STA 414 / STA2104

Lecture 5

Mark Ebden
Department of Statistics
University of Toronto

With thanks to Russ Salakhutdinov

Outline

- Unit 4’s Bayesian model comparison
- Three approaches to (binary- or multi-class-) classification:

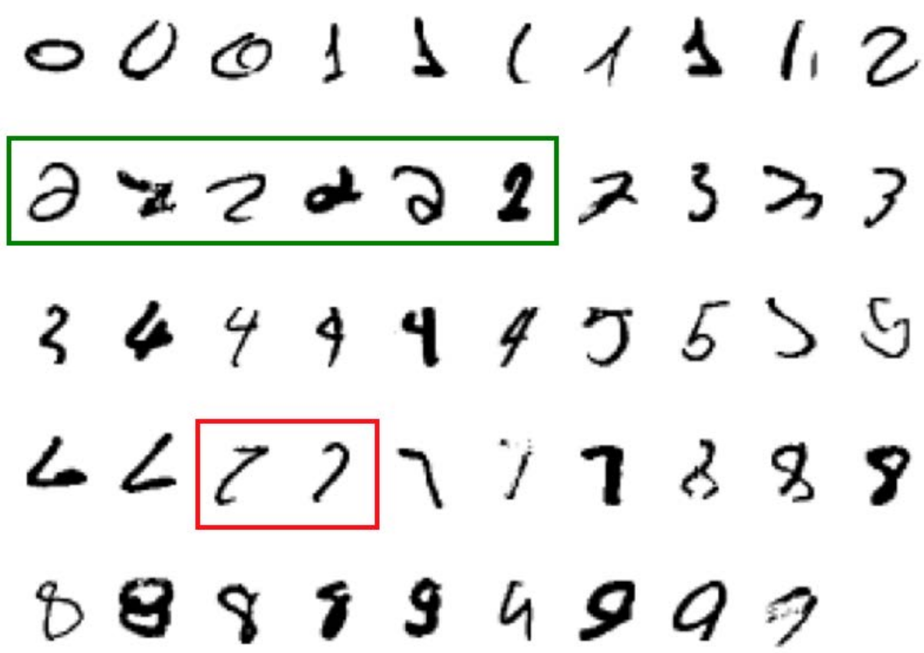
  1. Non-probabilistic: e.g. Least-squares, Fisher’s linear discriminant
  2. Discriminative: e.g. Logistic regression
  3. Generative: e.g. Gaussian discriminative analysis

- Midterm review

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 analogous class of models for solving **classification** problems



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

e.g. **x** is a 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 $k = 1, \ldots, K$

- 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 the parameters

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

adaptive parameters

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

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

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

making the same decisions

- 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

passing linear function through a nonlinear function

- These kinds of models are called generalized linear models

- Note that these models are no longer linear in the parameters, owing to the presence of a 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

perceptron algorithm

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

- Or we can model the conditional **probability** distribution $p(\mathcal{C}_k|\mathbf{x})$, and then use this distribution to make optimal decisions. There are two ways to do this:

  - 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).

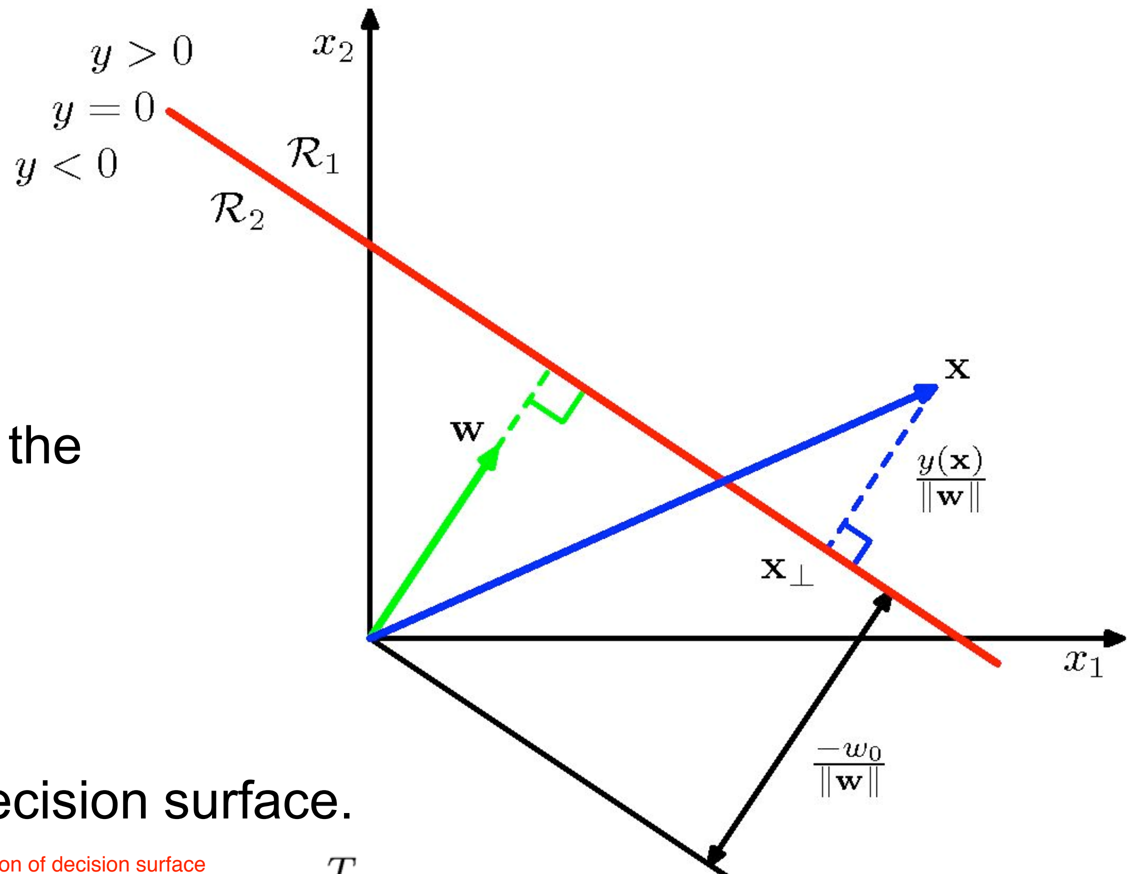
  - Generative Approach: Model class conditional densities $p(\mathbf{x}|\mathcal{C}_k)$ together with the prior probabilities $p(\mathcal{C}_k)$ for the classes. Infer posterior probability using Bayes' rule:

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

  - 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:

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

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

w determines orientation of decision surface

- If **x** is a point on decision surface, then: $\frac{\mathbf{w}^T\mathbf{x}}{||\mathbf{w}||} = -\frac{w_0}{||\mathbf{w}||}$ • w_0 determines the loation of the decision surface
- Hence w_0 determines the distance of the decision surface to the origin

Multiple Classes

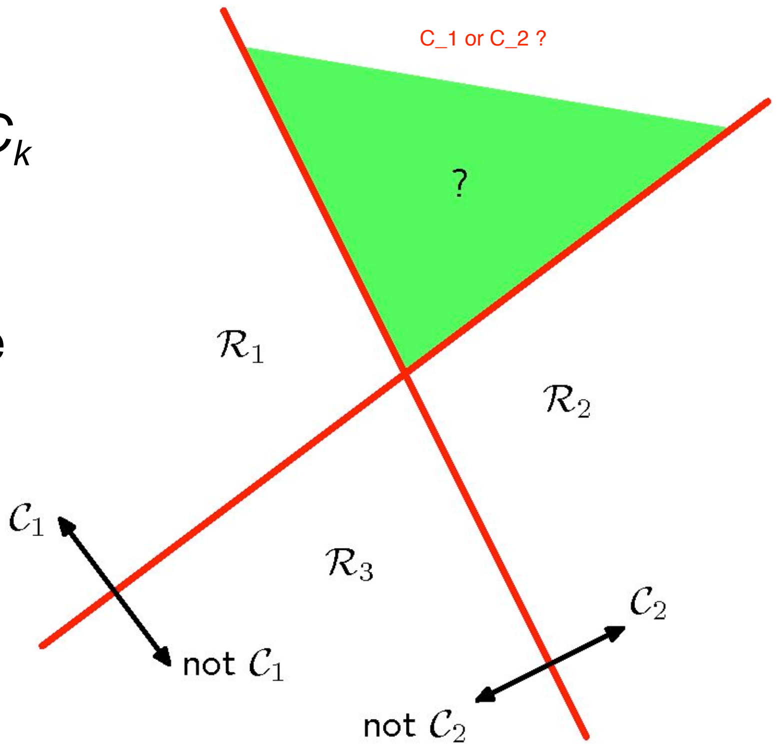
• Consider the extension of linear discriminants to K>2 classes.

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

- Separate the 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.

quadratic w.r.t. K

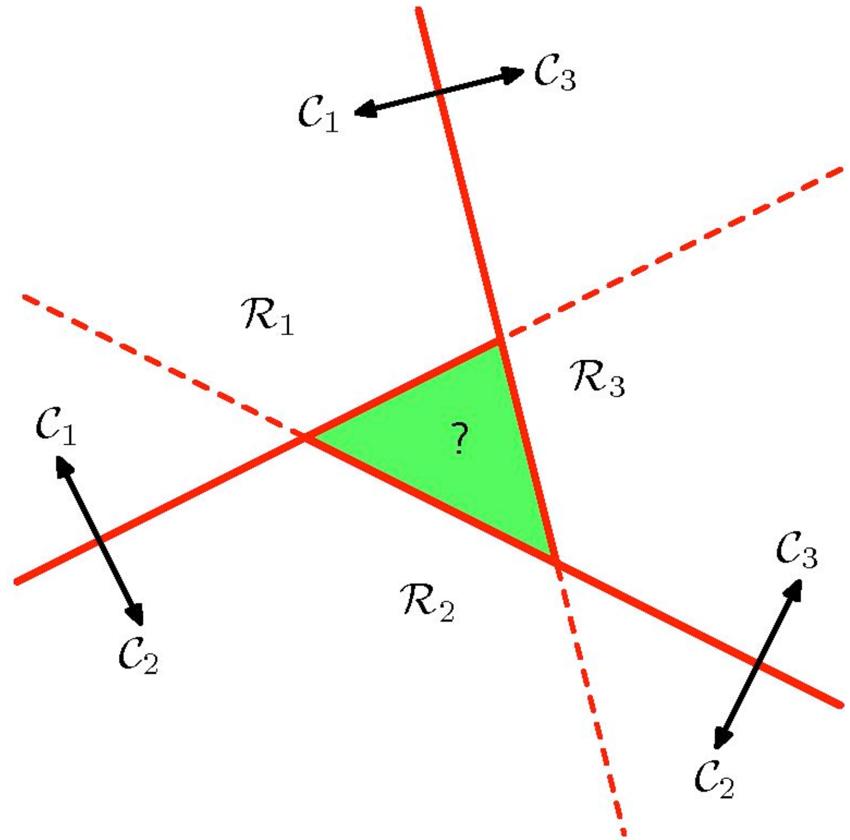
• An alternative is to use $K(K-1)/2$ binary discriminant functions.

for each pair of class C_k and C_j, there is a classifier

- Each function discriminates between two particular classes.
- A majority vote can be used

• 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).

decision boundary for any two class happens when y_k = y_j

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

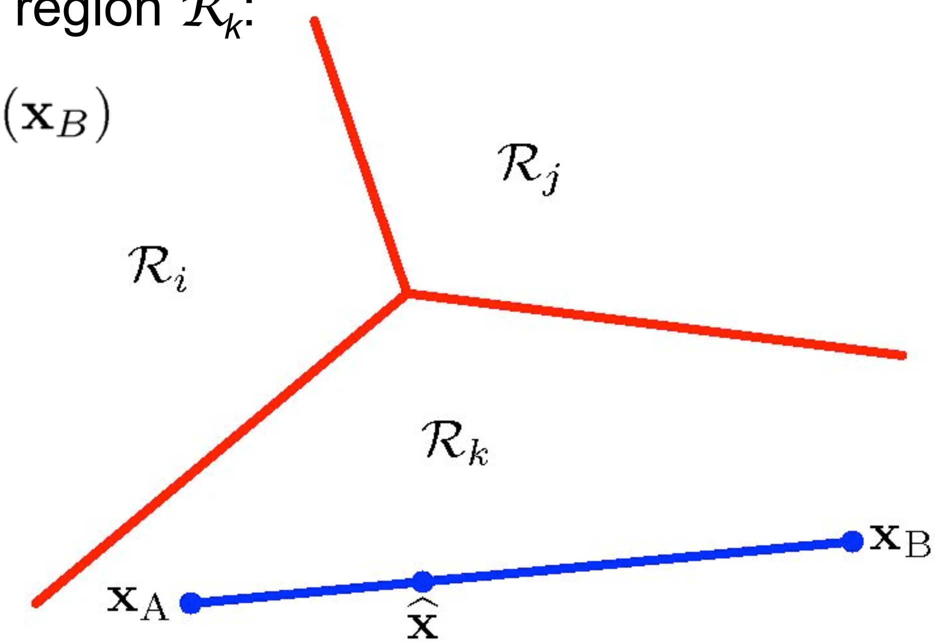
- For any two points that lie inside the region $\mathcal{R}_k$:

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

implies that for $0 < \alpha < 1$

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

owing to linearity of the discriminant functions.



proving that for any point on the line connecting X_a and X_b, also lies in R_k

Least Squares for Classification

- Consider a general classification problem with K classes using a 1-of-K encoding scheme for the target vector $\mathbf{t}$.

- 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:

transpose maps (D+1) -> K

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

(D+1) x K matrix whose k^{th} column is a (D+1)- dimensional vector:

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

a column vector

just incoporate bias term into the matrix notation

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 $n = 1, \ldots, N$.

• 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
that minimizes sum of squares error

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

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

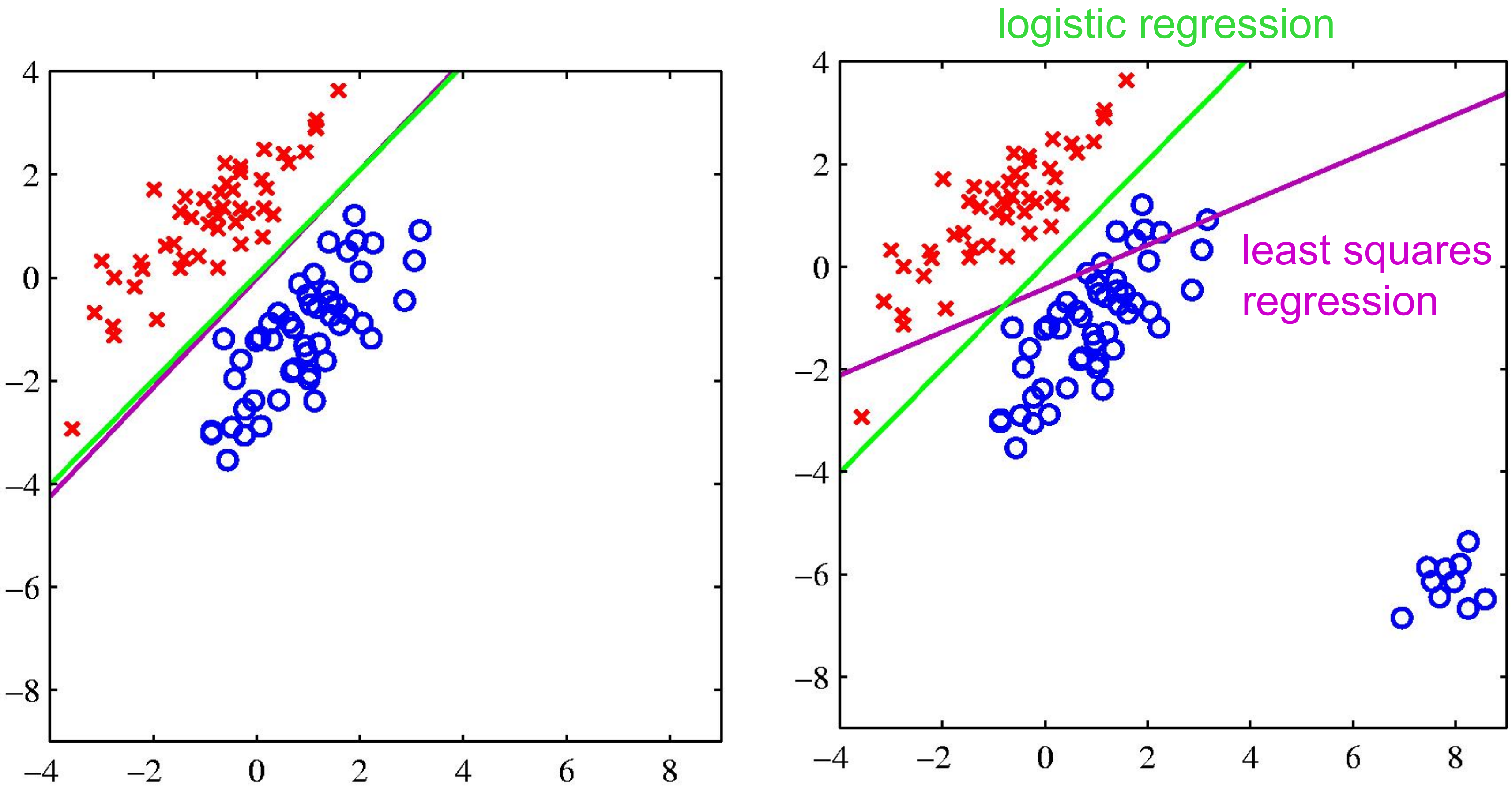
• A new input $\mathbf{x}$ is assigned to the class for which $y_k = \tilde{\mathbf{x}}^T\tilde{\mathbf{w}}_k$ is largest.

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

Problems using Least Squares

Least squares is highlighted: highly sensitive to outliers, unlike logistic regression

since least squares error penalizes predictions that are too correct, i.e. far from the correct side of decision boundary

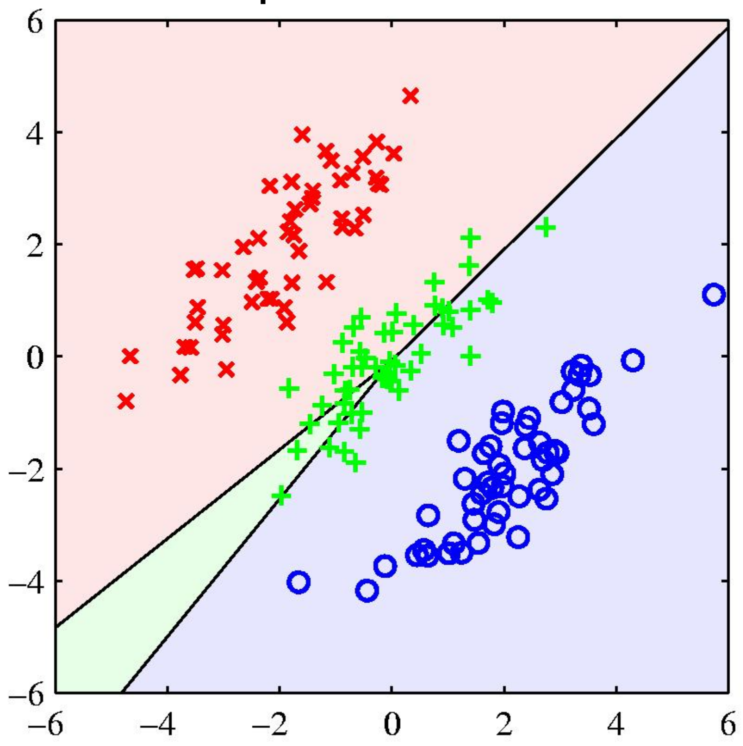


Problems using Least Squares

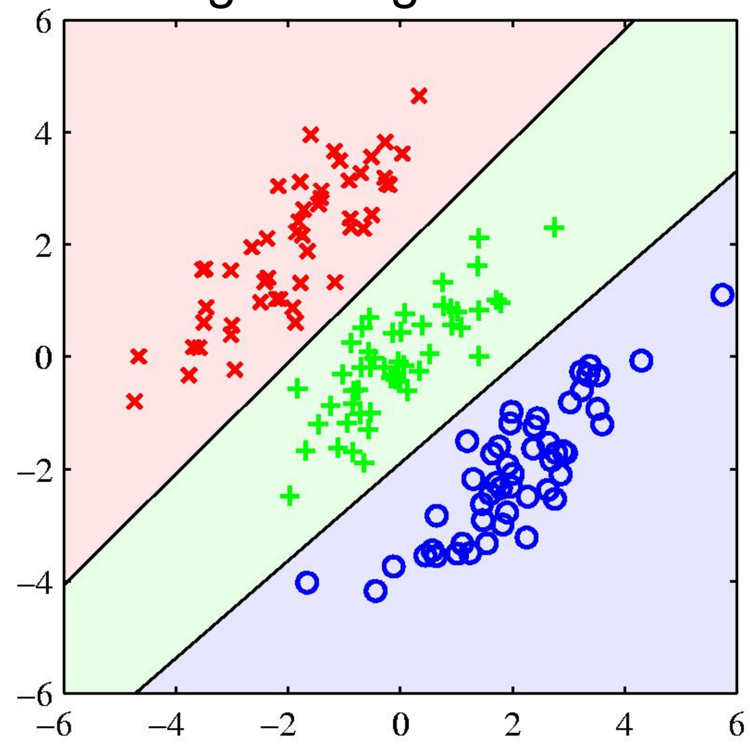
Example of synthetic dataset containing 3 classes, where lines denote decision boundaries.

least squares corresponds to MLE under gaussian conditional
but binary classification clearly has a distribution that is far from gaussian
need more appropriate probabilistic models

Least squares discriminant



Logistic regression



Many green points are misclassified.

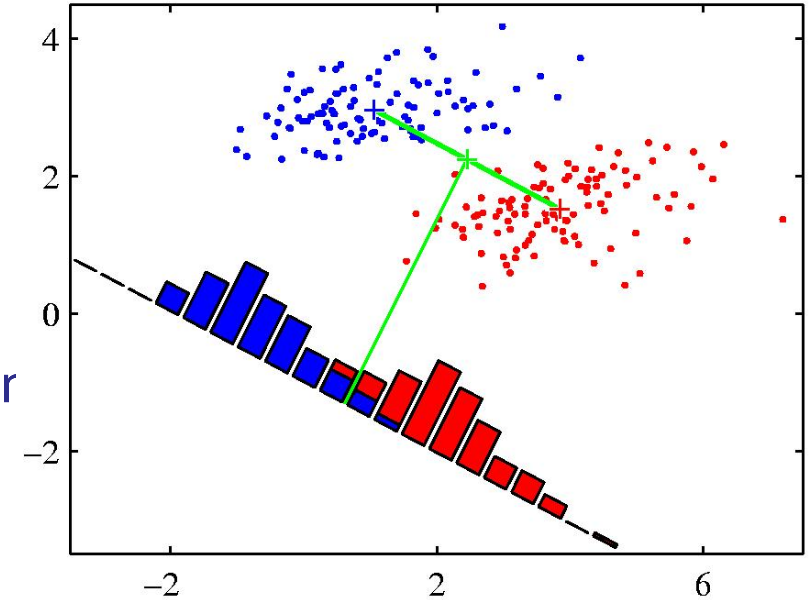
Fisher's Linear Discriminant

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

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

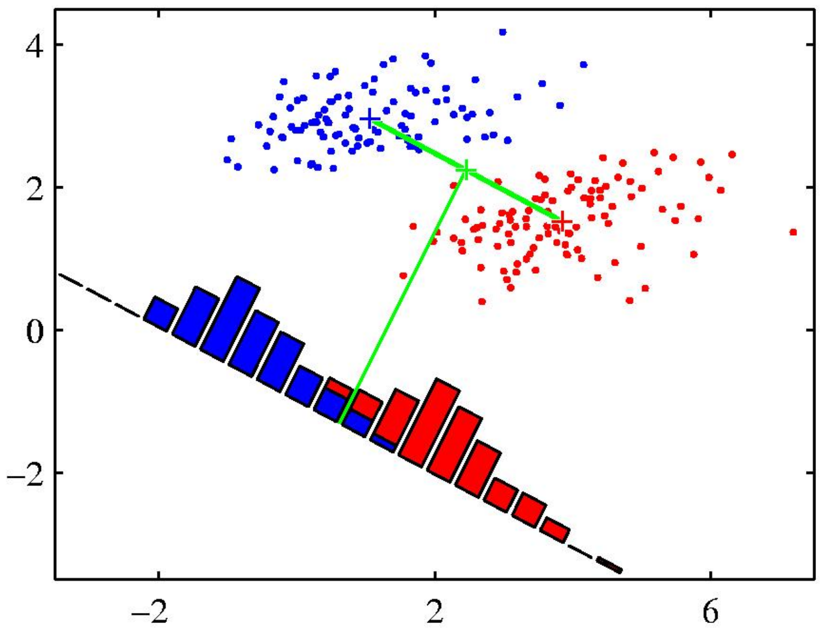
D -> 1 reduction in dimension

- 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.

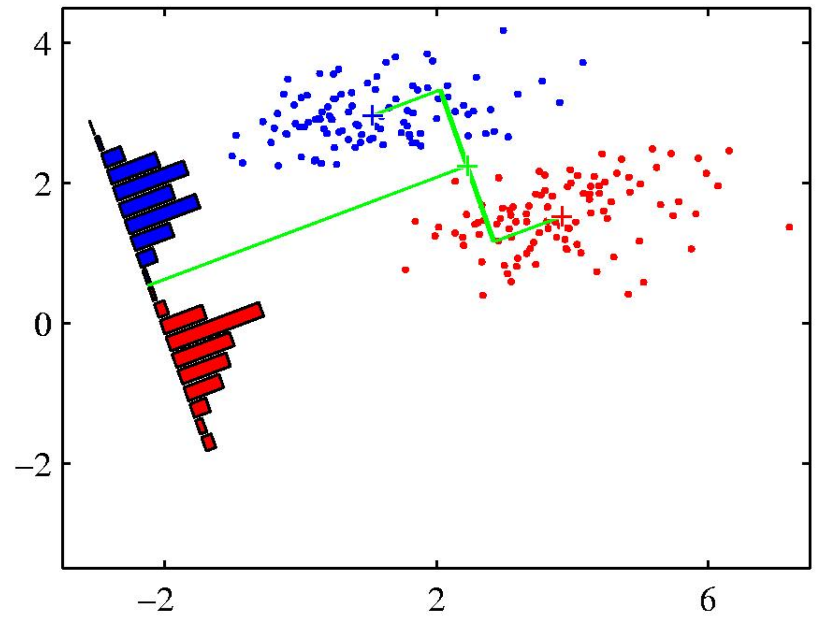


When projected onto the line joining the class means, the classes are not well separated.

Pictorial Illustration



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



Corresponding projection based on Fisher's linear discriminant.

Fisher's Linear Discriminant

- Let the mean of two classes be given by:

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

mean vectors

- It's not always good enough to project onto the vector separating the two classes: $\mathbf{w} \propto \mathbf{m}_1 - \mathbf{m}_2.$

in the projected y space

- We also want to minimize the within-class "variance":

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

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

where $m_k = \mathbf{w}^T \mathbf{m}_k.$ mean in projected space

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

- Fisher's criterion: maximize 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**:

J maximized when $$(\mathbf{w}^T S_b \mathbf{w}) S_w \mathbf{w} = (\mathbf{w}^T S_w \mathbf{w}) \underbrace{S_b \mathbf{w}}_{\text{is always in the direction of } (\mathbf{m}_2 - \mathbf{m}_1).}$$

scalar factors

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

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

Three Approaches to Classification

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

- Or we can model the conditional **probability** distribution $p(\mathcal{C}_k|\mathbf{x})$, and then use this distribution to make optimal decisions. There are two ways to do this:

  - 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'll consider this approach 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:

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

inverse of logistic function, the logit function

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.

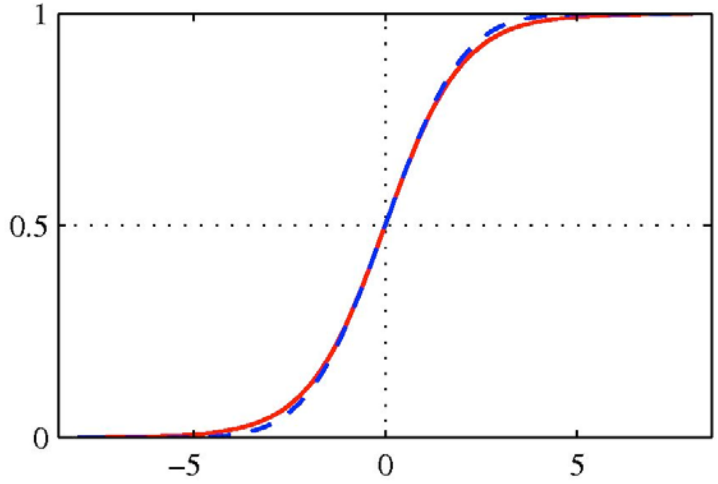
log ratio of posterior probability

Sigmoid Function

- The posterior probability of class C_1 is given by:

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

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



- The term *sigmoid* means S-shaped. The sigmoid function maps the whole real axis onto (0,1)

- It satisfies:

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

symmetric property

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:

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

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

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

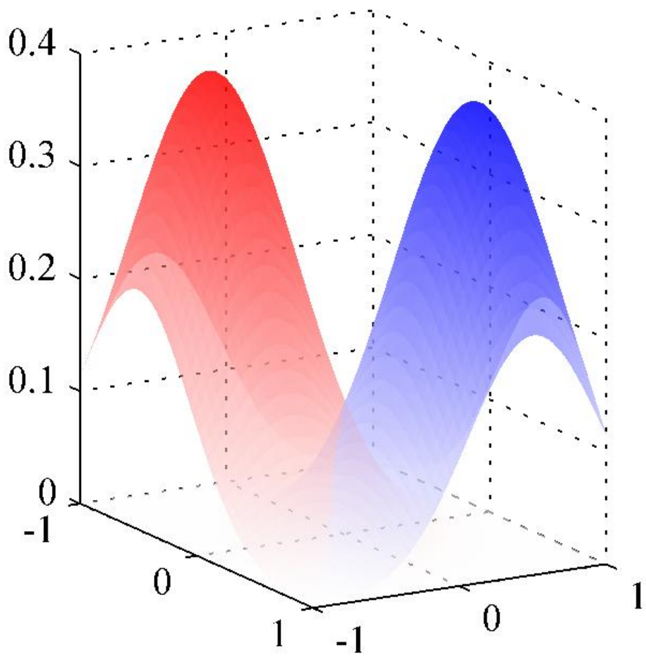
where we have defined:

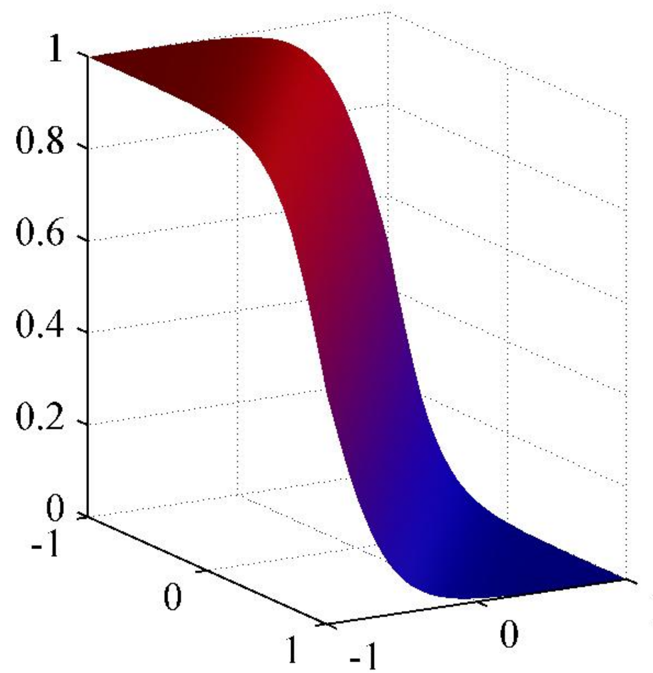
$$\mathbf{w} = \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2),$$

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

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

Example of Two Gaussian Models





Class-conditional densities for two classes

are two gaussians!

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:

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

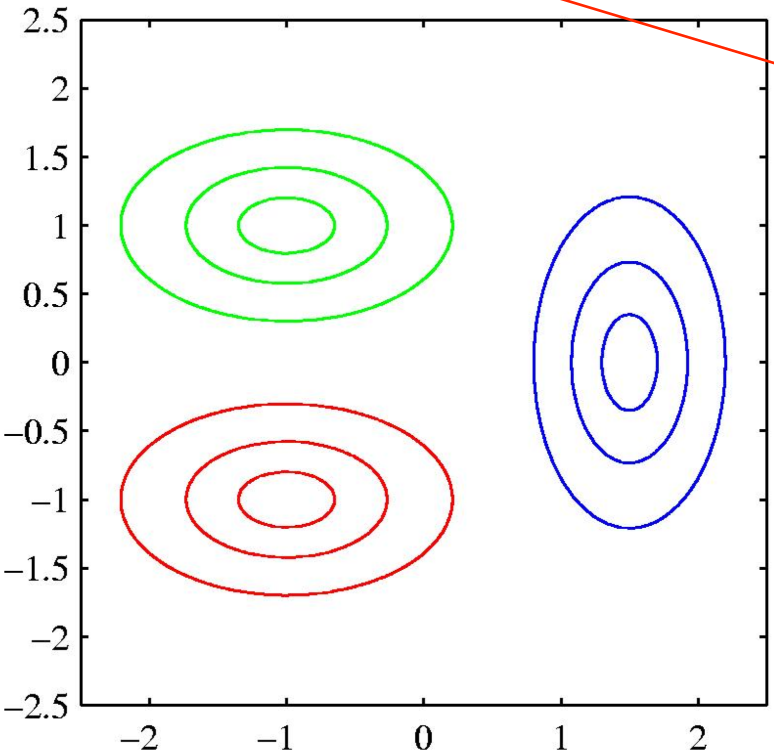
$$w_{k0} = -\frac{1}{2}\boldsymbol{\mu}_k^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k + \ln p(\mathcal{C}_k).$$

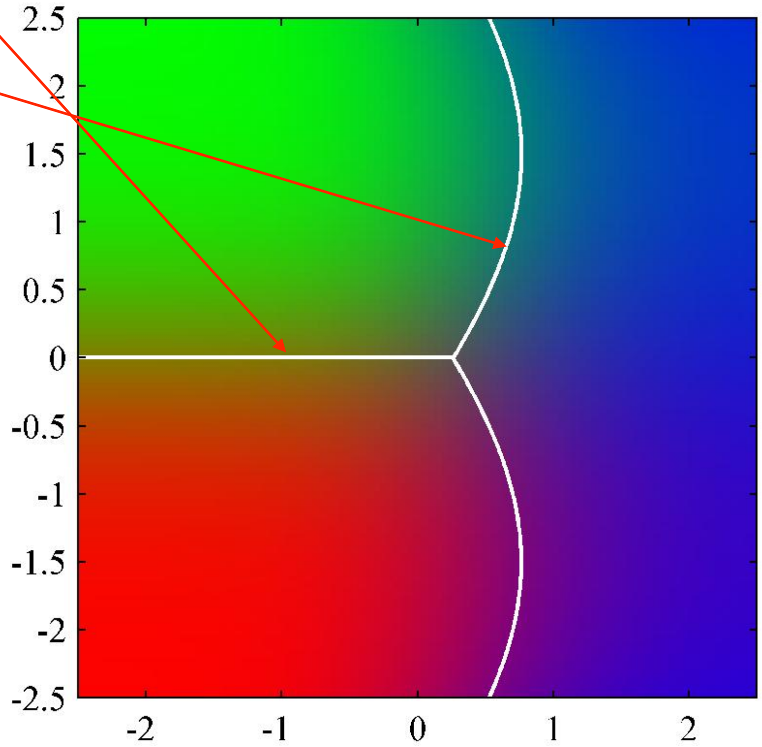
- Again, the decision boundaries are linear in input space.

- **Note:** 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.

covariance matrix same for green and red
blue covariance matrix is different

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 $p(\mathcal{C}_1) = \pi,\ \ p(\mathcal{C}_2) = 1 - \pi.$ specify prior

- The likelihood function takes the form:

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

Data points from class C_1.

Data points from class C_2.

exponent of t_n ensures that one of the expression is chosen for the right class designation

- 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 with respect to $\boldsymbol{\Sigma}$:

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

- Here we defined:

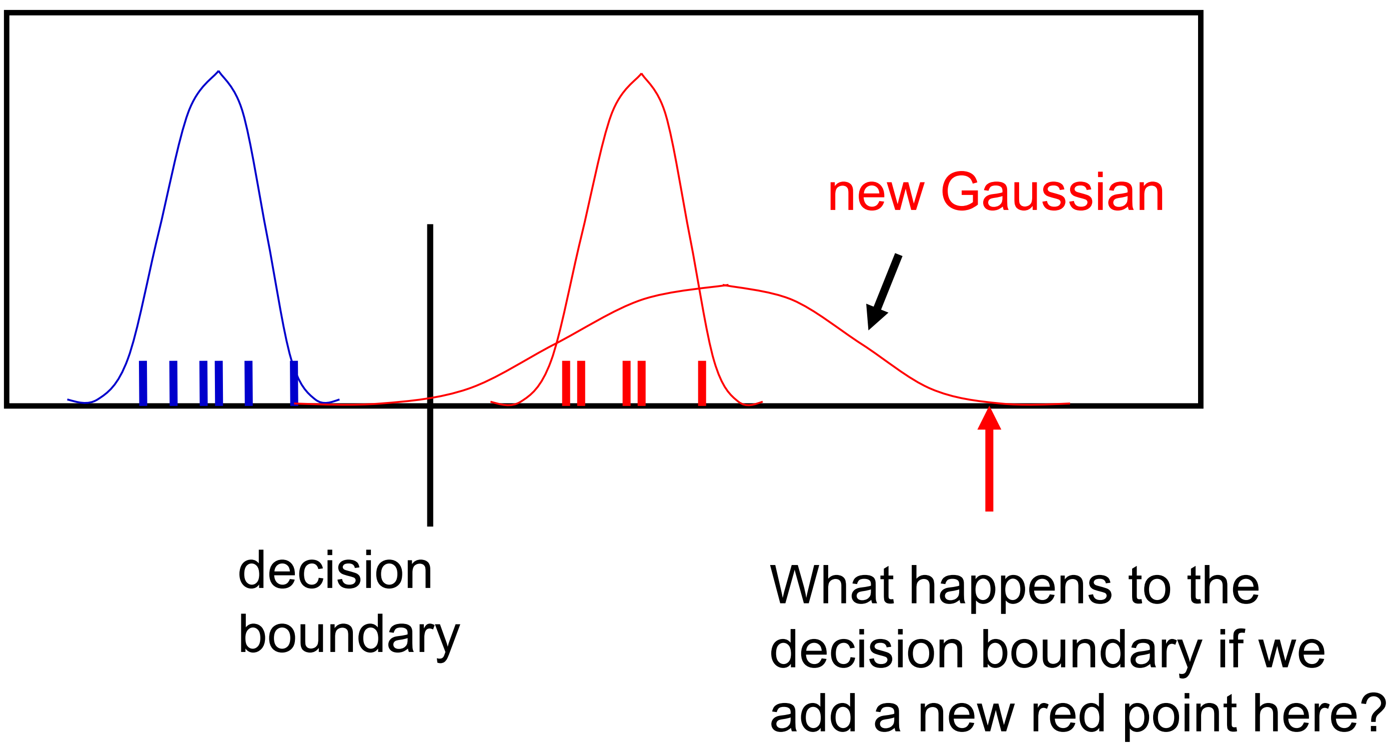
$$\mathbf{S} = \frac{N_1}{N}\mathbf{S}_1 + \frac{N_2}{N}\mathbf{S}_2,$$
$$\mathbf{S}_1 = \frac{1}{N_1}\sum_{n \in \mathcal{C}_1} (\mathbf{x}_n - \boldsymbol{\mu}_1)(\mathbf{x}_n - \boldsymbol{\mu}_1)^T,$$
$$\mathbf{S}_2 = \frac{1}{N_2}\sum_{n \in \mathcal{C}_2} (\mathbf{x}_n - \boldsymbol{\mu}_2)(\mathbf{x}_n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

- You are not responsible for showing that the solution is:

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

- The 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.

however if we are not sure of distribution of likelihood,
the performance is worse than discriminative probabilistic approach

- How can we fix this?

t distribution, robust to outliers

Three Approaches to Classification

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

- Or we can model the conditional **probability** distribution $p(\mathcal{C}_k|\mathbf{x})$, and then use this distribution to make optimal decisions. There are two ways to do this:

not interested in how x is generated

  - 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:

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

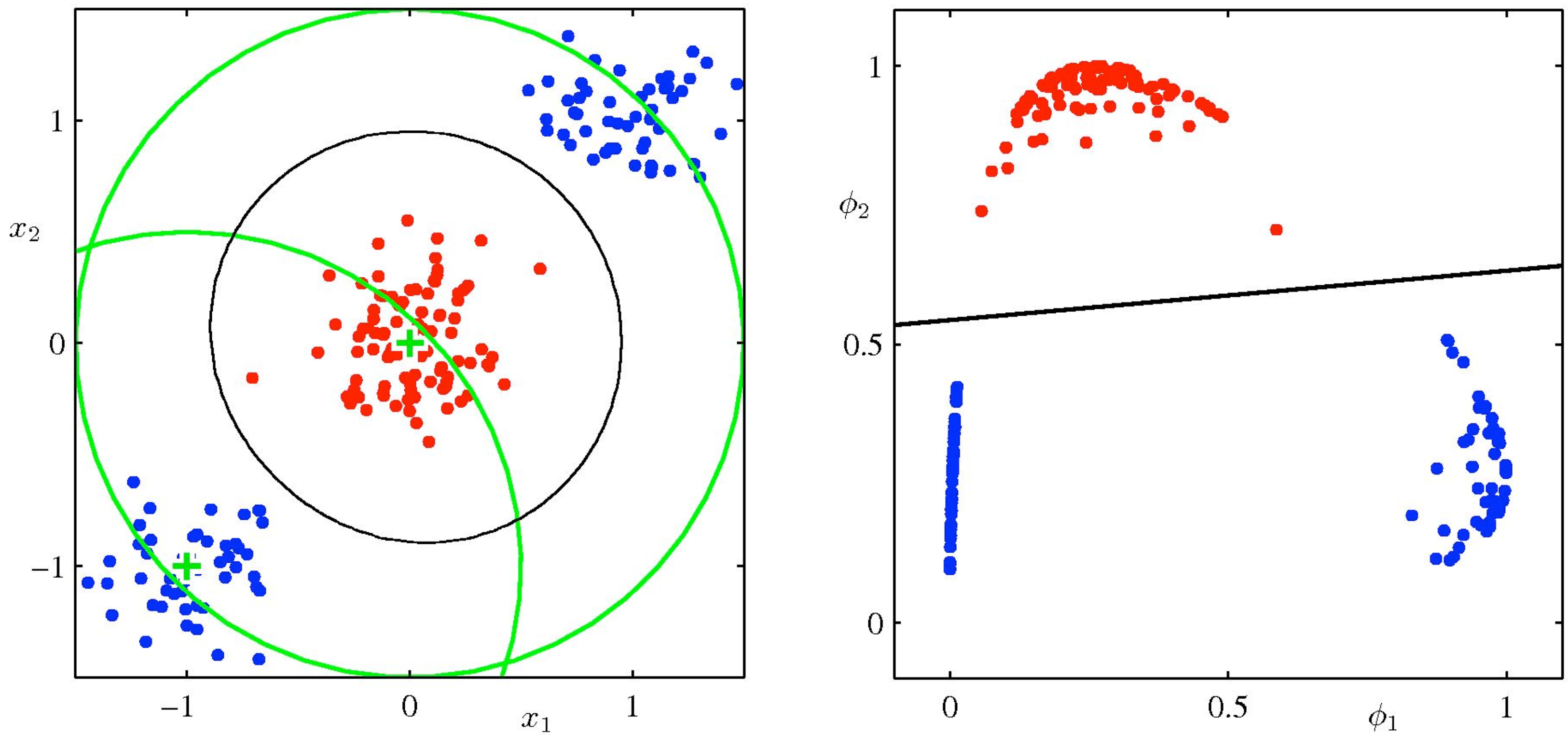
Fixed Basis Functions

- So far, we have considered classification models that work directly in the input space.
- All these models will still work in principle if we first make a fixed nonlinear transformation of the input space using a 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.

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

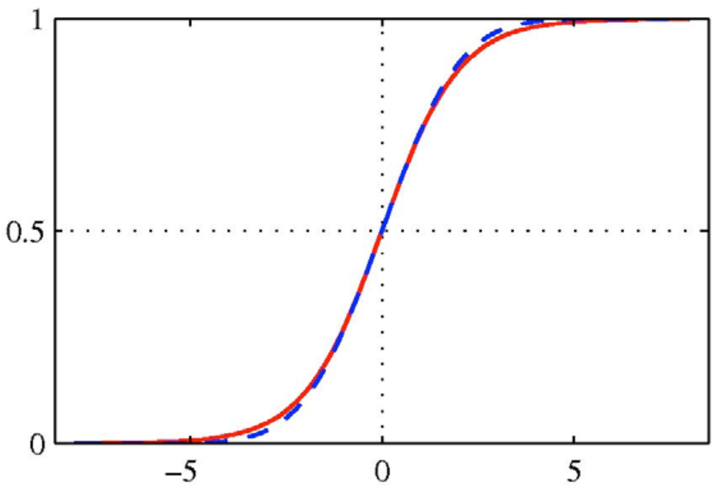
- Consider the problem of two-class classification.
- We have seen that the posterior probability of class C_1 can be written as a logistic sigmoid function:

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

where $p(\mathcal{C}_2|\mathbf{x}) = 1 - p(\mathcal{C}_1|\mathbf{x})$, 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.

i.e. we find optimal parameter (w) instead of estimating parameters for the class conditional and prior distribution neagtive log of maximum likelihood for logistic discriminant model is equivalent to cross-entropy errors

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 the 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:

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

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

stepwise differentiation using chain rules

- 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.

Not on the 12 February test

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:

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

Not on the 12 February test

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

- Taking the negative logarithm gives the cross-entropy error 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:

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

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.

knowing probability for 1 class is p implies probability for the other calss is 1-p

Recap

• Generative approach: Determine the class conditional densities and class-specific priors, and then use Bayes' rule to obtain the posterior probabilities $p(\mathcal{C}_k|\mathbf{x})$

- 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.

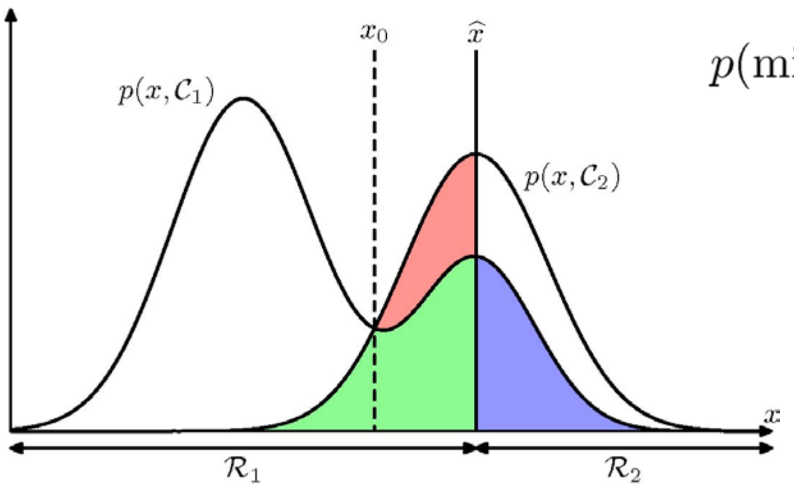
Model $p(\mathcal{C}_k|\mathbf{x})$ directly.

- Can be preferable when you're not confident about the data's generative mechanism

do not know how dataset is generated
or do not know what exactly is the class density distribution

Midterm Review

- Polynomial curve fitting – generalization, overfitting
- Decision theory:
  - Minimizing misclassification rate / Minimizing the expected loss



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

  - Loss functions for regression

$$\mathbb{E}[L] = \int\int \left(t - y(\mathbf{x})\right)^2 p(\mathbf{x}, t)\mathrm{d}\mathbf{x}\mathrm{d}t.$$

Midterm Review

- Bernoulli, Multinomial random variables (mean, variances)
- Multivariate Gaussian distribution (form, mean, covariance)
- Maximum likelihood estimation for these distributions.

- Exponential family / Maximum likelihood estimation / sufficient statistics for exponential family.

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

- Linear basis function models / maximum likelihood and least squares:

$$\ln p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta) = \sum_{i=1}^{N} \ln \mathcal{N}(t_n|\mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_n), \beta)$$

$$= -\frac{\beta}{2}\sum_{n=1}^{N}\left(t_n - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_n)\right)^2 + \frac{N}{2}\ln\beta - \frac{N}{2}\ln(2\pi).$$

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

Midterm Review

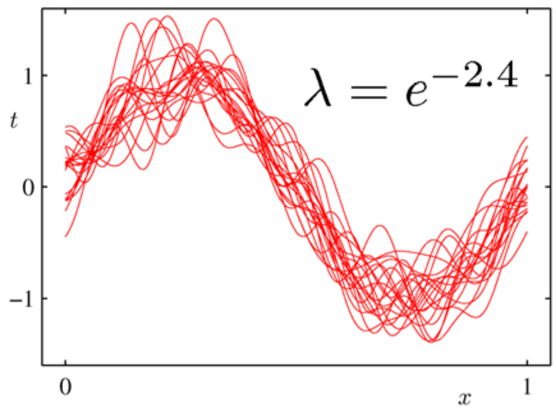
- Regularized least squares:

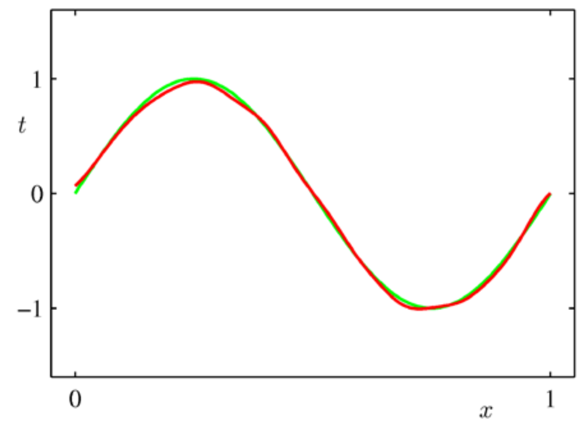
Ridge regression

$$\frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\phi(\mathbf{x}_n)\}^2 + \frac{\lambda}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} \qquad \mathbf{w} = \left(\lambda\mathbf{I} + \mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi}\right)^{-1}\mathbf{\Phi}^{\mathrm{T}}\mathbf{t}.$$

- Bias-variance decomposition.

High variance





Low bias

Midterm Review

• Bayesian Inference: likelihood, prior, posterior:

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

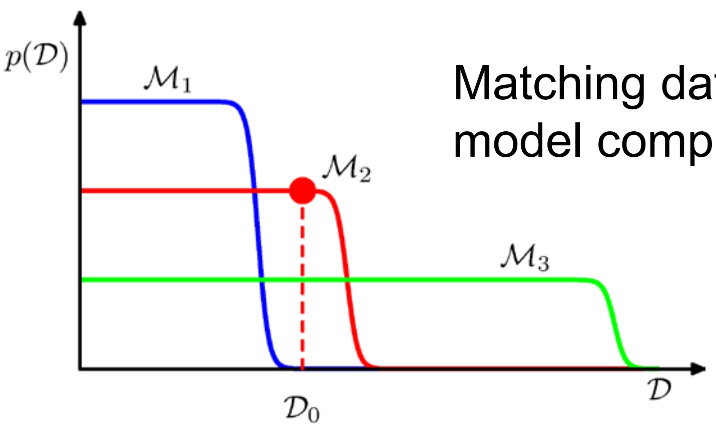
Marginal likelihood (normalizing constant):

$$P(\mathcal{D}) = \int p(\mathcal{D}|\mathbf{w})P(\mathbf{w})\mathrm{d}\mathbf{w}$$

• Marginal likelihood / predictive distribution.

• Bayesian linear regression / parameter estimation / posterior distribution / predictive distribution

• Bayesian model comparison



Matching data and model complexity

$$\ln p(\mathcal{D}) \simeq \ln p(\mathcal{D}|\mathbf{w}_{\mathrm{MAP}}) + M \ln\left(\frac{\Delta w_{\mathrm{posterior}}}{\Delta w_{\mathrm{prior}}}\right).$$

Midterm Review

- Classification models:
  - Discriminant functions
  - Fisher's linear discriminant
  - Logistic regression

- Probabilistic Generative Models / Gaussian class conditionals / Maximum likelihood estimation:

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

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

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

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