{-1}(\mathbf{m}_2 - \mathbf{m}_1).$$

Fisher's Linear Discriminant

- Notice that the objective $J(\mathbf{w})$ is invariant with respect to rescaling of the vector $\mathbf{w} \rightarrow \alpha\mathbf{w}$.
- Maximizing

$$J(\mathbf{w}) = \frac{\mathbf{w}^T S_b \mathbf{w}}{\mathbf{w}^T S_w \mathbf{w}}$$

is equivalent to the following constraint optimization problem, known as the generalized eigenvalue problem:

$$\min_{\mathbf{w}} -\mathbf{w}^T S_b \mathbf{w}, \quad \text{subject to } \mathbf{w}^T S_w \mathbf{w} = 1.$$

- Forming the Lagrangian:

$$L = -\mathbf{w}^T S_b \mathbf{w} + \lambda(\mathbf{w}^T S_w \mathbf{w} - 1).$$

- The following equation needs to hold at the solution:

$$2 S_b \mathbf{w} = 2\lambda S_w \mathbf{w}.$$

- The solution is given by the eigenvector of $S_w^{-1} S_b$ that correspond to the largest eigenvalue.

Three Approaches to Classification

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

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

We will consider next.

Probabilistic Generative Models

• Model class conditional densities $p(\mathbf{x}|\mathcal{C}_k)$ separately for each class, as well as the class priors $p(\mathcal{C}_k)$.

• Consider the case of two classes. The posterior probability of class C_1 is given by:

$$p(\mathcal{C}_1|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1) + p(\mathbf{x}|\mathcal{C}_2)p(\mathcal{C}_2)}$$

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

Logistic sigmoid function

where we defined:

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

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

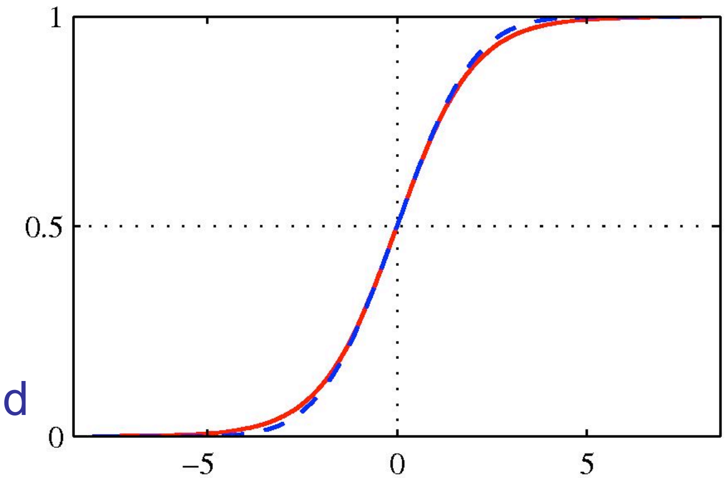
Sigmoid Function

- The posterior probability of class C_1 is given by:

$$p(\mathcal{C}_1|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1) + p(\mathbf{x}|\mathcal{C}_2)p(\mathcal{C}_2)}$$

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

Logistic sigmoid function



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

- It satisfies:

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

Softmax Function

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

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

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

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

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

Example of Continuous Inputs

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

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

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

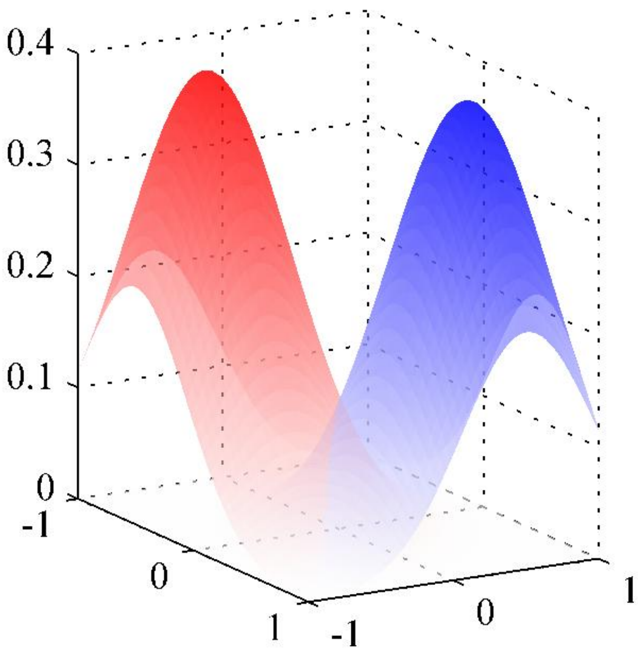
$$p(\mathcal{C}_k|\mathbf{x}) = \sigma(\mathbf{w}^T\mathbf{x} + w_0),$$

where we have defined:

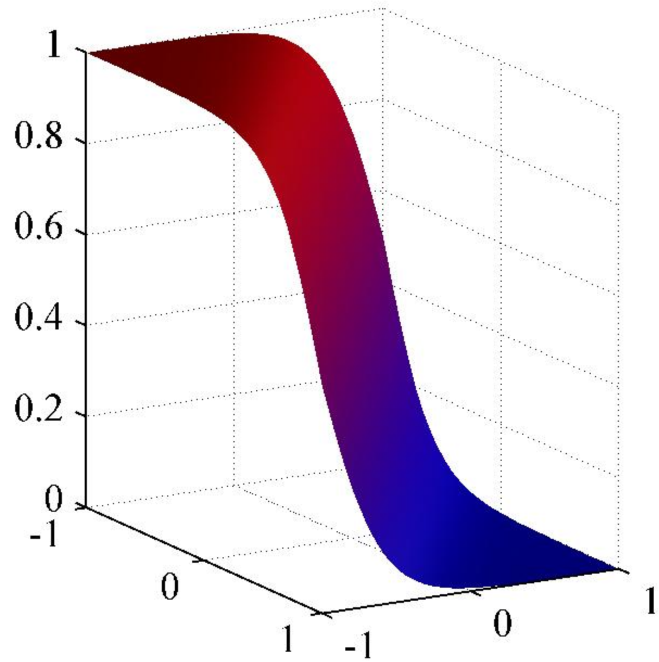
$$\mathbf{w} = \mathbf{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2),$$
$$w_0 = -\frac{1}{2}\boldsymbol{\mu}_1^T\mathbf{\Sigma}^{-1}\boldsymbol{\mu}_1 + \frac{1}{2}\boldsymbol{\mu}_2^T\mathbf{\Sigma}^{-1}\boldsymbol{\mu}_2 + \ln\frac{p(\mathcal{C}_1)}{p(\mathcal{C}_2)}.$$

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

Example of Two Gaussian Models



Class-conditional densities for two classes



The corresponding posterior probability $p(\mathcal{C}_1|\mathbf{x})$, given by the sigmoid function of a linear function of **x**.

Case of K Classes

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

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

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

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

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

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

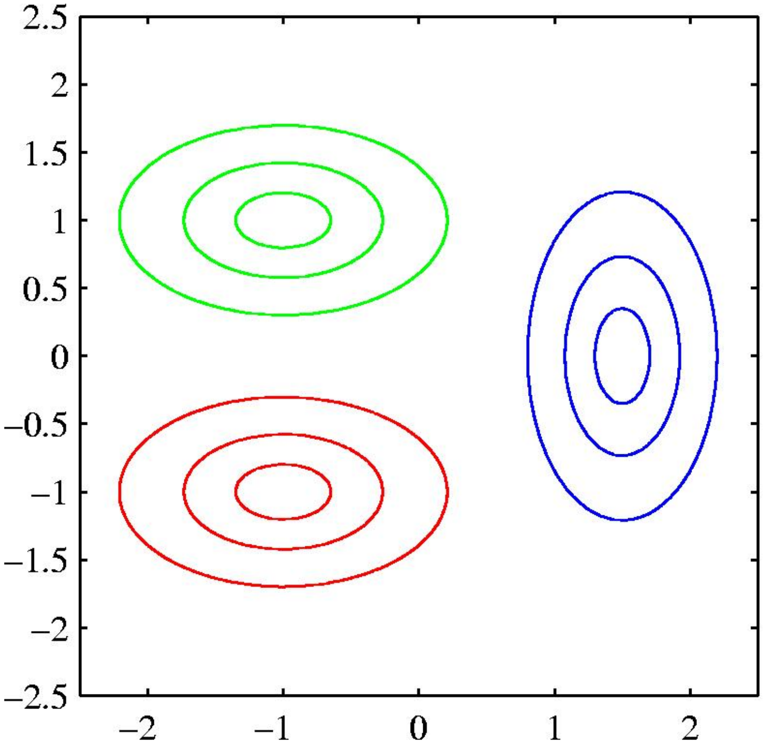
- Again, the decision boundaries are linear in input space.

- If we allow each class-conditional density to have its own covariance, we will obtain quadratic functions of **x**.

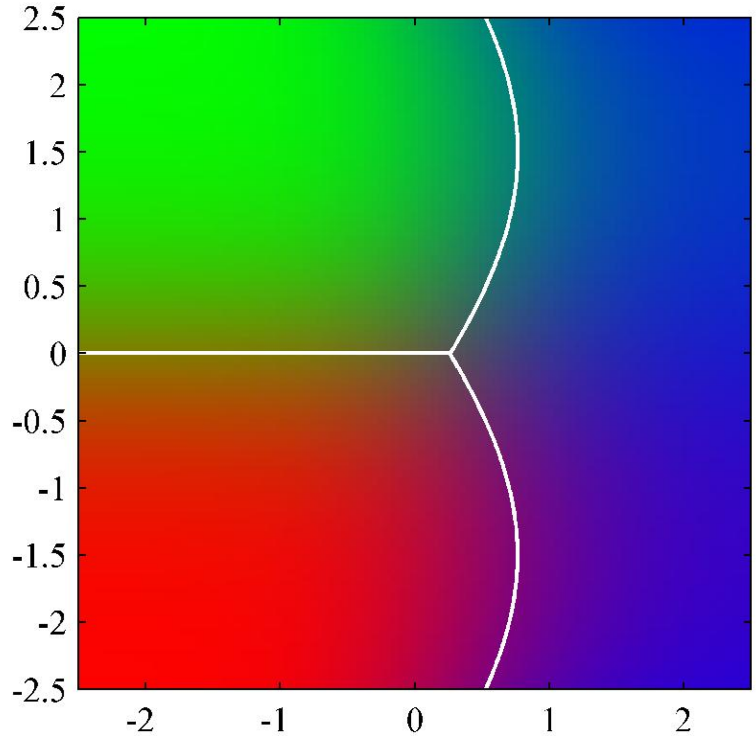
- This leads to a quadratic discriminant.

Quadratic Discriminant

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



Class-conditional densities for three classes



The corresponding posterior probabilities for three classes.

Maximum Likelihood Solution

- Consider the case of two classes, each having a Gaussian class-conditional density with shared covariance matrix.

- We observe a dataset $\{\mathbf{x}_n, t_n\},\ n = 1, .., N.$
  - Here t_n=1 denotes class C_1, and t_n=0 denotes class C_2.
  - Also denote $p(\mathcal{C}_1) = \pi,\ \ p(\mathcal{C}_2) = 1 - \pi.$

- The likelihood function takes form:

$$p(\mathbf{t}, \mathbf{X} | \pi, \boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \boldsymbol{\Sigma}) = \prod_{n=1}^{N} \Big[\underbrace{\pi \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_1, \boldsymbol{\Sigma})}_{\text{Data points from class } C_1.} \Big]^{t_n} \Big[\underbrace{(1-\pi) \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_2, \boldsymbol{\Sigma})}_{\text{Data points from class } C_2.} \Big]^{1-t_n}.$$

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

Maximum Likelihood Solution

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

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

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

Differentiating, we get:

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

- Maximizing the respect to μ_1, we look at the terms of the log-likelihood functions that depend on μ_1:

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

Differentiating, we get:

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

And similarly:

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

Maximum Likelihood Solution

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

- Maximizing the respect to Σ:

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

- Here we defined:

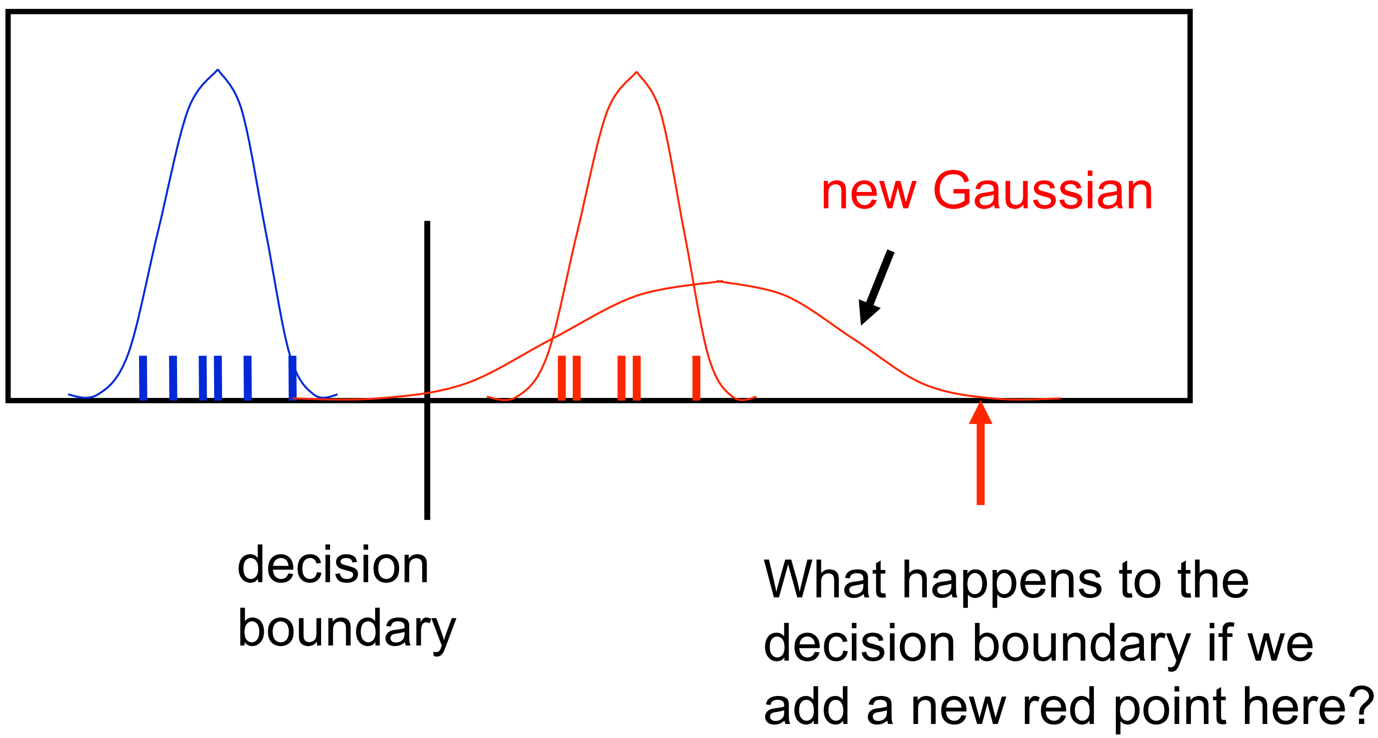
$$\begin{aligned}
\mathbf{S} &= \frac{N_1}{N}\mathbf{S}_1 + \frac{N_2}{N}\mathbf{S}_2, \\
\mathbf{S}_1 &= \frac{1}{N_1}\sum_{n\in\mathcal{C}_1}(\mathbf{x}_n - \boldsymbol{\mu}_1)(\mathbf{x}_n - \boldsymbol{\mu}_1)^T, \\
\mathbf{S}_2 &= \frac{1}{N_2}\sum_{n\in\mathcal{C}_2}(\mathbf{x}_n - \boldsymbol{\mu}_2)(\mathbf{x}_n - \boldsymbol{\mu}_2)^T.
\end{aligned}$$

- Using standard results for a Gaussian distribution we have:

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

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

Example



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

Three Approaches to Classification

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

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

We will consider next.

Fixed Basis Functions

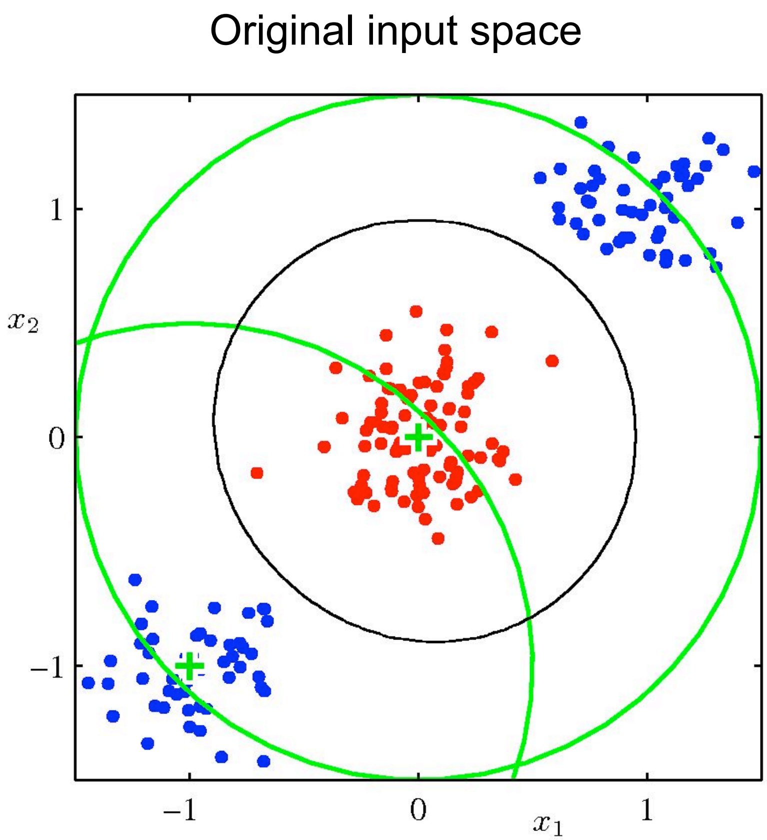
• So far, we have considered classification models that work directly in the input space.

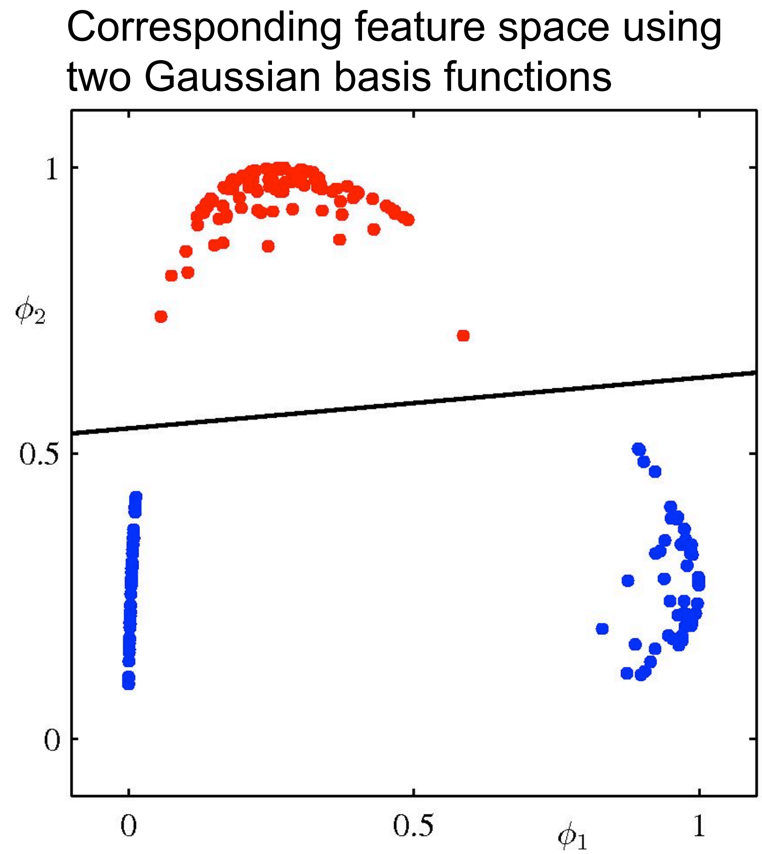
• All considered algorithms are equally applicable if we first make a fixed nonlinear transformation of the input space using vector of basis functions $\phi(\mathbf{x})$.

• Decision boundaries will be linear in the feature space $\phi,$ but would correspond to nonlinear boundaries in the original input space **x**.

• Classes that are linearly separable in the feature space $\phi(\mathbf{x})$ need not be linearly separable in the original input space.

Linear Basis Function Models





- We define two Gaussian basis functions with centers shown by green the crosses, and with contours shown by the green circles.

- Linear decision boundary (right) is obtained using logistic regression, and corresponds to nonlinear decision boundary in the input space (left, black curve).

Logistic Regression

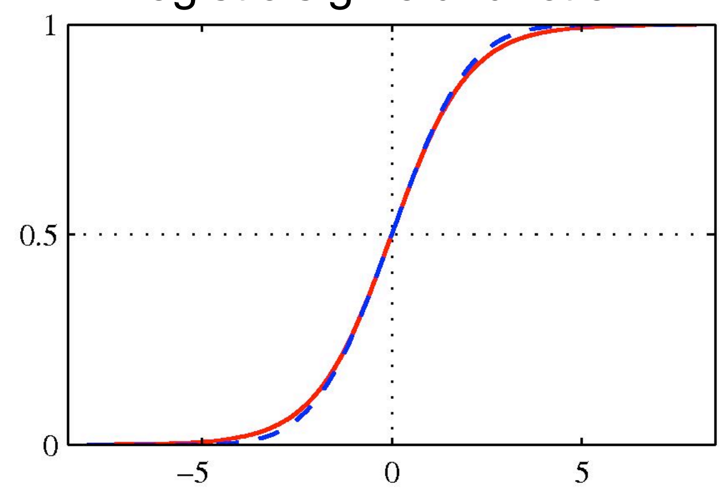
- Let us look at the two-class classification problem.
- We have seen that the posterior probability of class C_1 can be written as a sigmoid function:

$$p(\mathcal{C}_1|\mathbf{x}) = \frac{1}{1+\exp(-\mathbf{w}^T\mathbf{x})} = \sigma(\mathbf{w}^T\mathbf{x}),$$

where $p(\mathcal{C}_2|\mathbf{x}) = 1 - p(\mathcal{C}_1|\mathbf{x}),$ and we omit the bias term for clarity.

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

logistic sigmoid function



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

Here we model $p(\mathcal{C}_k|\mathbf{x})$ directly.

ML for Logistic Regression

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

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

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

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

- Differentiating and using the chain rule:

$$\frac{\mathrm{d}}{\mathrm{d}y_n} E_n = \frac{y_n - t_n}{y_n(1 - y_n)}, \qquad \frac{\mathrm{d}}{\mathrm{d}\mathbf{w}} y_n = y_n(1 - y_n)\mathbf{x}_n, \qquad \boxed{\frac{\mathrm{d}}{\mathrm{d}a}\sigma(a) = \sigma(a)(1 - \sigma(a)).}$$

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

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

ML for Logistic Regression

- We therefore obtain:

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

prediction target

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

Multiclass Logistic Regression

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

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

• Unlike in generative models, here we will use maximum likelihood to determine parameters of this discriminative model directly.

• As usual, we observed a dataset $\{\mathbf{x}_n, t_n\}, \; n = 1, .., N,$ where we use 1-of-K encoding for the target vector $\mathbf{t_n}$.

• So if $\mathbf{x_n}$ belongs to class C_k, then **t** is a binary vector of length K containing a single 1 for element k (the correct class) and 0 elsewhere.

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

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

Multiclass Logistic Regression

- We can write down the likelihood function:

$$p(\mathbf{T}|\mathbf{X}, \mathbf{w}_1, ..., \mathbf{w}_K) = \prod_{n=1}^{N} \underbrace{\left[\prod_{k=1}^{K} p(\mathcal{C}_k|\mathbf{x}_n)^{t_{nk}} \right]}_{\text{Only one term corresponding to correct class contributes.}} = \prod_{n=1}^{N} \left[\prod_{k=1}^{K} y_{nk}^{t_{nk}} \right]$$

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

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

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

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

- Taking the gradient:

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

Special Case of Softmax

- If we consider a softmax function for two classes:

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

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

Recap

- Generative approach: Determine the class conditional densities and class-specific priors, and then use Bayes' rule to obtain the posterior probabilities.
  - Different models can be trained separately on different machines.
  - It is easy to add a new class without retraining all the other classes.

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

- Discriminative approach: Train all of the model parameters to maximize the probability of getting the labels right.

Model $p(\mathcal{C}_k|\mathbf{x})$ directly.

Bayesian Logistic Regression

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

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

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

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

- However, the posterior distribution

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

is no longer Gaussian, and we cannot analytically integrate over the model parameters **w**.

- We need to introduce some approximations.

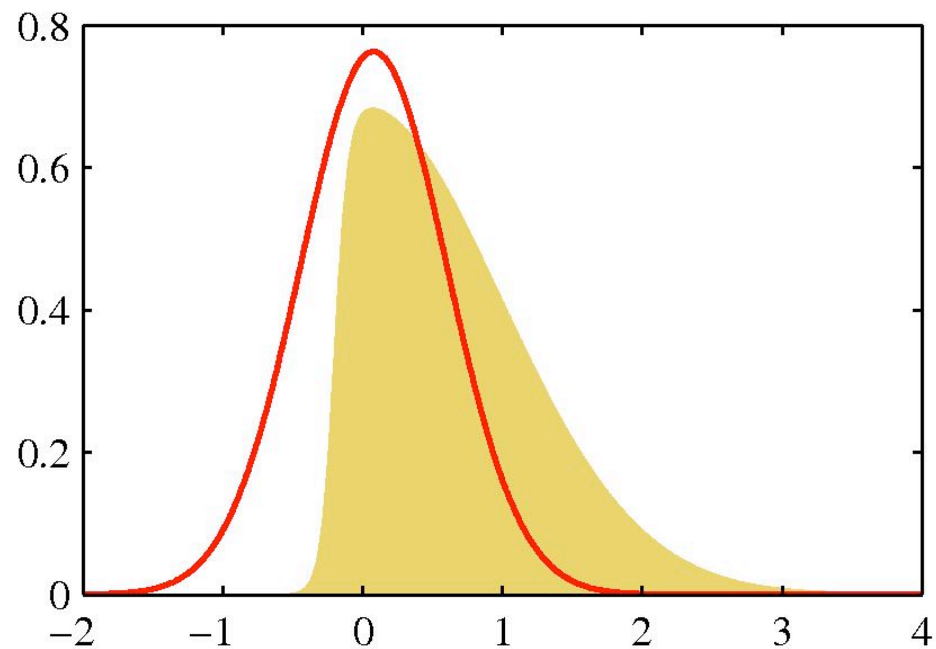
Pictorial illustration

• Consider a simple distribution:

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

• The plot shows the normalized distribution (in yellow), which is not Gaussian.

• The red curve displays the corresponding Gaussian approximation.



Recap: Computational Challenge of Bayesian Framework

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

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

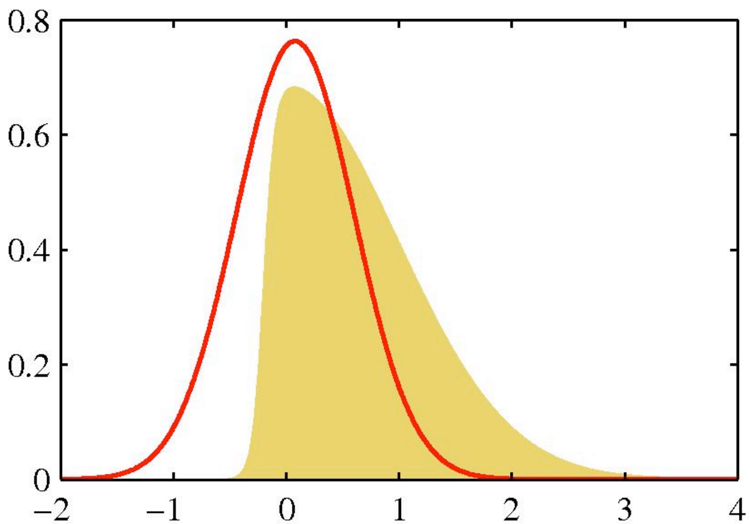
We will consider Laplace approximation next.

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

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

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

Laplace Approximation



- We will use the following notation:

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

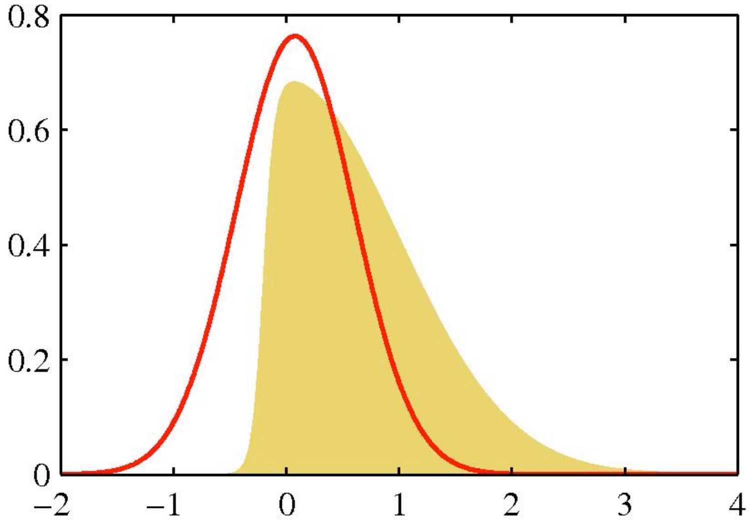
- We can evaluate $\tilde{p}(\mathbf{z})$ point-wise but cannot evaluate $\mathcal{Z}$.

- For example

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

- Goal: Find a Gaussian approximation q(z) which is centered on a mode of the distribution p(z).

Laplace Approximation



- We will use the following notation:

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

- At the stationary point $\mathbf{z}_0$, the gradient $\nabla \tilde{p}(\mathbf{z}_0)$ vanishes.

- Consider a Taylor approximation $\ln \tilde{p}(\mathbf{z})$ around $\mathbf{z}_0$.

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

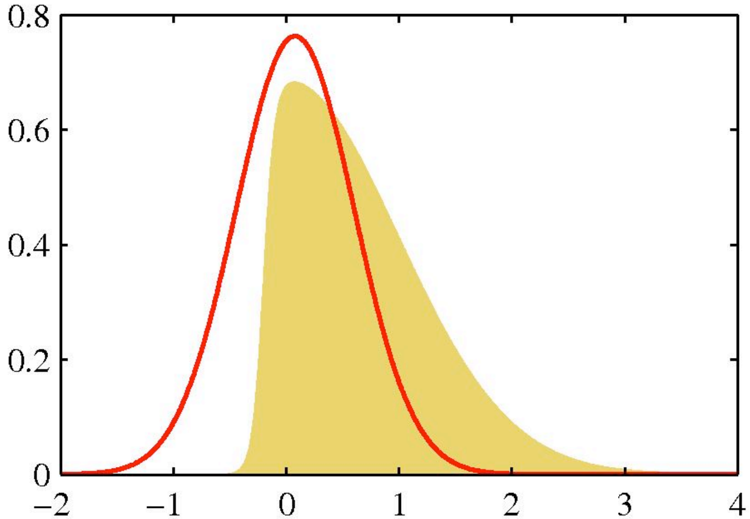
where A is a Hessian matrix:

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

- Exponentiating both sides:

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

Laplace Approximation



- We will use the following notation:

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

- Using Taylor approximation, we get:

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

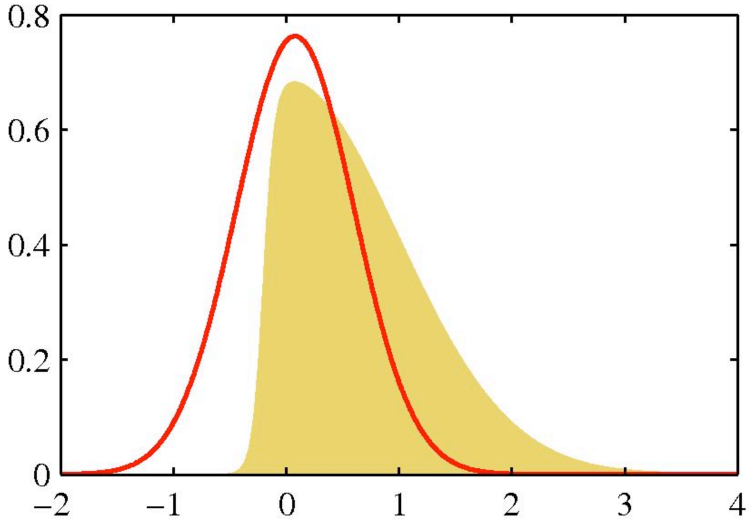
- Hence a Gaussian approximation for $p(\mathbf{z})$ is:

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

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

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

Laplace Approximation



- We will use the following notation:

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

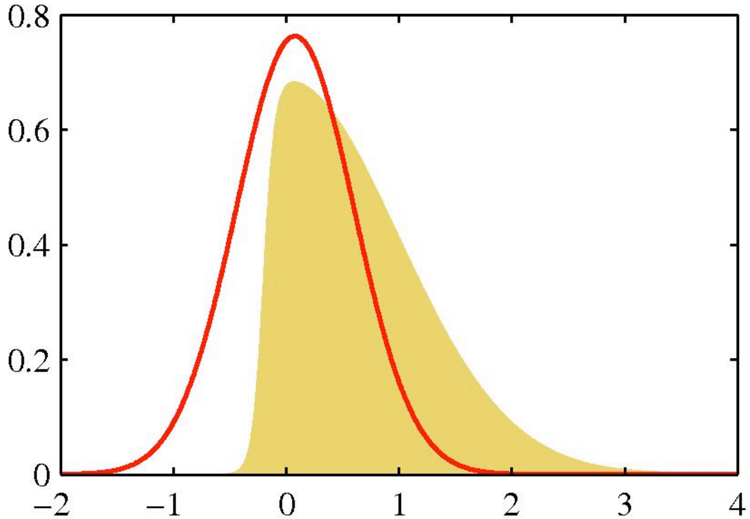
- Using Taylor approximation, we get:

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

- Bayesian inference: $p(\mathbf{w}|\mathcal{D}) = \dfrac{p(\mathcal{D}|\mathbf{w})p(\mathbf{w})}{p(\mathcal{D})}.$
- Identify: $\tilde{p}(\theta|\mathcal{D}) = p(\mathcal{D}|\theta)p(\theta), \;\; \mathcal{Z} = \int p(\mathcal{D}|\theta)p(\theta)\mathrm{d}\theta.$
- The posterior is approximately Gaussian around the MAP estimate:

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

Laplace Approximation



- We will use the following notation:

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

- Using Taylor approximation, we get:

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

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

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

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

Bayesian Information Criterion

- BIC can be obtained from the Laplace approximation:

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

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

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

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

Bayesian Logistic Regression

- Remember the likelihood:

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

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

Log-prior term

- The log of the posterior takes form:

Log-likelihood term

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

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

$$\mathbf{S}_N^{-1} = - \nabla \nabla \ln p(\mathbf{w}|\mathbf{X},\mathbf{t}) = S_0^{-1} + \sum_n y_n (1 - y_n) \mathbf{x}_n \mathbf{x}_n^T.$$

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

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

Predictive Distribution

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

$$\begin{aligned} p(\mathcal{C}_1|\mathbf{x}^*, \mathbf{t}, \mathbf{X}) &= \int p(\mathcal{C}_1|\mathbf{x}^*, \mathbf{w}) p(\mathbf{w}|\mathbf{t}, \mathbf{X}) \mathrm{d}\mathbf{w} \\ &\approx \int \sigma(\mathbf{w}^T \mathbf{x}^*) q(\mathbf{w}) \mathrm{d}\mathbf{w}, \end{aligned}$$

Still not tractable.

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

Predictive Distribution

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

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

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

where δ is the Dirac delta function. Hence

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

1-dimensional integral.

• Let us characterize p(a).

• The delta function imposes a linear constraint on **w**. It forms a marginal distribution from the joint q(**w**) by marginalizing out all directions orthogonal to **x***.

• Since q(**w**) is Gaussian, the marginal is also Gaussian.

Predictive Distribution

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

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

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

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

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

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

- Hence we obtain approximate predictive:

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

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

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