STA414/2104

Statistical Methods for Machine Learning II

Murat A. Erdogdu

Department of Computer Science
Department of Statistical Sciences

Lecture 4

UNIVERSITY OF
TORONTO

Announcements

- Homework 1 is due on Feb 8, 13:59.
  - You should have received your crowdmark invitation already. If not, let me know.
  - TA office hours are posted on the webpage.

Last time

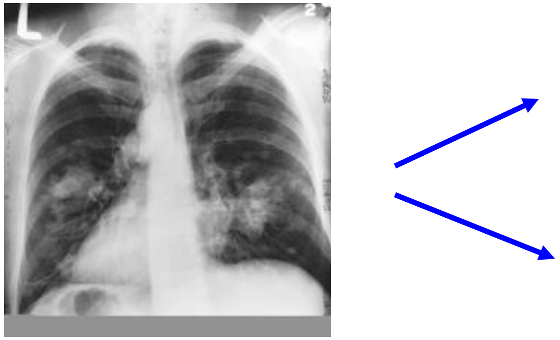
- Basis function models
- Geometry of least squares
- Lasso and ridge regression
- Bayesian Linear Regression
- Equivalent kernel

- **Today:** 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 to only 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:

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

parameters

- 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.

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

fixed nonlinear function:
activation 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.

- These class of models are called generalized linear models

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

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

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

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

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 attempt: Construct a discriminant function that directly maps each input vector to a specific class.

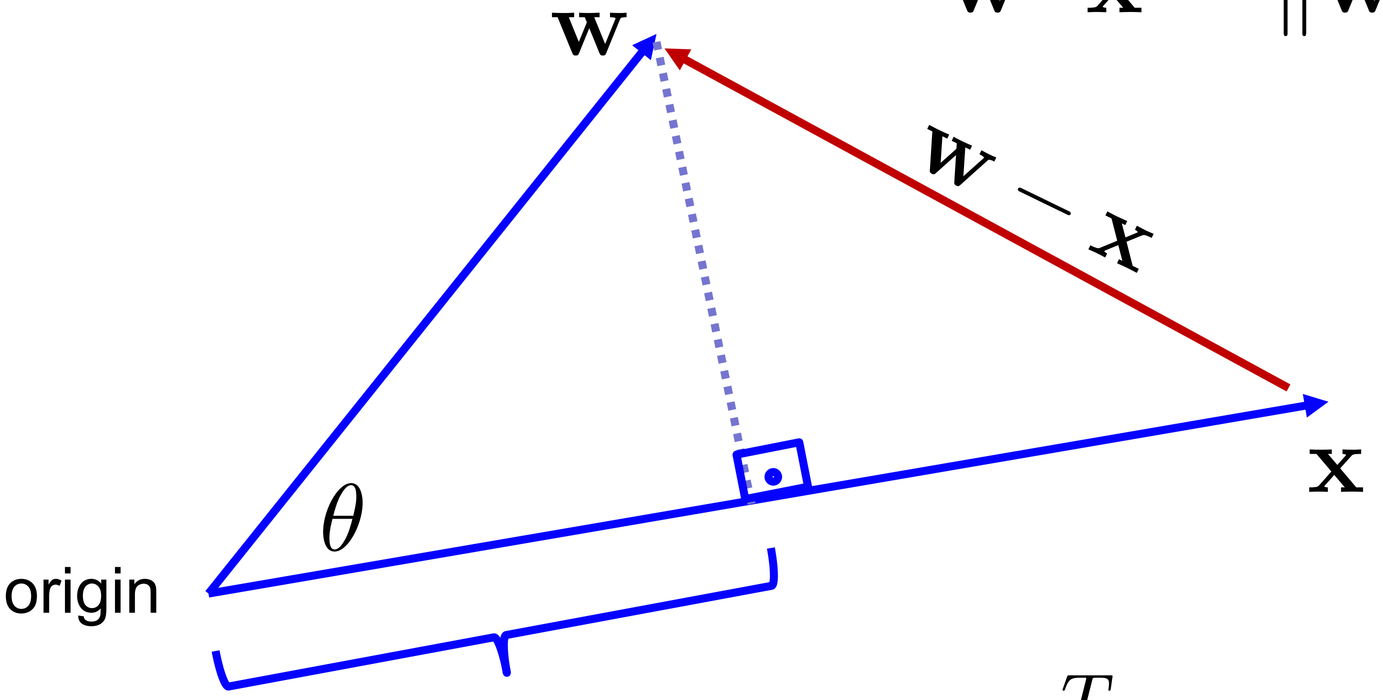
- There are two alternative approaches:
  - Discriminative Approach: Model $p(\mathcal{C}_k|\mathbf{x})$, directly, for example by representing them as parametric models, and optimize for parameters using the training set (e.g. logistic regression).
  - Generative Approach: Model class conditional densities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

Linear algebra refresher

$$\mathbf{w}^T\mathbf{x} = \|\mathbf{w}\|\|\mathbf{x}\|\cos(\theta)$$



$$\|\mathbf{w}\|\cos(\theta) = \frac{\mathbf{w}^T\mathbf{x}}{\|\mathbf{x}\|}$$

Discriminant Functions

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

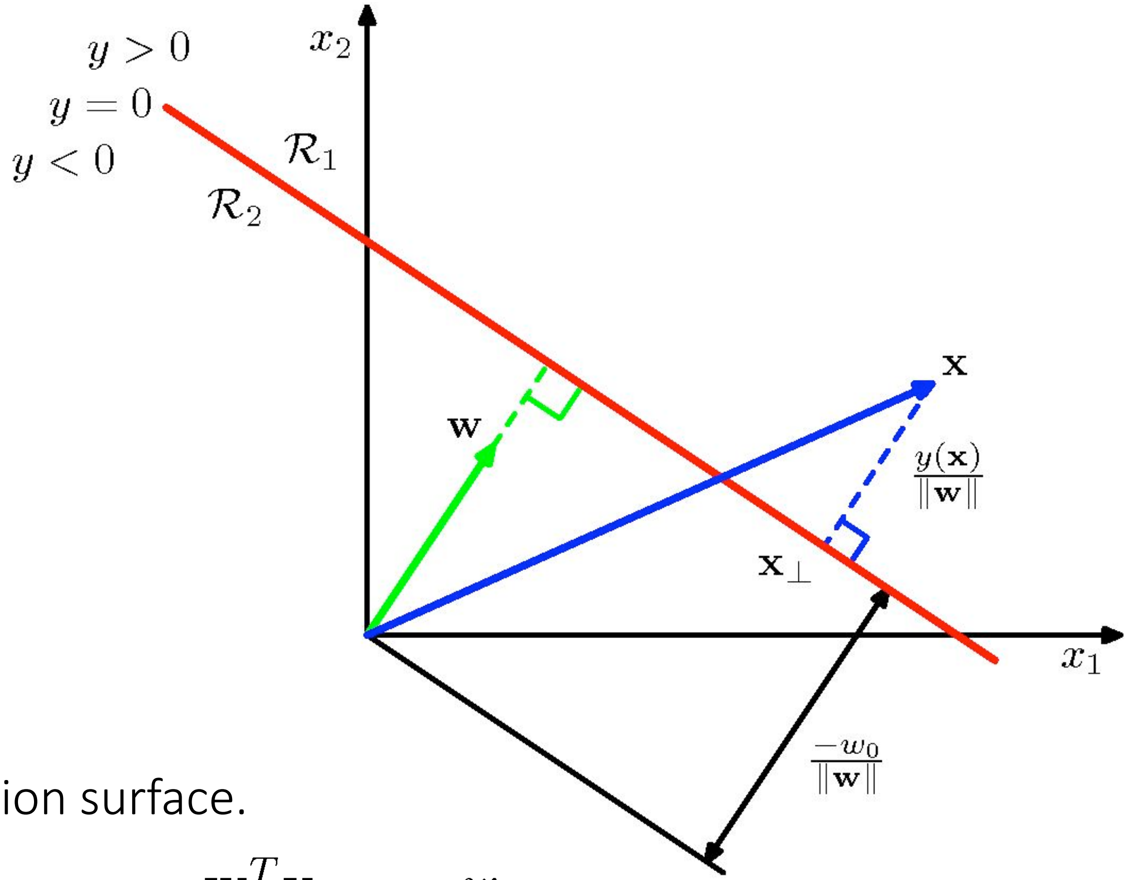
- Assign x to C_1 if $y(\mathbf{x}) \geq 0,$ and class C_2 otherwise.

- Decision boundary:

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

- If two points x_A and x_B lie on the decision surface, then:

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



- The w is orthogonal to the decision surface.

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

- Hence w_0 determines the location of the decision surface.

Multiple Classes

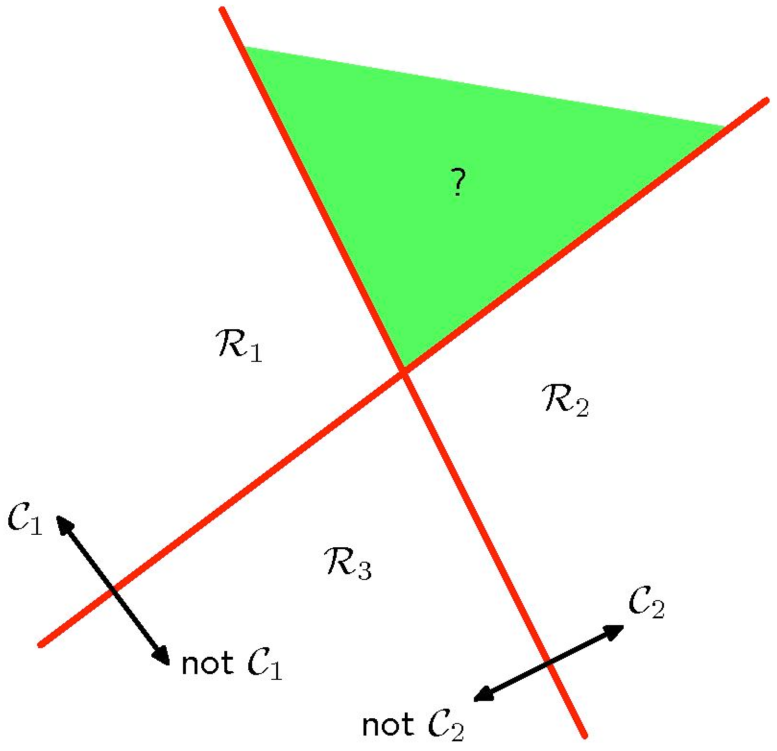
- Consider the extension of linear discriminants to K>2 classes.

- 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



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 connected and convex.

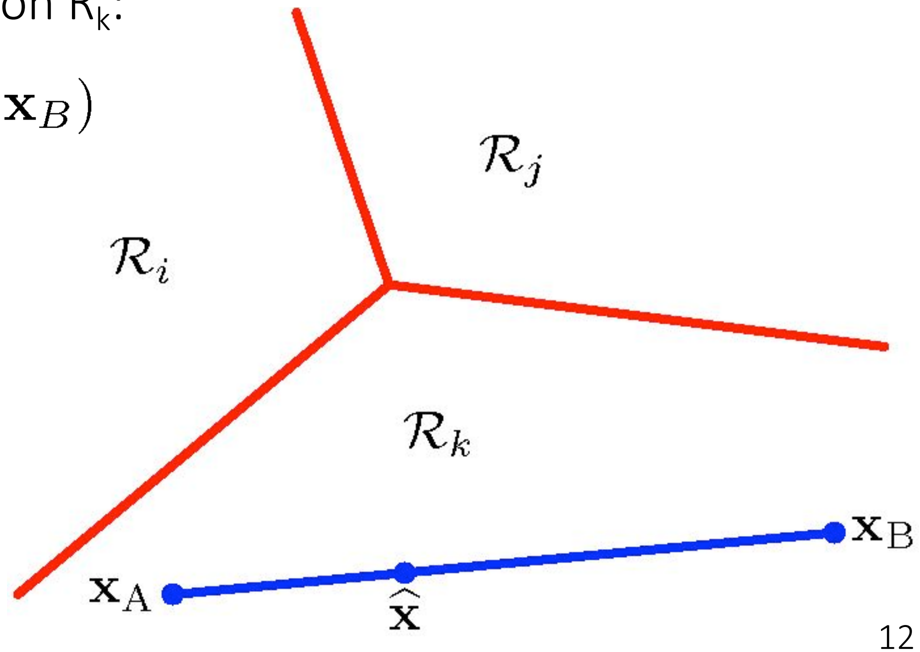
- For any two points that lie inside the region R_k:

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

implies that

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

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** = $[0,0,0,1,0,0]^T$.

- 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:

Kx1 target

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

(D+1) x 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. $(\mathbf{x}_n, \mathbf{t}_n)$, where n=1,…,N.
- Define a matrix $\mathbf{T}$ with n-th row $\mathbf{t}_n^T$, and a matrix $\tilde{\mathbf{X}}$ with n-th row $\mathbf{x}_n^T$
- We have already seen how to do least squares. Least squares minimizes sum of the square of the errors.

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

- 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

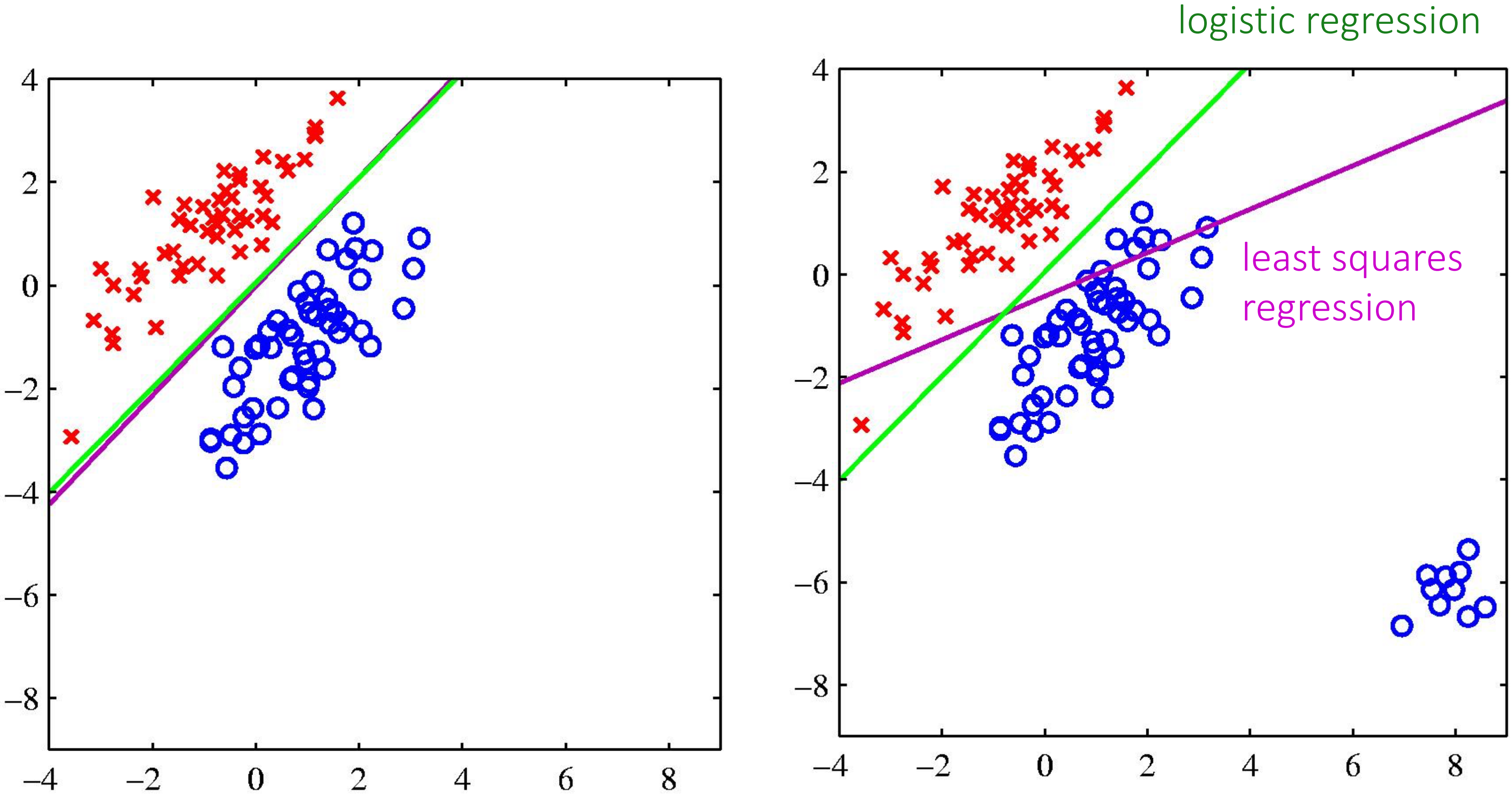
N x (D+1) input matrix whose nth row is $\tilde{\mathbf{x}}_n^T$.

N x K target matrix whose nth 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.

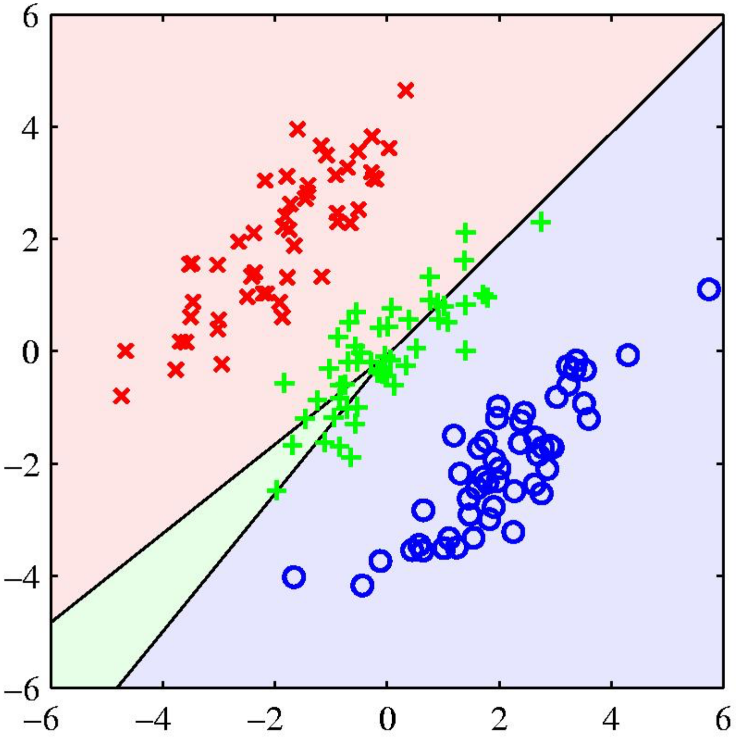
Problems using Least Squares

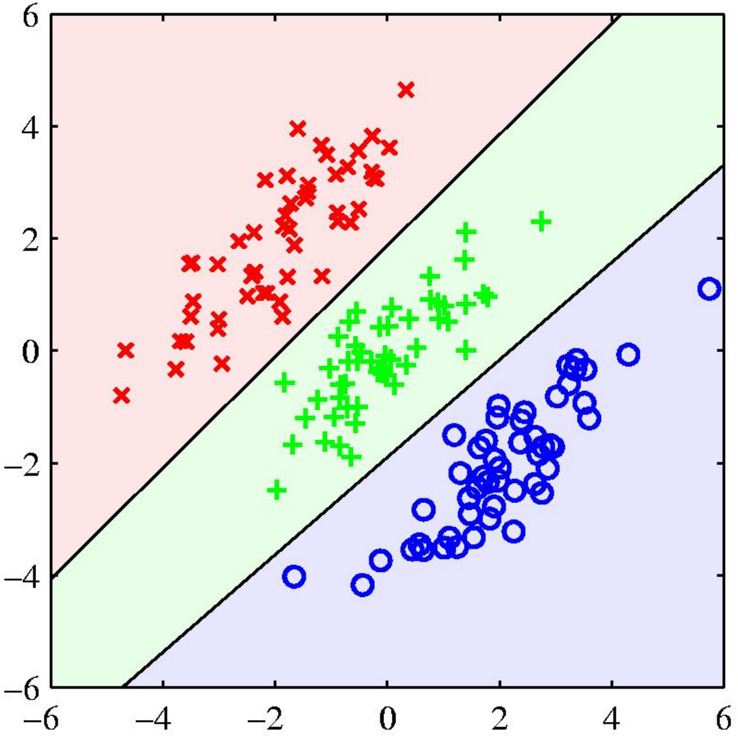
Least squares is highly sensitive to `outliers’, unlike logistic regression



Problems using Least Squares

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


Least squares discriminant


Logistic regression

Many green points are misclassified. Is this surprising?

Fisher’s Linear Discriminant

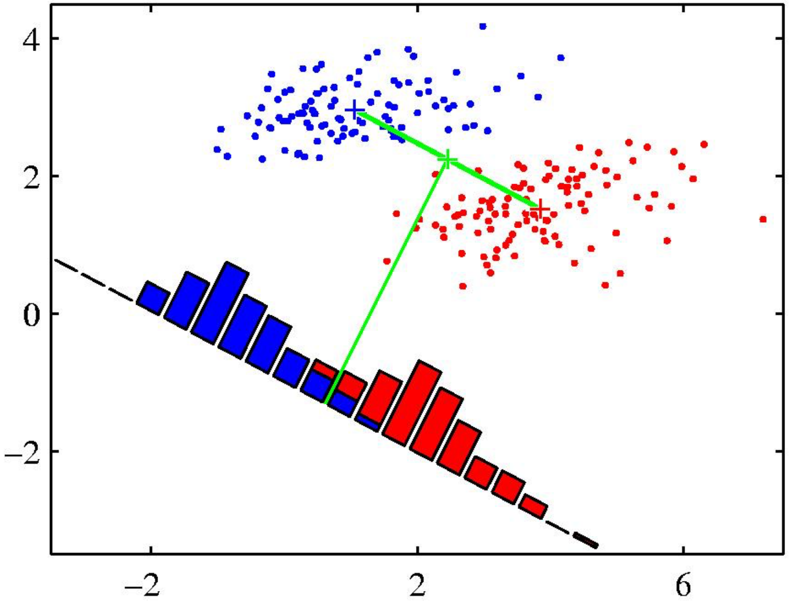
- Dimensionality reduction: Suppose we take a D-dim input vector and project it down to one dimension using:

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

- 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.

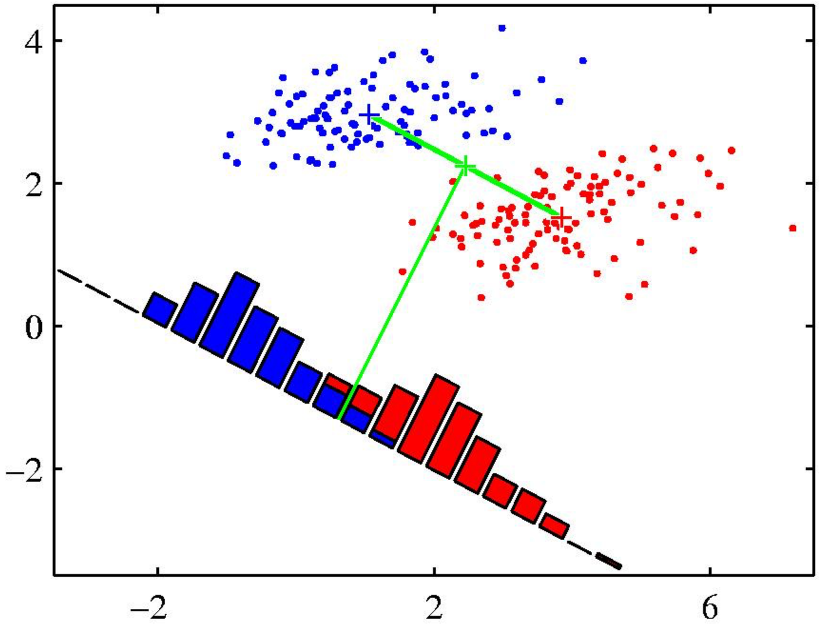
- The problem arises from strongly non-diagonal covariance of the class distributions.

- 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 class overlap.

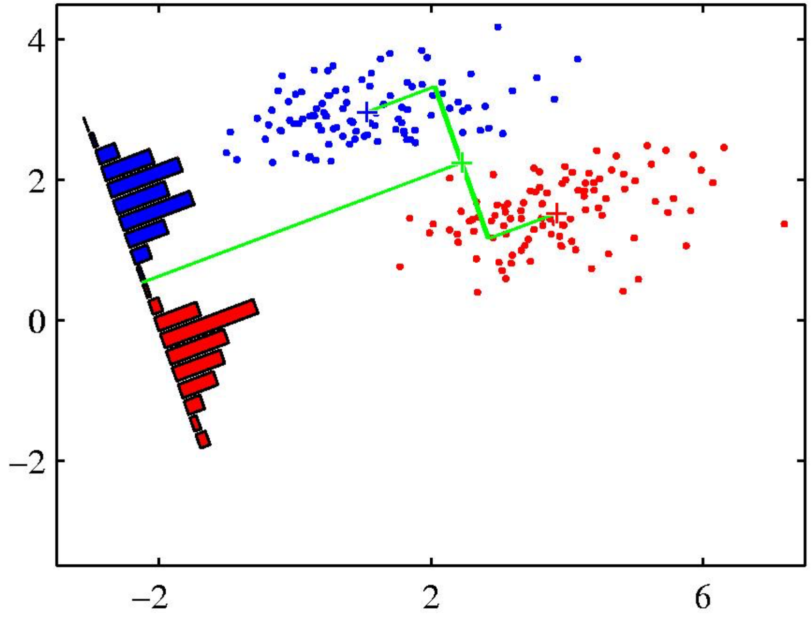


When projected onto the line joining the class means, the classes are not well separated.

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 (say it is a unit vector for now):

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

- But we also want to minimize the 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 ratio of the between-class variance to within-class variance:

between

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

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:

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

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

- Notice that the objective J(w) is invariant with respect to rescaling of the vector $\mathbf{w} \rightarrow \alpha\mathbf{w}$.
- Intuition: differentiating J(w) with respect to w and setting it equal to 0:

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

scalar factors

$$\mathbf{w} \propto S_w^{-1} S_b \mathbf{w}$$
$$= S_w^{-1} (\mathbf{m}_2 - \mathbf{m}_1) \underbrace{(\mathbf{m}_2 - \mathbf{m}_1)^T \mathbf{w}}_{\text{scalar factor}}$$

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 $(\mathbf{m}_2 - \mathbf{m}_1)$.

- The optimal direction 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. We get the same direction as in previous slide.

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:

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

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 ratio of probabilities of two classes, also known as the log-odds.

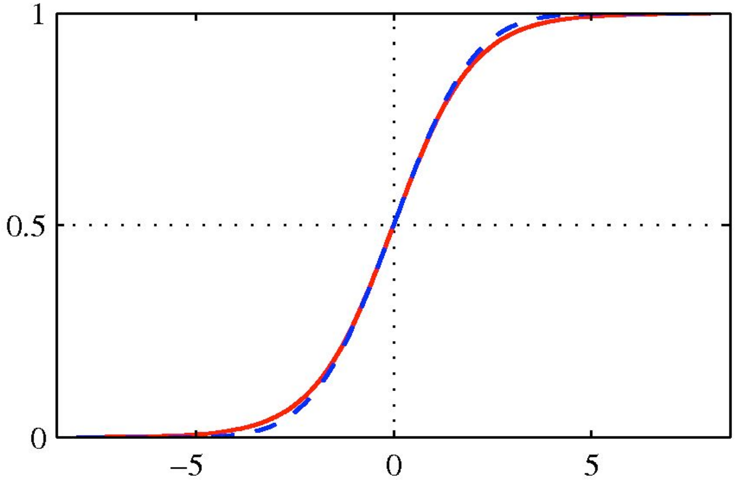
Sigmoid Function

- 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:

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

verify

Softmax Function

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

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

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

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

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

Example of Continuous Inputs

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

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

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

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

where we have defined:

verify

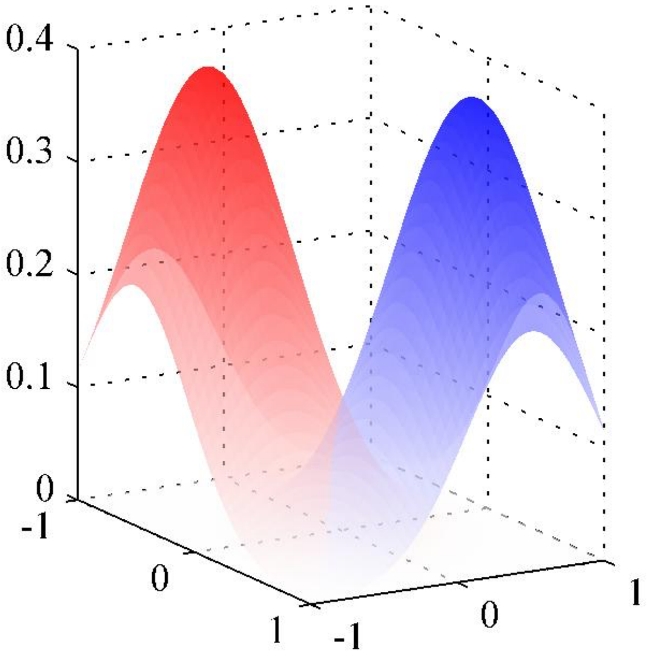
Compare to Fisher's discriminant!

$$\mathbf{w} = \mathbf{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2),$$

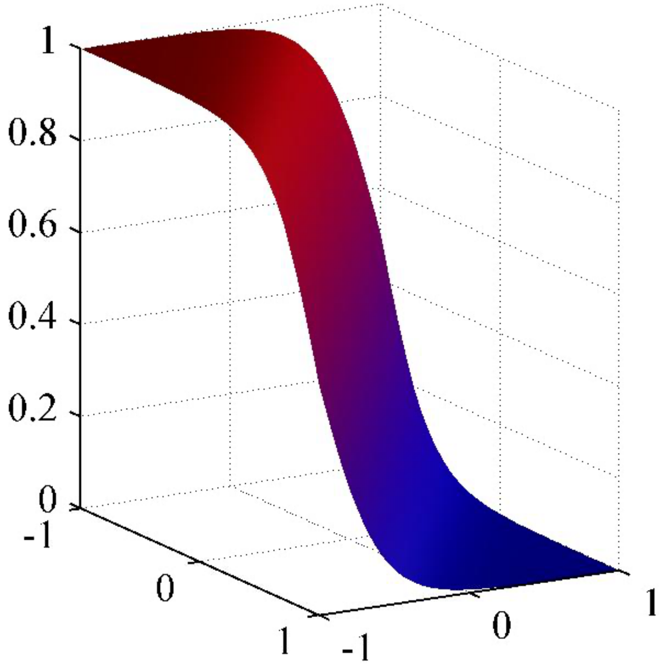
$$w_0 = -\frac{1}{2}\boldsymbol{\mu}_1^T\mathbf{\Sigma}^{-1}\boldsymbol{\mu}_1 + \frac{1}{2}\boldsymbol{\mu}_2^T\mathbf{\Sigma}^{-1}\boldsymbol{\mu}_2 + \ln\frac{p(\mathcal{C}_1)}{p(\mathcal{C}_2)}.$$

- 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:

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

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

verify

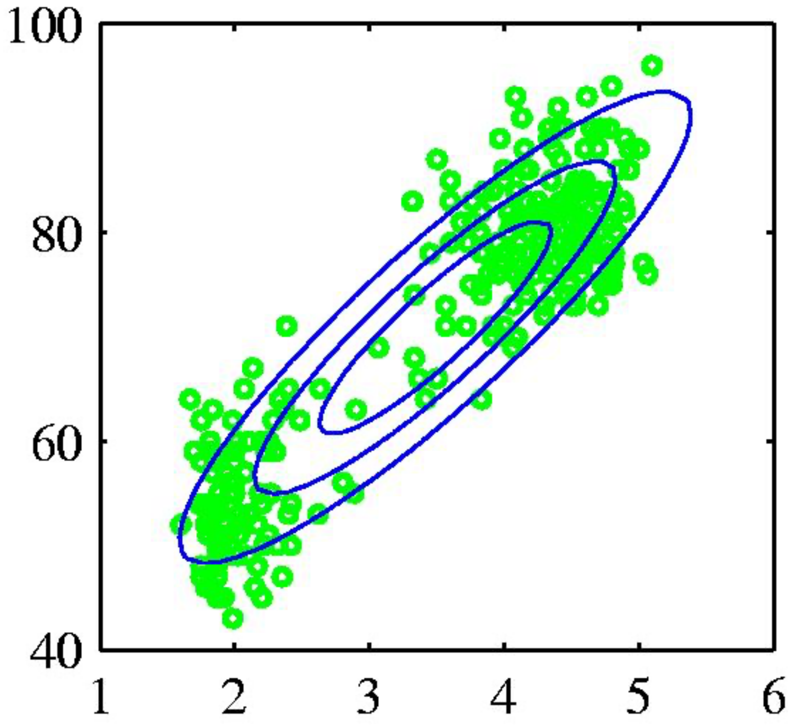
- Again, the decision boundaries are linear in input space.

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

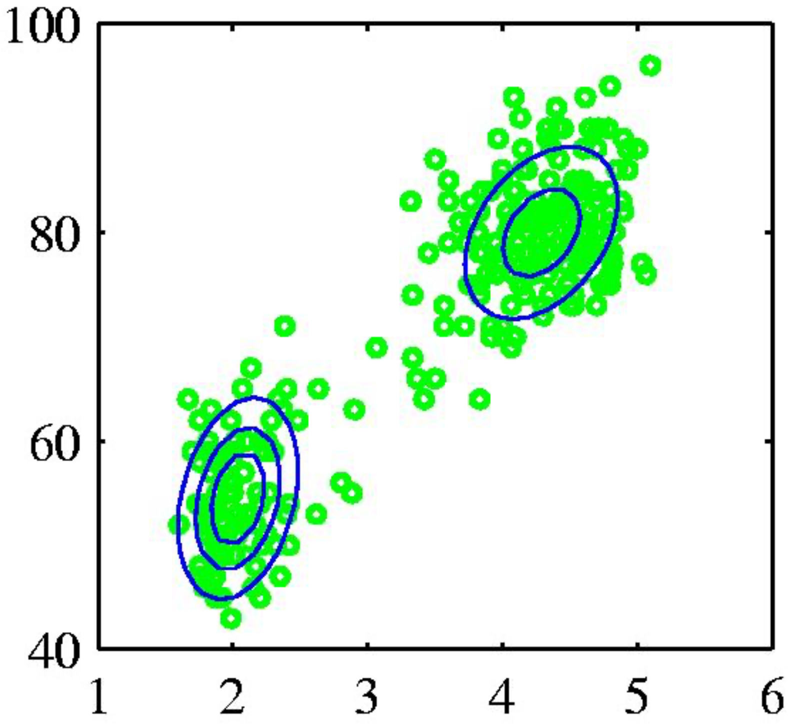
- This leads to a quadratic discriminant.

Mixture of Gaussians

- When modeling real-world data, Gaussian assumption may not be appropriate.
- Consider the following example: Old Faithful Dataset



Single Gaussian



Mixture of two Gaussians

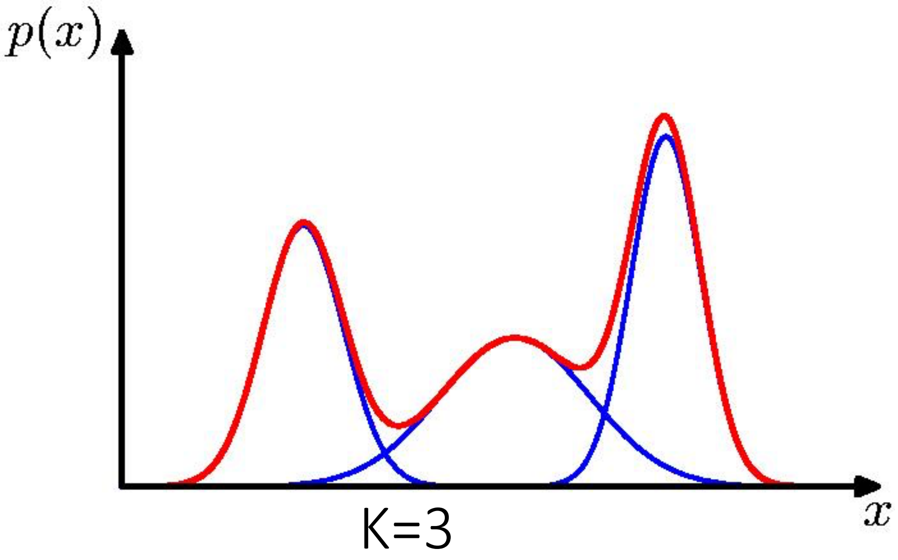
Mixture of Gaussians

- We can combine simple models into a complex model by defining a superposition of K Gaussian densities of the form:

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \underbrace{\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}_{\text{Component}}$$

Mixing coefficient

$$\forall k : \pi_k \geqslant 0 \qquad \sum_{k=1}^{K} \pi_k = 1$$

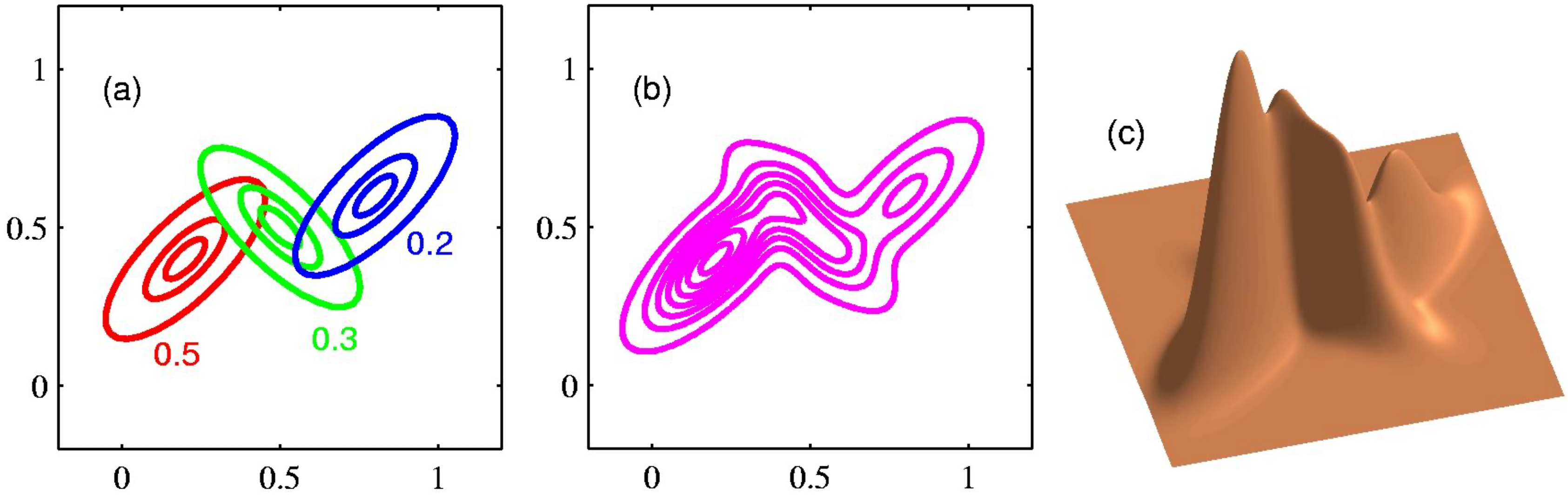


- Note that each Gaussian component has its own mean and covariance. The parameters π_k are called mixing coefficients.

- Mote generally, mixture models can comprise linear combinations of other distributions.

Mixture of Gaussians

- Illustration of a mixture of 3 Gaussians in a 2-dimensional space:



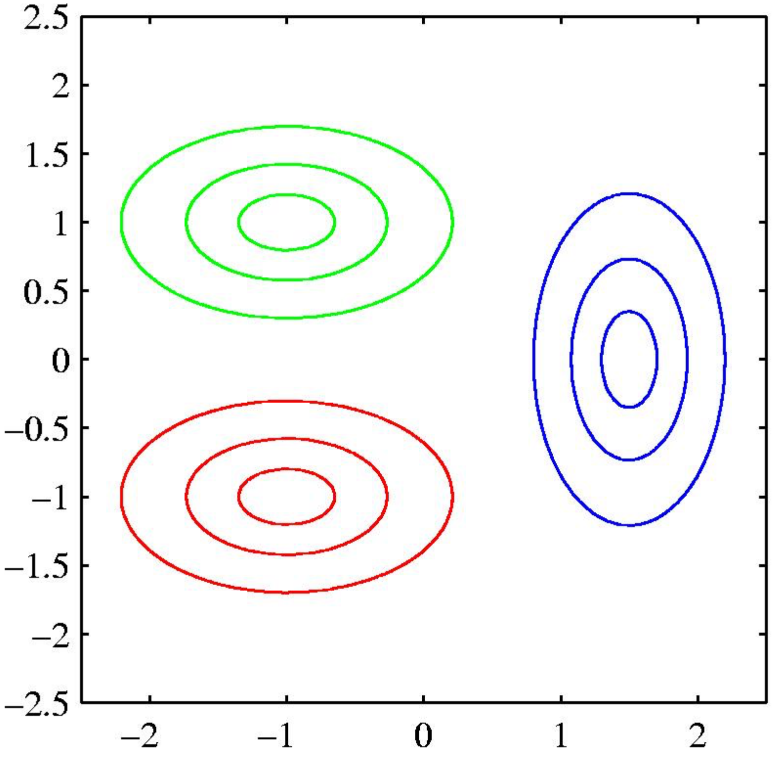
(a) Contours of constant density of each of the mixture components, along with the mixing coefficients

(b) Contours of marginal probability density $p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$

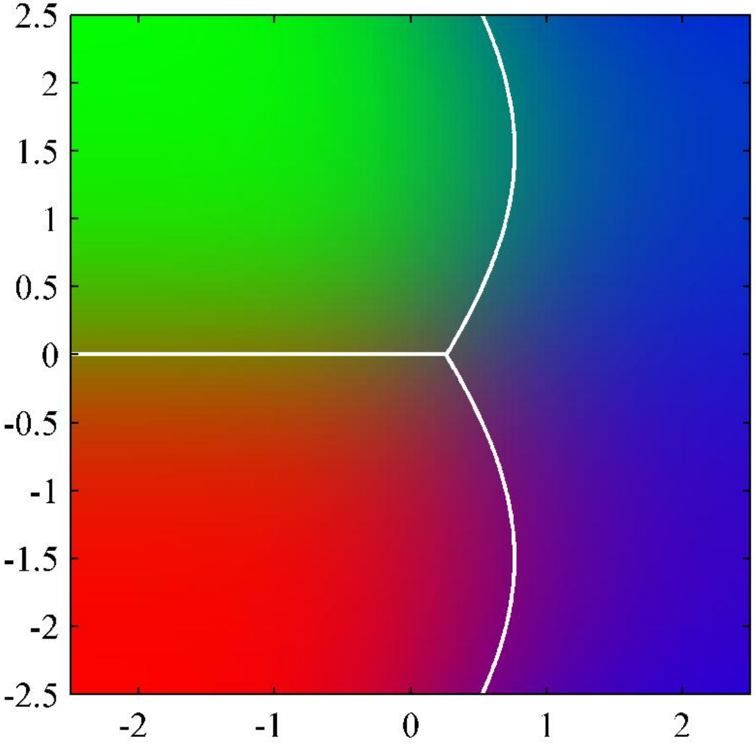
(c) A surface plot of the distribution p(x).

Quadratic Discriminant

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

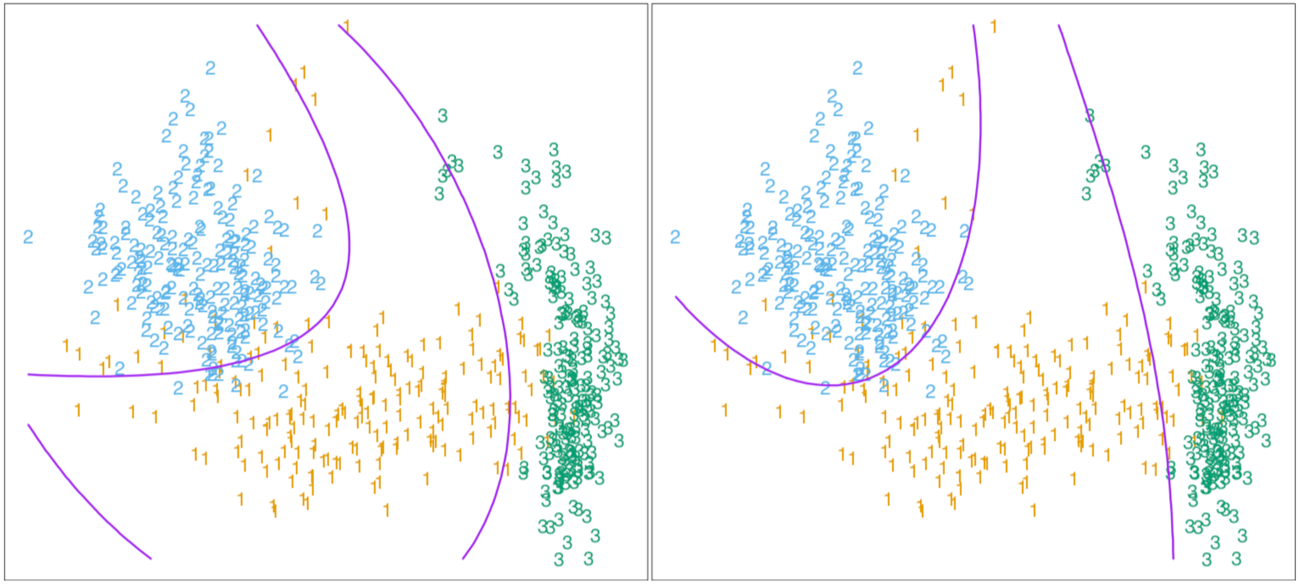


Class-conditional densities for three classes



The corresponding posterior probabilities for three classes.

Quadratic decision boundaries



Two methods for fitting quadratic boundaries. [Left] Quadratic decision boundaries, obtained using LDA in the five-dimensional "quadratic" space. [Right] Quadratic decision boundaries found by QDA. The differences are small, as is usually the case.

Maximum Likelihood Estimation

- Consider the case where each class having a Gaussian class-conditional density with shared covariance matrix.

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

- 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 $p(\mathcal{C}_1) = \pi,$ and $p(\mathcal{C}_2) = 1 - \pi.$
- The likelihood function takes form:

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

Data points from class C_1.

Data points from class C_2.

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

Maximum Likelihood Solution

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

- Maximizing with 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 with 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 : The log-likelihood reads,

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

- Here we defined:

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

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

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

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:

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

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

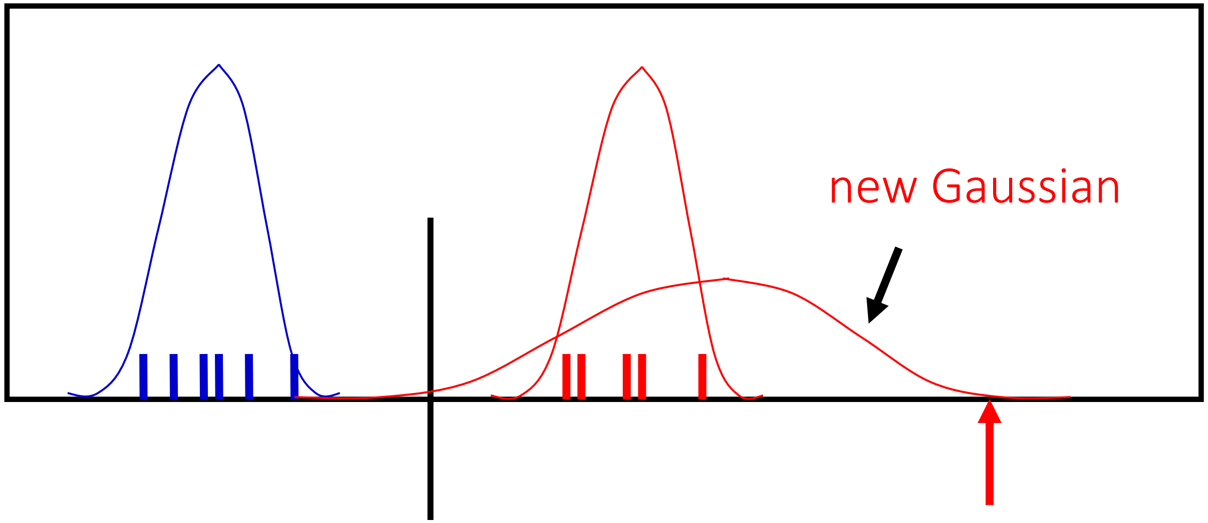
$$\mathbf{S}_2 = \frac{1}{N_2}\sum_{n\in\mathcal{C}_2}(\mathbf{x}_n - \boldsymbol{\mu}_2)(\mathbf{x}_n - \boldsymbol{\mu}_2)^T.$$

- Using standard matrix derivative rules, we have:

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

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

Example



decision
boundary

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

- For generative fitting, the red mean moves right but the decision boundary moves left! If you believe the data is Gaussian, this is reasonable.

Three Approaches to Classification

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

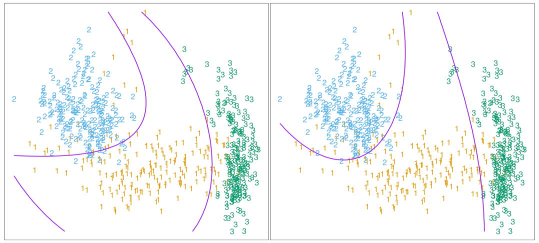
$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\mathbf{x})}.$$

We will consider next.

Fixed Basis Functions

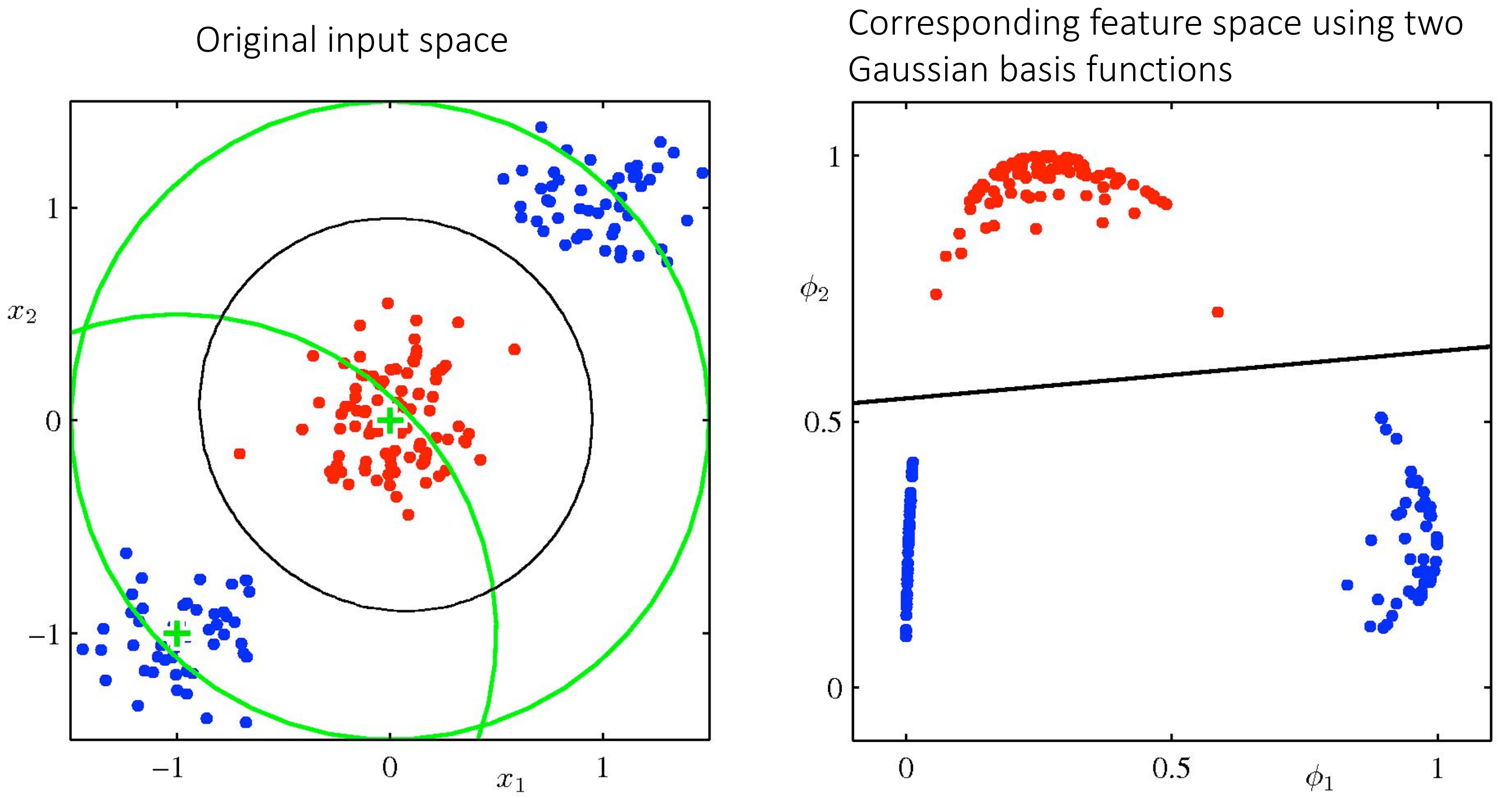
- So far, we have considered classification models that work directly in the input space.
- All considered algorithms are equally applicable if we first make a fixed nonlinear transformation of the input space using vector of basis functions $\phi(\mathbf{x})$.
- Decision boundaries will be linear in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

recall →



Two methods for fitting quadratic boundaries. [Left] Quadratic decision boundaries, obtained using LDA in the five-dimensional "quadratic" space. [Right] Quadratic decision boundaries found by QDA. The differences are small, as is usually the case.

Linear Basis Function Models



Original input space

Corresponding feature space using two Gaussian basis functions

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

Logistic Regression

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

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

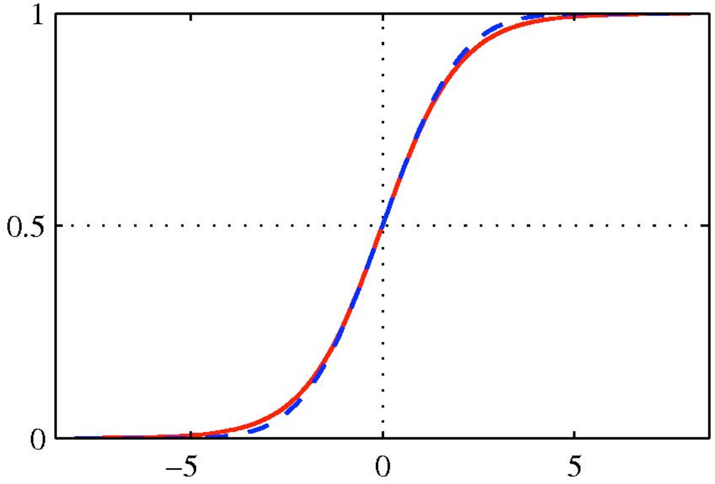
where $p(\mathcal{C}_2|\mathbf{x}) = 1 - p(\mathcal{C}_1|\mathbf{x}),$ we omit the bias term for clarity.

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

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

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

logistic sigmoid function



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 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:

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

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

verify

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

ML for Logistic Regression

- We therefore obtain:

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

prediction target

- This takes 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.
- Coefficients diverge if data is perfectly separable.

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 t_n.

- So if 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 x K binary matrix of target variables.

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

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

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

- 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.