# M7. Classification: Linear Models (and Kernel Methods)

Manikandan Narayanan

Week 11 (Oct 6-, 2025)

PRML Jul-Nov 2025 (Grads Section)

# Acknowledgment of Sources

- Slides based on content from related
  - Courses:
    - IITM – Profs. Arun/Harish/Chandra's PRML offerings (slides, quizzes, notes, etc.), Prof. Ravi's "Intro to ML" slides – cited respectively as [AR], **[HR]**, [CC], [BR] in the bottom right of a slide.
    - India – NPTEL PR course by IISc Prof. PS. Sastry (slides, etc.) – cited as [PSS] in the bottom right of a slide.
  - Books:
    - PRML by Bishop. (content, figures, slides, etc.) – cited as **[CMB]**
    - Pattern Classification by Duda, Hart and Stork. (content, figures, etc.) – [DHS]
    - Mathematics for ML by Deisenroth, Faisal and Ong. (content, figures, etc.) – [DFO]
    - Information Theory, Inference and Learning Algorithms by David JC MacKay – [DJM]

# Outline for Module M7

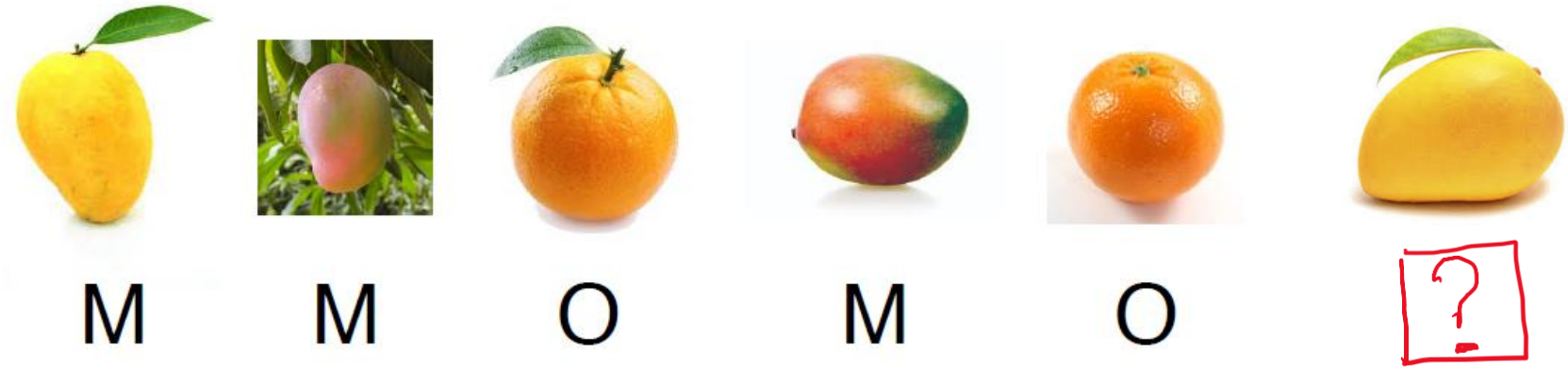
- M7. Classification (Linear models)
  - **M7.0 Introduction**
  - M7.1 Generative model ((Naïve) Bayes classifier) (brief as already seen)
  - M7.2 Discriminative model (Logistic regression)
  - M7.3 Discriminant function (Perceptron, *Linear/Fisher's Discriminant Analysis*) (very brief)
  - M7.4 Kernel methods
  - M7.5 Concluding thoughts

# Outline for Module M7 (detailed)

- M7. Classification (Linear models)
  - **M7.0 Introduction**
    - **Motivating examples / Birds-eye view**
  - M7.1 Generative model (brief as already seen)
    - Naïve Bayes classifier: Gaussian, identity covariance
    - Bayes classifier: Gaussian, general (shared) covariance
    - From generative model → Logistic fn., Linear decision surfaces
  - M7.2 Discriminative model
    - Logistic regression: MLE and Gradient-descent solution
    - Ridge Logistic regression: Bayesian MAP/regularization
  - M7.3 Discriminant function (very brief)
    - Geometry of decision surfaces
    - Perceptron objective function
    - Unifying view of objective functions
  - M7.4 Kernel methods
    - Representer theorem
    - Kernel logistic regression / etc. (also Kernel regression (as Assignment qn), Kernel PCA, leading to Kernel SVM, etc.)
  - M7.5 Concluding thoughts

# Recall: Classification – standard examples

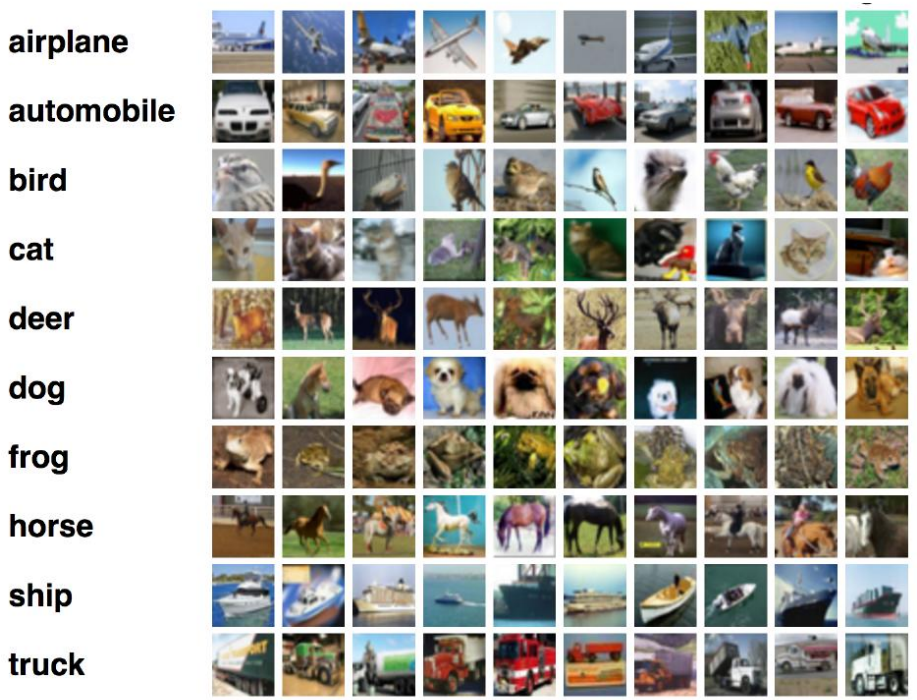
- Identify a given image as mango or orange, given many example images of mango and orange.




[HR]

# Recall: Classification – popular examples

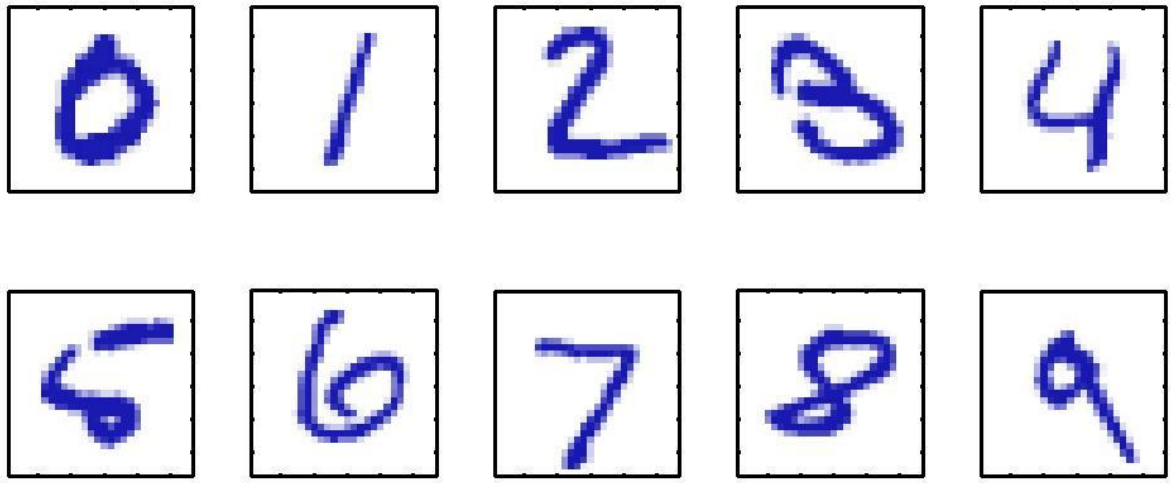
Classify an image into one of 10 classes,
given a “training set” of images with classes.




[HR]

# Recall: Example classification problem

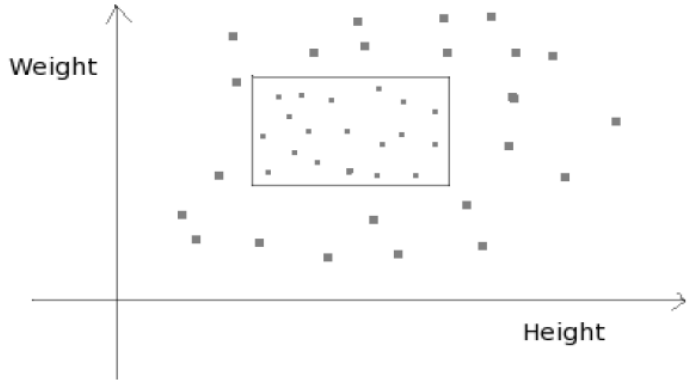
- Handwritten Digit Recognition:
  - Take a digit's 28 x 28 pixel image, represented by a vector of 784 real numbers, as input feature vector **x**, and output the identify of the digit 0,…,9.
  - Non-trivial problem due to the wide variability of handwriting.



[CMB]

# Recall: Example (non-linear) classification problem

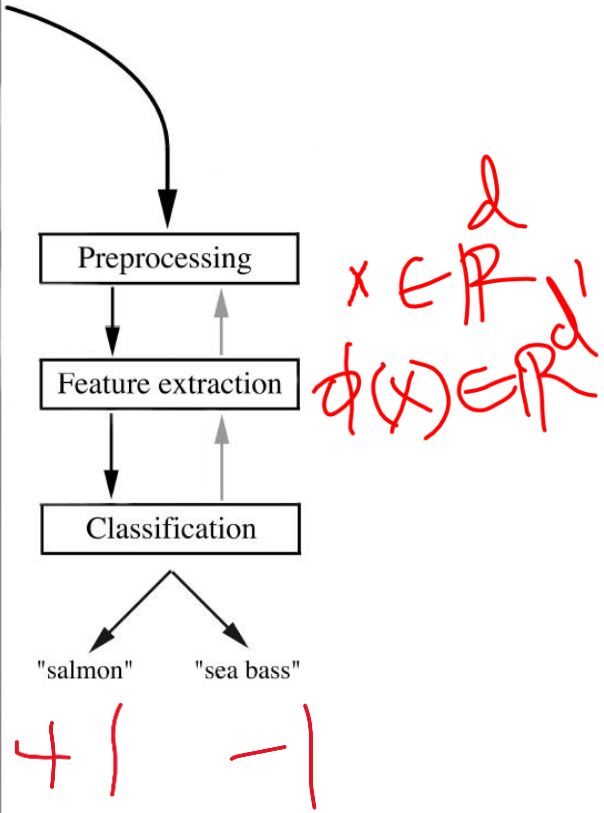
- Problem: recognize persons of ‘medium build’
- Features: Height and Weight




The classifier is ‘nonlinear’ here

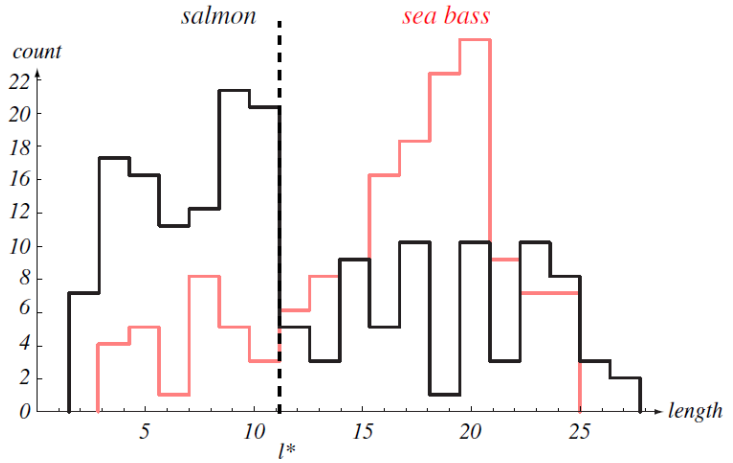
[PSS]

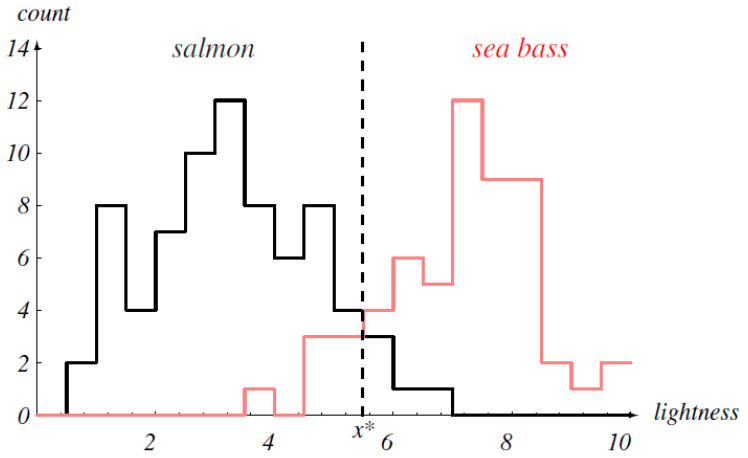
# Recall: Classification – standard examples




$x \in \mathbb{R}^d$

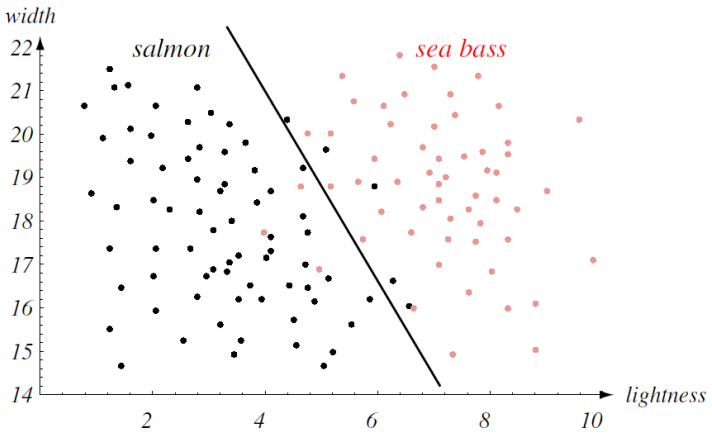
$\phi(x) \in \mathbb{R}^{d'}$

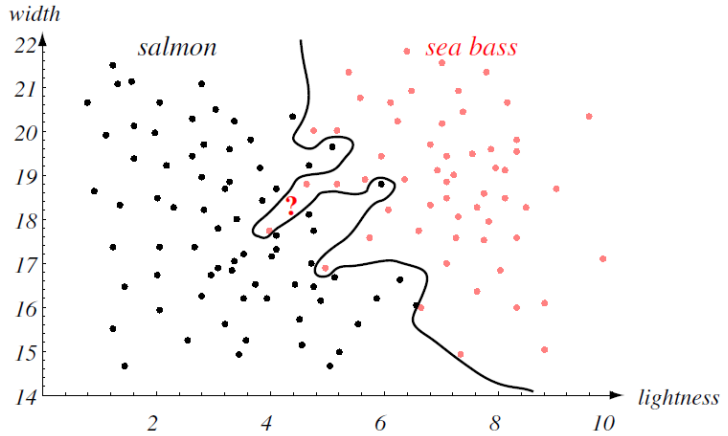


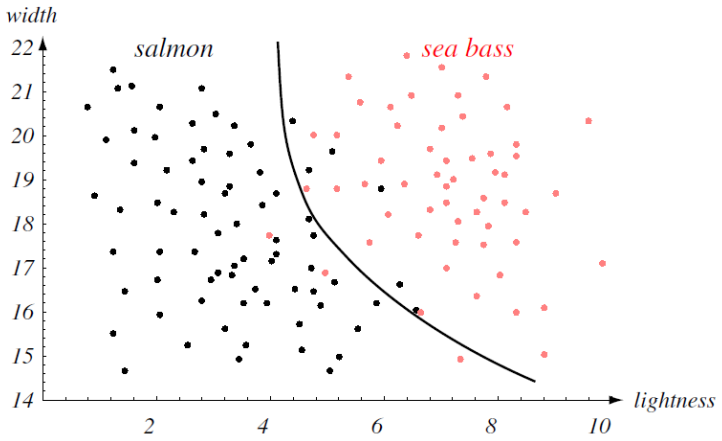




[DHS]

# Recall: Decision boundary – from simple to complex



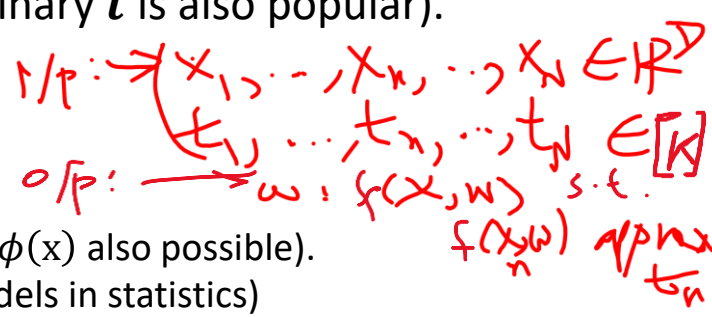




[DHS]

# Classification

- Learn a map from input variables (features **x**) to a discrete/categorical output variable (target $\boldsymbol{t} \in [\boldsymbol{K}] \coloneqq \{\mathbf{1}, \mathbf{2}, \dots, \boldsymbol{K}\} \coloneqq \{\boldsymbol{C_k}\}_{\boldsymbol{k}=\mathbf{1},\dots,\boldsymbol{K}}$; **1-of-K (one-hot)** encoding of $\boldsymbol{t}$ (or) **-1/+1** for binary $\boldsymbol{t}$ is also popular).
  **Given $\{\boldsymbol{x_n}, \boldsymbol{t_n}\}_{\boldsymbol{n}=\mathbf{1}\dots\boldsymbol{N}}$ pairs, we seek a function $\boldsymbol{f_w}(\boldsymbol{x})$ that approximates the map from x → t.**



  **Linear** if $\mathrm{f_w}(\mathrm{x}) \coloneqq f(\, z(\mathrm{x}, w)\,) = f\left(w^T \mathrm{x} + w_0\right)$ where
  - argument of f i.e., **z(x,w) is linear** in adjustable parameters $w$ and x (but non-linear basis fns. $\phi(\mathrm{x})$ also possible).
  - **f is a non-linear** *activation function* (its inverse is called *link function* in generalized linear models in statistics)
  - decision surfaces correspond to $f_w(\mathrm{x}) = const.$, so that $w^T \mathrm{x} + w_0 = const.$ (so **decision surfaces are <u>linear in $\boldsymbol{x}$</u>**, even if $f(.)$ is non-linear, as long as its one-to-one)

- A basic supervised learning problem/algorithm like linear regression:
  - Practical limitations for complex data, but basic linear classification methods (**Naïve Bayes classifier and Logistic regression**) is a good starting point for other sophisticated learning algorithms. Linear models being simpler are highly efficient to learn, highly interpretable, etc.
  - Optimization problem doesn't result in a closed-form solution as in linear regression; but objective function is convex and gradient-descent methods work well.

- Our approach in this lecture: Mostly [Harish's logistic regression notes; CMB, Chapter 4].

[CMB]

# Example f(.) (activation functions)

- Binary (0-1) classifn.: Let $z \coloneqq w^T \mathrm{x} + w_0$. Then,

$$f(z) = \mathrm{logistic}(z) \coloneqq \sigma(z) \coloneqq \frac{1}{1+\exp(-z)}.$$

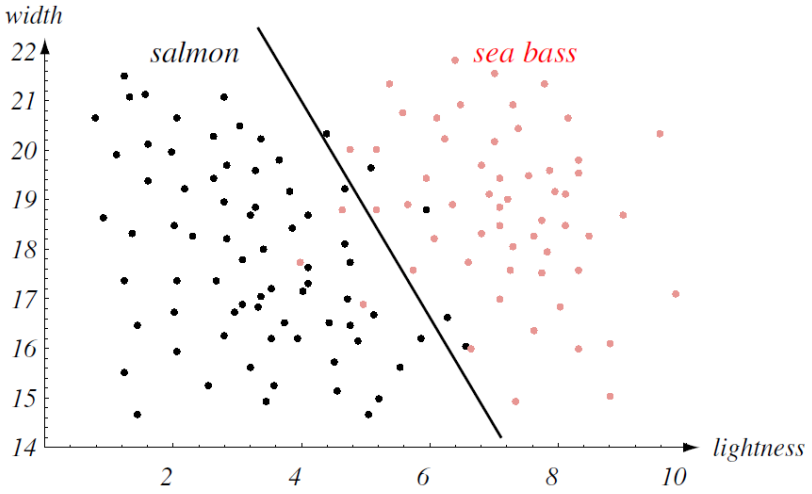
- Mulit-class (1-of-K one-hot) classifn.: Let $z_k \coloneqq w_k^T \mathrm{x} + w_{k0}$. Then,

$$f(z_1, z_2, \ldots, z_K) = \mathrm{soft(arg)max}(z_1, z_2, \ldots, z_K) \coloneqq \left\{\frac{\exp(z_k)}{\sum_i \exp(z_i)}\right\}_{k=1\ to\ K}.$$

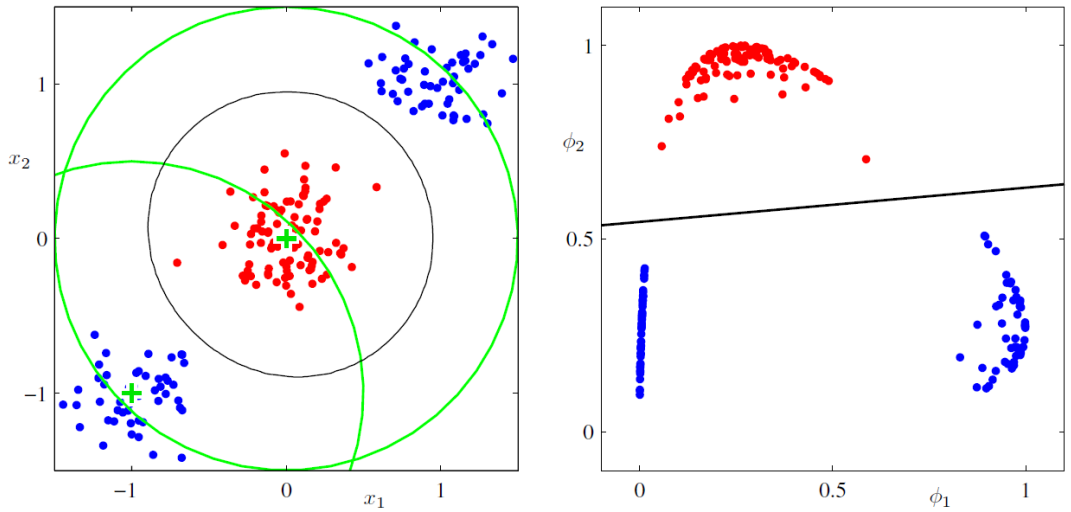
# A note on DB (Decision Boundary): linear in $x$ or $\phi(x)$?

If $f$ one-to-one, then $f\left(w^T \mathrm{x} + w_0\right)$ = cutoff (say 0.5) $\Rightarrow w^T \mathrm{x} + w_0$ = const. (x could've been $\phi(x)$ too).

Linear in $x$




Linear in $\phi(x)$, but not $x$

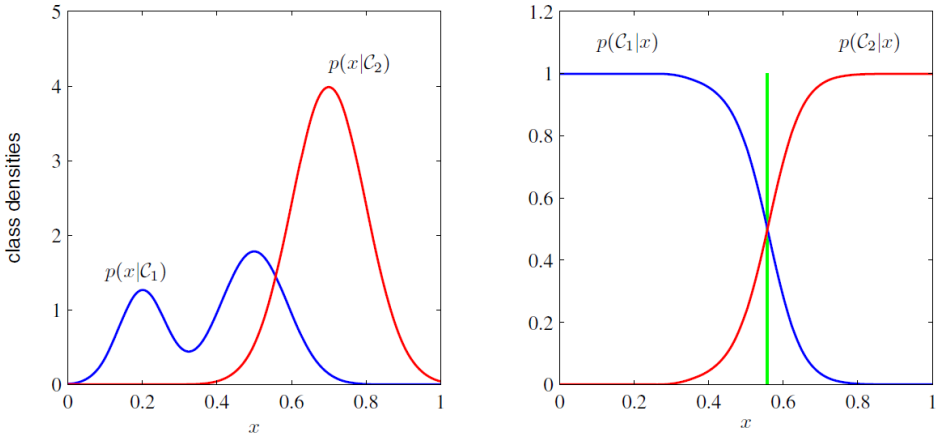



DB also linear in $w$, but of less use here than in linear regression (where obj. fn linear in $w$ led to closed-form solution).
In linear models of classification, obj. fn. (empirical expected loss or empirical risk) is still convex in $w$ to enable nice properties for optmzn. via gradient descent or similar methods (but closed-form solution for $w$ not possible).

# Recall: Inference and decision: three approaches for classification

- Generative model approach:
  - (I) Model $p(x, C_k) = p(x|C_k)p(C_k)$
  - (I) Use Bayes' theorem $p(C_k|\mathbf{x}) = \frac{p(\mathbf{x}|C_k)p(C_k)}{p(\mathbf{x})}$
  - (D) Apply optimal decision criteria

- Discriminative model approach:
  - (I) Model $p(C_k|\mathbf{x})$ directly
  - (D) Apply optimal decision criteria




- Discriminant function approach:
  - (D) Learn a function that maps each x to a class label directly from training data
  - Note: No posterior probabilities!

[CMB]

# Outline for Module M7

- M7. Classification (Linear models)
  - M7.0 Introduction
  - **M7.1 Generative model**
    - **Naïve Bayes classifier: Gaussian, identity covariance**
    - **Bayes classifier: Gaussian, general (shared) covariance**
    - **From generative model → Logistic fn., Linear decision surfaces**
  - M7.2 Discriminative model (Logistic regression)
  - M7.3 Discriminant function (Perceptron) (very brief)
  - M7.4 Kernel methods
  - M7.5 Concluding thoughts

# K=2 Bayes classifier – Gaussian, identity covariance ($y := t$)

Recall: Generative Models for Classification

$X, Y \in \mathbb{R}^d \times \{\pm 1\}$

$X \mid Y = +1 \sim N(\mu_+, I)$ ; $P(Y = +1) = a$

$X \mid Y = -1 \sim N(\mu_-, I)$ ; $P(Y = -1) = 1 - a$

$$P(Y = 1 \mid X = x) = \frac{f_{X|Y}(x \mid +1)\, P(Y = +1)}{f_{X|Y}(x \mid +1)\, P(Y = +1) + f_{X|Y}(x \mid -1)\, P(Y = -1)}$$

$$= \frac{a \cdot \exp\left(-\frac{1}{2}\|x - \mu_+\|^2\right)}{a \exp\left(-\frac{1}{2}\|x - \mu_+\|^2\right) + (1-a)\exp\left(-\frac{1}{2}\|x - \mu_-\|^2\right)}$$

[HR]

$$= \frac{1}{1 + \exp\left(x^T \mu_- - x^T \mu_+ - \frac{1}{2}\|\mu_-\|^2 + \frac{1}{2}\|\mu_+\|^2 + \ln\left(\frac{1-a}{a}\right)\right)}$$

$$= \frac{1}{1 + \exp\left(-(w^T x + b)\right)}$$

where $w = \mu_+ - \mu_-$

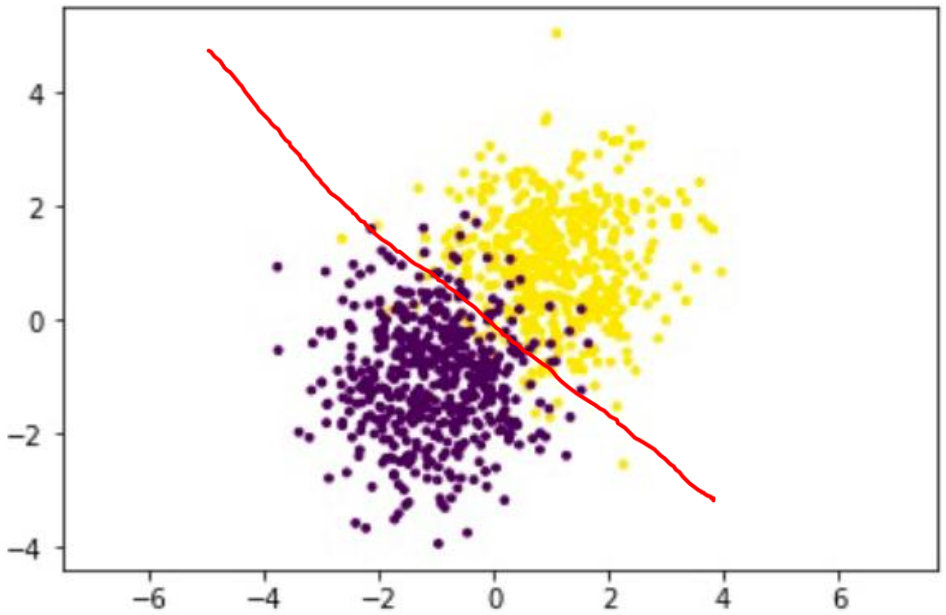
$$b = -\frac{1}{2}\|\mu_+\|^2 + \frac{1}{2}\|\mu_-\|^2 + \ln\frac{a}{1-a}$$

$$P(y=1|x) = \sigma(w^T x + b)$$

where $\sigma(u) = \frac{1}{1+\exp(-u)}$ $\Rightarrow$


[HR]

# Example data 1




$X \mid Y = -1 \sim N\left(\begin{bmatrix} -1 \\ -1 \end{bmatrix}, I\right)$

$X \mid Y = +1 \sim N\left(\begin{bmatrix} 1 \\ 1 \end{bmatrix}, I\right)$

$P(Y = 1) = 1/2$

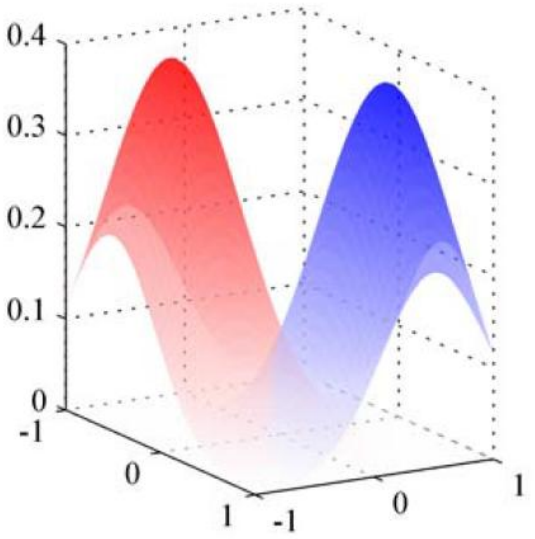
$P(Y = 1 \mid X = x) = \sigma([2,2]x)$

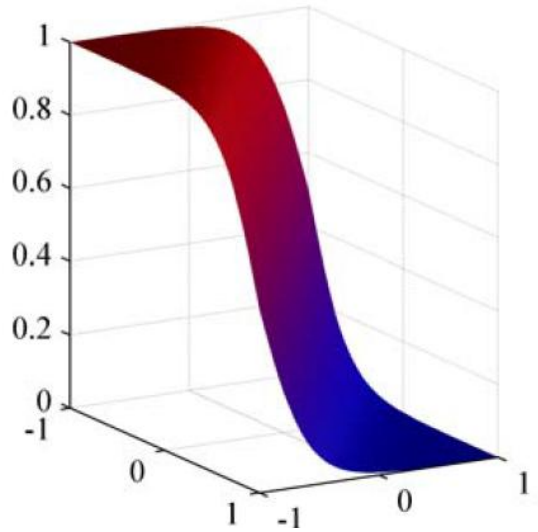
[HR]

# K=2 Bayes classifier – Gaussian, general (shared) covariance

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

$$p(\mathcal{C}_1|\mathbf{x}) = \sigma(\mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0)$$




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

[CMB]

# K>2 Bayes classifier – Gaussian, general (shared) covariance

- Normalized exponential or softmax:

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

- Gaussian:

  (same as before)

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

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

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

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

[CMB]

# Sanity check

That’s all there is to generative models of classification…

…or am I missing something?

# How do we learn the generative model? Density estimation (MLE) of Gaussian (K=2 shown; K>2 is similar)

$$\hat{\pi}_{ML} = \frac{1}{N}\sum_{n=1}^{N} \mathbf{1}_{\{\mathbf{y}_n = C_1\}} = \frac{N_1}{N} = \frac{N_1}{N_1 + N_2}$$

$$\hat{\mu}_{1\,ML} = \frac{1}{N_1}\sum_{n=1}^{N} \mathbf{1}_{\{\mathbf{y}_n = C_1\}}\mathbf{x}_n \qquad \hat{\mu}_{2\,ML} = \frac{1}{N_2}\sum_{n=1}^{N} \mathbf{1}_{\{\mathbf{y}_n = C_2\}}\mathbf{x}_n$$

$$\begin{aligned}
\hat{\Sigma}_{ML} = \quad \mathbf{S} &= \frac{N_1}{N}\mathbf{S}_1 + \frac{N_2}{N}\mathbf{S}_2 \\
\mathbf{S}_1 &= \frac{1}{N_1}\sum_{n \in C_1} (\mathbf{x}_n - \hat{\mu}_{1\,ML})(\mathbf{x}_n - \hat{\mu}_{1\,ML})^{\mathrm{T}} \\
\mathbf{S}_2 &= \frac{1}{N_2}\sum_{n \in C_2} (\mathbf{x}_n - \hat{\mu}_{2\,ML})(\mathbf{x}_n - \hat{\mu}_{2\,ML})^{\mathrm{T}}.
\end{aligned}$$

$$L(\theta) = \prod_{n=1}^{N} P(x_n, y_n)$$

$$\theta = (\pi, \mu_1, \mu_2, \Sigma)$$

$$= \prod_n P(y_n|\pi) P(x_n|y_n, \mu_1, \mu_2, \Sigma)$$

$$LL(\theta) = \sum_n \log P(y_n|\pi) + \sum_{n: y_n = C_1} \log P(x_n|y_n, \mu_1, \Sigma) + \sum_{n: y_n = C_2} \log P(x_n|y_n, \mu_2, \Sigma)$$

[CMB]

# What if covariance matrix is not shared across classes?

- Then, $LL(\theta)$ in last slide becomes: $LL(\theta) = \sum_n \log P(y_n|\pi) + \sum_{n:y_n=C_1} \log P(x_n|y_n,\mu_1,\Sigma_1) + \sum_{n:y_n=C_2} \log P(x_n|y_n,\mu_2,\Sigma_2)$

- and each of these three summation terms can be optimized separately!
- Therefore, all params' MLE remain the same, except for these params.:

$\hat{\Sigma}_{1(ML)} = S_1$ & $\hat{\Sigma}_{2(ML)} = S_2$.

# **Recall:** Inference & Decision (steps in detail for a generative model when covariance matrix is **not** shared):

Setup (training data):

Learning the Bayes classifier from data:

- $S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ with $x_i \in \mathcal{X}$, $y_i \in \{1\}$

---

Inference (density est.):

- Assume $P_{X|Y}(x|1)$ and $P_{X|Y}(x|-1)$ are from P.
- Estimate $P_{X|Y}(x|1)$ from positively labelled samples and $P_{X|Y}(x|-1)$ from negatively labelled samples using Maximum Likelihood.
- Estimate $P(Y=1)$ & $P(Y=-1)$
- Use Bayes Rule to make an estimate of the posterior probability $P(Y=1|X=x) = \eta(x)$

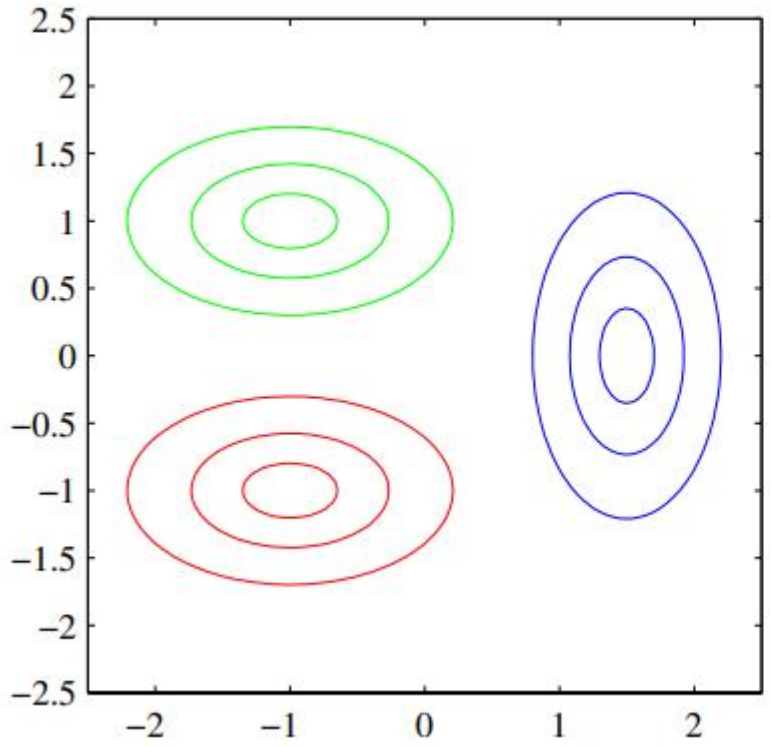
---

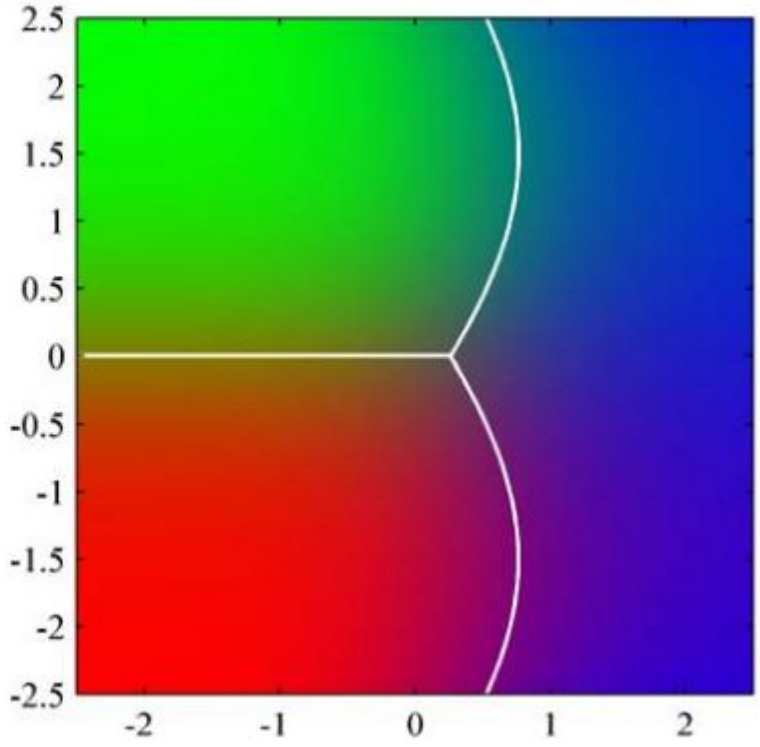
Decision:

- Compute $h^*(x) = \text{sign}(2\eta(x)-1)$

[HR]

# Shared vs. non-shared covariance matrix: linear vs. quadratic discriminant (DB – quadratic terms do not cancel out in non-shared case!)



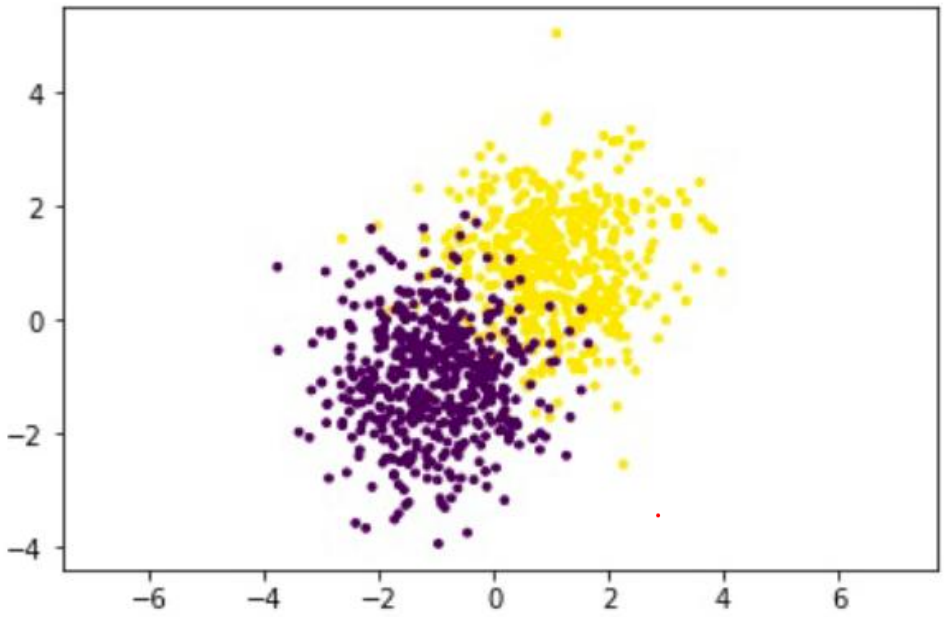

[CMB]

# Sanity check

Now, we're really done with generative models of classification…

…and let's motivate the discriminative model approach.

## Example data 1



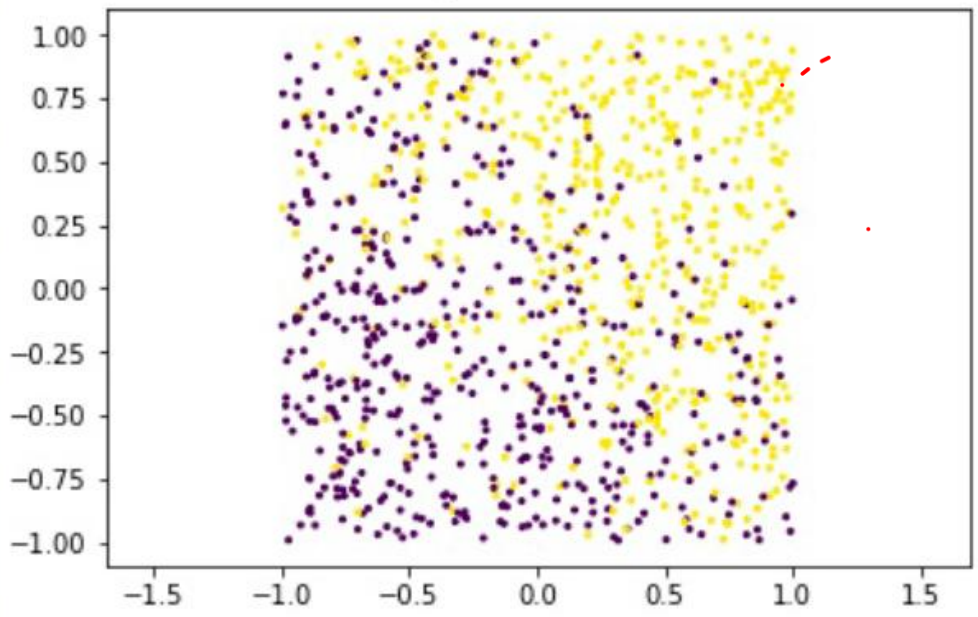

$X \mid Y = -1 \sim N\left(\begin{bmatrix} -1 \\ -1 \end{bmatrix}, I\right)$

$X \mid Y = +1 \sim N\left(\begin{bmatrix} 1 \\ 1 \end{bmatrix}, I\right)$

$P(Y = 1) = 1/2$

$P(Y = 1 \mid X = x) = \sigma([2,2]x)$

## Example data 2




$P(Y = 1 \mid X = x) = \sigma([2,2]x)$

Compared to generative models:
Discriminative (logistic regn.): less restrictive (with enough data)
Discriminant (linear decision surface): even less restrictive (but no probabs.)

[HR]

# Outline for Module M7

- M7. Classification (Linear models)
  - M7.0 Introduction
  - M7.1 Generative model ((Naïve) Bayes classifier)
  - **M7.2 Discriminative model**
    - **Logistic regression: MLE and Gradient-descent solution**
    - **Ridge Logistic regression: Bayesian MAP/regularization**
  - M7.3 Discriminant function (Perceptron) (very brief)
  - M7.4 Kernel methods
  - M7.5 Concluding thoughts

# Classification :

## Discriminative models :

: $X, Y \in \mathbb{R}^d \times \{\pm 1\} \sim D$

$X \sim D_x$

$P(Y=1 \mid X=x) = \sigma(w^T x)$ ;

$P(Y=-1 \mid X=x) = 1 - \sigma(w^T x) = \sigma(-w^T x)$

$\Big\} P(Y=y \mid X=x) = \sigma(y w^T x)$

where $w$ is some unknown constant.

1. Given $(x_1, y_1), \ldots, (x_m, y_m)$ i.i.d samples drawn from $D$. Can $w$ be discovered ?
2. If $w$ is known, how to Predict for a given $x$?

[HR]

$$P(y_1, y_2, \cdots, y_m \mid x_1 \cdots x_m, w) = \mathcal{L}(w)$$

$$= \prod_{i=1}^{m} P(Y_i = y_i \mid X_i = x_i, w)$$

$$= \prod_{i=1}^{m} \left(\sigma(w^T x_i)\right)^{I(y_i=1)} \cdot \left(\sigma(-w^T x_i)\right)^{I(y_i=-1)} = \boxed{\prod_{i=1}^{m} \sigma(y_i w^T x_i)}$$

$$\mathcal{LL}(w) = \sum_{i=1}^{m} \log \sigma(y_i w^T x_i)$$

$$= \sum_{i=1}^{m} \log \frac{1}{1 + e^{-y_i w^T x_i}}$$

[HR]

$$-\ell\ell(w) = \sum_{i=1}^{m} \log\left(1+\exp(-y_i w^T x_i)\right) = \hat{R}(w)$$

$\hat{R}(w)$ : Empirical logistic loss

$$\nabla \hat{R}(w) = \sum_{i=1}^{m} \frac{1}{1+\exp(-y_i w^T x_i)} \left(\exp(-y_i w^T x_i)\right)(-y_i x_i)$$

$$= \sum_{i=1}^{m} \sigma(-y_i w^T x_i)(-y_i x_i)$$

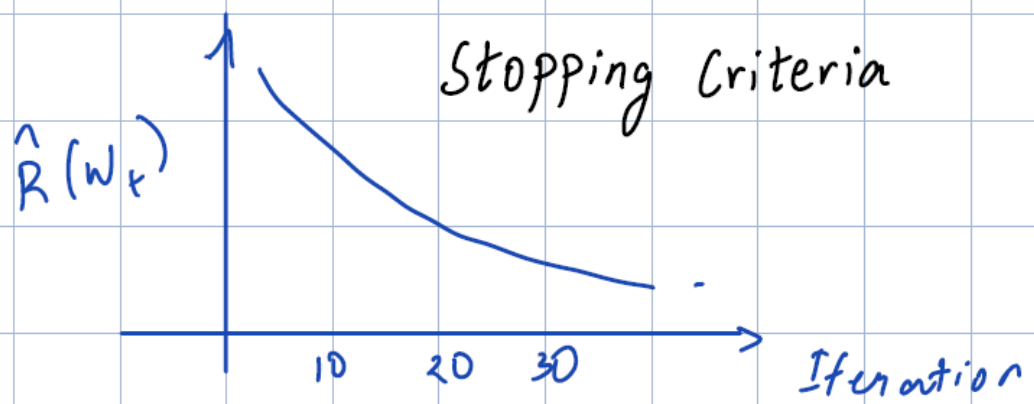
[HR]

Gradient Descent:

$$w_{t+1} = w_t - \eta \nabla \hat{R}(w_t)$$

Apply Gradient descent to Empirical Logistic Loss:

$$w \mapsto w + \eta \sum_{i=1}^{m} \sigma(-y_i w^T x_i) y_i x_i$$




NOTE: Iterated reweighted least squares method (Newton-Raphson instead of gradient-descent) also possible and common with logistic regn.

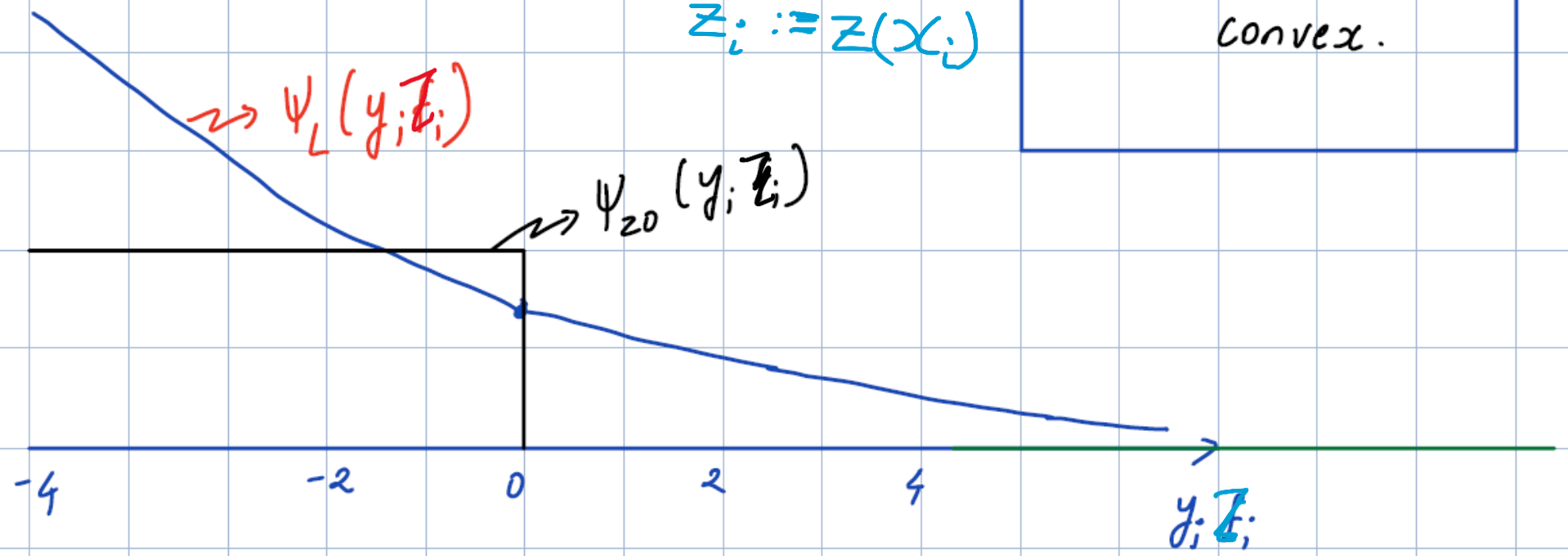
[HR]

# Empirical Logistic loss Minimisation.

$$\hat{R}(w) = \sum_{i=1}^{n} \log\left(1 + \exp\left(-y_i w^T x_i\right)\right)$$

$$= \sum_{i=1}^{n} \psi_L\left(y_i \underbrace{w^T x_i}_{z_i := z(x_i)}\right)$$

P.T $\psi_L : \mathbb{R} \to \mathbb{R}$ is convex.




[HR]

# Regularised/MAP Logistic Regression

$w \sim N(0, \tau^2 I)$ $\quad x \sim D_x$

$P(Y = +1 \mid x) = \sigma(w^T x)$

$$\mathcal{L}^P(w) = P(w \mid x_1, \cdots x_n, y_1, \cdots y_n)$$

$$= \frac{P(y_1 \cdots y_n \mid x_1, \cdots x_n, w)\, P(w \mid x_1 \cdots x_n)}{P(y_1 \cdots y_n \mid x_1 \cdots x_n)}$$

Steps here

$$-\mathcal{L}\mathcal{L}^P(w) = \sum_{i=1}^{n} \log(1 + \exp(-y_i w^T x_i)) + \frac{\lambda}{2}\|w\|^2$$

(Subject to additive & multiplicative constants)

[HR]

# Outline for Module M7

- M7. Classification (Linear models)
  - M7.0 Introduction
  - M7.1 Generative model ((Naïve) Bayes classifier)
  - M7.2 Discriminative model (Logistic regression)
  - **M7.3 Discriminant function (very brief)**
    - **Geometry of decision surfaces**
    - Perceptron objective function
    - Unifying view of objective functions
  - M7.4 Kernel methods
  - M7.5 Concluding thoughts

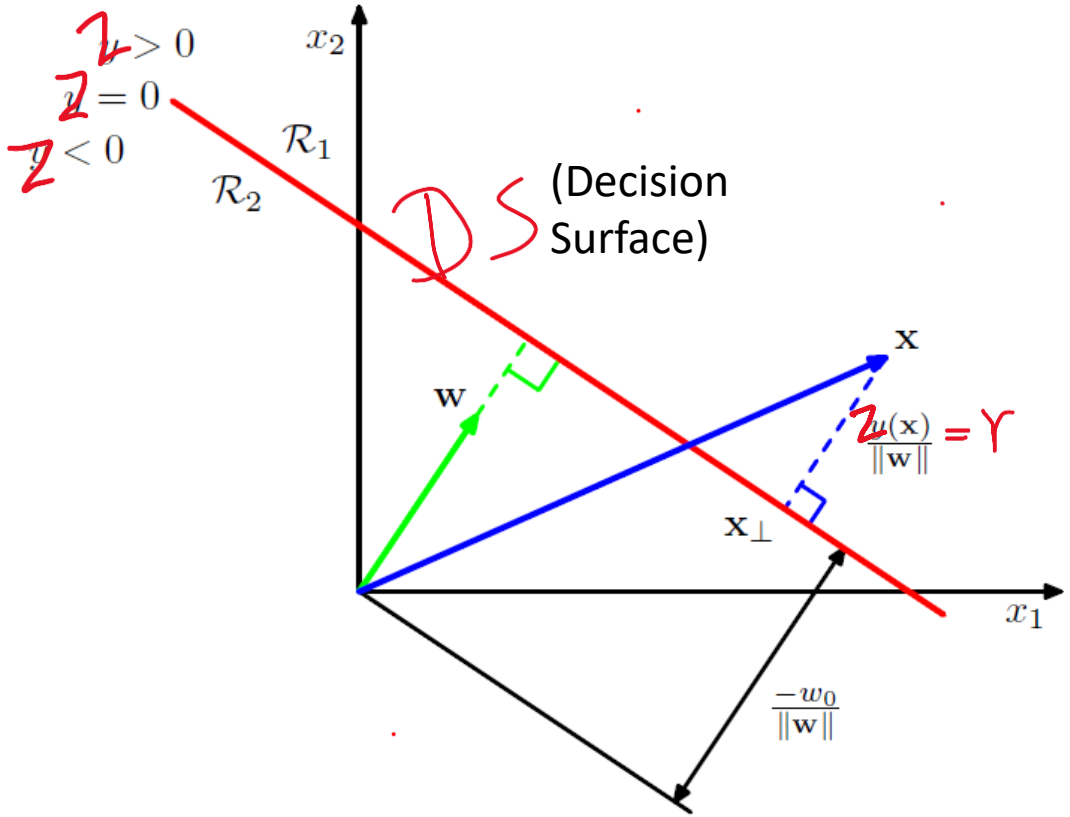
# Discriminant

- Discriminant is a function that takes an input vector $x \in \mathbb{R}^d$ and assigns it to one of the $K$ classes
  - we will assume K=2 henceforth for simplicity!

- We focus only on linear discriminants (i.e., those for which DB is a hyperplane wrt $x$ (or $\phi(x)$)**.**
  - $z(x) = w^T x + w_0$ (or $w^T \phi(x)$)
  - DB: $z(x) = 0$ (hyperplane)
  - Prediction: $f\big(z(x)\big)$ = $\text{sign}(z(x))$ (i.e., Predict $C_1$ if $z(x) \geq 0$, & $C_2$ if $z(x) < 0$)

Recall defn. of hyperplane $\{x \in \mathbb{R}^d : w^T x = b\}$, which is a $(d-1)$-dimensional (affine) subspace of a $d$-dim. vector space.

# Geometry of decision surfaces: 2-way or binary classification




DB is $w^T \mathrm{x} + w_0$ = *const.*, but *const.* can be absorbed into $w_0$ to get DB as $\boldsymbol{w^T x + w_0 = 0}$.
Let $z(w, x) = w^T \mathrm{x} + w_0$ with the *const.* absorbed.

Signed dist. $r$ of $x$ to DB:

Express $X = X_\perp + r \cdot \frac{w}{\|w\|}$

Mult. by $W^T$ & add $W_0$ on both sides:

$\Rightarrow z(X) = 0 + r\|w\|$

$\Rightarrow \boxed{r = \frac{z(X)}{\|w\|}}$

[CMB]

# Outline for Module M7

- M7. Classification (Linear models)
  - M7.0 Introduction
  - M7.1 Generative model ((Naïve) Bayes classifier)
  - M7.2 Discriminative model (Logistic regression)
  - **M7.3 Discriminant function (very brief)**
    - Geometry of decision surfaces
    - **Perceptron objective function**
    - Unifying view of objective functions
  - M7.4 Kernel methods
  - M7.5 Concluding thoughts

# Perceptron
## (artificial neuron - elementary unit in ANNs)

- General form of linear models of classification:

$$f_{\boldsymbol{w}}(\mathbf{x}) = f\left(\boldsymbol{w}^T\mathbf{x} + w_0\right), \text{ or}$$
$$f_{\boldsymbol{w}}(\mathbf{x}) = f(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}))$$

- Choice of function $f$ for perceptron (or any linear discriminant for that matter):

$$f(a) = \begin{cases} +1, & a \geqslant 0 \\ -1, & a < 0. \end{cases}$$

- Choice of objective function **specific to perceptron** to learn w:

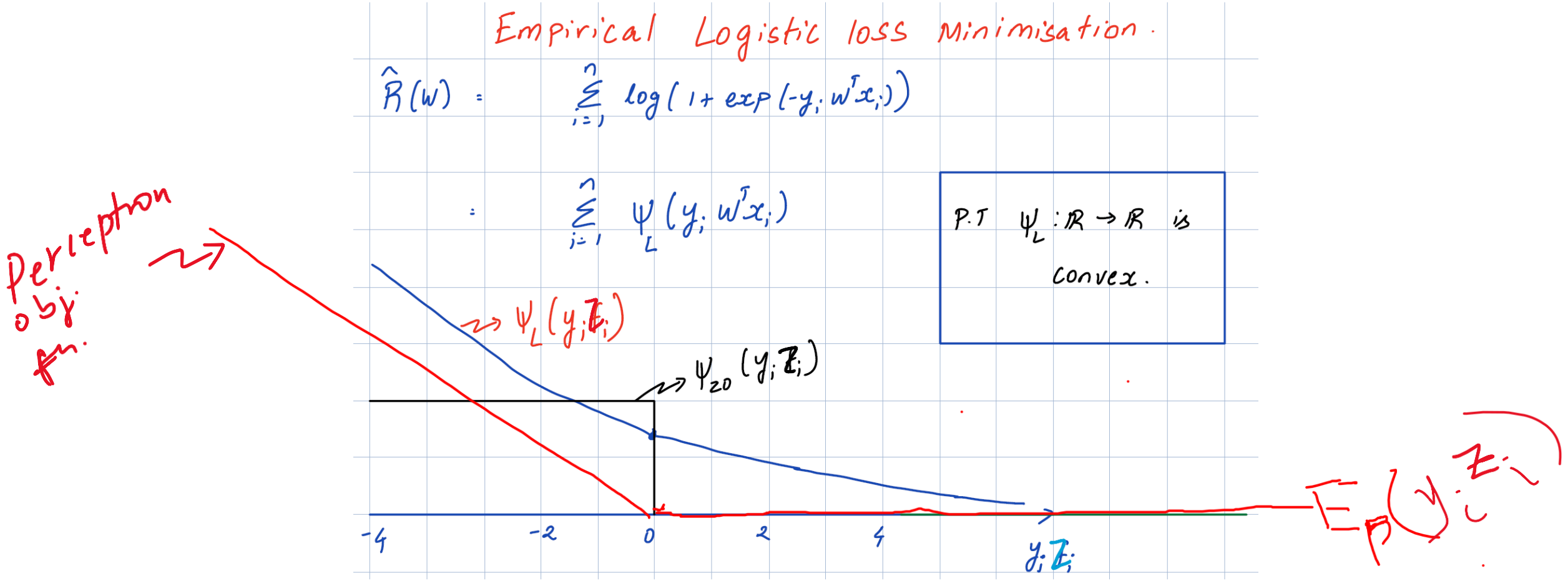
$$E_P(w) = \sum_{i \in \mathcal{M}} (-\, y_i \;\; \boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x_i}))$$

misc/assif.

$\propto$ distance of $\phi(x_i)$ from DS/DB

[CMB]

# Recall: Logistic regression – direct (non-probabilistic) motivation using obj. fn.




Perceptron – simple (stochastic grad. descent) training algo. exists, but only of historical importance, as it suffers from many issues (doesn't converge for non-linearly separable dataset, multiple decision surfaces possible for linearly separable datasets, convergence in practice may take many steps,…)

# Outline for Module M7

- M7. Classification (Linear models)
  - M7.0 Introduction
  - M7.1 Generative model ((Naïve) Bayes classifier)
  - M7.2 Discriminative model
  - **M7.3 Discriminant function (very brief)**
    - Geometry of decision surfaces
    - Perceptron objective function
    - **Unifying view of objective functions**
  - M7.4 Kernel methods
  - M7.5 Concluding thoughts

# The larger context: loss fn view offers a ***unified motivation*** of many classifn. methods (thereby allowing us to condense a "laundry" list of methods into a single framework)

I. LOSS FUNCTIONS FOR BINARY CLASSIFICATION

Given data : $(x_1, y_1) \cdots (x_n, y_n) \in \mathbb{R}^d \times \{\pm 1\}$

Most algorithms we see are "surrogate risk" minimisation algorithms.

(i.e) They want to find an $f: \mathbb{R}^d \to \mathbb{R}$

$$\text{sign}(f(x_i)) = y_i$$

or equivalently minimise

$$\mathcal{Z}_0[f] = \sum_{i=1}^{n} \psi_{z_0}(y_i f(x_i))$$

where $\psi_{z_0}(u) = 1$ if $u \leq 0$

$= 0$ otherwise.

[HR]

# Surrogate loss fns

But due to the difficulty of minimising the zero one error we minimise one of these:

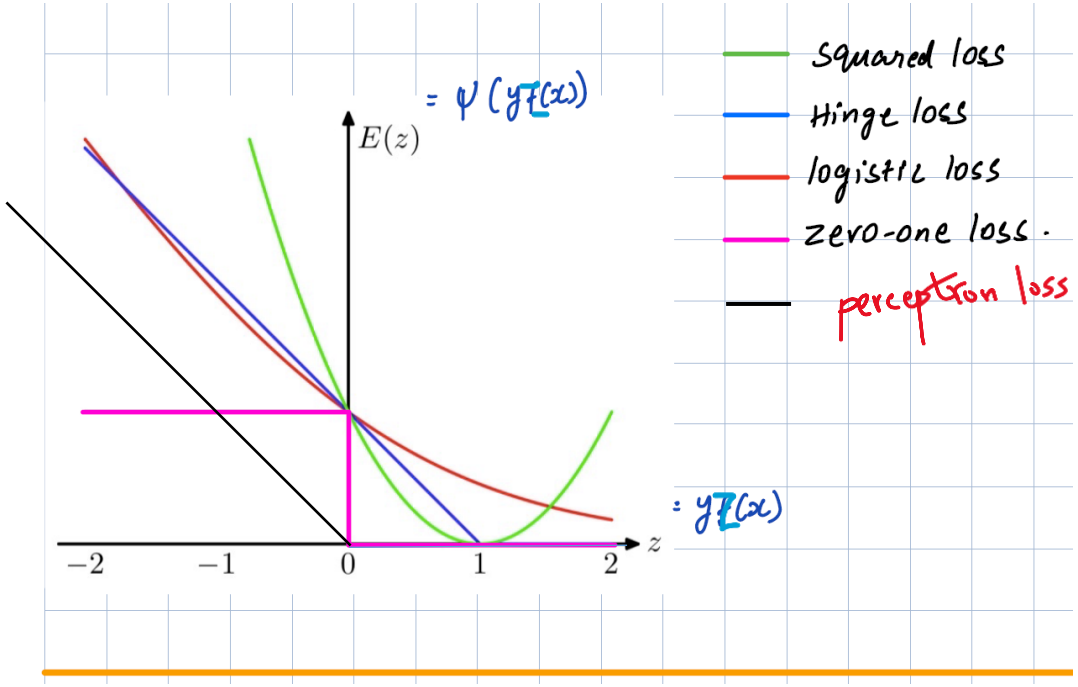
$$sq[f] = \sum_{i=1}^{n} \psi_Q(y_i f(x_i)) = \sum_{i=1}^{n} (1 - y_i f(x_i))^2 = \sum_{i=1}^{n} (y_i - f(x_i))^2$$

$$\text{logistic}[f] = \sum_{i=1}^{n} \psi_L(y_i f(x_i)) = \sum_{i=1}^{n} \log(1 + \exp(-y_i f(x_i)))$$

$$\text{Hinge}[f] = \sum_{i=1}^{n} \psi_H(y_i f(x_i)) = \sum_{i=1}^{n} \max(0, 1 - y_i f(x_i))$$

$$\exp[f] = \sum_{i=1}^{n} \psi_E(y_i f(x_i)) = \sum_{i=1}^{n} \exp(-y_i f(x_i))$$

$$\text{perc}[f] = \sum_{i=1}^{n} \psi_{\text{perc}}(y_i f(x_i)) = \sum_{i=1}^{n} \max(0, -y_i f(x_i))$$




[HR]

# More on Hinge loss and (soft-margin) SVM

$$\min_{w,b,\xi} \ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \quad \text{s.t} \quad y_i(w^T x_i + b) \geq 1 - \xi_i, \ \xi_i \geq 0$$

$$\Leftrightarrow \min_{w,b} \ C\sum_{i=1}^{n}\psi_H(y_i(w^T x_i + b)) + \frac{1}{2}\|w\|^2$$

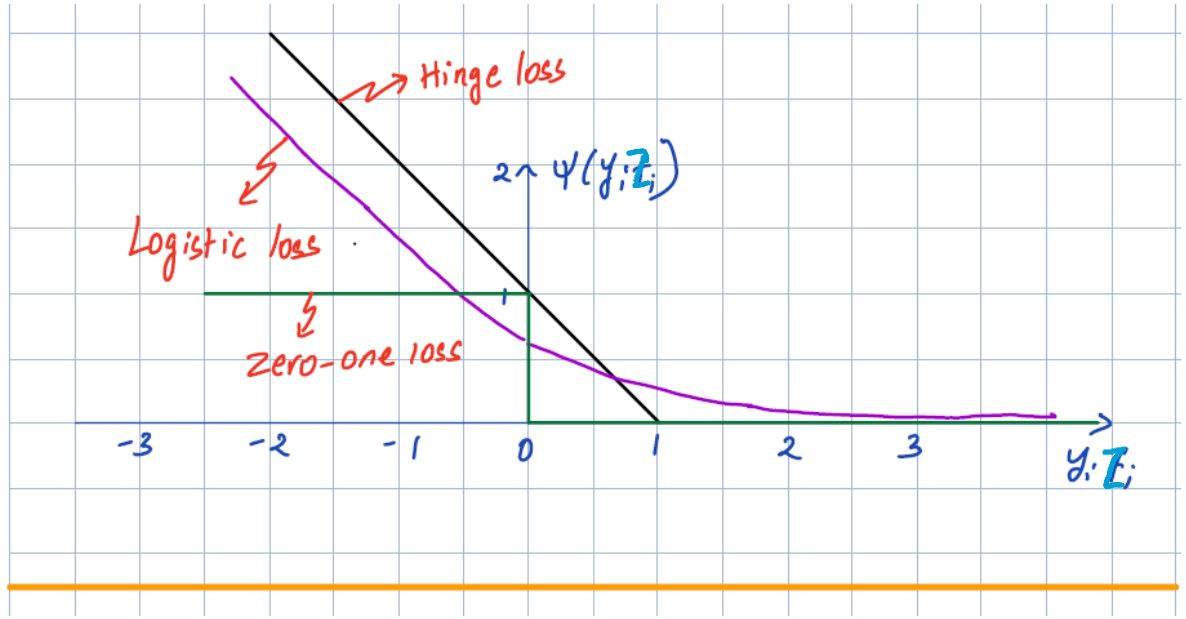
Express $\xi_i$ as a function of $w, b$:

$$\xi_i \geq 1 - y_i(w^T x_i + b) \ ; \ \xi_i \geq 0$$

$$\therefore \ \xi_i = \max\left(1 - y_i(w^T x_i + b), 0\right)$$

$$= \max(1 - y_i z_i, 0) \quad \text{where } z_i = w^T x_i + b$$

$$= \psi_H(y_i z_i)$$




$$\psi_H(u) = \max(1-u, 0)$$

[HR]

# Outline for Module M7

- M7. Classification (Linear models)
  - M7.0 Introduction
  - M7.1 Generative model ((Naïve) Bayes classifier)
  - M7.2 Discriminative model (Logistic regression)
  - M7.3 Discriminant function (Perceptron) (very brief)
  - **M7.4 Kernel methods**
    - **Representer theorem**
    - **Kernel logistic regression / Kernel regression (as Asst. qn.) / etc. (leading to Kernel SVM, etc.)**
  - M7.5 Concluding thoughts

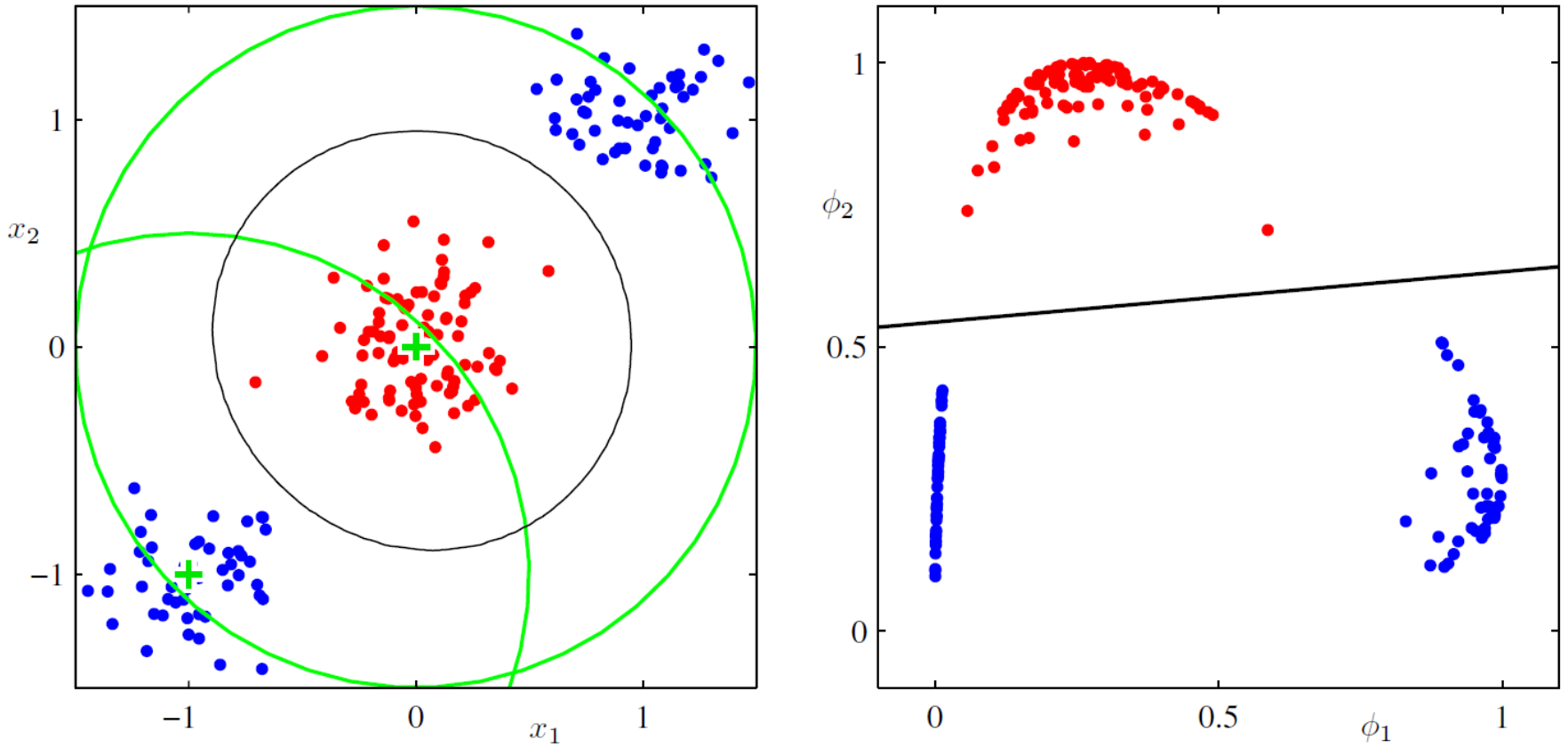



**Figure 4.12** Illustration of the role of nonlinear basis functions in linear classification models. The left plot shows the original input space $(x_1, x_2)$ together with data points from two classes labelled red and blue. Two 'Gaussian' basis functions $\phi_1(\mathbf{x})$ and $\phi_2(\mathbf{x})$ are defined in this space with centres shown by the green crosses and with contours shown by the green circles. The right-hand plot shows the corresponding feature space $(\phi_1, \phi_2)$ together with the linear decision boundary obtained given by a logistic regression model of the form discussed in Section 4.3.2. This corresponds to a nonlinear decision boundary in the original input space shown by the black curve in the left-hand plot.

[CMB]

$\phi : \mathbb{R}^d \to \mathbb{R}^{d'}$ Kernel Logistic Regression

$$\min_{w \in \mathbb{R}^{d'}} \sum_{i=1}^{n} \log\left(1+\exp\left(-y_i w^T \phi(x_i)\right)\right) + \frac{\lambda}{2}\|w\|^2$$

$$\Phi = \begin{bmatrix} \phi(x_1)^T \\ \vdots \\ \phi(x_n)^T \end{bmatrix}_{n \times d'}$$

$$w = \Phi^T \alpha$$

$$K = \Phi \Phi^T \quad (n \times n)$$

$\Updownarrow$ RT

$$\min_{\alpha \in \mathbb{R}^n} \sum_{i=1}^{n} \log\left(1+\exp\left(-y_i \alpha^T \Phi \phi(x_i)\right)\right) + \frac{\lambda}{2} \alpha^T \Phi \Phi^T \alpha$$

$\Updownarrow$ subst.

$$\min_{\alpha \in \mathbb{R}^n} \sum_{i=1}^{n} \log\left(1+\exp\left(-y_i \alpha^T K_{:,i}\right)\right) + \frac{\lambda}{2} \alpha^T K \alpha$$

The $i^{th}$ column of $K$.

[HR]

# Representer theorem

$$\min_{w \in \mathbb{R}^d} \; f(\Phi w) + \frac{\lambda}{2}\|w\|^2$$

$$\Phi \in \mathbb{R}^{n \times d}$$

$$w = u + v \qquad u \in NS(\Phi),\ v \in RS(\Phi)$$

$$\min_{\substack{u \in NS(\Phi) \\ v \in RS(\Phi)}} f(\Phi(u+v)) + \frac{\lambda}{2}\|u+v\|^2$$

$$\min_{v \in RS(\Phi)} f(\Phi v) + \frac{\lambda}{2}\|u\|^2 + \frac{\lambda}{2}\|v\|^2$$

$\|u\|^2 \to 0$

[HR]

# Representer theorem (contd.)

$$\min_{v \in RS(\Phi)} f(\Phi v) + \frac{\lambda}{2}\|v\|^2$$

$$\Updownarrow$$

$$\min_{\alpha \in \mathbb{R}^n} f(\Phi \Phi^T \alpha) + \frac{\lambda}{2}\|\Phi^T \alpha\|^2$$

$$\Updownarrow$$

$$\min_{\alpha \in \mathbb{R}^n} f(K\alpha) + \frac{\lambda}{2}\alpha^T K \alpha$$

[HR]

# Gradient descent for Kernel Logistic Regression:

$$R(\alpha) = \sum_{i=1}^{n} \log\left(1 + \exp\left(-y_i \alpha^T K_{:,i}\right)\right) + \frac{\lambda}{2} \alpha^T K \alpha$$

$$\nabla R(\alpha) = \sum_{i=1}^{n} \frac{\exp\left(-y_i \alpha^T K_{:,i}\right)}{1 + \exp\left(-y_i \alpha^T K_{:,i}\right)} \cdot \left(-y_i K_{:,i}\right) + \lambda K \alpha$$

$$= \sum_{i=1}^{n} \sigma\left(-y_i \alpha^T K_{:,i}\right)\left(-y_i K_{:,i}\right) + \lambda K \alpha$$

$$\alpha^* = \min_{\alpha} R(\alpha)$$

## Gradient Descent:

1. Initialise $\alpha \in \mathbb{R}^n$, Choose $\eta > 0$.
2. Repeat:
   (a) $\alpha \leftarrow \alpha - \eta \nabla R(\alpha)$
   (b) Stop if $R(\alpha)$ has not decreased much in the last few iterations.
3. Return $\alpha$.

[HR]

## Predicting the class at a given $x$:

Predicted value of $P(y=1 \mid x=x) = \sigma(w^{*T}\phi(x))$

$$\sigma(w^{*T}\phi(x)) = \sigma(\alpha^{*T}\Phi\phi(x))$$

$$= \sigma\left(\sum_{i=1}^{n} \alpha_i^* K(x_i, x)\right)$$

Class Predicted : $\text{sign}\left(\sigma(w^{*T}\phi(x)) - 0.5\right)$

$$= \text{Sign}\left(w^{*T}\phi(x)\right)$$

[HR]

# Brief Mention: Kernel (linear) regression

Representer Theorem:

$$\min_{w \in \mathbb{R}^{d'}} \frac{1}{2} \|\Phi w - y\|^2 + \frac{\lambda}{2} \|w\|^2 \qquad \Phi \in \mathbb{R}^{n \times d'}$$

$$\Phi^T \alpha = \sum_{i=1}^{n} \alpha_i \phi(x_i)$$

$\Updownarrow$ RT

$$\min_{\alpha \in \mathbb{R}^n} \frac{1}{2} \|\Phi \Phi^T \alpha - y\|^2 + \frac{\lambda}{2} \|\Phi^T \alpha\|^2$$

Why? Any $w \in \mathbb{R}^{d'} = v + u$ for some $u \in \text{Null}(\Phi)$, $v \in \text{Row}(\Phi)$

and $\|w\|^2 = \|v\|^2 + \|u\|^2$

Let $v = \Phi^T \alpha$

**(will be explored further in Assignment)** [HR]

# Kernel functions and examples

$K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is called kernel function, if $K(u, v)$ represents $\phi(u)^T \phi(v)$ for some $\phi : \mathbb{R}^d \to \mathbb{R}^{d'}$

Example Kernel:

$$K(u, v) = \exp\left(-\frac{\|u - v\|^2}{\sigma^2}\right)$$

$u \in \mathbb{R}^d$, $v \in \mathbb{R}^d$

$\exists \ \phi : \mathbb{R}^d \to \mathcal{H}$

$$K(u, v) = \langle \phi(u), \phi(v) \rangle$$

Another Example Kernel:

$$K(u, v) = (u^T v)^{73}$$

[HR]

# Illustration of Kernel Function:

To show: $K(u,v) = (u^T v)^3$ is a valid Kernel.

$$X = \begin{bmatrix} t_1 & t_2 \\ u_1 & u_2 \\ v_1 & v_2 \end{bmatrix} \quad ; \quad \Phi = \begin{bmatrix} t_1^3 & \sqrt{3}t_1^2 t_2 & \sqrt{3}t_1 t_2^2 & t_2^3 \\ u_1^3 & \sqrt{3}u_1^2 u_2 & \sqrt{3}u_1 u_2^2 & u_2^3 \\ v_1^3 & \sqrt{3}v_1^2 v_2 & \sqrt{3}v_1 v_2^2 & v_2^3 \end{bmatrix} = \begin{bmatrix} \phi(t)^T \\ \phi(u)^T \\ \phi(v)^T \end{bmatrix}$$

$\exists \phi$ s.t. ...

2d data & Homogenous 3rd ~~degree~~ Polynomial regression

3 data points.

$$\phi(u) = [u_1^3, \sqrt{3}u_1^2 u_2, \sqrt{3}u_1 u_2^2, u_2^3]$$

$$\Phi\Phi^T = \begin{bmatrix} \phi(t)^T\phi(t) & \phi(t)^T\phi(u) & \phi(t)^T\phi(v) \\ \phi(u)^T\phi(t) & \phi(u)^T\phi(u) & \phi(u)^T\phi(v) \\ \phi(v)^T\phi(t) & \phi(v)^T\phi(u) & \phi(v)^T\phi(v) \end{bmatrix}$$

[HR]

$$\phi(u)^T \phi(v) = u_1^3 v_1^3 + 3 u_1^2 u_2 v_1^2 v_2 + 3 u_1 u_2^2 v_1 v_2^2 + u_2^3 v_2^3$$

$$= (u_1 v_1 + u_2 v_2)^3$$

$$= (u^T v)^3 = K(u, v)$$

$\downarrow$

Homogenous 3rd degree polynomial Kernel.

(i.e) you can compute $\phi(u)^T \phi(v)$ without computing $\phi(u)$ & $\phi(v)$

[HR]

Qn: What functions $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ are valid kernel functions?

Exercise: Identify if following fns are kernels:

1) These kernels are for $d = 1$
   (a) $K(x,y) = x+y$
   (b) $K(x,y) = x-y$
   (c) $K(x,y) = xy$
   (d) $K(x,y) = x^2$
   (e) $K(x,y) = x^2 + y^2$
2) These kernels are for $d > 1$
   (a) $K(x,y) = x^T y$
   (b) $K(x,y) = \sum_{i=1}^{d} x_i^2 y_i^2$
   (c) $K(x,y) = \exp(x^T y)$
   (d) $K(x,y) = (1 + x^T y)^p$
   (e) $K(x,y) = \exp\left(-\frac{\|x-y\|^2}{2\sigma^2}\right)$

Solution:

i) Find $\phi : \mathbb{R}^d \to \mathbb{R}^{d'}$ s.t $K(x,y) = \phi(x)^T \phi(y)$ to show $K$ is a valid kernel.

ii) For all sets of data points $x_1, \ldots x_n$ with $x_i \in \mathbb{R}^d$, the $n \times n$ Kernel matrix must be symmetric and PSD for $K$ to be a valid Kernel.

[HR]

# Concluding thoughts

- Linear models for classification seen from all three approaches - generative/discriminative/direct.
  - Has similar pros/cons as linear regression - non-linear basis functions, along with the kernel method/trick, can allow for non-linear boundaries between classes, but still suffers from fundamental limitation of fixed basis functions (fixed before seeing the training data).

- Next steps:
  - Non-linear models of classification: Use training data that starts with a set of non-linear basis functions, and either selects a subset of it (kernel SVM; convex objective fn.) or adapts their parameters (multi-layer perceptron or neural network; non-convex obj.).

[CMB]

# Thank you! (Appendix follows)

# Appendix

# Kernel PCA

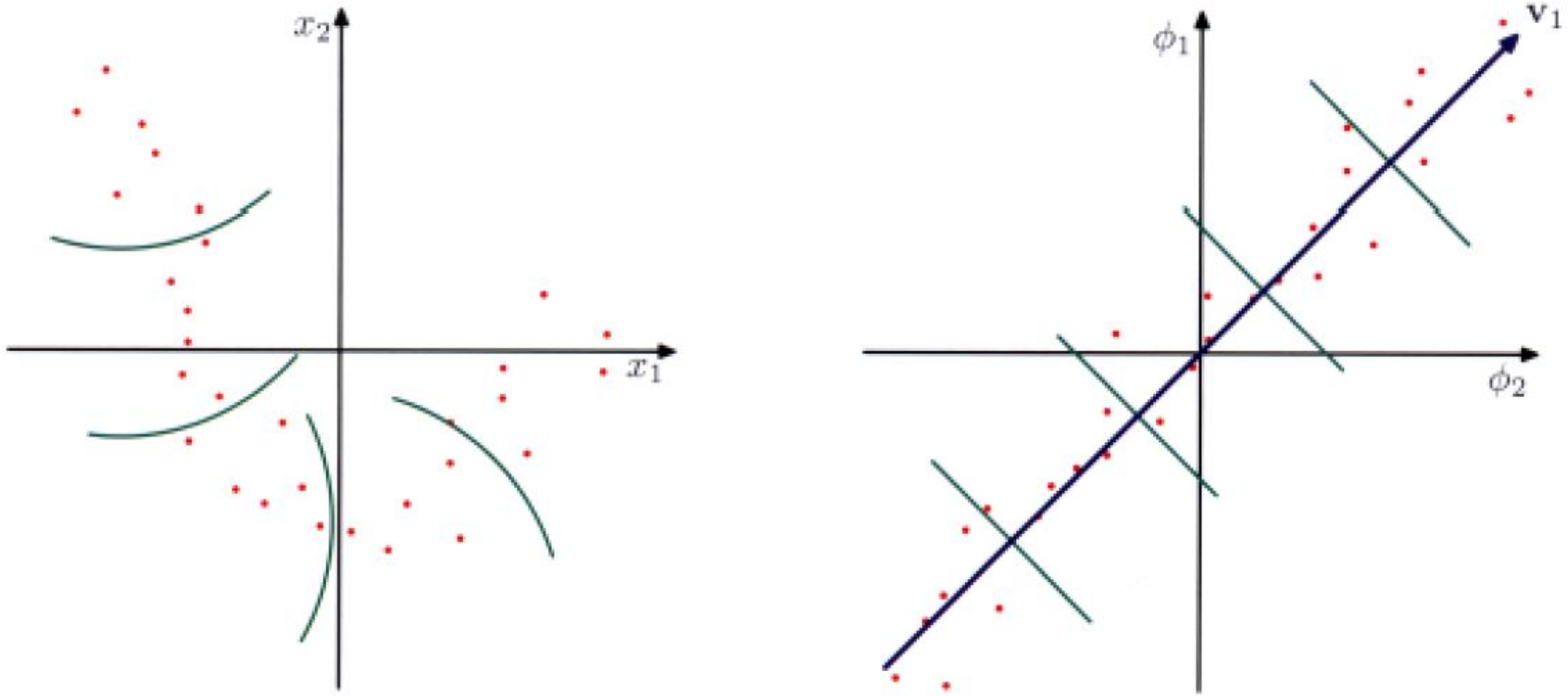
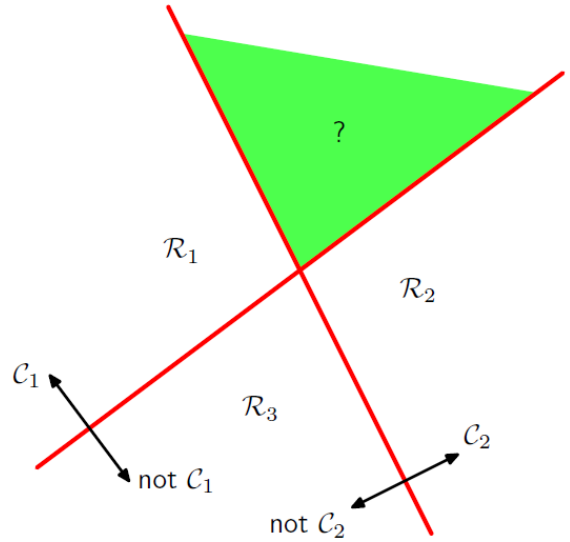


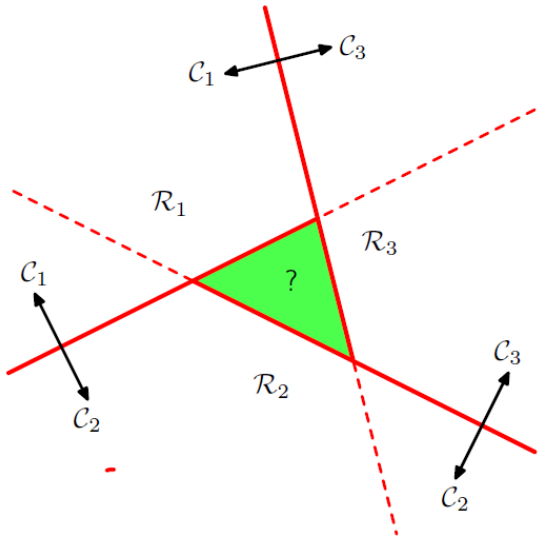

**Figure 12.16** Schematic illustration of kernel PCA. A data set in the original data space (left-hand plot) is projected by a nonlinear transformation $\phi(\mathbf{x})$ into a feature space (right-hand plot). By performing PCA in the feature space, we obtain the principal components, of which the first is shown in blue and is denoted by the vector $\mathbf{v}_1$. The green lines in feature space indicate the linear projections onto the first principal component, which correspond to nonlinear projections in the original data space. Note that in general it is not possible to represent the nonlinear principal component by a vector in $\mathbf{x}$ space.

[CMB]

# Discriminant (DBs) for K>2 classes: Simple 2-way → multi-way constructions don't work! However, multi-way → 2-way works!



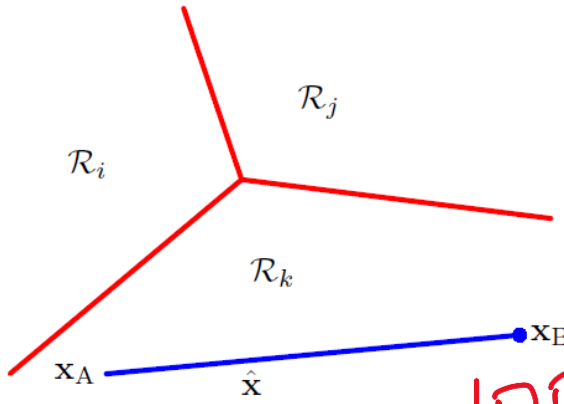

1-vs-rest




$C_i$ vs $C_j$

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

$\hat{k}(x) = \arg\max_{k \in [K]} z_k(x)$




(cvx)

$z_i(x) = z_j(x)$ DS/DB with similar geom.

[CMB]

# Perceptron training (learning **w**)

- Stochastic gradient descent: $\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E_{\mathrm{P}}(\mathbf{w}) = \mathbf{w}^{(\tau)} + \eta \boldsymbol{\phi}_n t_n$
  - Only interested in the direction of $w$. So $\eta$ can be set to 1.
  - That is, perceptron function $y(x, w)$ unchanged if $w$ scaled by a constant.

- Then, for a single misclassified data point, add $\phi(x_n)$ to **w** if $t = +1$, and subtract it from **w** o.w.
  - This will lead to decrease in error for a single update, because:

$$-\mathbf{w}^{(\tau+1)\mathrm{T}} \boldsymbol{\phi}_n t_n = -\mathbf{w}^{(\tau)\mathrm{T}} \boldsymbol{\phi}_n t_n - (\boldsymbol{\phi}_n t_n)^{\mathrm{T}} \boldsymbol{\phi}_n t_n < -\mathbf{w}^{(\tau)\mathrm{T}} \boldsymbol{\phi}_n t_n$$

  - If data is linearly separable, proof of convergence in finite steps exists.

- Only of historical importance, as it suffers from many issues: training doesn't converge for non-linearly separable dataset, multiple decision surfaces possible for linearly separable datasets, convergence in practice may take very many steps, etc.

[CMB]

# Perceptron training in action

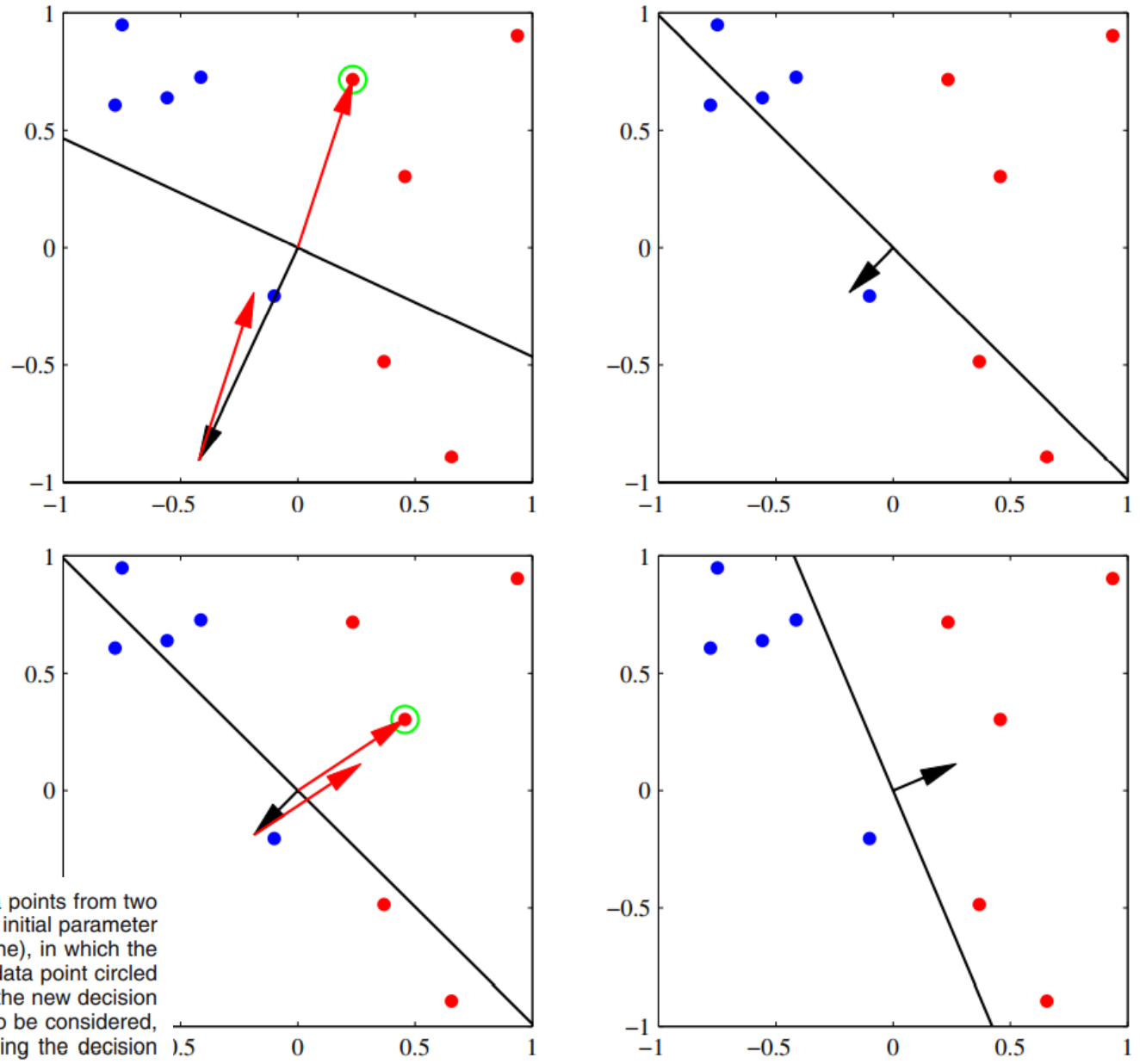


**Figure 4.7** Illustration of the convergence of the perceptron learning algorithm, showing data points from two classes (red and blue) in a two-dimensional feature space $(\phi_1, \phi_2)$. The top left plot shows the initial parameter vector $\mathbf{w}$ shown as a black arrow together with the corresponding decision boundary (black line), in which the arrow points towards the decision region which classified as belonging to the red class. The data point circled in green is misclassified and so its feature vector is added to the current weight vector, giving the new decision boundary shown in the top right plot. The bottom left plot shows the next misclassified point to be considered, indicated by the green circle, and its feature vector is again added to the weight vector giving the decision boundary shown in the bottom right plot for which all data points are correctly classified.

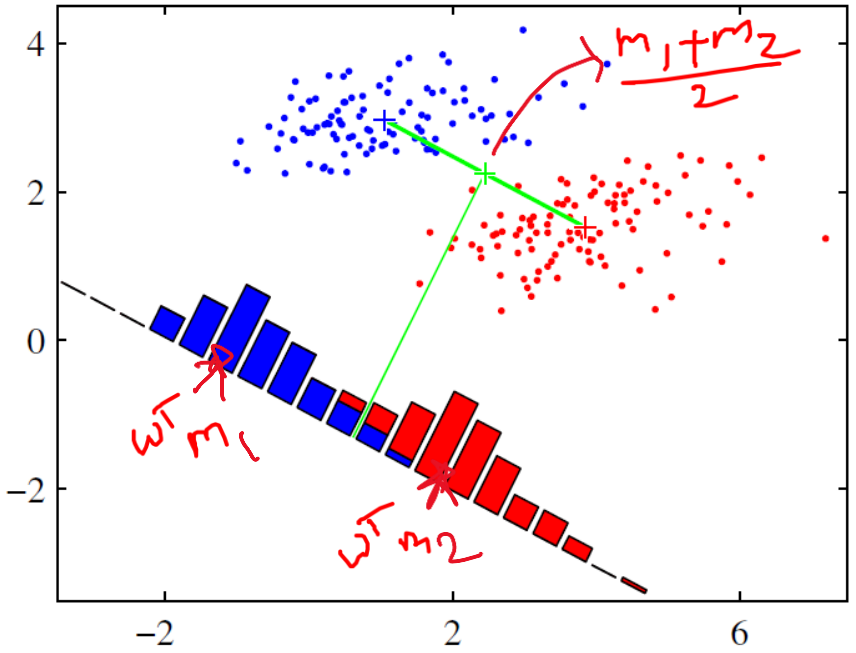
[CMB]

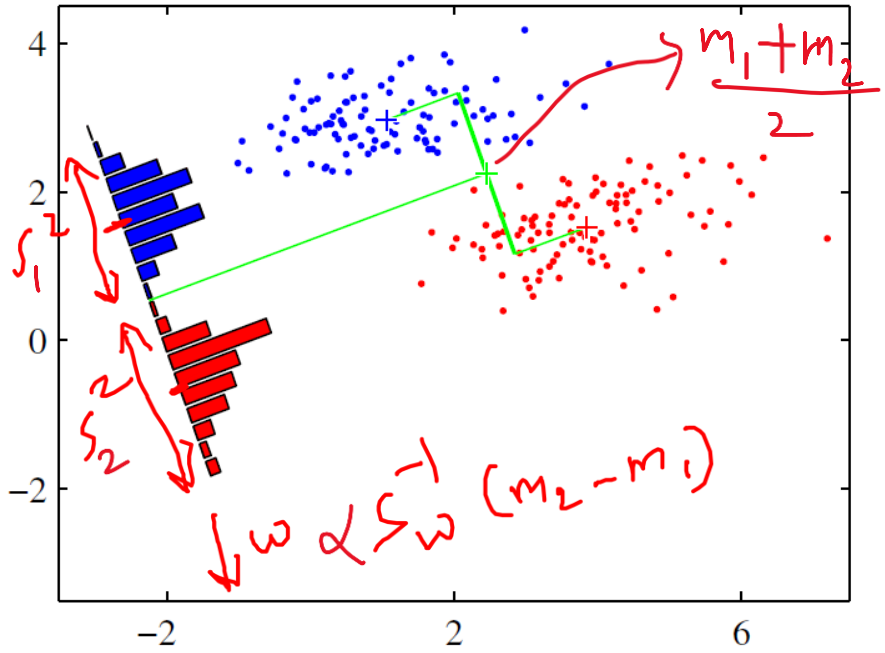
# Outline for Module M5

- M5. Classification (Linear models)
  - M5.0 Introduction
  - M5.1 Generative model ((Naïve) Bayes classifier)
  - M5.2 Discriminative model (Logistic regression)
  - **M5.3 Discriminant function**
    - Geometry of decision surfaces
    - Perceptron objective function
    - **Linear/Fisher's Discriminant Analysis**
  - M5.4 Kernel methods
  - M5.5 Concluding thoughts

# Fisher's discriminant analysis

- Reduce points to one dimension using a linear function, and then apply a threshold (step fn. $f(.)$ to classify as in perceptron).
- So same choice of $f(.), f(\boldsymbol{w}^{\boldsymbol{T}} x)$ as in perceptron, but choice of objective function to learn **w** differs (between/within class variance).




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

[CMB]

# FDA - Choice of obj. fn. – attempt #1

Maximize $\mathbf{w}^{\mathrm{T}}(\mathbf{m}_2 - \mathbf{m}_1)$ s.t. $\|w\| = 1$

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

yields $w \propto (m_2 - m_1)$

[CMB]

# FDA - Choice of obj. fn. – attempt #2

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

BCW / WCW

$$m_k = \mathbf{w}^{\mathrm{T}}\mathbf{m}_k = y(m_k)$$

$$s_k^2 = \sum_{n \in \mathcal{C}_k} (y_n - m_k)^2$$

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

$y(x) = w^T x$

[CMB]

# Maximizing J(w)

$$J(\mathbf{w}) = \frac{\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{B}}\mathbf{w}}{\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{W}}\mathbf{w}} \tag{4.26}$$

where $\mathbf{S}_{\mathrm{B}}$ is the *between-class* covariance matrix and is given by

$$\mathbf{S}_{\mathrm{B}} = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^{\mathrm{T}} \tag{4.27}$$

d x d

and $\mathbf{S}_{\mathrm{W}}$ is the total *within-class* covariance matrix, given by

$$\mathbf{S}_{\mathrm{W}} = \sum_{n\in\mathcal{C}_1}(\mathbf{x}_n - \mathbf{m}_1)(\mathbf{x}_n - \mathbf{m}_1)^{\mathrm{T}} + \sum_{n\in\mathcal{C}_2}(\mathbf{x}_n - \mathbf{m}_2)(\mathbf{x}_n - \mathbf{m}_2)^{\mathrm{T}}. \tag{4.28}$$

Differentiating (4.26) with respect to $\mathbf{w}$, we find that $J(\mathbf{w})$ is maximized when

$$(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{B}}\mathbf{w})\mathbf{S}_{\mathrm{W}}\mathbf{w} = (\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{W}}\mathbf{w})\mathbf{S}_{\mathrm{B}}\mathbf{w}. \tag{4.29}$$

[CMB]

# Maxima of J(w) -- identical to generative model (Bayes classifier; cf. next slide)!

From (4.27), we see that $\mathbf{S}_{\mathrm{B}}\mathbf{w}$ is always in the direction of $(\mathbf{m}_2 - \mathbf{m}_1)$. Furthermore, we do not care about the magnitude of $\mathbf{w}$, only its direction, and so we can drop the scalar factors $(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{B}}\mathbf{w})$ and $(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{W}}\mathbf{w})$. Multiplying both sides of (4.29) by $\mathbf{S}_{\mathrm{W}}^{-1}$ we then obtain

$$\mathbf{w} \propto \mathbf{S}_{\mathrm{W}}^{-1}(\mathbf{m}_2 - \mathbf{m}_1). \tag{4.30}$$

- What if data is axis-aligned? (think about identity as well as diagonal $S_w$)
- What to do once you've projected data points i.e., what is a good threshold to apply?

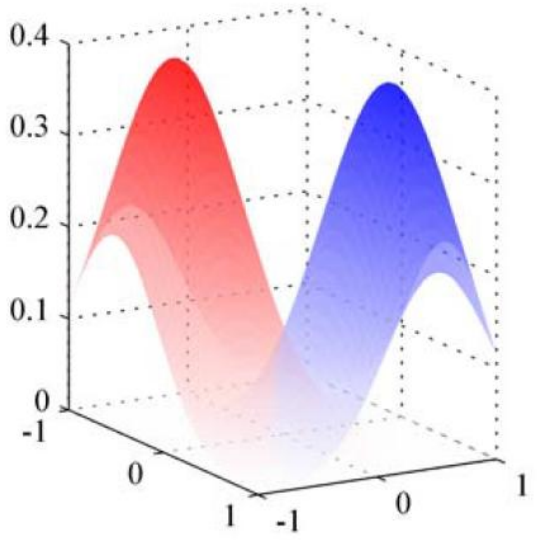
→ $w^T\left(\frac{m_2+m_1}{2}\right)$ $(:= -w_0)$

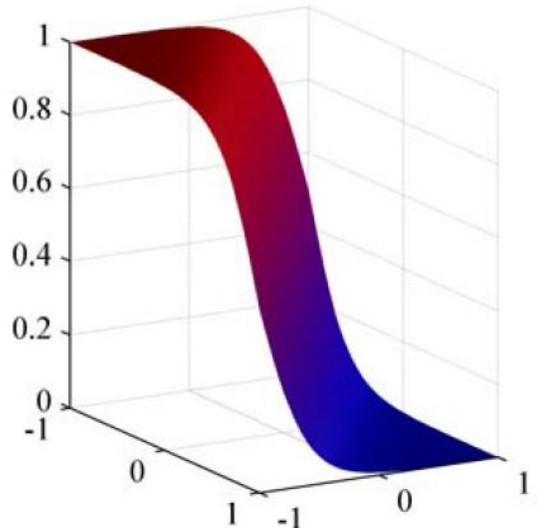
[CMB]

# Recall: K=2 Bayes classifier – Gaussian, general (shared) covariance

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

$$p(\mathcal{C}_1|\mathbf{x}) = \sigma(\mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0)$$

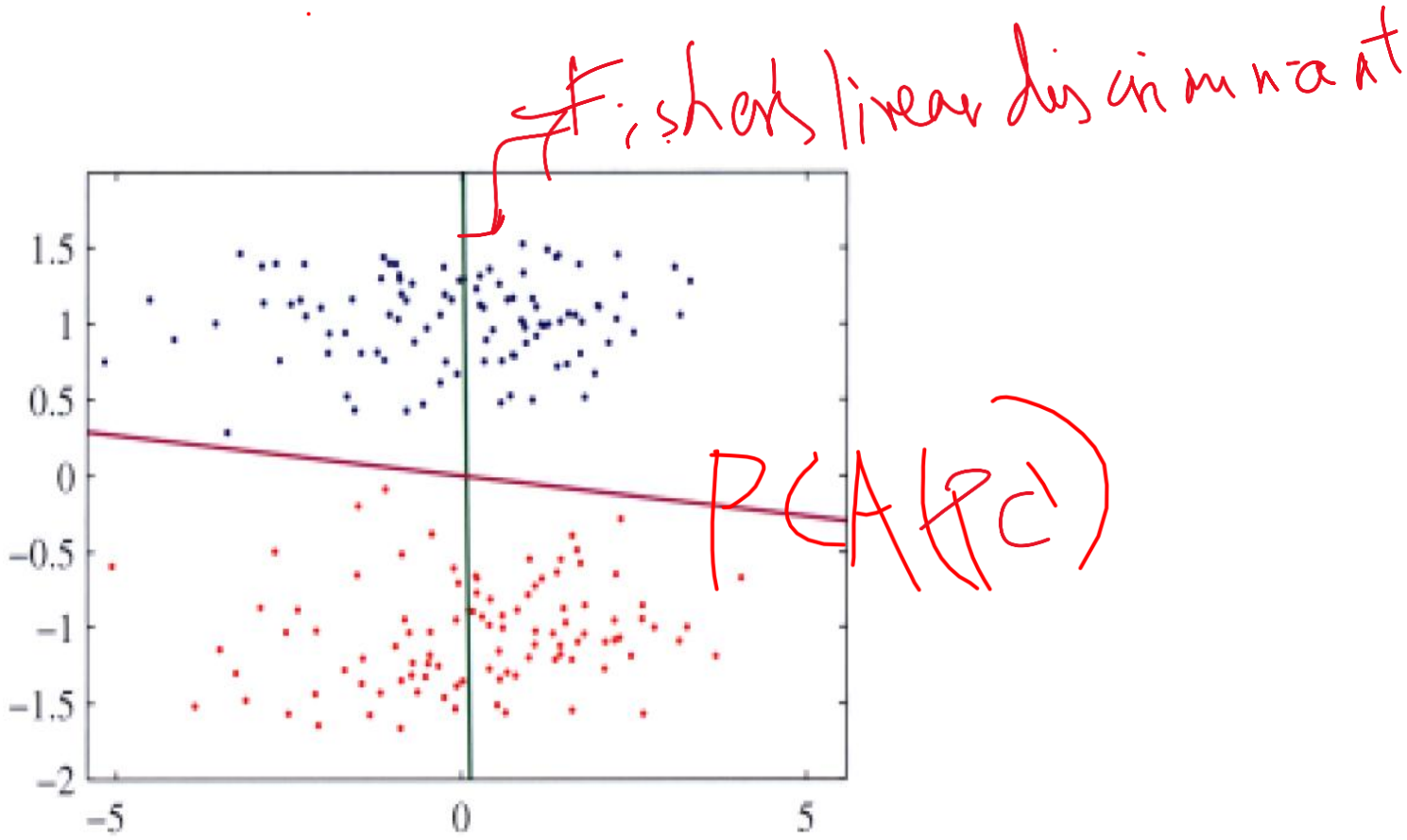



$$\begin{aligned}
\mathbf{w} &= \mathbf{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) \\
w_0 &= -\frac{1}{2}\boldsymbol{\mu}_1^{\mathrm{T}}\mathbf{\Sigma}^{-1}\boldsymbol{\mu}_1 + \frac{1}{2}\boldsymbol{\mu}_2^{\mathrm{T}}\mathbf{\Sigma}^{-1}\boldsymbol{\mu}_2 + \ln\frac{p(\mathcal{C}_1)}{p(\mathcal{C}_2)}.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

[CMB]

# A comparison: PCA vs FDA




[CMB]