Visual Computing Institute
RWTH AACHEN UNIVERSITY

Elements of Machine Learning & Data Science

Winter semester 2023/24

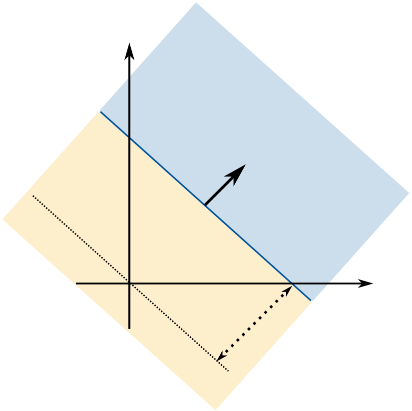
Lecture 14 – Linear Discriminants

01.12.2023

Prof. Bastian Leibe

Machine Learning Topics

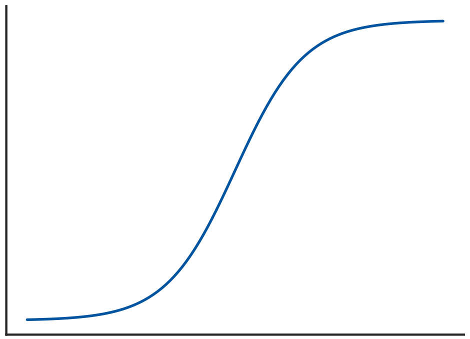
1. Introduction to ML
2. Probability Density Estimation
3. **Linear Discriminants**
4. Linear Regression
5. Logistic Regression
6. Support Vector Machines
7. AdaBoost
8. Neural Network Basics



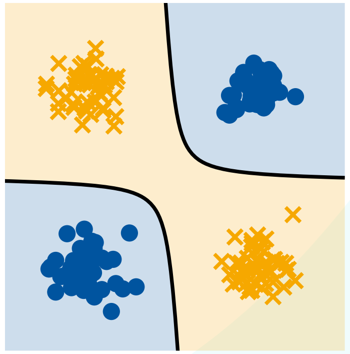
Linear Discriminant Functions

$$E(\mathbf{w}) = \frac{1}{2}\sum_n (y(\mathbf{x}_n; \mathbf{w}) - t_n)^2$$

Least-Squares Classification



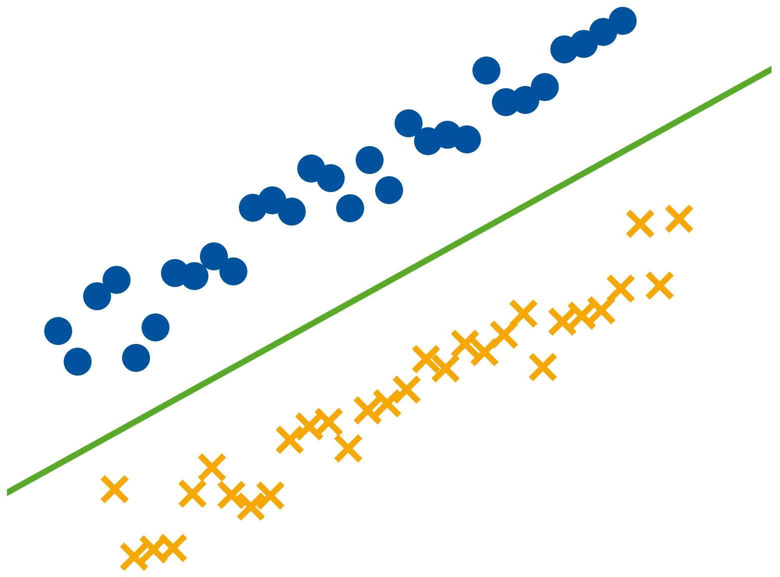
Activation Functions



Basis Functions

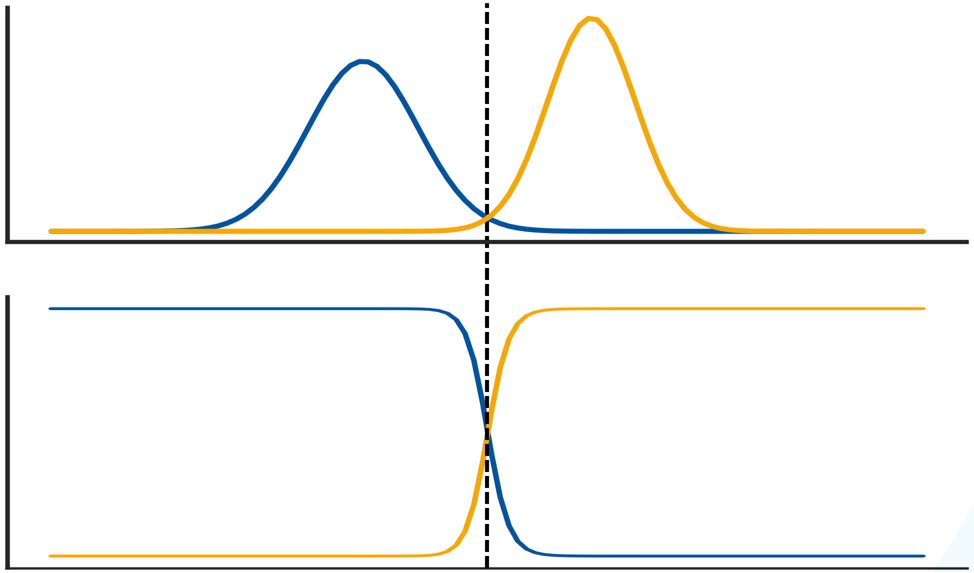
Linear Discriminants

1. **Motivation: Discriminant Functions**
2. Linear Discriminant Functions
3. Least-Squares Classification
4. Generalized Linear Discriminants
5. Basis Functions



Discriminant Functions

- Remember Bayes Decision Theory
  - Model the likelihood $p(\mathbf{x}|\mathcal{C}_k)$ and the prior $p(\mathcal{C}_k)$
  - From this, we can compute the posterior $p(\mathcal{C}_k|\mathbf{x})$
  - Bayes optimal decision: Decide for class $\mathcal{C}_1$ if

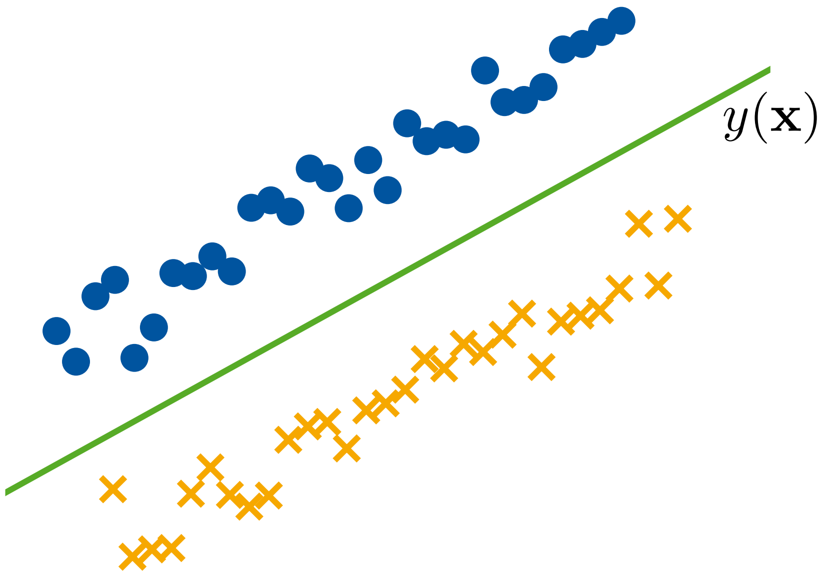


$$p(\mathcal{C}_1|\mathbf{x}) > p(\mathcal{C}_2|\mathbf{x}) \quad \Leftrightarrow \quad \frac{p(\mathbf{x}|\mathcal{C}_1)}{p(\mathbf{x}|\mathcal{C}_2)} > \frac{p(\mathcal{C}_2)}{p(\mathcal{C}_1)}$$

by comparing posteriors *by comparing likelihoods and priors*

Discriminant Functions

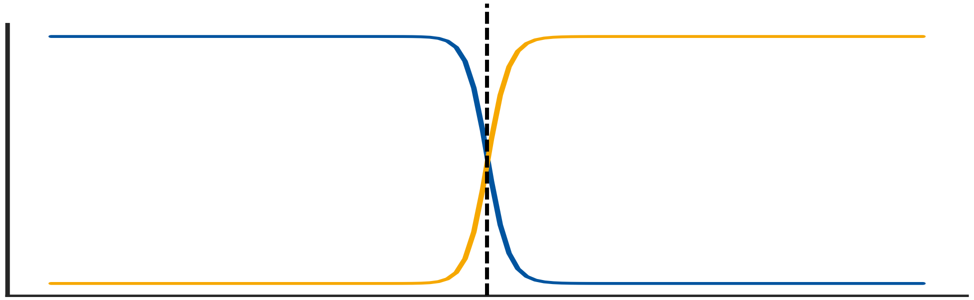
- This chapter: Different approach
  - Directly represent the decision boundary with a discriminant function $y(\mathbf{x})$
  - Without explicit modeling of probability densities!
    - *We will learn ways to define* $y(\mathbf{x})$ *such that we still make a decision based on posteriors…*
    - *…but we don't have to. This framework gives us more flexibility.*



Idea

- Formulate classification in terms of comparisons
  - Bayes Decision Theory:

$$\begin{aligned} & p(\mathcal{C}_1|\mathbf{x}) && > p(\mathcal{C}_2|\mathbf{x}) \\ \Longleftrightarrow\ & p(\mathcal{C}_1|\mathbf{x}) - p(\mathcal{C}_2|\mathbf{x}) && > 0 \\ \Longleftrightarrow\ & y(\mathbf{x}) && > 0 \end{aligned}$$



  - More general:
    - Define a discriminant function $y(\mathbf{x})$
    - Classify $\mathbf{x}$ as class $\mathcal{C}_1$ if $y(\mathbf{x}) > 0$
  - Advantage: more flexibility

E.g., we could now define

$$y(\mathbf{x}) = p(\mathcal{C}_1|\mathbf{x}) - p(\mathcal{C}_2|\mathbf{x})$$

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

Idea

- **Multi-class case**
  - Define discriminant functions

$$y_1(\mathbf{x}), \ldots, y_K(\mathbf{x})$$

  - Classify $\mathbf{x}$ as class $\mathcal{C}_k$ if

$$y_k(\mathbf{x}) > y_j(\mathbf{x}) \quad \forall j \neq k$$

- Again, this is compatible with Bayes Decision Theory:
  - E.g., $y_k(\mathbf{x}) = p(\mathcal{C}_k|\mathbf{x})$

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

$$y_k(\mathbf{x}) = \log p(\mathbf{x}|\mathcal{C}_k) + \log p(\mathcal{C}_k)$$

Problem Formulation

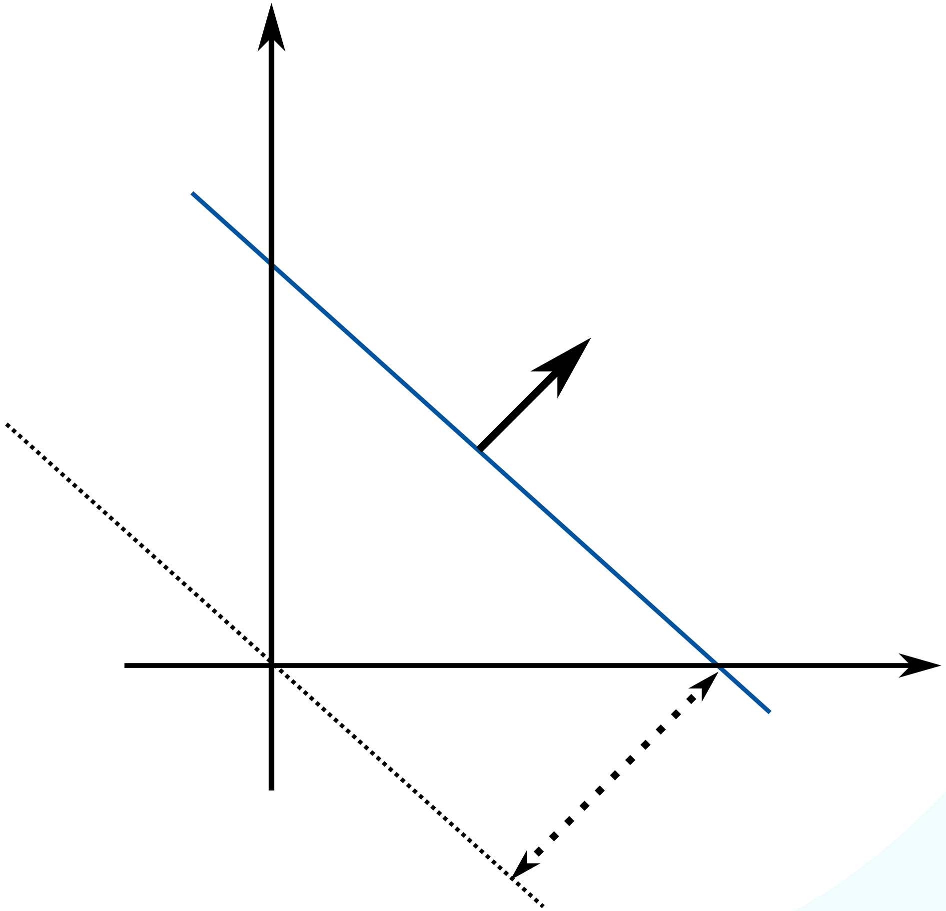
- **General classification problem**
  - Goal: take a new input $\mathbf{x}$ and assign it to one of K classes $\mathcal{C}_k$
  - Given training set $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
    with target values $\mathcal{T} = \{\mathbf{t}_1, \ldots, \mathbf{t}_N\}$

  $\Rightarrow$ Learn a discriminant function $y(\mathbf{x})$ to perform the classification

- **2-class problem**
  - Binary target values $t_n \in \{-1, 1\}$ or $t_n \in \{0, 1\}$
- ***K*-class problem**
  - 1-of-*K* coding scheme, e.g. $\mathbf{t}_n = (0, 1, 0, 0, 0)^{\mathsf{T}}$

Linear Discriminants

1. Motivation: Discriminant Functions
2. **Linear Discriminant Functions**
3. Least-Squares Classification
4. Generalized Linear Discriminants
5. Basis Functions



Linear Discriminant Functions

- 2-class problem
  - $y(\mathbf{x}) > 0$: Decide for class $\mathcal{C}_1$, else for $\mathcal{C}_2$
- In the following, we focus on linear discriminant functions:

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

weight vector

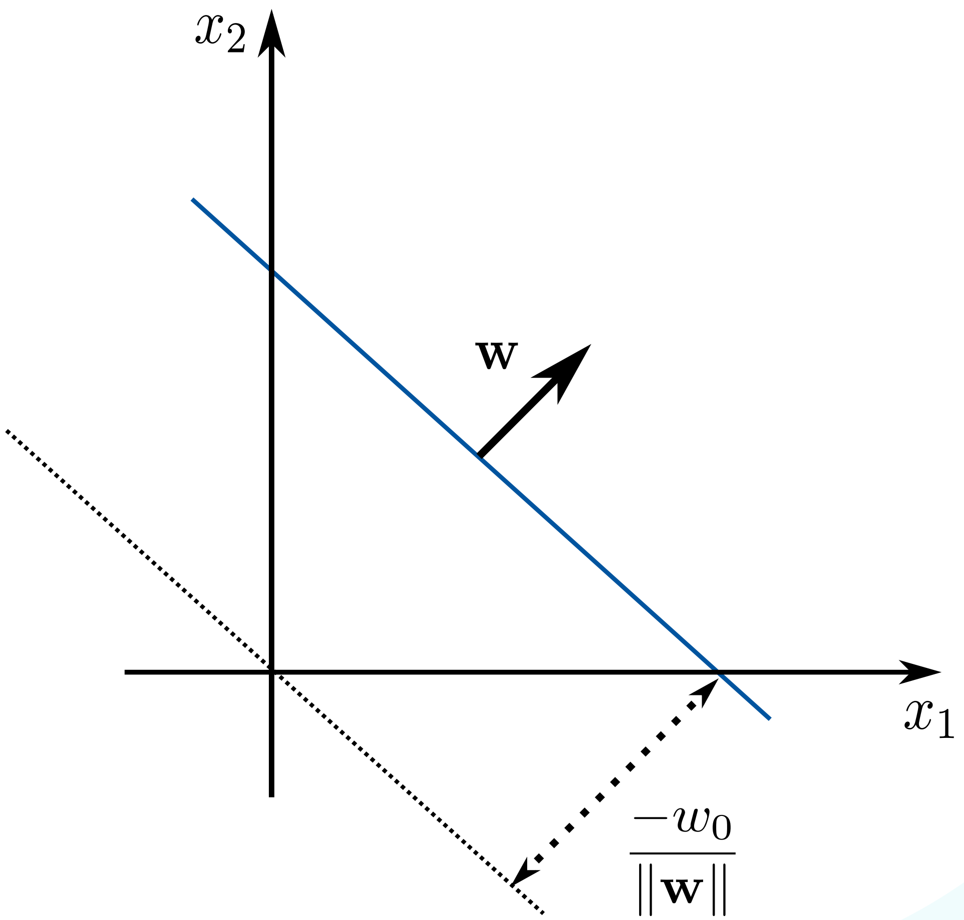
“bias”
(= threshold)

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

Intuition

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

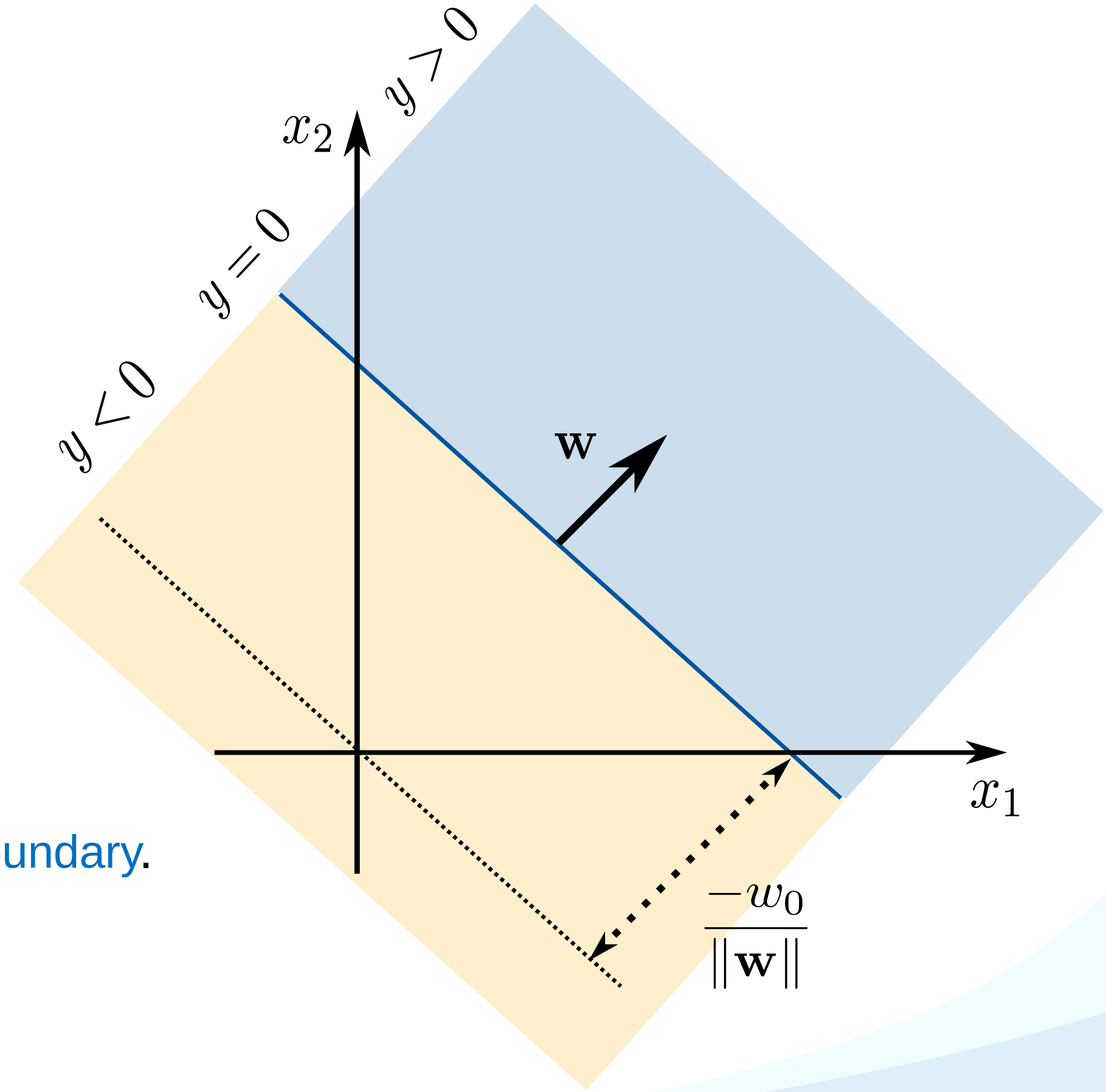
- Graphical interpretation:
  - Normal equation of a hyperplane
  - Normal vector: $\mathbf{w}$
  - Offset: $\frac{-w_0}{\|\mathbf{w}\|}$



Intuition

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

- Graphical interpretation:
  - Normal equation of a hyperplane
- The hyperplane is given by $y(\mathbf{x}) = 0$
  - One side: $y(\mathbf{x}) > 0$
  - Other side: $y(\mathbf{x}) < 0$
- This hyperplane defines the decision boundary.



Notation

- Let's look at the equation in detail

$$y(\mathbf{x}) = \mathbf{w}^{\mathsf{T}}\mathbf{x} + w_0 = \sum_{i=1}^{D} w_i x_i + w_0$$

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

- Alternative notation with extended vector

$$y(\mathbf{x}) = \sum_{i=0}^{D} w_i x_i \quad \text{with } x_0 = 1$$
$$= \widetilde{\mathbf{w}}^{\mathsf{T}}\tilde{\mathbf{x}}$$

$$\tilde{\mathbf{x}} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_D \end{bmatrix} \quad \widetilde{\mathbf{w}} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_D \end{bmatrix}$$

D : Number of Dimensions

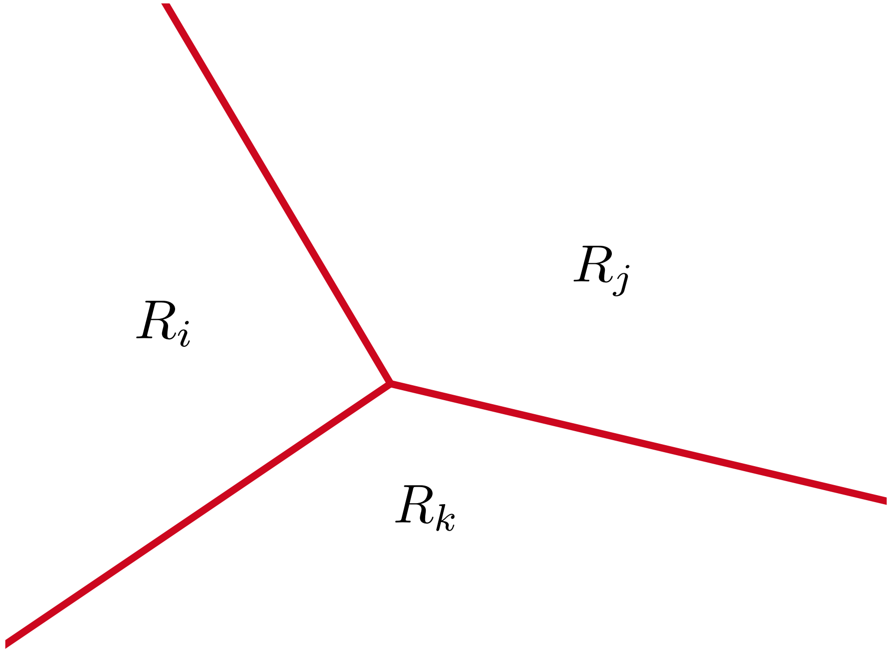
Extension to Multiple Classes

- ***K*-class discriminant**
  - Combination of K linear functions

$$y_k(\mathbf{x}) = \mathbf{w}_k^\mathsf{T}\mathbf{x} + w_{k0}, \quad k = 1, \ldots, K$$

- Interpretation
  - *Decide for* $\mathcal{C}_k$ *iff* $y_k(\mathbf{x}) > y_j(\mathbf{x}) \quad \forall j \neq k$
  - Resulting decision hyperplanes:

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



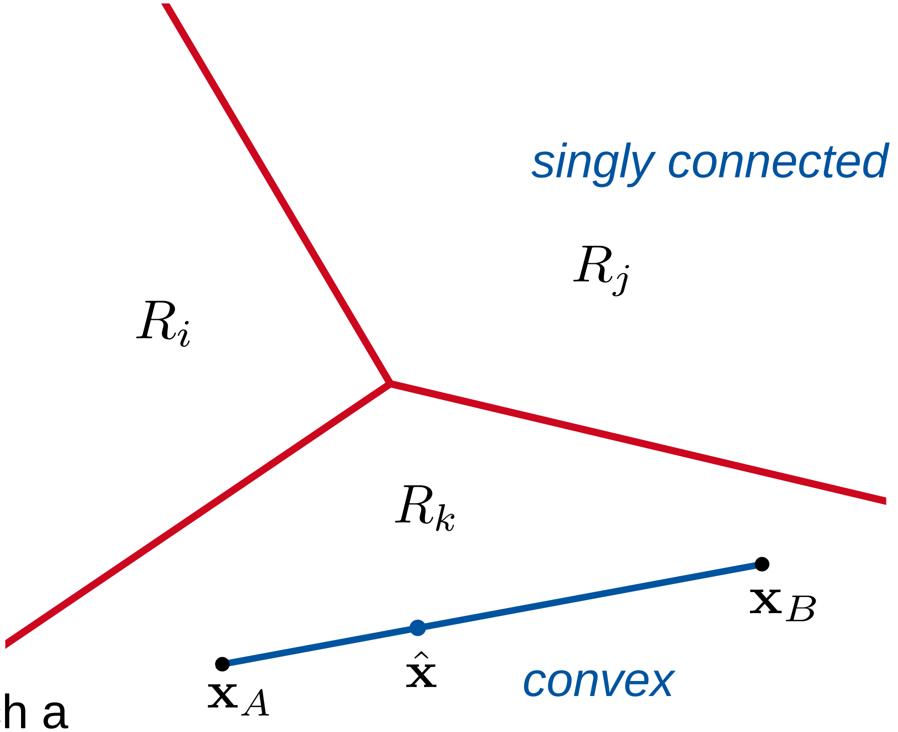
Extension to Multiple Classes

- ***K*-class discriminant**
  - Combination of K linear functions

$$y_k(\mathbf{x}) = \mathbf{w}_k^{\mathsf{T}}\mathbf{x} + w_{k0}, \quad k = 1, \ldots, K$$

- Interpretation
  - Decide for $\mathcal{C}_k$ iff $y_k(\mathbf{x}) > y_j(\mathbf{x}) \quad \forall j \neq k$
  - Resulting decision hyperplanes:

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

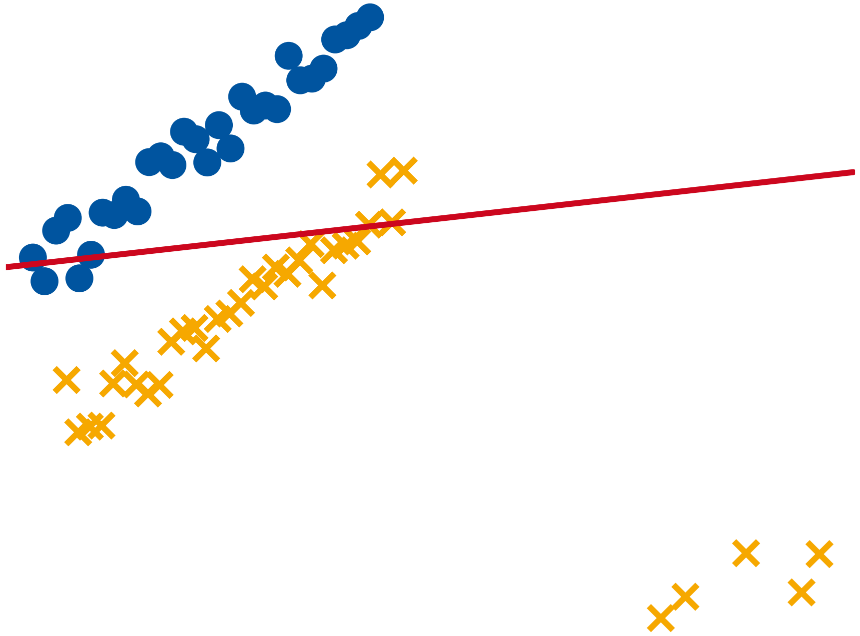


  - It can be shown that the decision regions of such a discriminant are always singly connected and convex.

$\Rightarrow$ Particularly suitable for problems with unimodal conditional densities $p(\mathbf{x}|\mathbf{w}_i)$

Linear Discriminants

1. Motivation: Discriminant Functions
2. Linear Discriminant Functions
3. **Least-Squares Classification**
4. Generalized Linear Discriminants
5. Basis Functions



General Classification Problem

- K Classes described by linear discriminant models:

$$y_k(\mathbf{x}) = \mathbf{w}_k^{\mathsf{T}}\mathbf{x} + w_{k0}, \quad k = 1, \ldots, K$$

- Group them together using vector notation:

$$\mathbf{y}(\mathbf{x}) = \widetilde{\mathbf{W}}^{\mathsf{T}}\widetilde{\mathbf{x}} = \begin{bmatrix} y_1(\mathbf{x}) \\ y_2(\mathbf{x}) \\ \vdots \\ y_k(\mathbf{x}) \end{bmatrix}$$

where

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

$$\widetilde{\mathbf{x}} = [1, x_1, \ldots, x_D]^{\mathsf{T}}$$

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

$\Rightarrow$ We can directly compare it to the target value

$$\mathbf{t} = [t_1, \ldots, t_k]^{\mathsf{T}}$$

Classification Problem for the Entire Dataset

- Write the classification output for the whole dataset:

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

where

$$\widetilde{\mathbf{W}} = [\tilde{\mathbf{w}}_1, \ldots, \tilde{\mathbf{w}}_K] = \begin{bmatrix} w_{01} & \cdots & w_{0K} \\ \vdots & \ddots & \vdots \\ w_{D1} & \cdots & w_{DK} \end{bmatrix}$$

  - Using the data matrix

$$\widetilde{\mathbf{X}} = \begin{bmatrix} \tilde{\mathbf{x}}_1^\mathsf{T} \\ \vdots \\ \tilde{\mathbf{x}}_N^\mathsf{T} \end{bmatrix} = \begin{bmatrix} x_{10} & \cdots & x_{1D} \\ \vdots & \ddots & \vdots \\ x_{N0} & \cdots & x_{ND} \end{bmatrix}$$

  - Similarly, we group the target vectors in a matrix

$$\mathbf{T} = \begin{bmatrix} \mathbf{t}_1^\mathsf{T} \\ \vdots \\ \mathbf{t}_N^\mathsf{T} \end{bmatrix} = \begin{bmatrix} t_{11} & \cdots & t_{1K} \\ \vdots & \ddots & \vdots \\ t_{N1} & \cdots & t_{NK} \end{bmatrix}$$

  - We can now compare the classifier output with the target matrix: $\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T}$

Defining the Classification Problem

- Comparing the classifier output with the target matrix:

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

- Goal: Choose $\widetilde{\mathbf{W}}$ such that this difference becomes minimal
  - What does *minimal* mean here?
  - *How strongly do we want to penalize deviations from the ideal target value?*

- Idea: define an error function that specifies the loss for each deviation

$$E(\widetilde{\mathbf{W}}) = \sum_{n=1}^{N} \sum_{k=1}^{K} L\left(y_k(\mathbf{x}_n; \mathbf{w}_k), t_{nk}\right)$$

Least-Squares Classification

- Simplest approach: minimize Sum-of-squares error

$$E(\mathbf{W}) = \frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{K}(y_k(\mathbf{x}_n;\mathbf{w}_k) - t_{nk})^2$$
$$= \frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{K}(\mathbf{w}_k^{\mathsf{T}}\mathbf{x}_n - t_{nk})^2$$

Simplified notation:
Leaving out the $\tilde{\mathbf{x}}$ *…*

- How do we minimize this function?
  - *Take the derivative and set it to zero…*

Derivation

- Let's concentrate on the two-class case first:

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}(\mathbf{w}^\top \mathbf{x}_n - t_n)^2 \qquad t_n \in \{-1, 1\}$$

- Taking the derivative:

$$\frac{\partial E(\mathbf{w})}{\partial \mathbf{w}} = \sum_{n=1}^{N}(\mathbf{w}^\top \mathbf{x}_n - t_n)\mathbf{x}_n$$

$$= \mathbf{X}^\top(\mathbf{X}\mathbf{w} - \mathbf{t})$$

with $\mathbf{X} = \begin{pmatrix} \mathbf{x}_1^\top \\ \vdots \\ \mathbf{x}_N^\top \end{pmatrix}$ $\quad \mathbf{t} = (t_1, \ldots, t_N)^\top$

Linear Algebra textbook:

$$\frac{\partial \mathbf{a}^\top \mathbf{b}}{\partial \mathbf{a}} = \mathbf{b}$$

- Setting the derivative to zero:

$$\frac{\partial E(\mathbf{w})}{\partial \mathbf{w}} = \mathbf{X}^\top (\mathbf{X}\mathbf{w} - \mathbf{t}) \stackrel{!}{=} 0$$

$$\mathbf{X}^\top \mathbf{X}\mathbf{w} = \mathbf{X}^\top \mathbf{t}$$

$$\mathbf{w} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{t}$$

$$= \mathbf{X}^\dagger \mathbf{t}$$

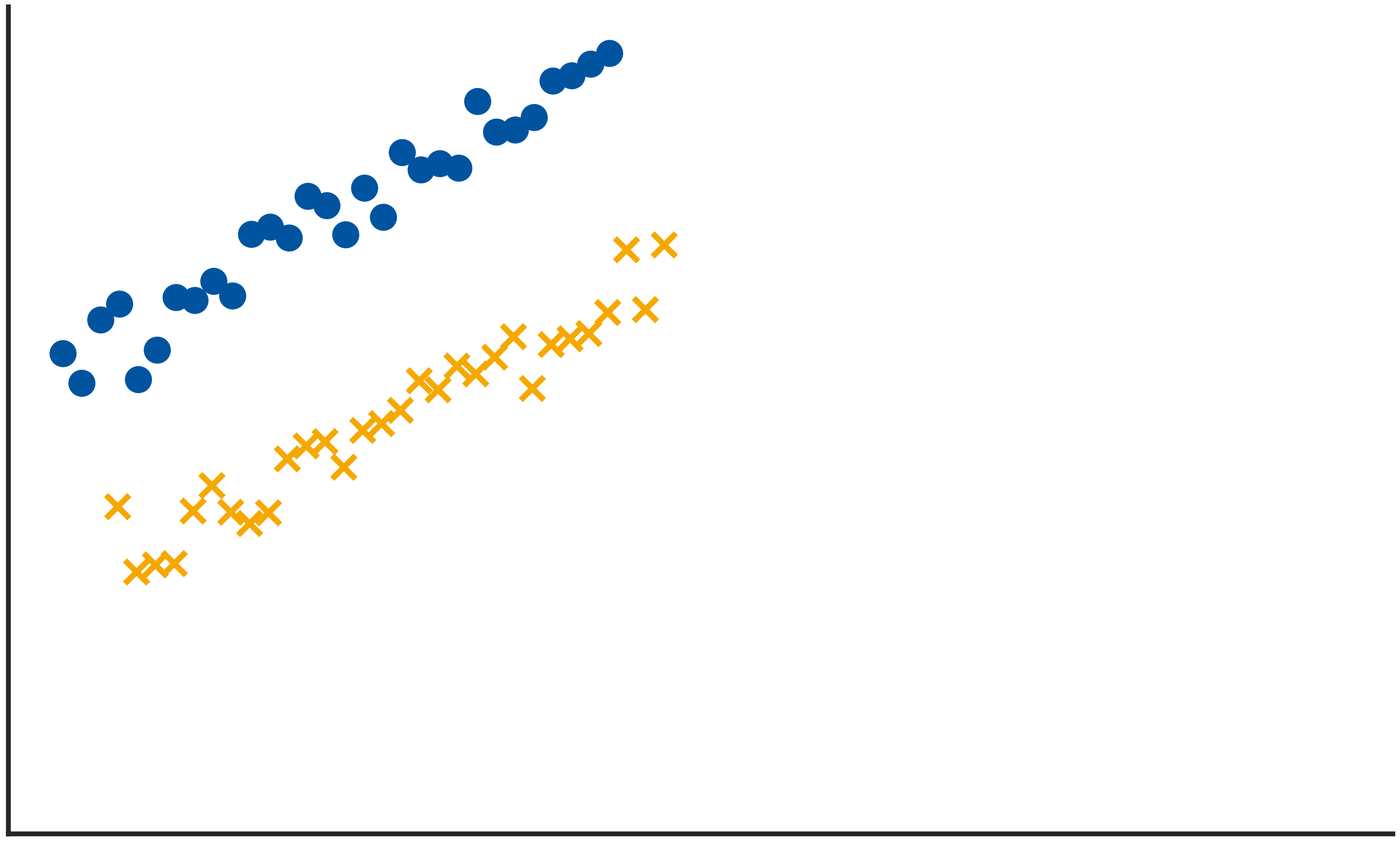
"pseudo-inverse"

- We then obtain the discriminant function as

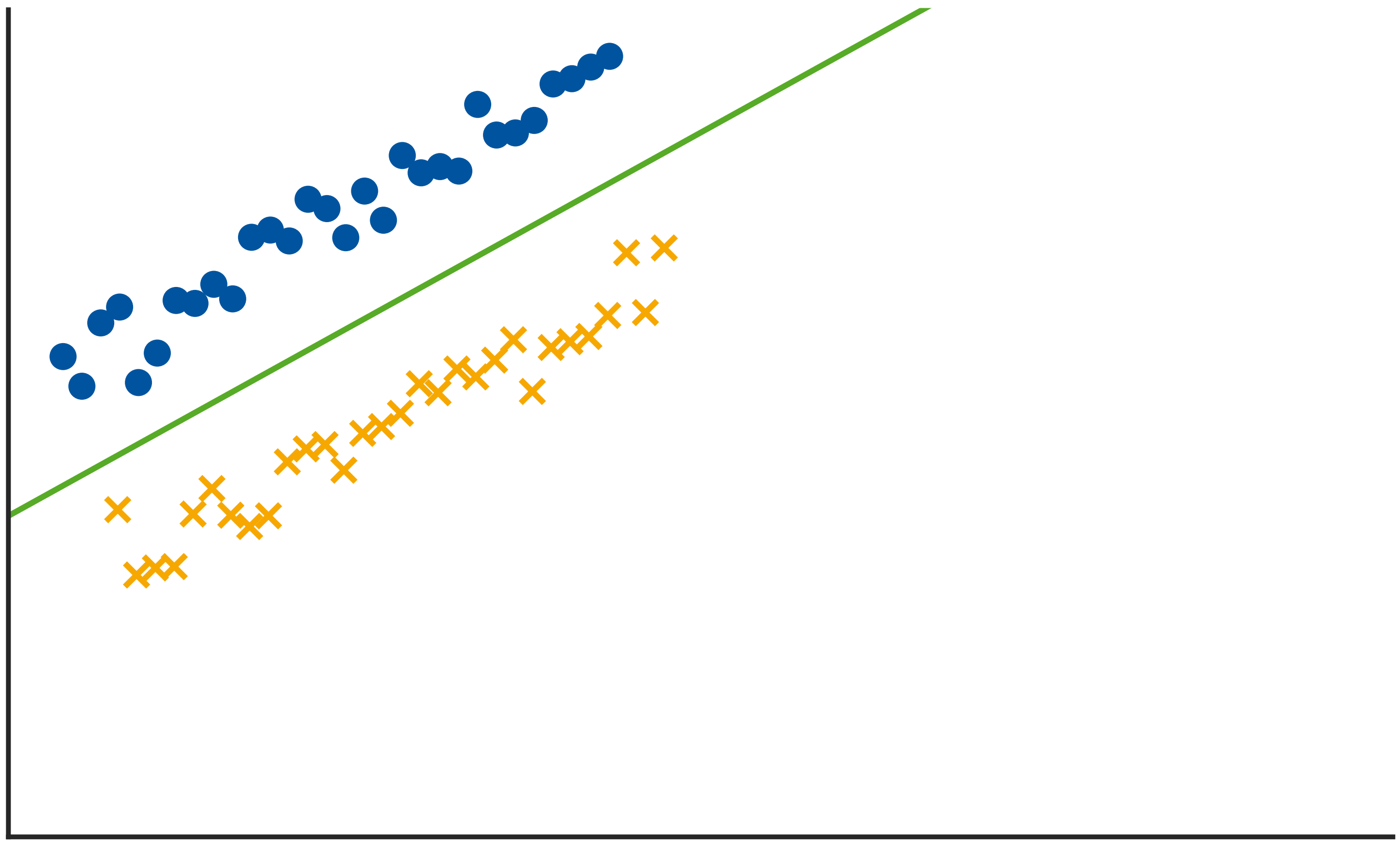
$$y(\mathbf{x}; \mathbf{w}) = \mathbf{w}^\top \mathbf{x} = \mathbf{t}^\top \left(\mathbf{X}^\dagger\right)^\top \mathbf{x}$$

Exact, closed-form solution!

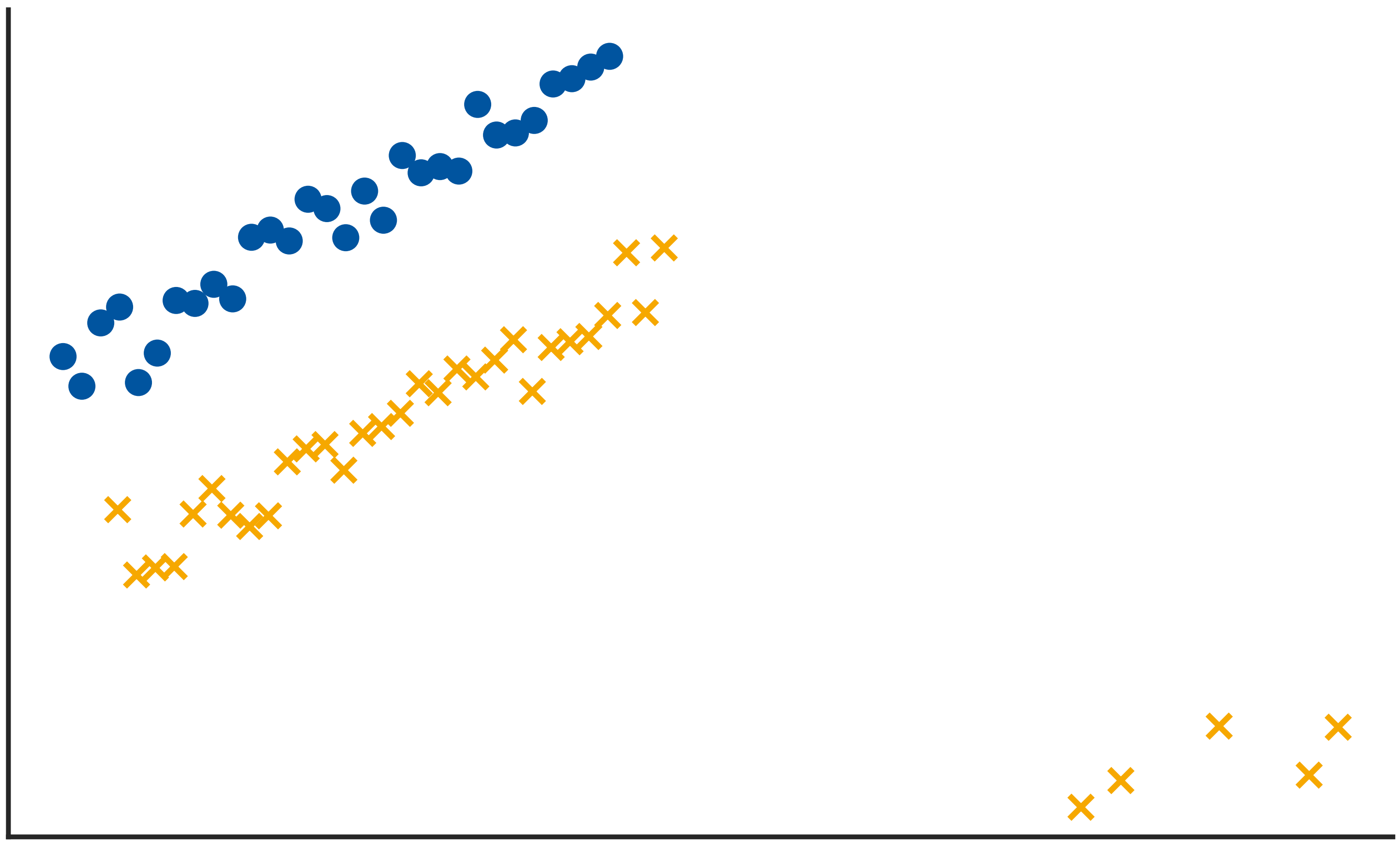
Example: Two Classes



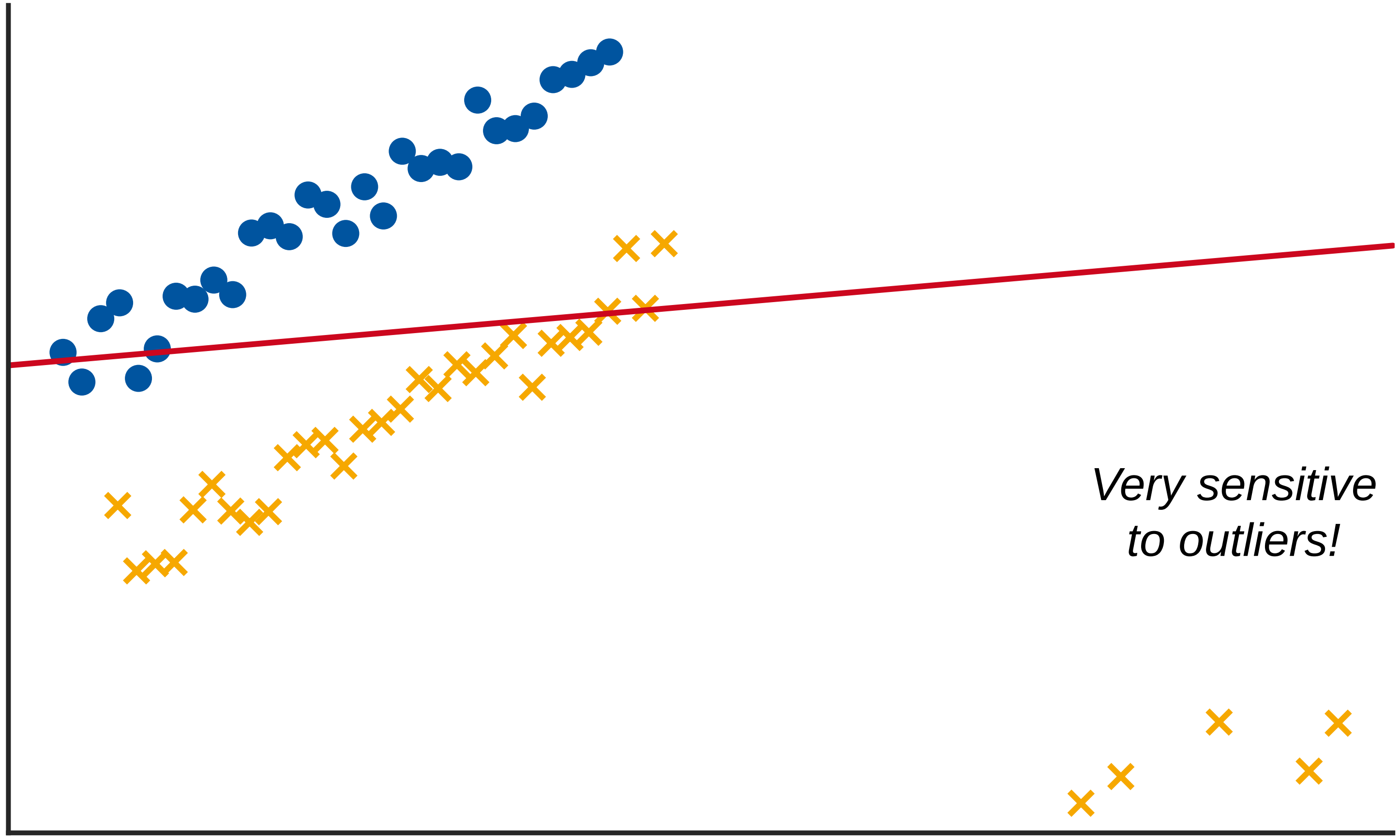
Example: Two Classes



Example: Two Classes



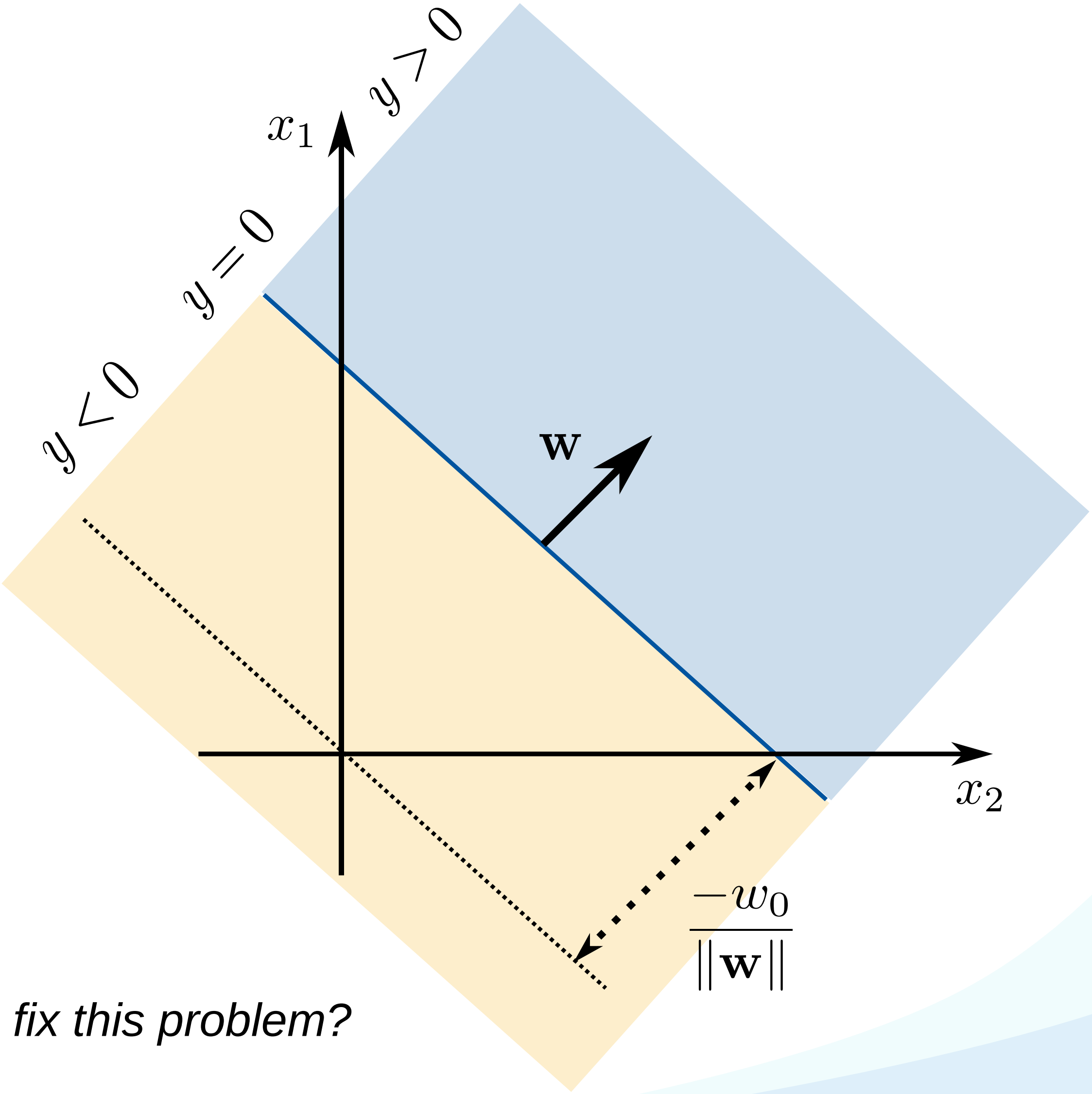
Example: Two Classes



Why Does This Happen?

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

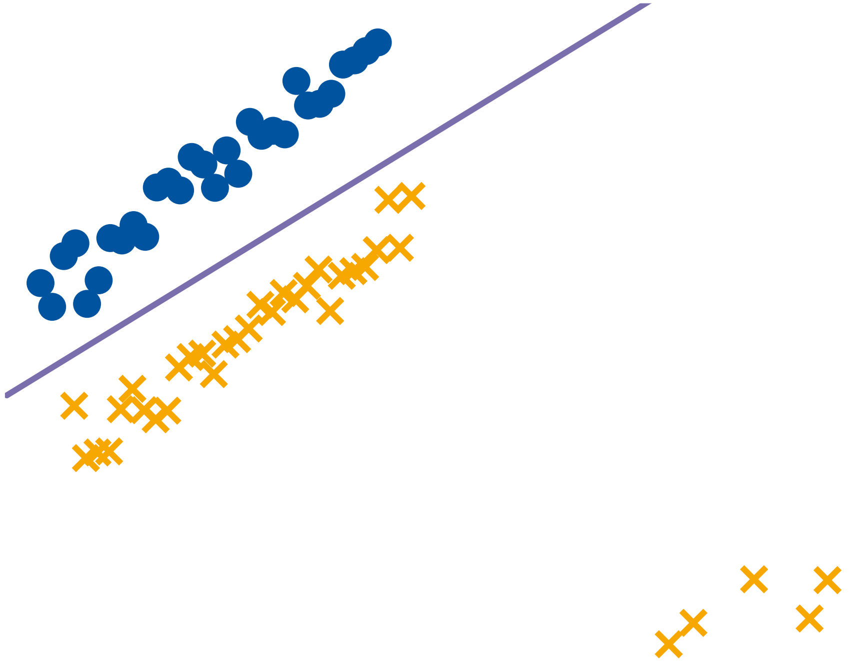
- Remember the interpretation of $y(\mathbf{x})$
  - Normal equation of a hyperplane
- $y(\mathbf{x})$ measures the (signed) distance of the point $\mathbf{x}$ from the hyperplane.
- However, we now compare it to a target value of $t_n \in \{-1, 1\}$ …



How can we fix this problem?

Linear Discriminants

1. Motivation: Discriminant Functions
2. Linear Discriminant Functions
3. Least-Squares Classification
4. **Generalized Linear Discriminants**
5. Basis Functions



Generalized Linear Models

- So far: model classification by linear discriminant function

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

- Generalize this with an activation function $g(\cdot)$:

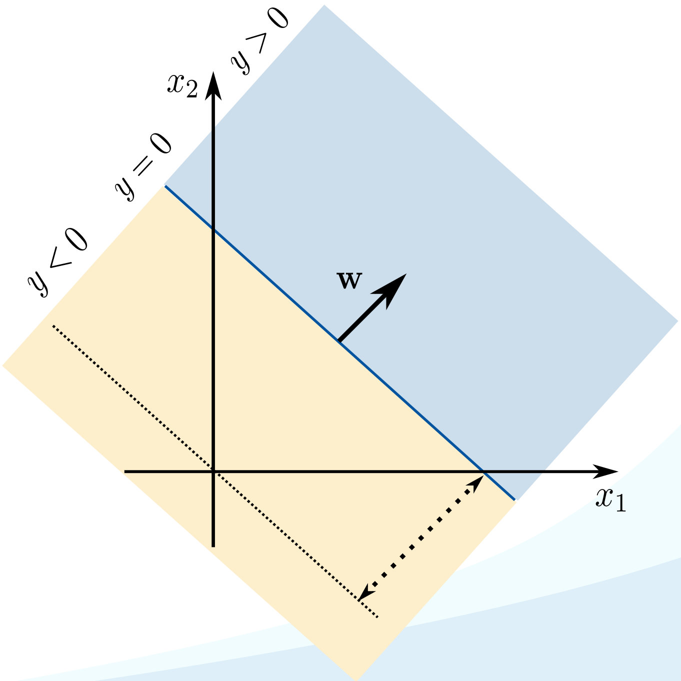
$$y(\mathbf{x}) = g(\mathbf{w}^\mathsf{T}\mathbf{x} + w_0)$$

- Remarks
  - $g(\cdot)$ may be non-linear.
  - Decision surfaces correspond to

$$y(\mathbf{x}) = \text{const} \iff \mathbf{w}^\mathsf{T}\mathbf{x} + w_0 = \text{const}$$

$\Rightarrow$ *If $g(\cdot)$ is monotonous (which is typically the case), the decision boundaries are still linear functions of $\mathbf{x}$.*

Generalized Linear Model



Activation Functions

- Recall least-squares classification:
  - Outliers have strong influence

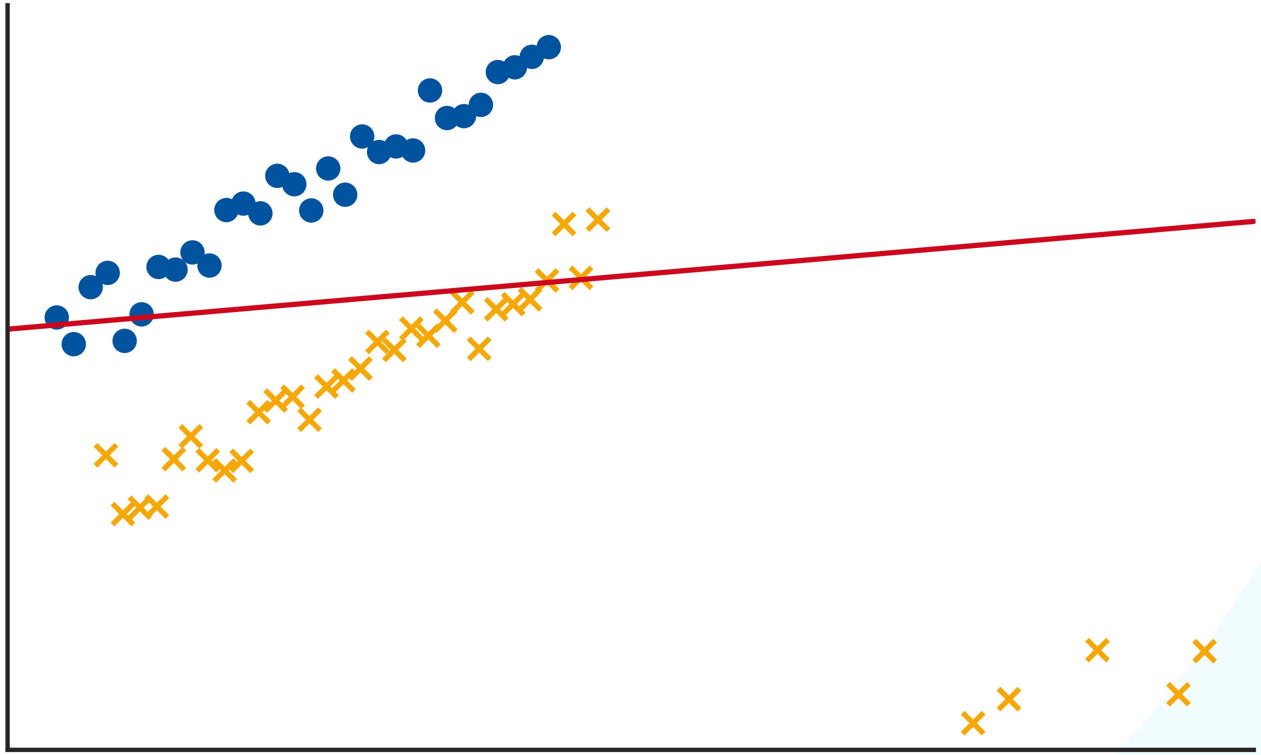
$$E(\mathbf{w}) = \sum_{n=1}^{N} (y(\mathbf{x}_n; \mathbf{w}) - t_n)^2$$

  - This is because the output $y(\mathbf{x}; \mathbf{w})$ can grow arbitrarily large:

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

  - Choosing a suitable nonlinearity can limit those influences:

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



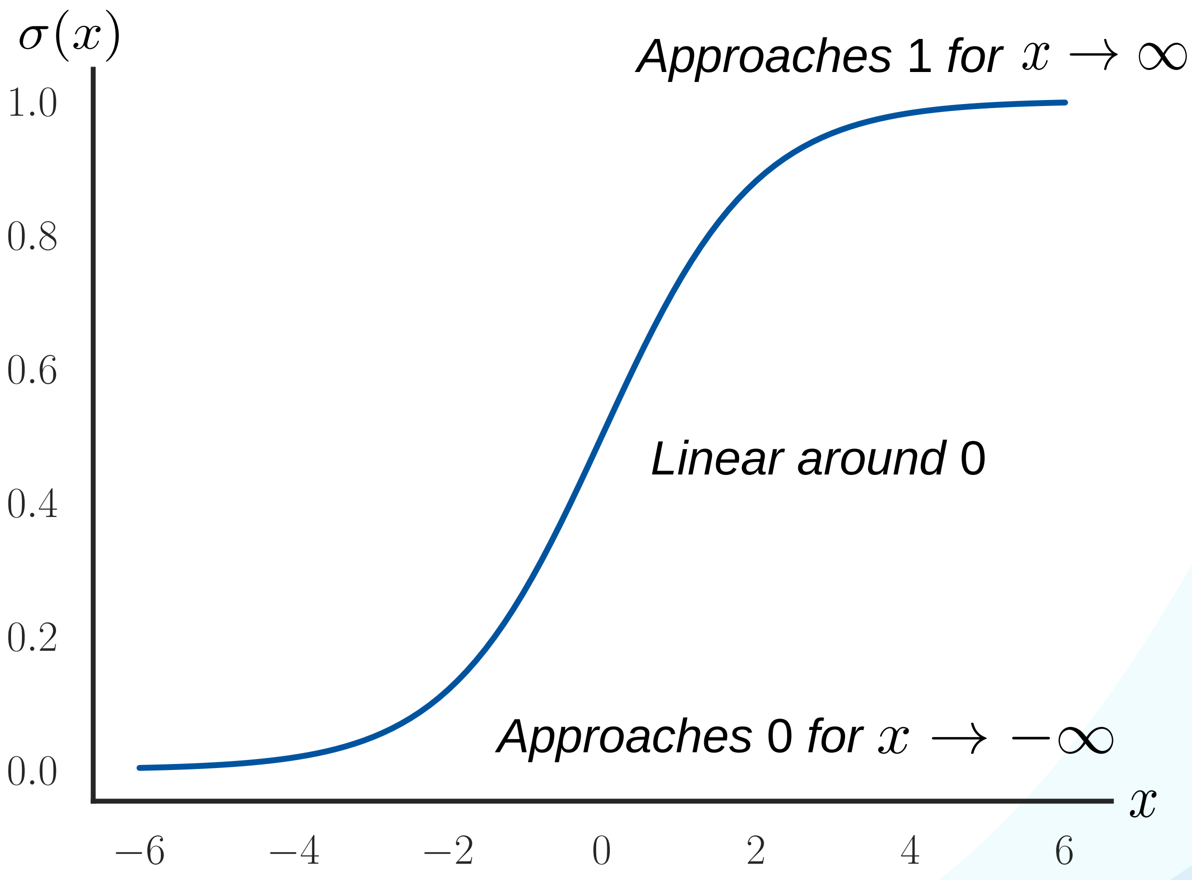
The Logistic Sigmoid

- To limit the influence of outliers, we can use the logistic sigmoid function:

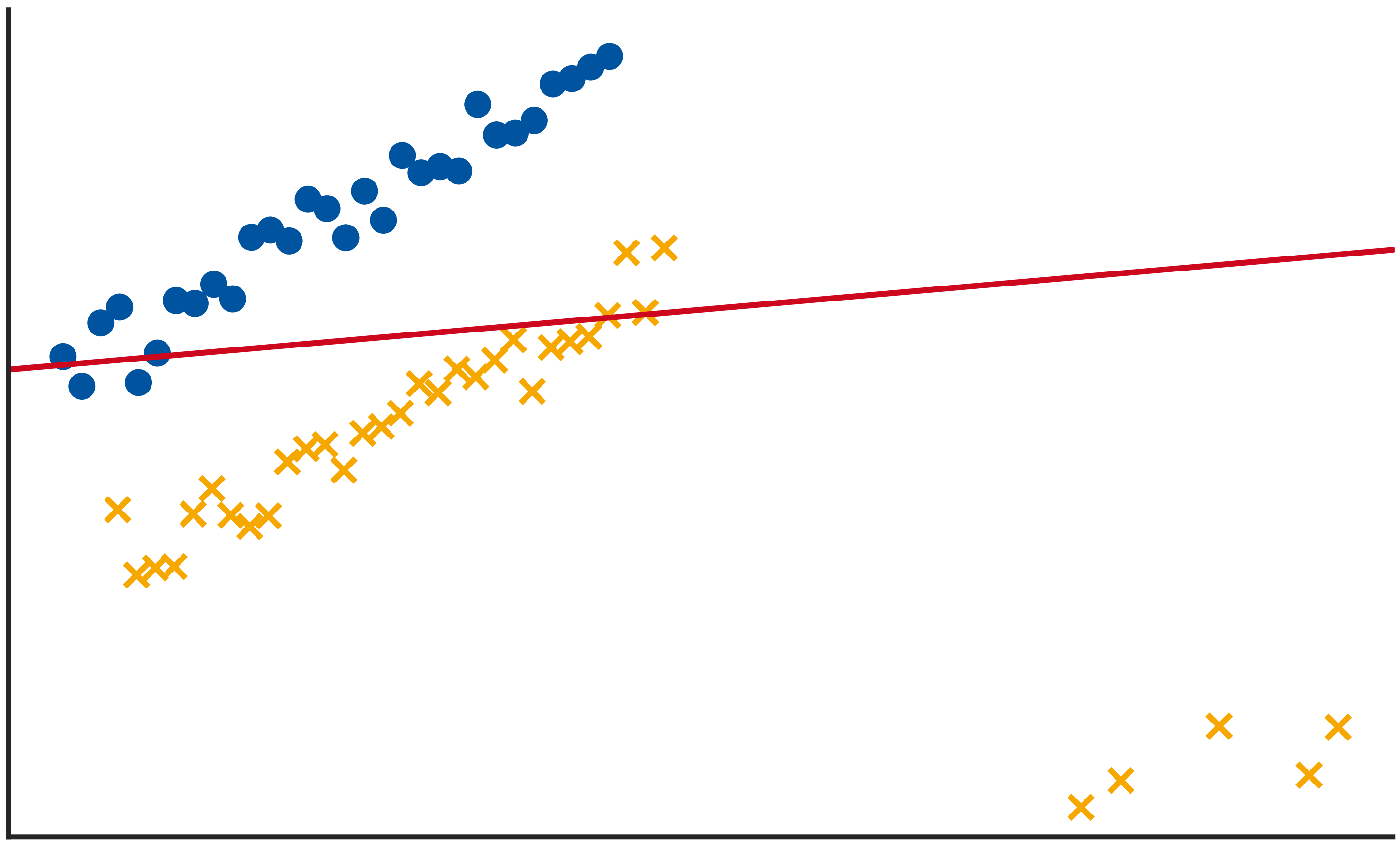
$$\sigma(x) = \frac{1}{1 + \exp(-x)}$$

- For 2-class problems, we scale it to the output range (-1,1)
(known as tangens hyperbolicus):

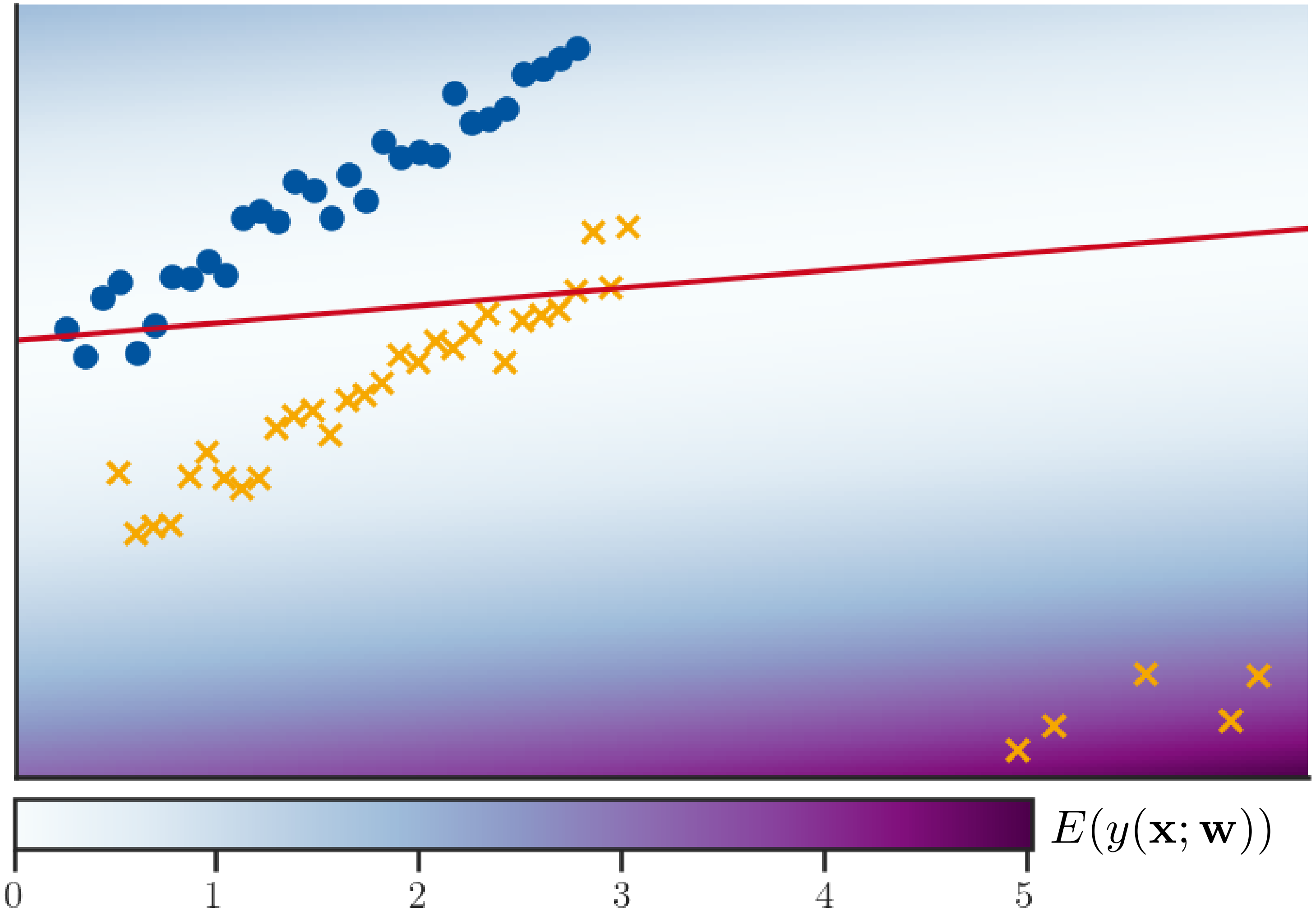
$$\tanh(x) = 2\sigma(x) - 1$$



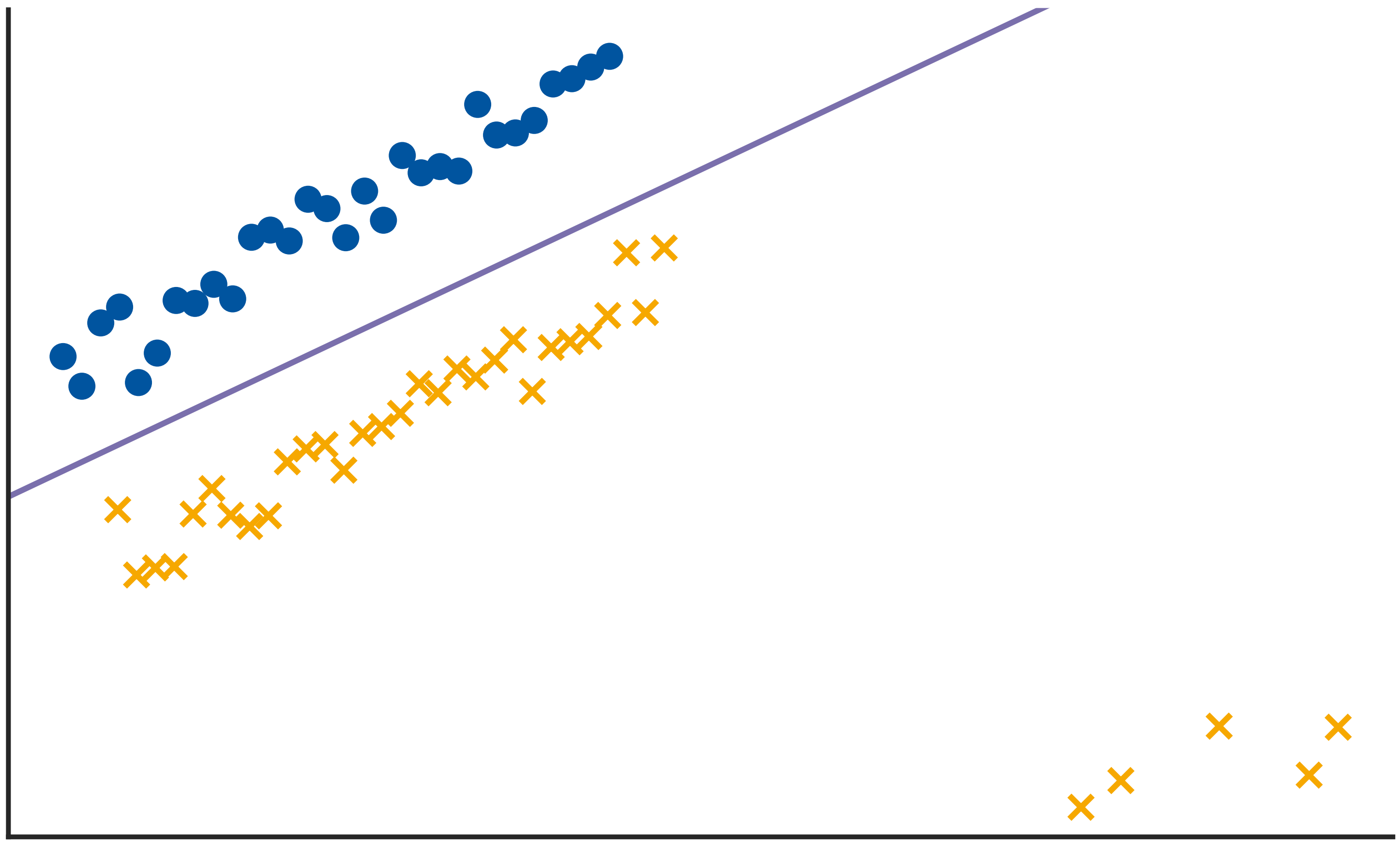
Example



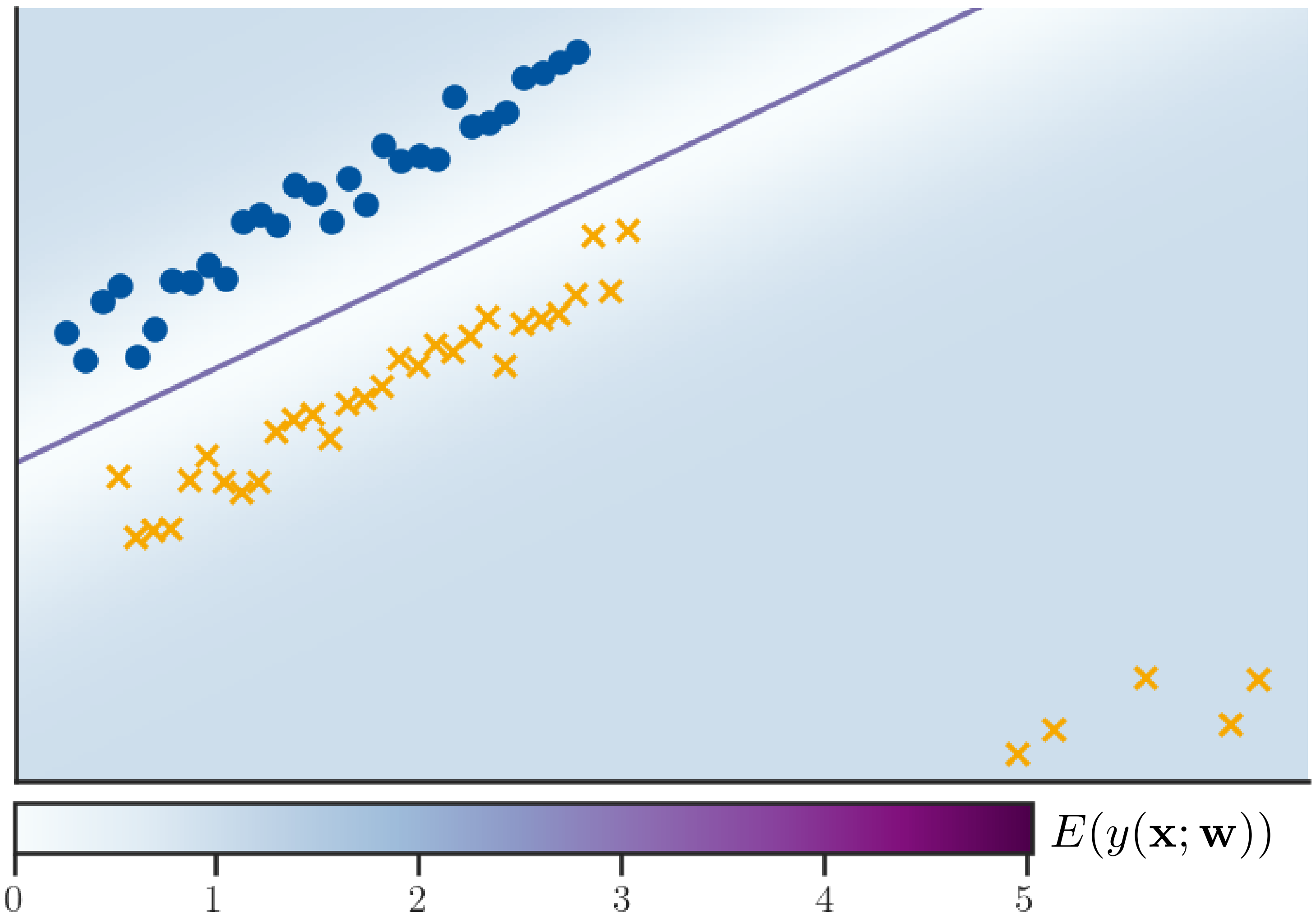
Example



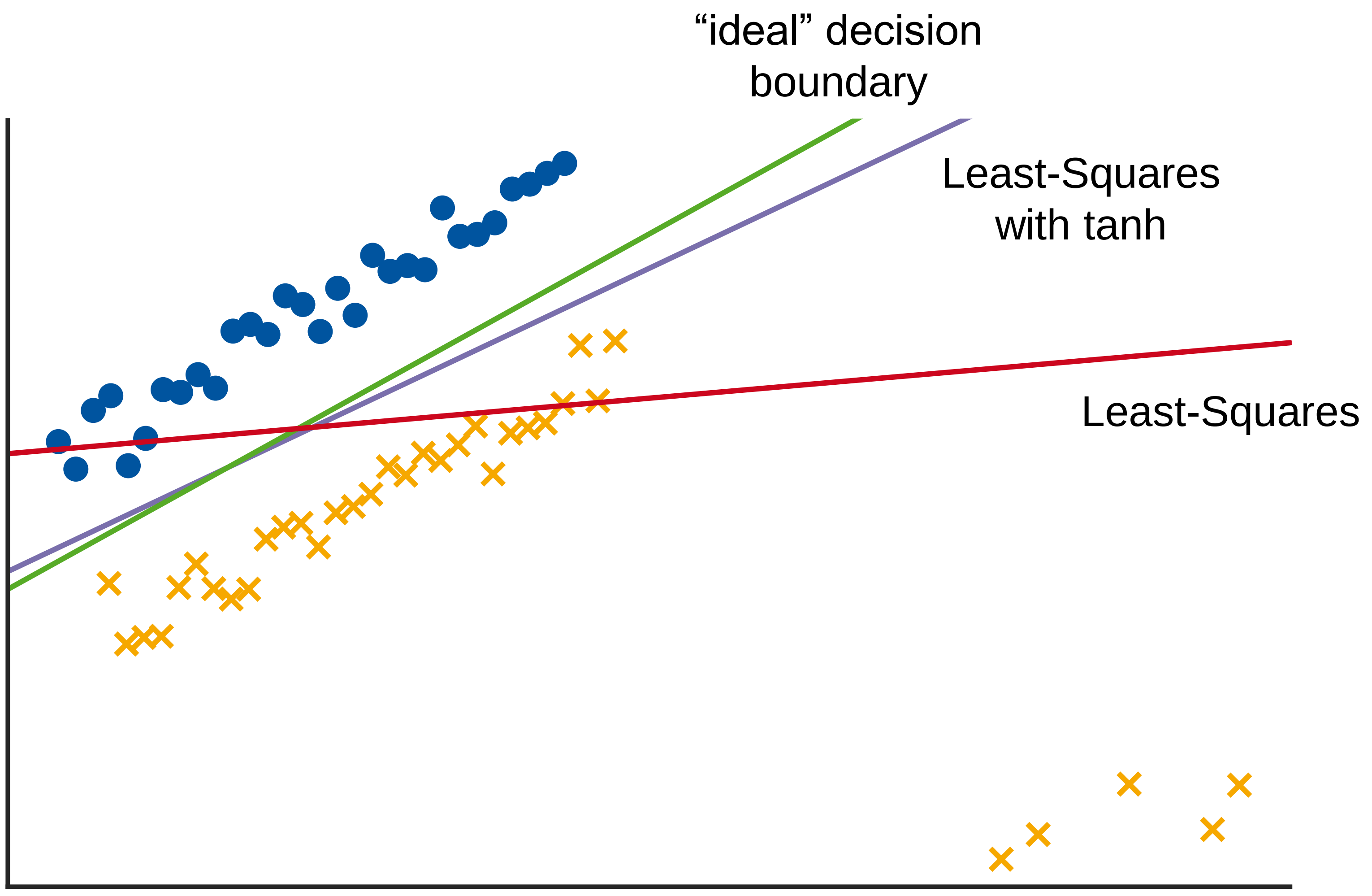
Example



Example



Example



“ideal” decision boundary
Least-Squares with tanh
Least-Squares

Discussion: Activation Functions

Advantages

- Nonlinearity gives more flexibility.
- Can be used to limit the effect of outliers.
- Choice of sigmoid actually has a nice probabilistic interpretation.

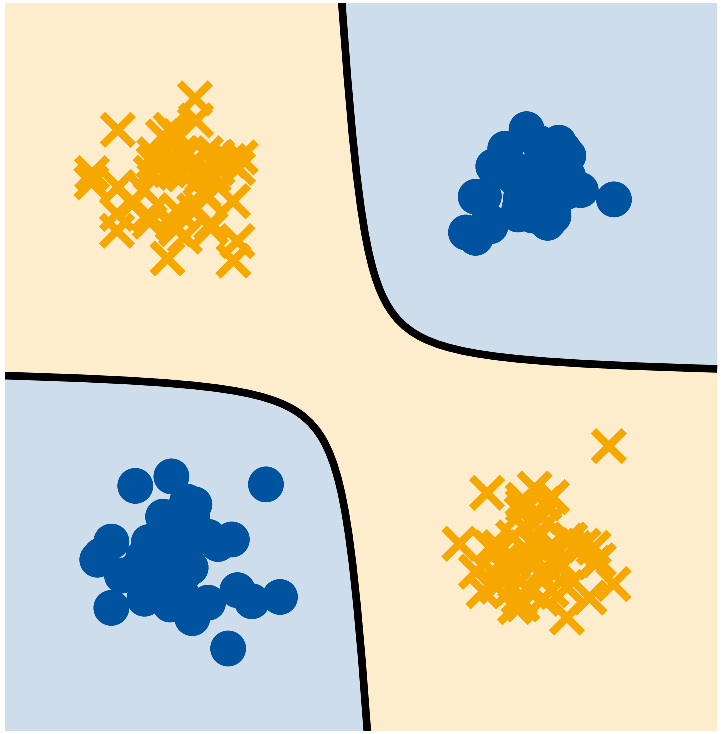
Limitations

- Least-squares minimization in general no longer leads to a closed-form analytical solution.

  $\Rightarrow$ Need to apply iterative methods.

Linear Discriminants

1. Motivation: Discriminant Functions
2. Linear Discriminant Functions
3. Least-Squares Classification
4. Generalized Linear Discriminants
5. **Basis Functions**

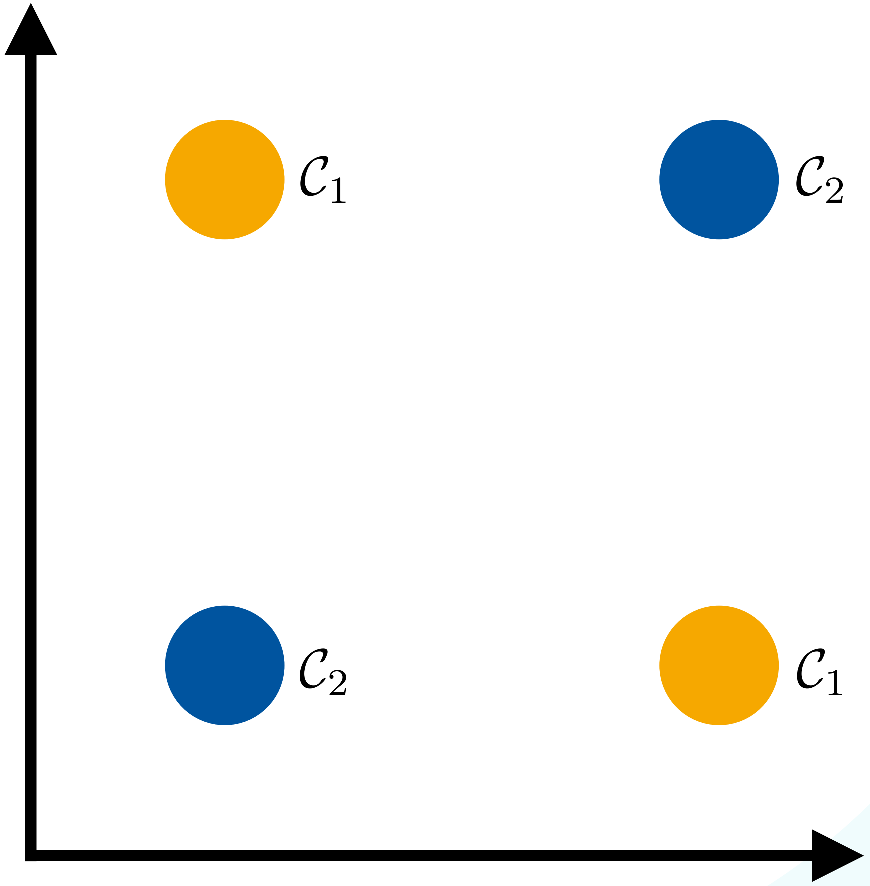


Basis Functions

- So far: assumed linear separability
  - Very restrictive assumption,
    classical counterexample: XOR
  - We need non-linear decision boundaries...
- Solution: use non-linear basis functions $\phi_j(\mathbf{x})$:

$$y(\mathbf{x}) = \sum_{j=1}^{M} w_j \phi_j(\mathbf{x}) + w_0$$

- By choosing the right ϕ, every continuous function can (in principle) be approximated with arbitrary accuracy



Intuition

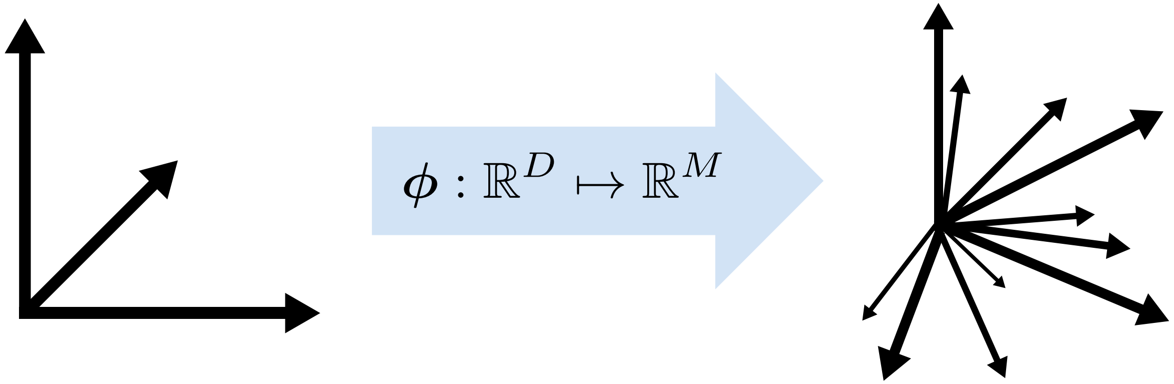
$$y_k(\mathbf{x}) = \sum_{j=0}^{M} w_{kj}\phi_j(\mathbf{x}) = \mathbf{w}_k^\mathsf{T}\boldsymbol{\phi}(\mathbf{x})$$

This is still a linear problem in $\boldsymbol{\phi}(\mathbf{x})$.

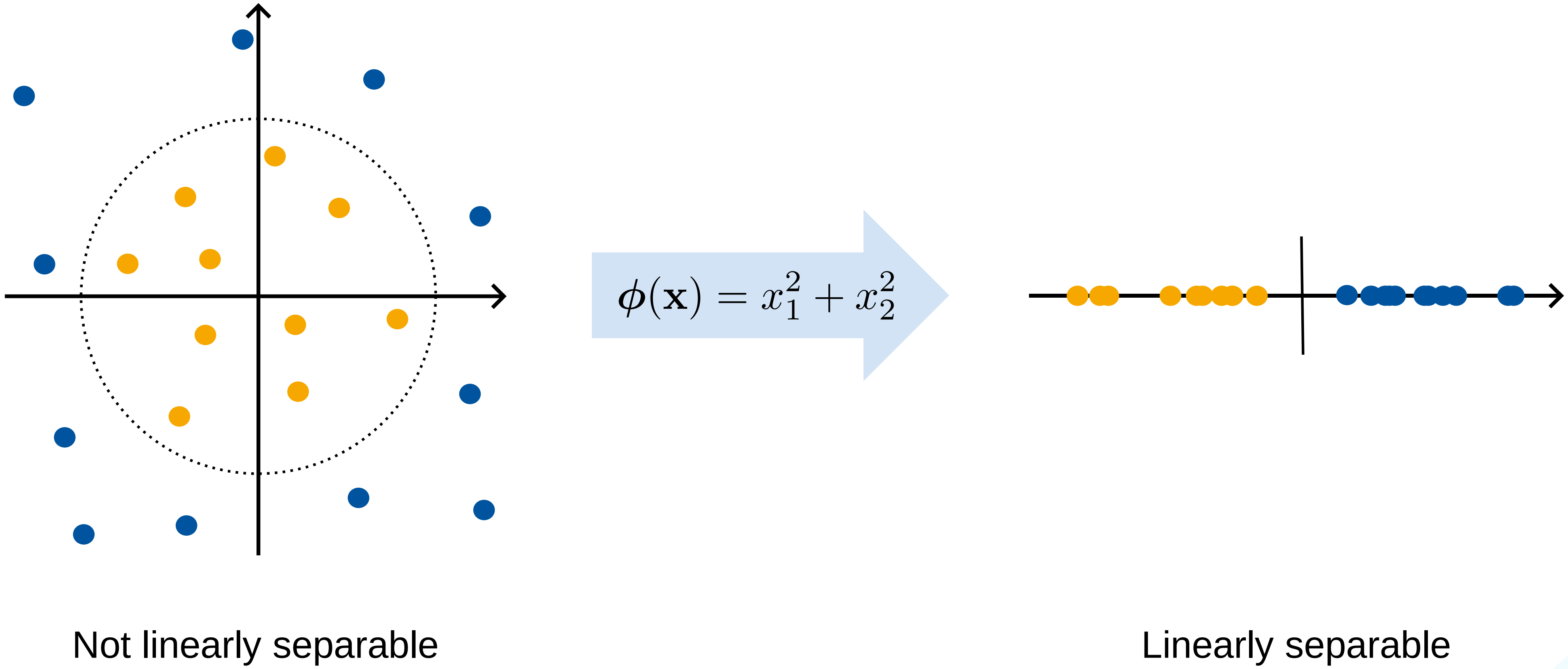
$\phi_j(\mathbf{x})$ are called basis functions.

But, depending on $\phi(\cdot)$, it may now be a nonlinear problem in $\mathbf{x}$.

Typically, $\phi_0(\mathbf{x}) = 1$ so that w_0 acts as a bias.



Usually, $\boldsymbol{\phi}$ maps into a higher-dimensional space.


$\phi(\mathbf{x}) = x_1^2 + x_2^2$
Not linearly separable
Linearly separable

Example: Polynomial Basis Functions

- Polynomial basis functions map x to powers of x:

$$\phi(x) = (x^m, x^{m-1}, \dots, x, 1)^\mathsf{T}$$

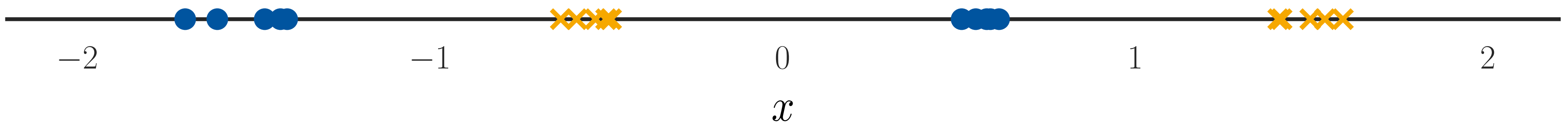
- When we optimize $\mathbf{w}^\mathsf{T}\phi(x)$ with polynomial basis functions, we implicitly optimize the coefficients of a polynomial in x:

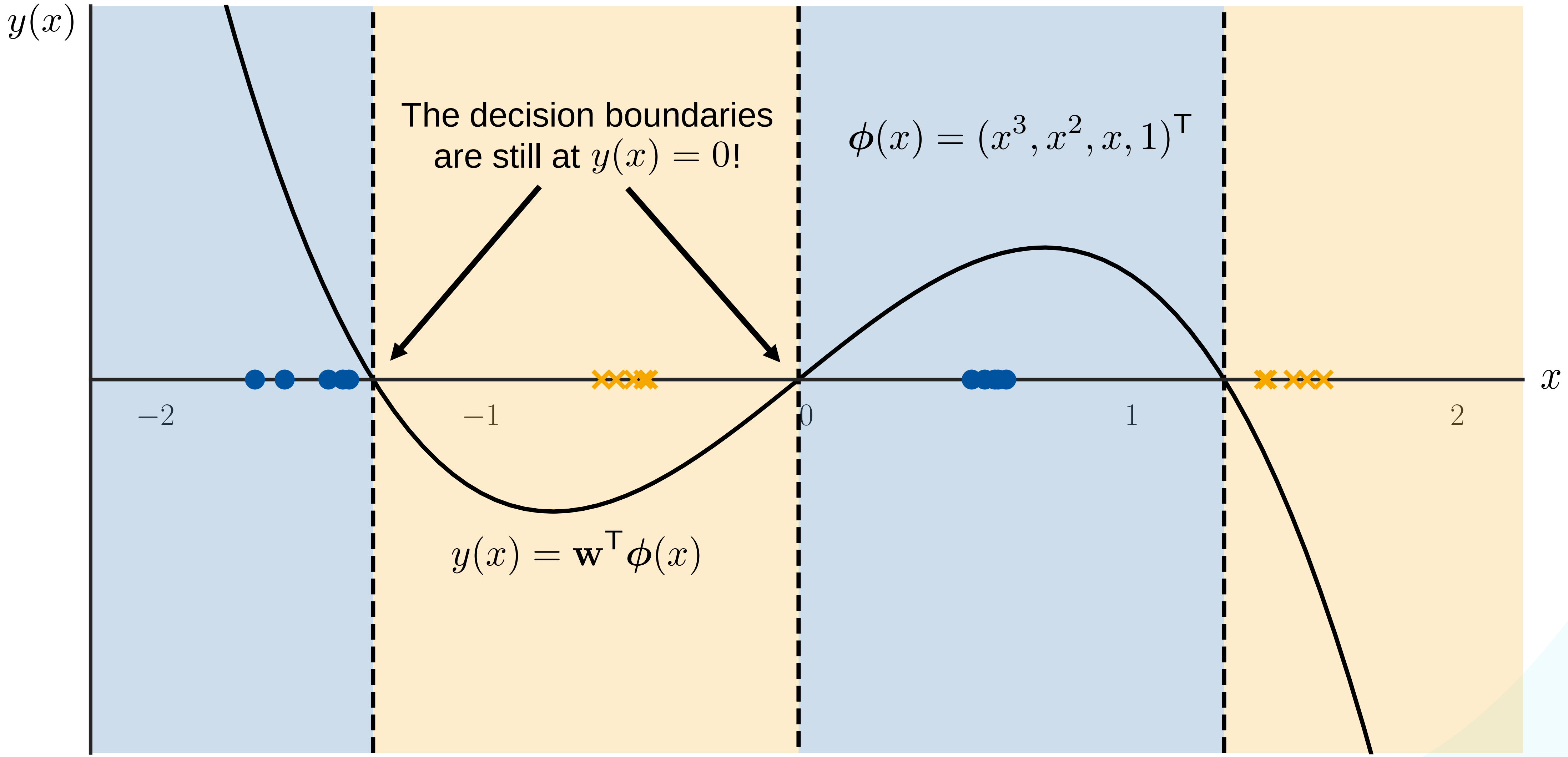
$$y(x) = \mathbf{w}^\mathsf{T}\phi(x)$$

$$= w_m x^m + w_{m-1}x^{m-1} + \dots + w_1 x + w_0$$

- As before, we decide for $\mathcal{C}_1$ if $y(x) > 0$.

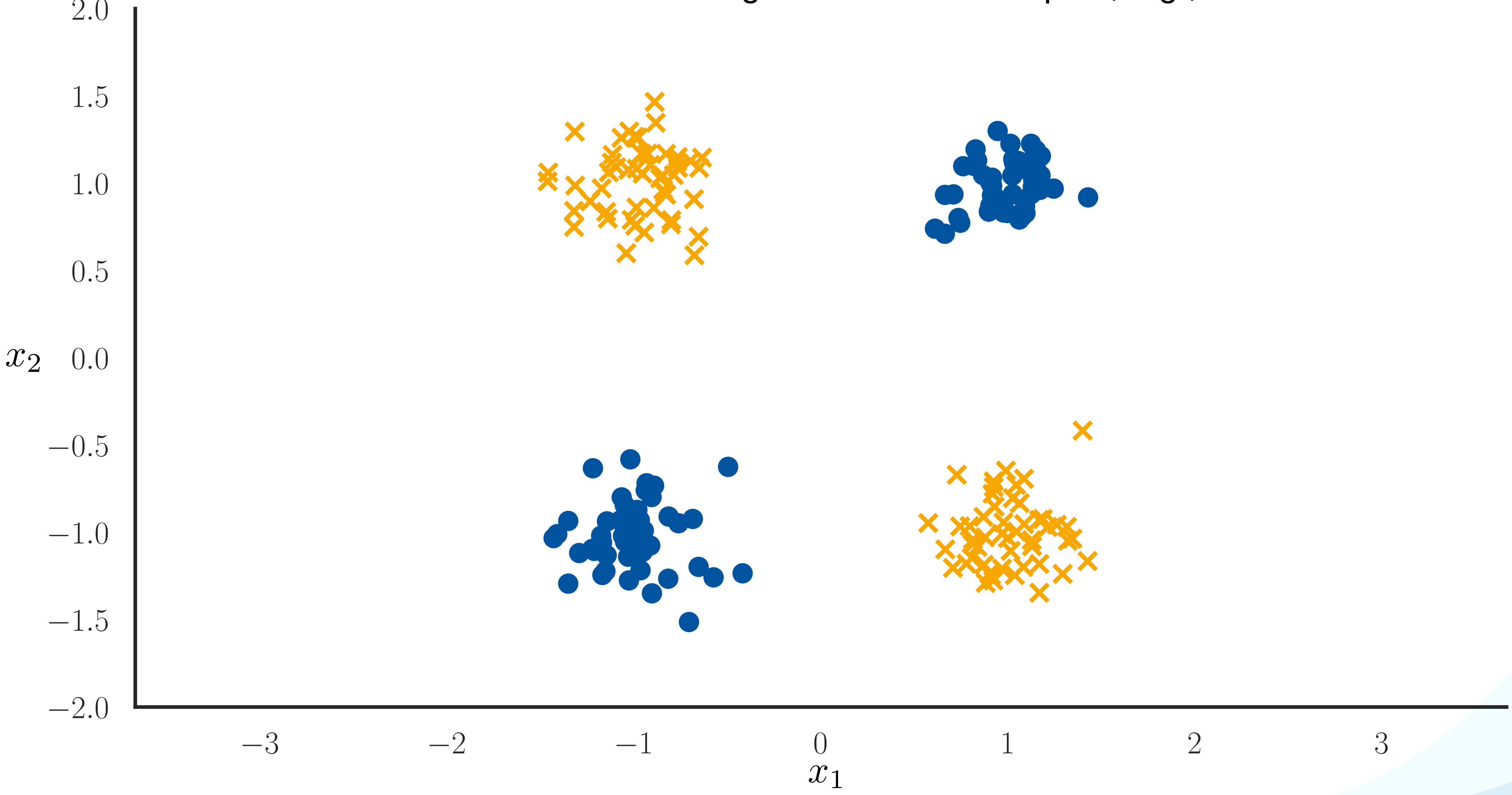
Let's use a third-degree polynomial: $\phi(x) = (x^3, x^2, x, 1)^\top$



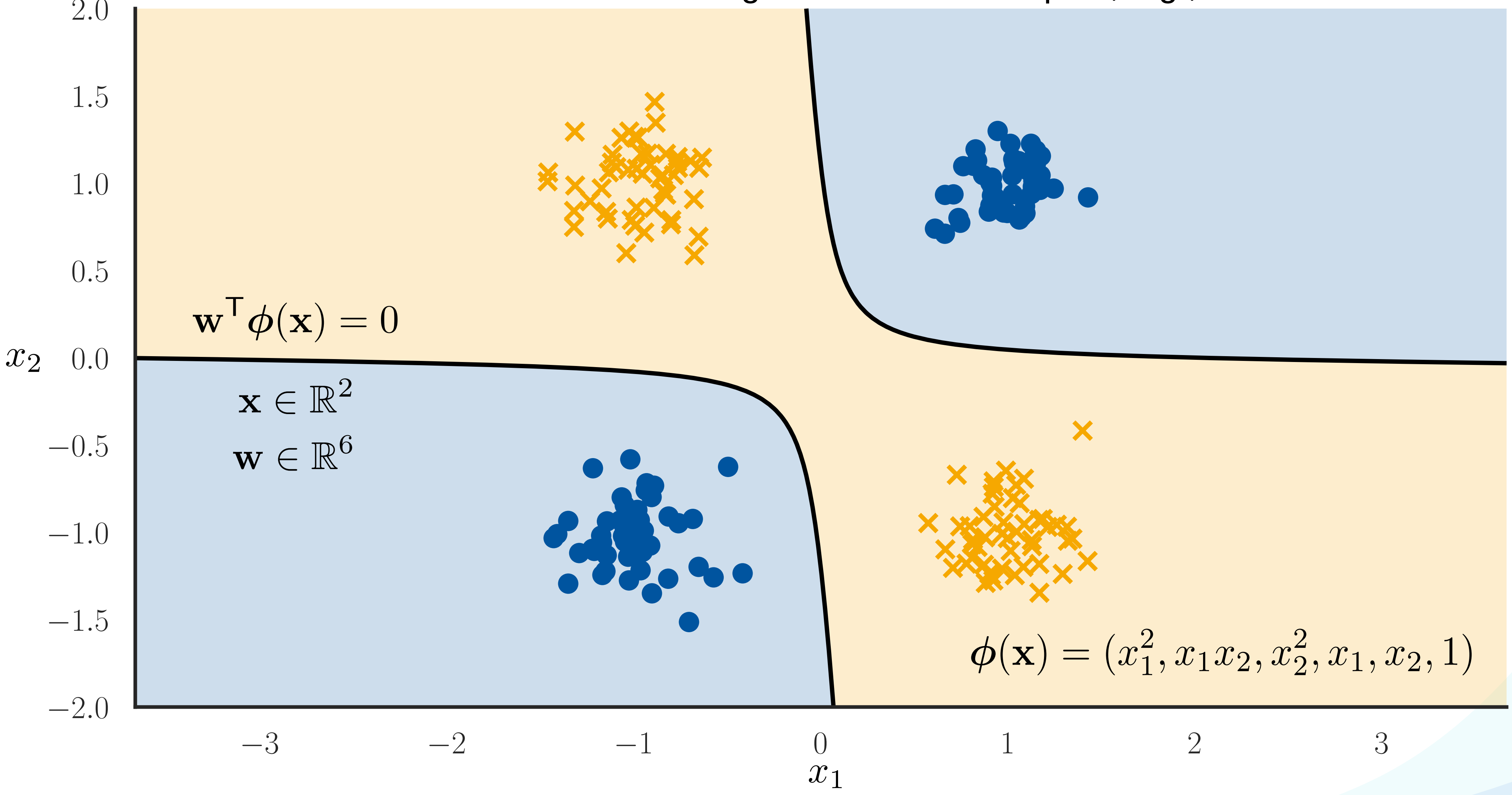


$y(x)$
The decision boundaries are still at $y(x) = 0$!
$\phi(x) = (x^3, x^2, x, 1)^\mathsf{T}$
-2
-1
0
1
2
x
$y(x) = \mathbf{w}^\mathsf{T}\phi(x)$

We can also extend this for higher dimensional inputs, e.g., 2D:



We can also extend this for higher dimensional inputs, e.g., 2D:



Discussion: Basis Functions

Advantages

- Basis functions allow us to address linearly non-separable problems
- The problem is still linear in $\phi(\mathbf{x})$ (but may be nonlinear in $\mathbf{x}$).
- We can think of $\phi(\mathbf{x})$ as transforming the data into a feature space in which the problem is easier to solve.
- In general, it is easier to find a separating hyperplane in higher-dimensional spaces.

Limitations

- The right choice of $\phi(\mathbf{x})$ depends on the problem and is another hyperparameter to optimize.
- Flexibility is limited by the curse of dimensionality. Evaluating $\mathbf{w}^\top\phi(\mathbf{x})$ can be expensive in high-dimensional spaces.
- Choosing a higher-dimensional feature space $\phi(\mathbf{x})$ increases the capacity of the classifier and may lead to overfitting.

References and Further Reading

- More information about Linear Discriminants is available in Chapter 4.1 of Bishop’s book.

Christopher M. Bishop
Pattern Recognition and Machine Learning
Springer, 2006