Machine Learning

Linear Discriminant Functions - Lecture V

Course Outline

Basic Concepts

- Parametric Method,
- Bayesian Learning and Nonparametrics Methods
- Clustering and Mixture of Gaussians

Classification Approaches

- Linear Discriminants
- Ensemble Methods and Boosting
- Randomized Trees, Forest

Deep Learning

- Foundations
- Optimization

Reinforcement Learning

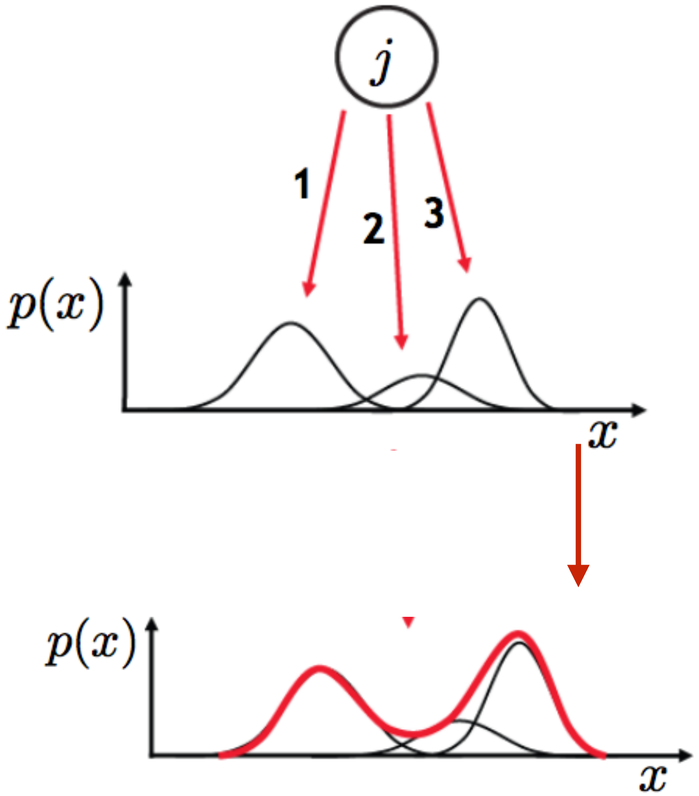
- Classical Reinforcement Learning
- Deep Reinforcement Learning

Videos for This Lecture

- Repetition Video (Part 0)
- Linear discriminant function (Part 1)
- Least-squares classification (Part 2)
- Generalized linear models (Part 3)

Recap: Mixture of Gaussians (MoG)

Generative Model



$$p(j) = \pi_j$$ **WEIGHT of mixture component**

$$p(x|\theta_j)$$ **Mixture component**

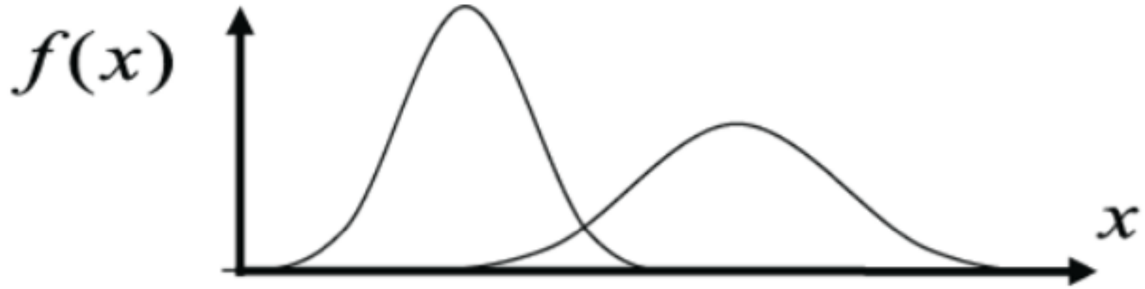
Mixture density

$$p(x|\theta) = \sum_{j=1}^{M} p(x|\theta_j)p(j)$$

Image source: B. Leibe Slides ML, 2015

Recap: Mixture of Gaussians

Assume we knew the values of the hidden variable…



ML for Gaussian #1 **ML for Gaussian #2**

assumed known → **1 111 22 2 2 j**

$h(j=1|x_n) =$ ***1 111 00 0 0***

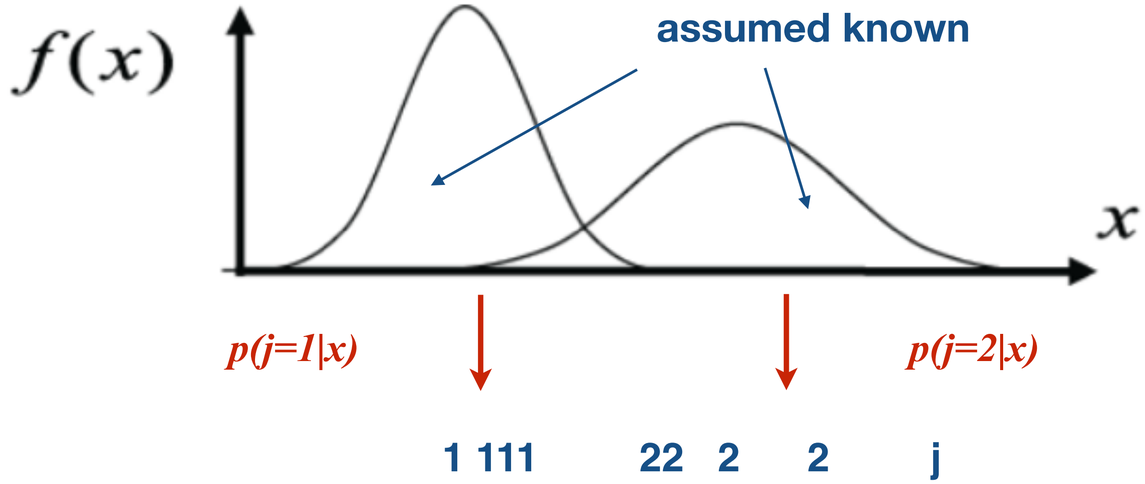
$h(j=2|x_n) =$ ***0 000 11 1 1***

$$\mu_1 = \frac{\sum_{n=1}^{N} h(j=1|x_n)x_n}{\sum_{i=1}^{N} h(j=1|x_n)} \qquad \mu_2 = \frac{\sum_{n=1}^{N} h(j=2|x_n)x_n}{\sum_{i=1}^{N} h(j=2|x_n)}$$

Slide is taken from B. Leibe, ML, 2017

Recap: Estimating MoGs-Iterative Strategy

Assume we knew the values of the hidden variable…



Bayes decision rule:

$$\text{Decide } j = 1 \text{ if } p(j = 1|x_n) > p(j = 2|x_n)$$

Slide is taken from B. Leibe, ML, 2017

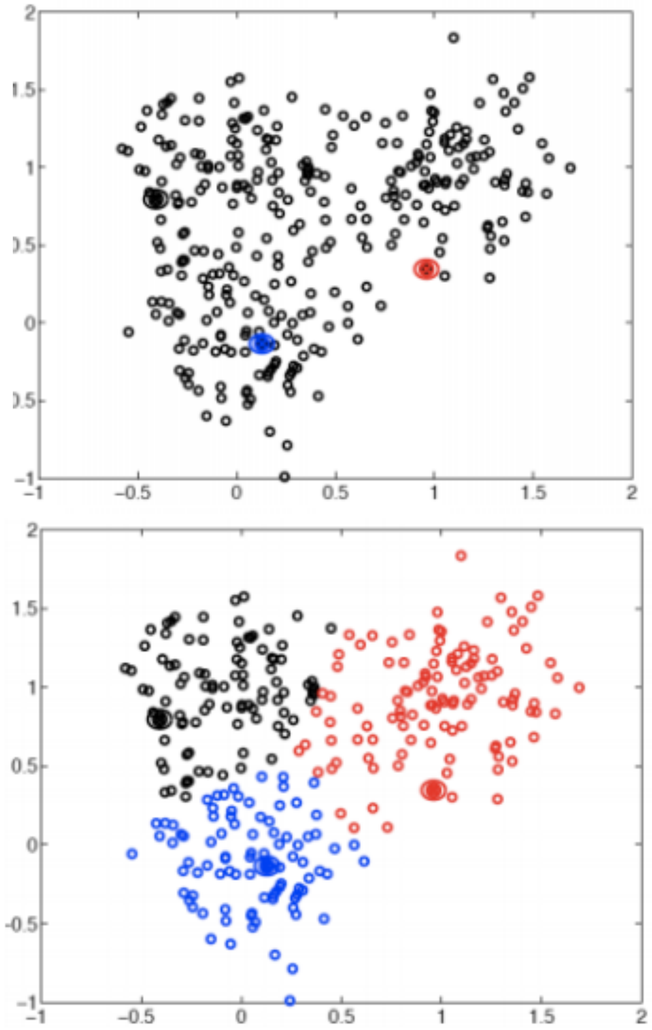
Recap: K-Means Clustering

Iterative procedure

1. Initialization: pick K arbitrary centroids (cluster means)
2. Assign each sample to the closest centroid.
3. Adjust the centroids to be the means of the samples assigned to them.
4. Go to step 2 (until no change)

Algorithm is guaranteed to converge

- Local optimum
- Final result depends on initial estimate



Slide is taken from B. Leibe, ML, 2017

Recap: EM Algorithm

Expectation-Maximization (EM) Algorithm

- **E-step:** softly assign samples to mixture components

$$\gamma_j(\mathbf{x}_n) \leftarrow \frac{\pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)} \qquad \forall j = 1, \ldots, K, \;\; n = 1, \ldots, N$$

- **M-step:** re-estimate the parameters (separately for each mixture component) based on the soft assignments

$$\hat{N}_j \leftarrow \sum_{n=1}^{N} \gamma_j(\mathrm{x}_n) = \text{soft number of samples labeled } j$$

$$\hat{\pi}_j^{new} \leftarrow \frac{\hat{N}_j}{N}$$

$$\hat{\mu}_j^{new} \leftarrow \frac{1}{\hat{N}_j} \sum_{n=1}^{N} \gamma_j(x_n) x_n$$

$$\hat{\Sigma}_j^{new} \leftarrow \frac{1}{\hat{N}_j} \sum_{n=1}^{N} \gamma_j(x_n)(x_n - \hat{\mu}_j^{new})(x_n - \hat{\mu}_j^{new})^T$$

Slide is taken from B. Leibe, ML, 2017

Part 1, Video LinearDiscriminantFunction_p1

- Definition
- Extension to multiple classes

Today's Topics

Linear discriminant function

- Definition
- Extension to multiple classes

Least-squares classification

- Derivation
- Shortcomings

Generalized linear models

- Connection to neural networks
- Generalized linear discriminants & gradient descent

Slide source: B. Leibe Slides ML, 2015

Discriminant Functions

Bayesian Decision Theory

$$p(C_k|x) = \frac{p(x|C_k)p(C_k)}{p(x)}$$

- Model condition probability densities $p(x|C_k$ and priors $p(C_k)$
- Compute posteriors $p(C_k|x)$ (using Bayes' rule)
- Minimize probability of misclassification by maximising $p(C|x)$

New Approach

- Directly encode decision boundary
- Without explicit modelling of probability densities
- Minimize misclassification probability directly

Slide source: B. Leibe Slides ML, 2015

Recap: Discriminant Functions

Formulate classification in terms of comparisons

- Discriminant functions $y_1(x), ..., y_K(x)$
- Classify x as class C_k if $y_k(x) > y_j(x) \; \forall j \neq j$

Examples (Bayes Decision Theory)

$$y_k(x) = p(C_k|x)$$

$$y_k(x) = p(x|C_k)p(C_k)$$

$$y_k(x) = \log p(x|C_k) + \log p(C_k)$$

Slide source: B. Leibe Slides ML, 2015

Discriminant Functions

Example: 2 classes

$$
\begin{aligned}
& y_1(x) > y_2(x) \\
\Leftrightarrow\ & y_1(x) - y_2(x) > 0 \\
\Leftrightarrow\ & \mathrm{y}(x) > 0
\end{aligned}
$$

Decision function (from Bayes Decision Theory)

$$y(x) = p(C_1|x) - p(C_2|x)$$

$$y(x) \models \ln \frac{p(x|C_1)}{p(x|C_2)} + \ln \frac{p(C_1)}{p(C_2)}$$

Slide source: B. Leibe Slides ML, 2015

Learning Discriminant Functions

General classification problem

- Goal: take a new input x and assign it to one of *K* classes C_k
- Given: training set $\mathrm{X}=\mathrm{x}_1, \ldots, \mathrm{x}_N$
  with target values $\mathrm{T}=\mathrm{t}_1, \ldots, \mathrm{t}_N$

  => Learn a discriminant function *y*(x) to perform the classification.

  2-class problem
  - Binary target values: $t_n \in \{0, 1\}$

  K-class problem
  - 1-of-K coding scheme, e.g. $t_n = (0, 1, 0, 0, 0)^T$

Learning Discriminant Functions

2-class problem

- $y(x)$>0: Decide for class C_1, else for class C_2

In the following, we focus on **linear discriminant functions**

$$y(\mathrm{x}) = \mathrm{w}^T \mathrm{x} + w_0$$

weight vector

"bias" (=threshold)

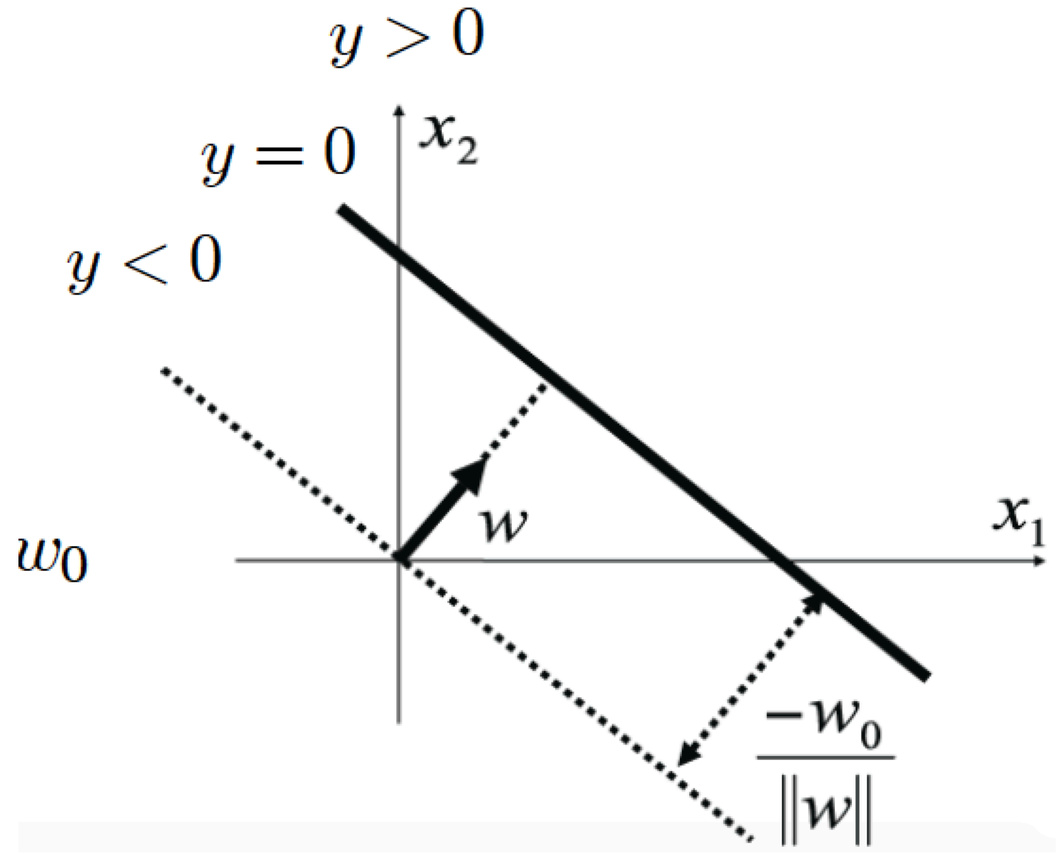
- If a data set can be perfectly classified by a linear discriminant, then we call it linearly separable.

Learning Discriminant Functions

Decision boundary y(x)=0 defines a hyperplane

- Normal vector: w
- Offset: $\frac{-w_0}{||\mathrm{w}||}$

$$y(\mathrm{x}) = \mathrm{w}^T\mathrm{x}+w_0$$



Slide source: B. Leibe Slides ML, 2015

Linear Discriminant Functions

Notation

- D : Number of dimensions

$$\mathrm{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_D \end{bmatrix} \quad \mathrm{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_D \end{bmatrix}$$

$$\begin{aligned} y(\mathrm{x}) &= \mathrm{w}^T \mathrm{x} + w_0 \\ &= \sum_{i=1}^{D} w_i x_i + w_0 \\ &= \sum_{i=0}^{D} w_i x_i \end{aligned}$$

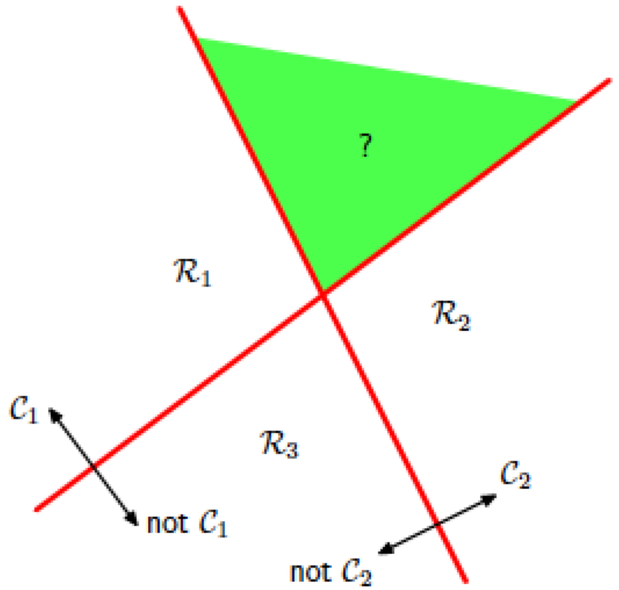
with x_0 =1 constant

Slide source: B. Leibe Slides ML, 2015

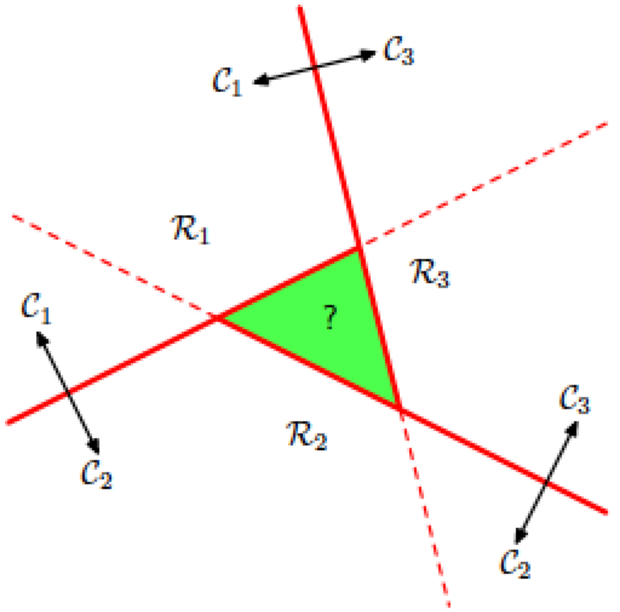
Extension to Multiple Classes

Two simple strategies

- *One-vs-all* classifiers



- *One-vs-one* classifiers



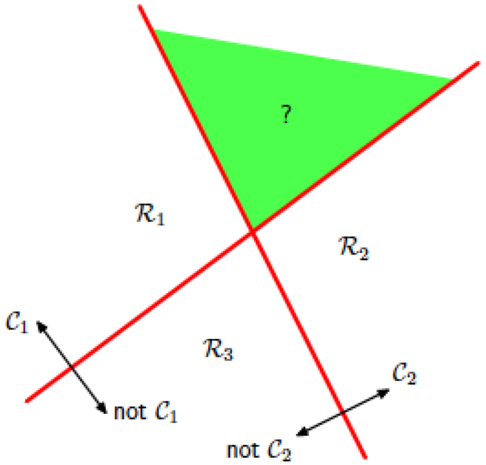
- What difficulties do you see for those strategies?

Slide source: B. Leibe Slides ML, 2015

Extension to Multiple Classes

Problem

- Both strategies in region for which the pure classification result $(y_k > 0)$ is ambiguous
- In the *one-vs-all* case, it is still possible to classify those inputs based on the continuous classifier output $y_k > y_j \ \forall j \neq k$



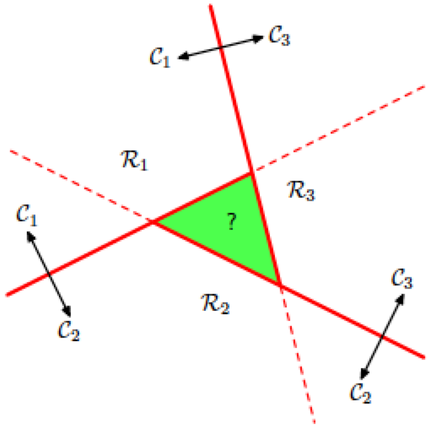
Solution

- We can avoid those difficulties by taking K linear functions of the form

$$y_k(\mathrm{x}) = \mathrm{w}_k^T \mathrm{x} + w_{k0}$$

and defining the decision boundaries directly by deciding for C_k iff $y_k > y_j \ \forall j \neq k$

This corresponds to a 1-of-K coding scheme



$$t_n = (0, 1, 0, 0, 0)^T$$

Slide source: B. Leibe Slides ML, 2015

Extension to Multiple Classes

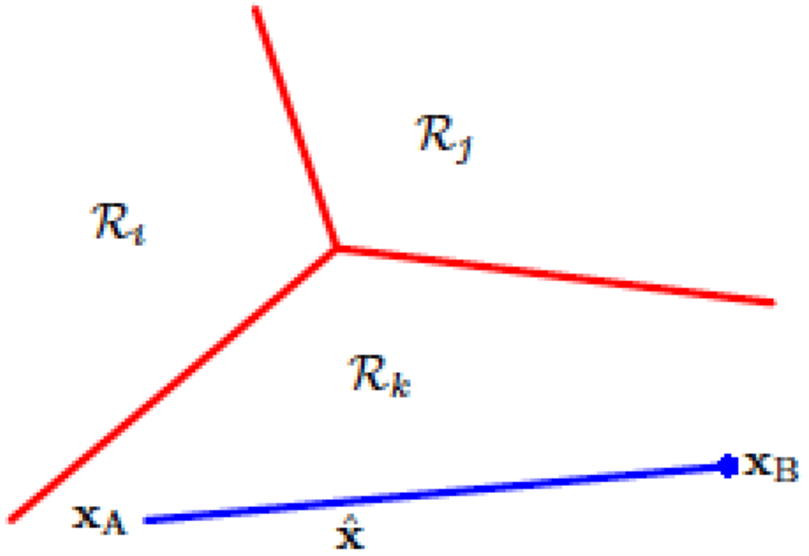
K-class discriminant

- Combination of K linear functions

$$y_k(\mathrm{x}) = \mathrm{w}_k^T \mathrm{x} + w_{k0}$$

- Resulting decision hyperplanes:

$$(\mathrm{w}_k - \mathrm{w}_j)^T \mathrm{x} + (w_{k0} - w_{j0}) = 0$$



- It can be shown that the decision regions of such a discriminant are always singly connected and convex.
- This makes linear discriminant models particularly suitable for problems for which the conditional densities $p(\mathrm{x}|w_i)$ are unimodal.

Slide source: B. Leibe Slides ML, 2015

Part 2, Video LinearDiscriminantFunction_p2

- Derivation of Least-squares classification
- Shortcomings

Today's Topics

Linear discriminant function

- Definition
- Extension to multiple classes

Least-squares classification

- **Derivation**
- **Shortcomings**

Generalized linear models

- Connection to neural networks
- Generalized linear discriminants & gradient descent

Slide source: B. Leibe Slides ML, 2015

General Classification Problem

Classification Problem

- Let's consider K classes described by linear models

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

- We can group those together using vector notation

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

where

$$\widetilde{\mathbf{W}} = [\widetilde{\mathbf{w}}_1, \ldots, \widetilde{\mathbf{w}}_K] = \begin{bmatrix} w_{10} & \cdots & w_{K0} \\ w_{11} & \cdots & w_{K1} \\ \vdots & \ddots & \vdots \\ w_{1D} & \cdots & w_{KD} \end{bmatrix}$$

- The output will again be in 1-of-K notation.

➡We can directly compare it to the target value $\mathrm{t} = [t_1, ..., t_k]^T$

Slide source: B. Leibe Slides ML, 2015

General Classification Problem

Classification Problem

- For the entire dataset, we can write

$$\mathbf{Y}(\widetilde{\mathbf{X}}) = \widetilde{\mathbf{X}}\widetilde{\mathbf{W}}$$

and compare to the target matrix T where

$$\widetilde{\mathbf{W}} = [\widetilde{\mathbf{w}}_1, \ldots, \widetilde{\mathbf{w}}_K]$$

$$\widetilde{\mathbf{X}} = \begin{bmatrix} \mathbf{x}_1^{\mathrm{T}} \\ \vdots \\ \mathbf{x}_N^{\mathrm{T}} \end{bmatrix} \qquad \mathbf{T} = \begin{bmatrix} \mathbf{t}_1^{\mathrm{T}} \\ \vdots \\ \mathbf{t}_N^{\mathrm{T}} \end{bmatrix}$$

- Result of the comparison: $\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T}$

Goal: Choose $\widetilde{\mathbf{W}}$ such that this is minimal!

Slide source: B. Leibe Slides ML, 2015

Least-Squares Classification

Simplest Approach

- Directly try to minimise the **sum-of-squares error**
- We could write this as

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{K}\left(y_k(\mathbf{x}_n;\mathbf{w}) - t_{kn}\right)^2$$

$$= \frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{K}\left(\mathbf{w}_k^T\mathbf{x}_n - t_{kn}\right)^2$$

- But let's stick with the matrix notation for now …
- (The result will be simpler to express and we'll learn some nice matrix algebra rules along the way …)

Slide source: B. Leibe Slides ML, 2015

Least-Squares Classification

Multi-class case

- Let's formulate the sum-of-squares error in matrix notation

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

using:

$$\sum_{i,j} a_{ij}^2 = \mathrm{Tr}\{\mathbf{A}^{\mathrm{T}}\mathbf{A}\}$$

  - Taking the derivative yields

$$\begin{aligned}
\frac{\partial}{\partial\widetilde{\mathbf{W}}}E_D(\widetilde{\mathbf{W}}) &= \frac{1}{2}\frac{\partial}{\partial\widetilde{\mathbf{W}}}\mathrm{Tr}\left\{(\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T})^{\mathrm{T}}(\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T})\right\} \\
&= \frac{1}{2}\frac{\partial}{\partial(\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T})^{\mathrm{T}}(\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T})}\mathrm{Tr}\left\{(\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T})^{\mathrm{T}}(\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T})\right\} \\
&\quad \cdot \frac{\partial}{\partial\widetilde{\mathbf{W}}}(\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T})^{\mathrm{T}}(\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T}) \\
&= \widetilde{\mathbf{X}}^{\mathrm{T}}(\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T})
\end{aligned}$$

chain rule:

$$\frac{\partial \mathbf{Z}}{\partial \mathbf{X}} = \frac{\partial \mathbf{Z}}{\partial \mathbf{Y}}\frac{\partial \mathbf{Y}}{\partial \mathbf{X}}$$

using:

$$\frac{\partial}{\partial \mathbf{A}}\mathrm{Tr}\{\mathbf{A}\} = \mathbf{I}$$

Slide source: B. Leibe Slides ML, 2015

Least-Squares Classification

Minimizing the sum-of-squares error

$$\frac{\partial}{\partial \widetilde{\mathbf{W}}} E_D(\widetilde{\mathbf{W}}) = \widetilde{\mathbf{X}}^{\mathrm{T}}(\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T}) \stackrel{!}{=} 0$$

$$\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} = \mathbf{T}$$

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

$$= \widetilde{\mathbf{X}}^{\dagger}\mathbf{T}$$ "pseudo-inverse"

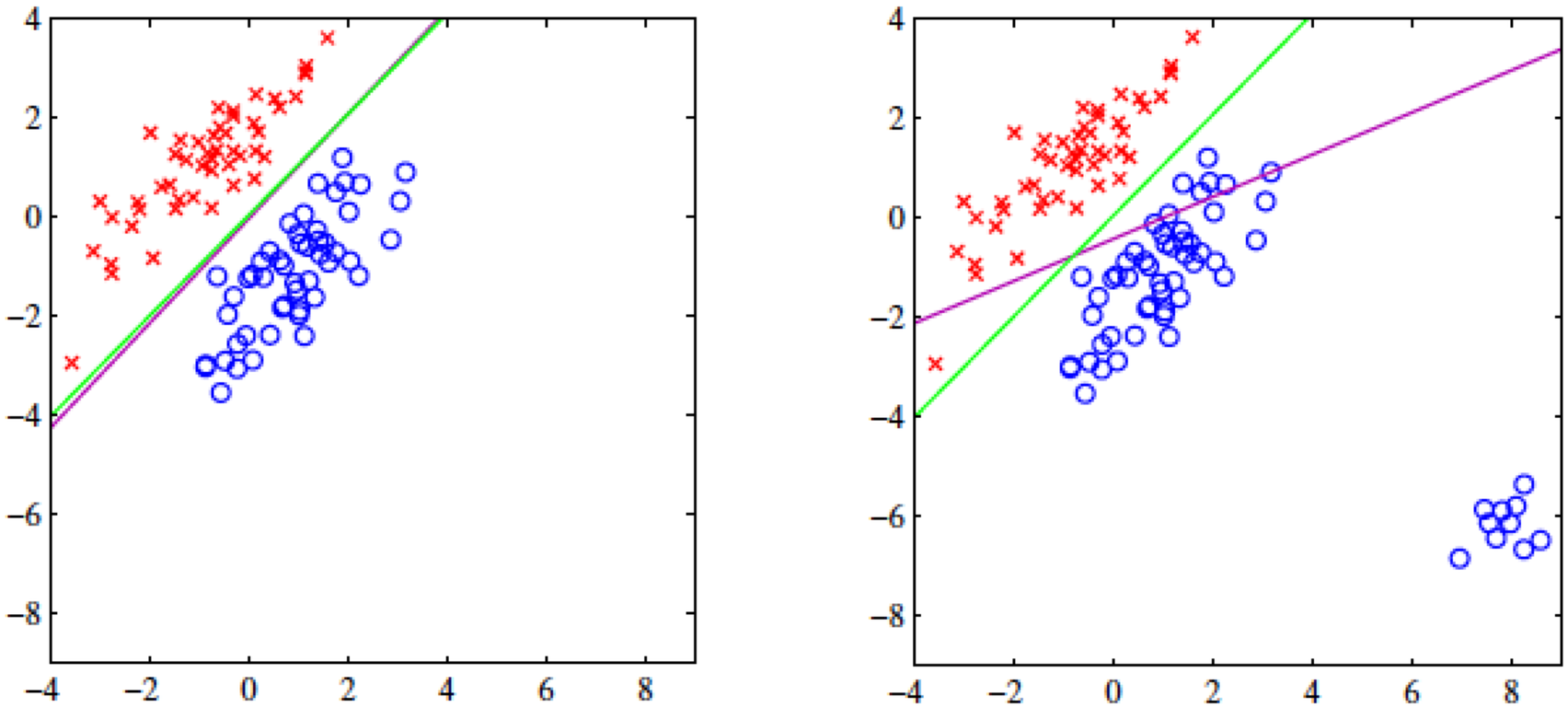
- We then obtain the discriminant function as

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

➡Exact, closed-form solution for the discriminant function parameters

Slide source: B. Leibe Slides ML, 2015

Problems with Least Squares



Least-squares is very sensitive to outliers!

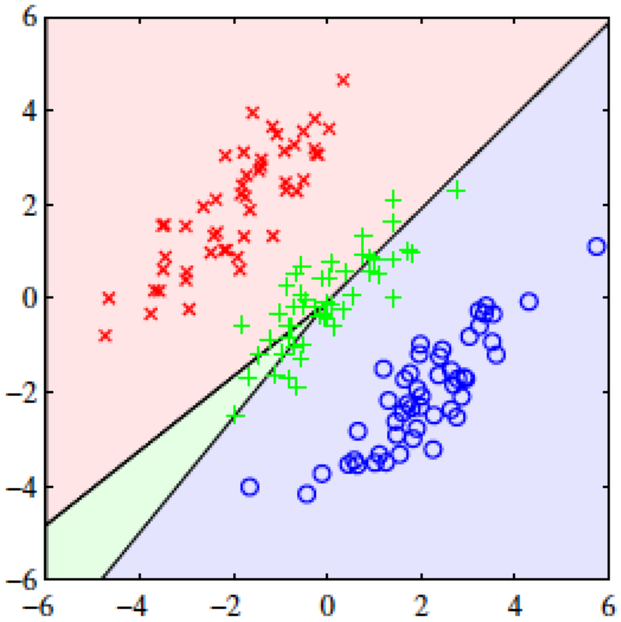
- The error function penalises predictions that are “too correct”

Slide source: B. Leibe Slides ML, 2015

Problems with Least Squares

Another example:

- 3 classes (red, green, blue)
- Linearly separable problem
- Least-squares solution:
  Most green points are misclassified!



Deeper reason for the failure

- Least-squares corresponds to Maximum Likelihood under the assumption of a Gaussian conditional distribution.
- However, our binary target vectors have a distribution that is clearly non-Gaussian!

➡Least-squares is the wrong probabilistic tool in this case!

Slide source: B. Leibe Slides ML, 2015

Part 3, Video LinearDiscriminantFunction_p3

- Connection to neural networks
- Generalized linear discriminants & gradient descent

Today's Topics

Linear discriminant function

- Definition
- Extension to multiple classes

Least-squares classification

- Derivation
- Shortcomings

Generalized linear models

- **Connection to neural networks**
- **Generalized linear discriminants & gradient descent**

Slide source: B. Leibe Slides ML, 2015

Generalized Linear Models

Linear model

$$y(\mathrm{x}) = \mathrm{w}^T \mathrm{x} + w_0$$

Generalized linear model

$$y(\mathrm{x}) = g(\mathrm{w}^T \mathrm{x} + w_0)$$

- *g(.)* is called an **activation function** and maybe nonlinear.
- The decision surface correspond to

$$y(\mathrm{x}) = const. \Leftrightarrow \mathrm{w}^T \mathrm{x} + w_0 = const.$$

- If *g* is monotonous (which is typically the case), the resulting decision boundaries are still linear functions of x.

Slide source: B. Leibe Slides ML, 2015

Generalized Linear Models

Consider 2 classes:

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

$$= \frac{1}{1 + \frac{p(\mathbf{x}|\mathcal{C}_2)p(\mathcal{C}_2)}{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1)}}$$

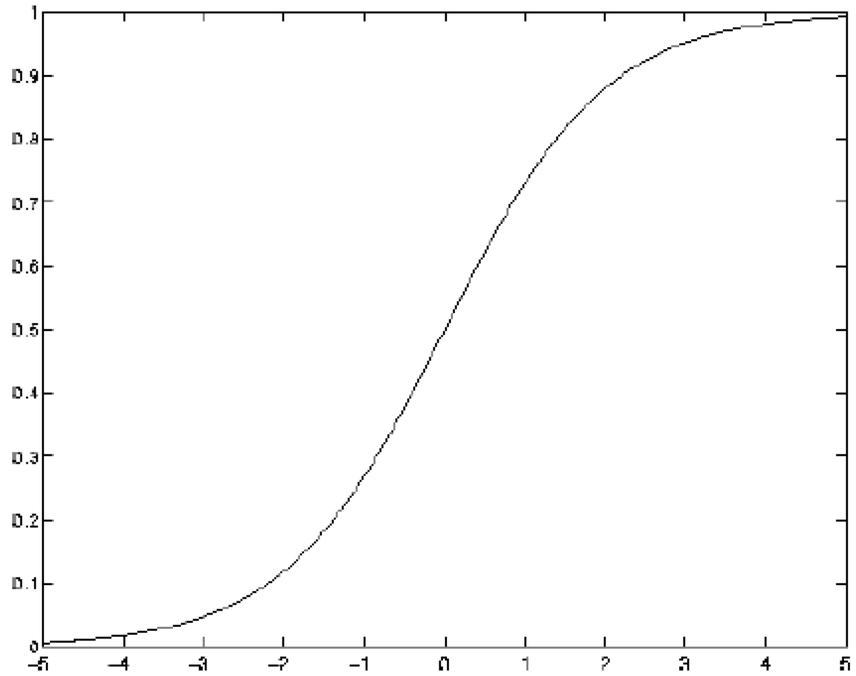
$$= \frac{1}{1 + \exp(-a)} \equiv g(a)$$

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

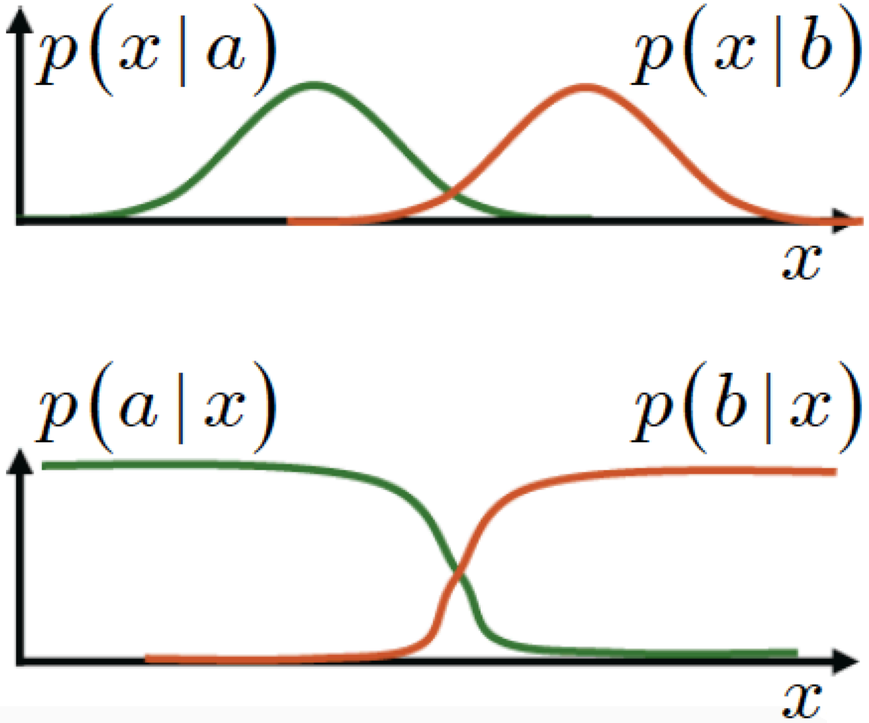
Slide source: B. Leibe Slides ML, 2015

Logistic Sigmoid Activation Function

$$g(a) \equiv \frac{1}{1+\exp(-a)}$$



Example: Normal distributions with identical covariance



Slide source: B. Leibe Slides ML, 2015

Normalized Exponential

General case of K>2 classes:

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

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

with $a_k = \ln p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)$

- This is known as the **normalized exponential** or **softmax** function
- Can be regarded as a multi class generalised of the logistic sigmoid

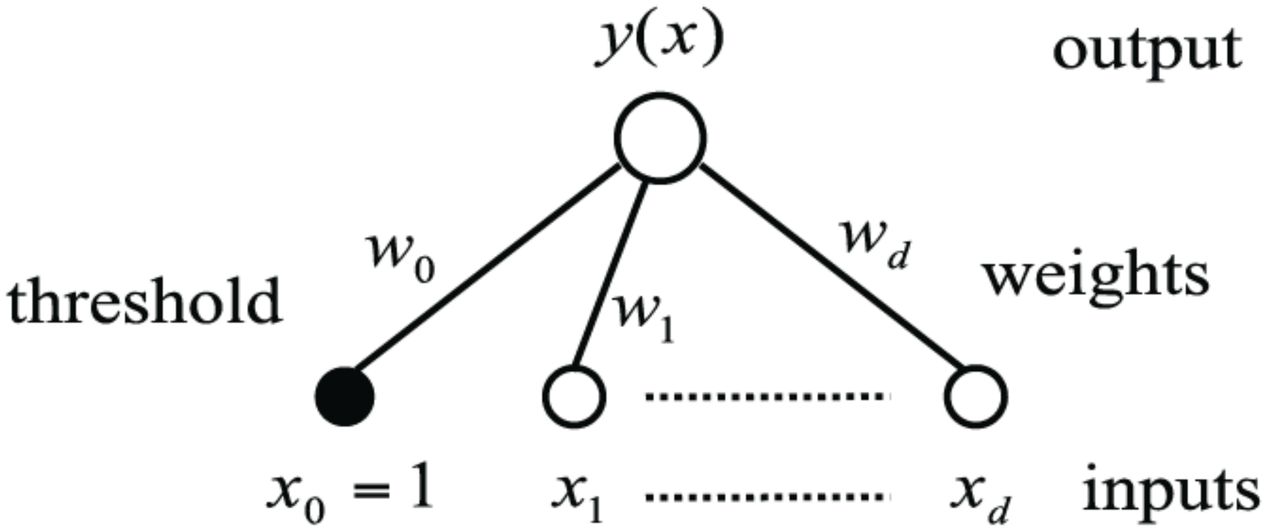
Slide source: B. Leibe Slides ML, 2015

Relationship to Neural Networks

2-Class case

$$y(\mathrm{x}) = g(\sum_{i=0}^{D} w_i x_i) \text{ with } x_0 = 1 \text{ constant}$$

Neural network (“single-layer perceptron”)



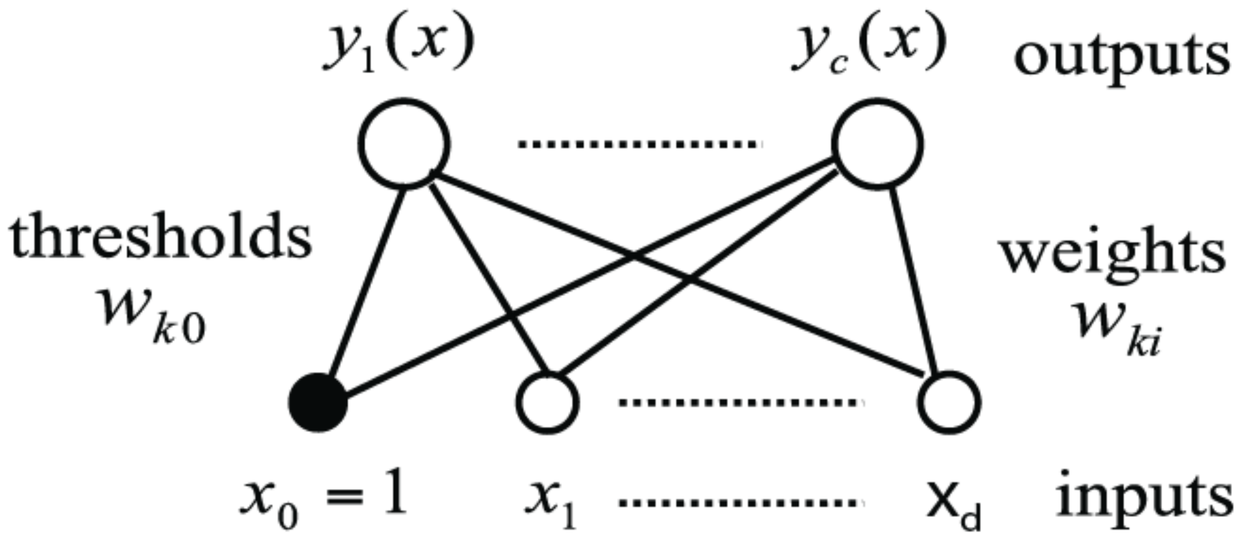
Slide source: B. Leibe Slides ML, 2015

Relationship to Neural Networks

Multi-class case

$$y_k(\mathrm{x}) = g(\sum_{i=0}^{D} w_{ki} x_i) \text{ with } x_0 = 1 \text{ constant}$$

Multi-class perceptron



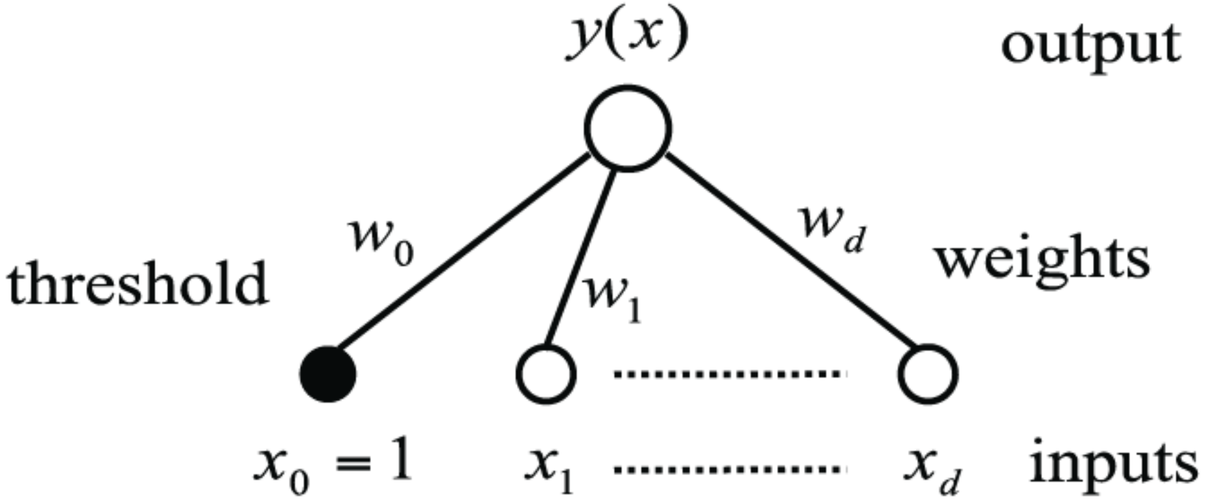
Slide source: B. Leibe Slides ML, 2015

Logistic Discrimination

If we use the logistic sigmoid activation function…

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

$$y(\mathrm{x}) = g(\mathrm{w}^T\mathrm{x} + w_0)$$



… then we can interpret the $y(x)$ as posterior probabilities!

Slide source: B. Leibe Slides ML, 2015

Other Motivation for Nonlinearity

Recall least-squares classification

- One of the problems was that data points that are “too correct” have a strong influence on the decision surface under a squared-error criterion.

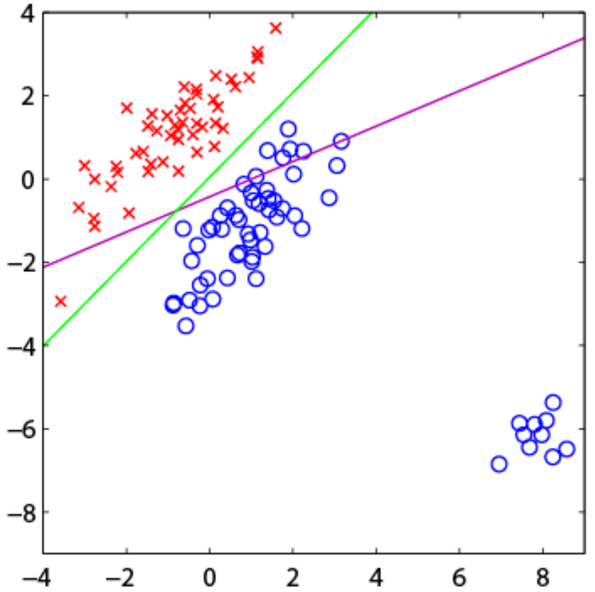
$$E(\mathrm{w}) = \sum_{n=1}^{N} (y(\mathrm{x}_n; \mathrm{w}) - \mathrm{t}_n)^2$$

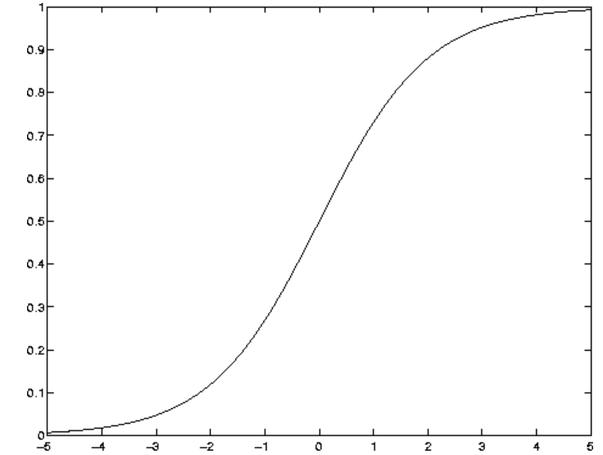
- Reason: the output of $y(\mathrm{x}_n; \mathrm{w})$ can grow arbitrarily large for some x_n :

$$y(\mathrm{x}; \mathrm{w}) = \mathrm{w}^T \mathrm{x} + w_0$$

- By choosing a suitable nonlinearity (e.g.a sigmoid), we can limit those influences

$$y(\mathrm{x}; \mathrm{w}) = g(\mathrm{w}^T \mathrm{x} + w_0)$$





Slide source: B. Leibe Slides ML, 2015

Discussion: Generalized Linear Models

Advantages

- The nonlinearity gives us more flexibility.
- Can be used to limit the effect of outliers
- Choice of a sigmoid leads to a nice probabilistic interpretation

Disadvantages

- Least-squares minimisation in general no longer leads to a close-form analytical solution.

➡Need to apply iterative methods.
➡Gradient descent.

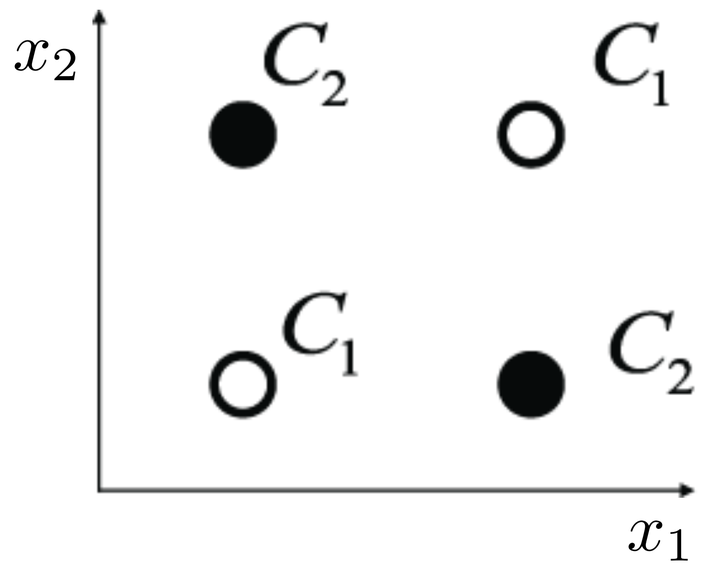
Slide source: B. Leibe Slides ML, 2015

Linear Separability

Up to now: restrictive assumption

- Only consider linear decision boundaries

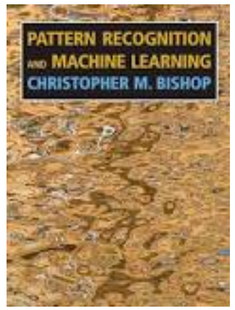
Classical counterexample: XOR



Slide source: B. Leibe Slides ML, 2015

Readings

Bishop’s book



More information on Linear Discriminant Functions can be found in Chapter 4 of Bishop’s book (in particular Chapter 4.1).

Slide is taken from B. Leibe, ML, 2017

You should be able to answer …

- What is 1 of K coding scheme?
- Formula for linear discriminant function, explain its entries.
- What does linearly separable mean?
- Extension of linear discriminant to multiple classes - two strategies. What are the difficulties with these strategies?
- What is a sum-of-squares error?
- Problems with least squares?
- Difference between linear model and generalised linear model.
- Sigmoid activation function: formula and how to implement it.
- What is softmax?
- What are advantages and disadvantages of generalised linear model?