EECS 553 Lecture 09
Machine Learning (ECE)
[class version]

Jeff Fessler

University of Michigan
©Laura Balzano and Jeff Fessler, UM, September 29, 2026, 13:12

F26

Logistics

© 2026 Fessler

- HW 4 will be due Wed, Oct. 7; mostly "non-graded" problems.
- Reproducibility project updates
  - 18 teams (12 with 4, 6 with 3, 2 students unassigned!?)
  - PA2: Fri. Oct. 9: reproducibility paper selected and brief paper justification submitted by each group
  - Can be a different paper from the 2-3 you submitted already
  - Exceptions to paper selection?
  - PA3: Fri. Nov. 9: draft report.
    Overleaf template linked in Canvas PA3
    Examples in Canvas → Files → Project → Example Reports

Recap / Outline

© 2026 Fessler

- Recap
  - Newton's method, Majorize-Minimize (MM) method
  - logistic regression
- Today
  - Logistic regression
    - Logistic regression: Likelihood
    - Logistic regression: Regularization
    - Multiclass logistic regression
  - Imbalanced data
  - Separating hyperplanes
    - Separating hyperplanes
  - Max-margin hyperplanes
  - Optimal soft-margin hyperplane: Prelude

Logistic regression: Between classification and regression ©

2026 Fessler

We first consider **logistic regression** in the binary case: $Y \in \{0, 1\}$. (There is a **multiclass** extension.)

This setting is typical for:

- Medical data:
  - The patient had a heart attack or did not.
  - The patient is alive or dead.
- Educational data:
  - The student passed or failed
  - The student was accepted to the program or not.
- Financial consumer data:
  - The transaction is legitimate or fraudulent.
  - The consumer defaulted on their loan or not.

Logistic regression: Classification or regression? © 2026 Fessler

- Consider the binary case for logistic regression: $Y \in \{0, 1\}$.
- Logistic regression *directly* builds and uses a model (or estimate) $\hat{\eta}(\boldsymbol{x}; \hat{\boldsymbol{\theta}})$ for the **posterior** probability (hence a kind of regression):

$$\eta(\boldsymbol{x}) \triangleq \mathsf{P}\{Y = 1 \mid \boldsymbol{X} = \boldsymbol{x}\}$$

- This estimate leads naturally to a **plug-in classifier** [1, §10.2.1]:

$$f(\boldsymbol{x}) \triangleq \begin{cases} 1, & \text{if } \hat{\eta}(\boldsymbol{x}; \hat{\boldsymbol{\theta}}) \geq \frac{1}{2} \\ 0, & \text{otherwise.} \end{cases}$$

- It combines two ideas:
  - Logistic probability estimation (a kind of **regression**)
  - "Linear" **classification**

Log. reg.: Probability model example

© 2026 Fessler

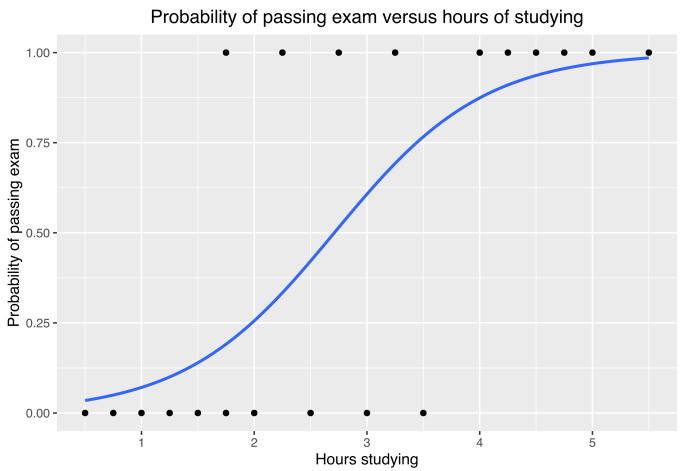
- Example taken from Wikipedia:
  A group of 20 students spends between 0 and 6 hours studying for an exam. How does the number of hours spent studying affect the probability of a student passing the exam?

Hours	0.50	0.75	1.00	1.25	1.50	1.75	1.75	2.00	2.25	2.50	2.75	3.00	3.25	3.50	4.00	4.25	4.50	4.75	5.00	5.50
Pass	0	0	0	0	0	0	1	0	1	0	1	0	1	0	1	1	1	1	1	1

If a (different) student studies for 3 hours, what is the probability of a pass?

Possible posterior probability model:

$$\begin{aligned} \mathrm{P}\{\text{Pass} \mid \text{Hours}\} &= \frac{1}{1 + \exp\Big(-\big(\, \boldsymbol{w}^\top \boldsymbol{x} + b \,\big)\Big)} \\ &= \frac{1}{1 + \exp(-(1.5 \times \text{Hours} - 4))} \\ &= \frac{1}{1 + \mathrm{e}^{-(1.5 \times (\text{Hours} - 2.7))}} \end{aligned}$$



Plot courtesy Canley

- The blue curve is a **sigmoid function** or "S-shaped" curve.
- Here we are using the **standard logistic function**:

$$\sigma(x) \triangleq \frac{1}{1 + \mathrm{e}^{-x}}.$$

- Feed linear combination of features into **logistic** probability model:

$$\eta(\boldsymbol{x};\boldsymbol{\theta}) \triangleq \sigma(\boldsymbol{w}^\top \boldsymbol{x} + b) = \frac{1}{1 + \mathrm{e}^{-(\boldsymbol{w}^\top \boldsymbol{x} + b)}} \approx \mathrm{P}\{Y = 1 \mid \boldsymbol{X} = \boldsymbol{x}\},$$

  where

$$\boldsymbol{\theta} = \begin{bmatrix} b \\ \boldsymbol{w} \end{bmatrix} \in \mathbb{R}^{d+1}.$$

- Somehow estimate $\hat{\boldsymbol{\theta}} = \begin{bmatrix} \hat{b} \\ \widehat{\boldsymbol{w}} \end{bmatrix}$. (Forthcoming...)
- Then plug-in method is a "linear" classifier:

$$\hat{y} = f(\boldsymbol{x}) = \mathbb{I}_{\{\eta(\boldsymbol{x};\hat{\boldsymbol{\theta}}) \geq 1/2\}}, \quad \eta(\boldsymbol{x};\hat{\boldsymbol{\theta}}) \geq \frac{1}{2} \iff \widehat{\boldsymbol{w}}^\top \boldsymbol{x} + \hat{b} \geq 0.$$

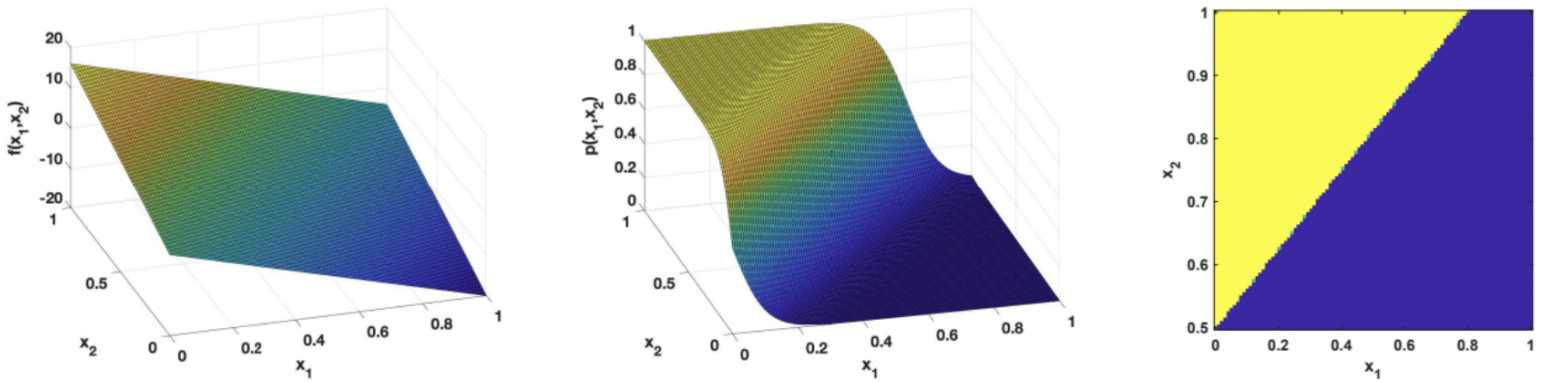
Logistic regression: Picture

© 2026 Fessler

- f is linear:

$$f(\boldsymbol{x}) = -20x_1 + 16x_2$$

$f(\boldsymbol{x})$ | $p(\boldsymbol{x}) = \sigma(f(\boldsymbol{x}))$ | classification regions



Above, "f" is our $\tilde{f}(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + b$
and $\sigma(\cdot)$ is the **standard logistic function**.

Figure courtesy Justin Romberg

- Given training data $\mathcal{D} = ((\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n))$, we must estimate the model parameter vector

$$\boldsymbol{\theta} = \begin{bmatrix} b \\ \boldsymbol{w} \end{bmatrix}.$$

- We need a criterion that quantifies how well

$$\eta(\boldsymbol{x}; \boldsymbol{\theta}) = \frac{1}{1 + \mathrm{e}^{-(\boldsymbol{w}^\top \boldsymbol{x} + b)}}$$

explains the training data for any candidate $\boldsymbol{\theta}$.

- Let $p(y \mid \boldsymbol{x}; \boldsymbol{\theta}) \triangleq \mathsf{P}\{Y = y \mid \boldsymbol{X} = \boldsymbol{x}; \boldsymbol{\theta}\}$ denote the conditional probability that $Y = y$ given $\boldsymbol{X} = \boldsymbol{x}$.
- By definition, and writing without braces using $y \in \{0, 1\}$:

$$p(y \mid \boldsymbol{x}; \boldsymbol{\theta}) = \begin{cases} \eta(\boldsymbol{x}; \boldsymbol{\theta}), & y = 1 \\ 1 - \eta(\boldsymbol{x}; \boldsymbol{\theta}), & y = 0 \end{cases}$$
$$= \eta(\boldsymbol{x}; \boldsymbol{\theta})^y \left(1 - \eta(\boldsymbol{x}; \boldsymbol{\theta})\right)^{1-y}.$$

- The **likelihood** of $\boldsymbol{\theta}$ is

$$L(\boldsymbol{\theta}) \triangleq \mathsf{p}(\mathcal{D}_{\text{train}}; \boldsymbol{\theta}) = \underbrace{\prod_{i=1}^{n} p(y_i \mid \boldsymbol{x}_i; \boldsymbol{\theta})}_{\text{independence}}$$
$$= \prod_{i=1}^{n} \eta(\boldsymbol{x}_i; \boldsymbol{\theta})^{y_i} \left(1 - \eta(\boldsymbol{x}_i; \boldsymbol{\theta})\right)^{1-y_i}.$$

- Notation (and rationale for the term "linear" vs affine):

$$\bar{\boldsymbol{x}}_i = \begin{bmatrix} 1 \\ \boldsymbol{x}_i \end{bmatrix} = [1 \; x_{i1} \; \cdots \; x_{id}]^\top$$
$$\boldsymbol{\theta} = \begin{bmatrix} b \\ \boldsymbol{w} \end{bmatrix} = [b \; w_1 \; \cdots \; w_d]^\top$$
$$\implies \underbrace{\tilde{f}(\boldsymbol{x}_i; \boldsymbol{\theta}) = \boldsymbol{w}^\top \boldsymbol{x}_i + b = \boldsymbol{\theta}^\top \bar{\boldsymbol{x}}_i}_{\text{discriminant}}.$$

- The **log-likelihood** of $\boldsymbol{\theta}$ is

$$\begin{aligned} \ell(\boldsymbol{\theta}) &\triangleq \log L(\boldsymbol{\theta}) \\ &= \sum_{i=1}^{n} \left[y_i \log\left(\frac{1}{1 + \mathrm{e}^{-\boldsymbol{\theta}^\top \bar{\boldsymbol{x}}_i}} \right) + (1 - y_i) \log\left(\frac{\mathrm{e}^{-\boldsymbol{\theta}^\top \bar{\boldsymbol{x}}_i}}{1 + e^{-\boldsymbol{\theta}^\top \bar{\boldsymbol{x}}_i}} \right) \right] \\ &= \sum_{i=1}^{n} \left[y_i \log\Big(\sigma(\tilde{f}(\boldsymbol{x}_i; \boldsymbol{\theta})) \Big) + (1 - y_i) \log\Big(1 - \sigma(\tilde{f}(\boldsymbol{x}_i; \boldsymbol{\theta})) \Big) \right]. \end{aligned}$$

- If the parameter dimension d is small, especially relative to the number of samples n, we can compute the estimate $\hat{\boldsymbol{\theta}}$ by maximizing the log-likelihood (*i.e.*, minimizing the negative log-likelihood):

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta} \in \mathbb{R}^{d+1}}{\arg\min} -\frac{1}{n} \ell(\boldsymbol{\theta}).$$

- Dividing by n aligns with ERM formulation.
- A HW problem writes it explicitly as a ERM with a surrogate loss function.
- This is a **convex** optimization problem (HW), but there is no closed-form solution, so iterative unconstrained optimization methods are required.

CQ: Logistic

© 2026 Fessler

Q09.1 Logistic regression (as formulated here) is:

A: **generative** B: **discriminative** C: neither D: both

Q09.2 Logistic regression (as formulated here) is:

A: **parametric** B: **nonparametric** C: neither D: both

- Unless $n \gg d$, it is typically preferable to minimize a **regularized** cost function like:

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta} \in \mathbb{R}^{d+1}}{\arg\min}\, J(\boldsymbol{\theta}), \quad J(\boldsymbol{\theta}) \triangleq \underbrace{-\frac{1}{n}\ell(\boldsymbol{\theta})}_{\text{empirical risk}} + \frac{\lambda}{2}\|\boldsymbol{\theta}\|_2^2, \quad \boldsymbol{\theta} = \begin{bmatrix} b \\ \boldsymbol{w} \end{bmatrix},$$

  where $\lambda > 0$ is a used-specified constant called a **regularization parameter**.

- Why introduce the regularization term?
  - If $n < d$, the Hessian $\nabla^2 \ell(\boldsymbol{\theta})$ will be singular.
  - $J(\boldsymbol{\theta})$ is **strictly convex** for $\lambda > 0$, so it has a unique minimizer. (HW)
  - The regularization term encourages the solution to have small norm, which can prevent **overfitting** to the training data - important when the sample size n is small relative to d.

Demo: JULIA & PYTHON

- No ellipses...
- Optimization via Quasi-Newton
- Decision boundary is

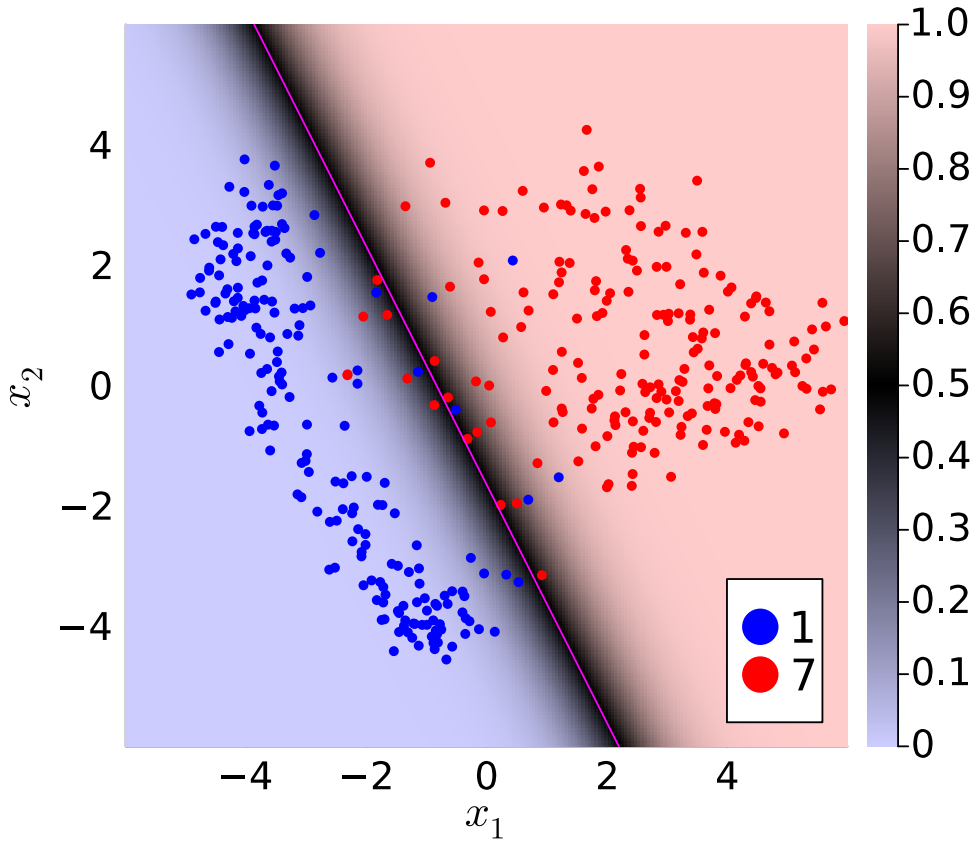
$$\{\boldsymbol{x} : \boldsymbol{w}^\top \boldsymbol{x} + b = 0\}$$

so (for $d = 2$):

$$x_2 = \frac{-(b + w_1 x_1)}{w_2}.$$

Here

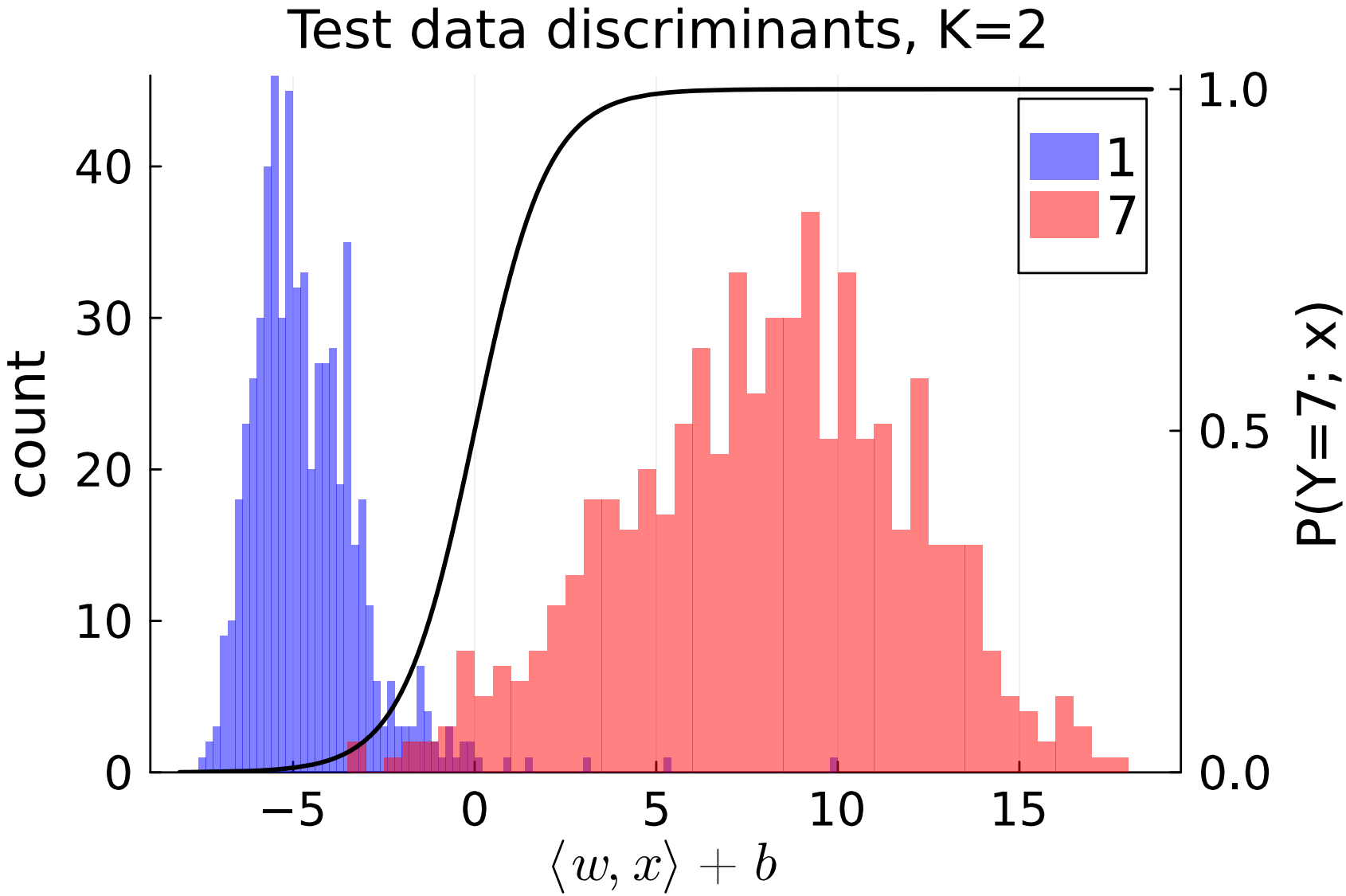
$$\hat{\boldsymbol{\theta}} = [1.76 \; 2.12 \; 1.07]^\top.$$



Color: predicted posterior prob's: $\hat{\eta}(\boldsymbol{x})$.
This is the "regression" in the name.

Discriminant histograms

© 2026 Fessler



CQ: Newton for logistic regression

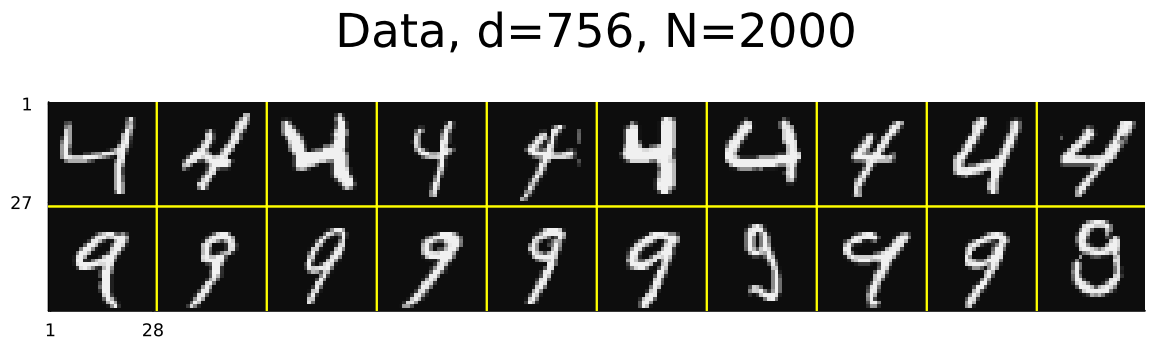
© 2026 Fessler

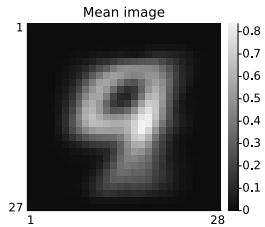
Q09.3 For this demo, what is the size of the Hessian matrix?

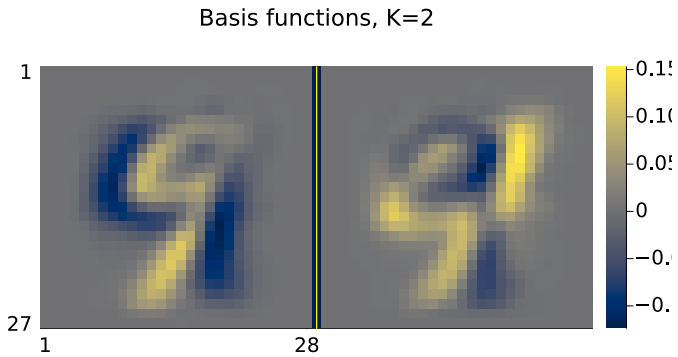
A: 1×1 B: 2×2 C: 3×3 D: 4×4 E: None of these

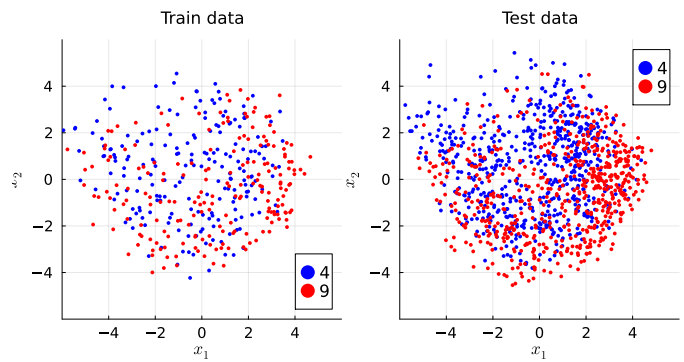
Harder case: Digits 4,9, $K = 2$

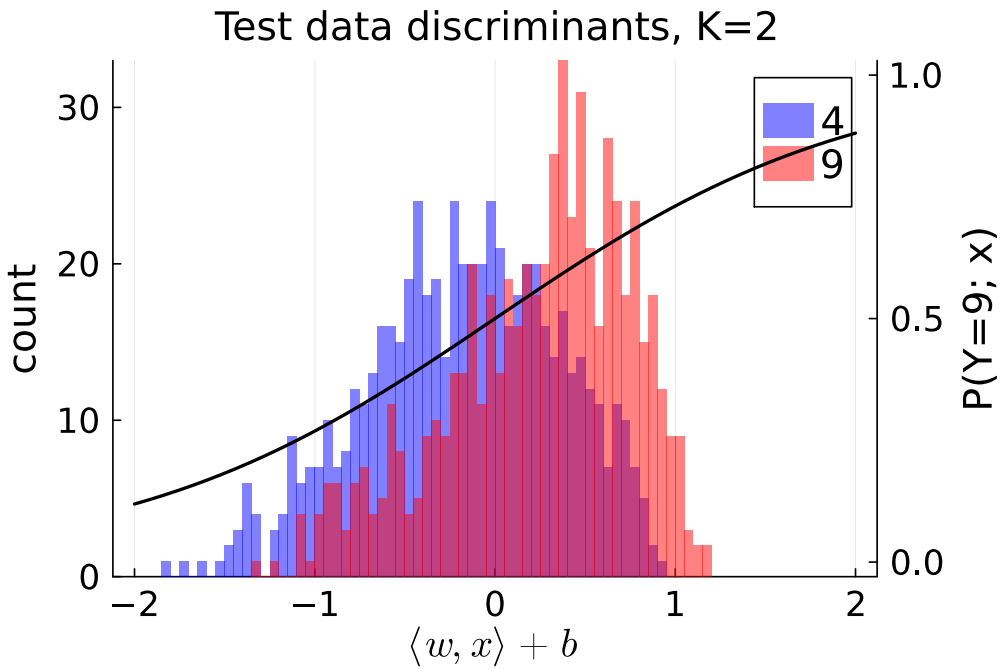
© 2026 Fessler





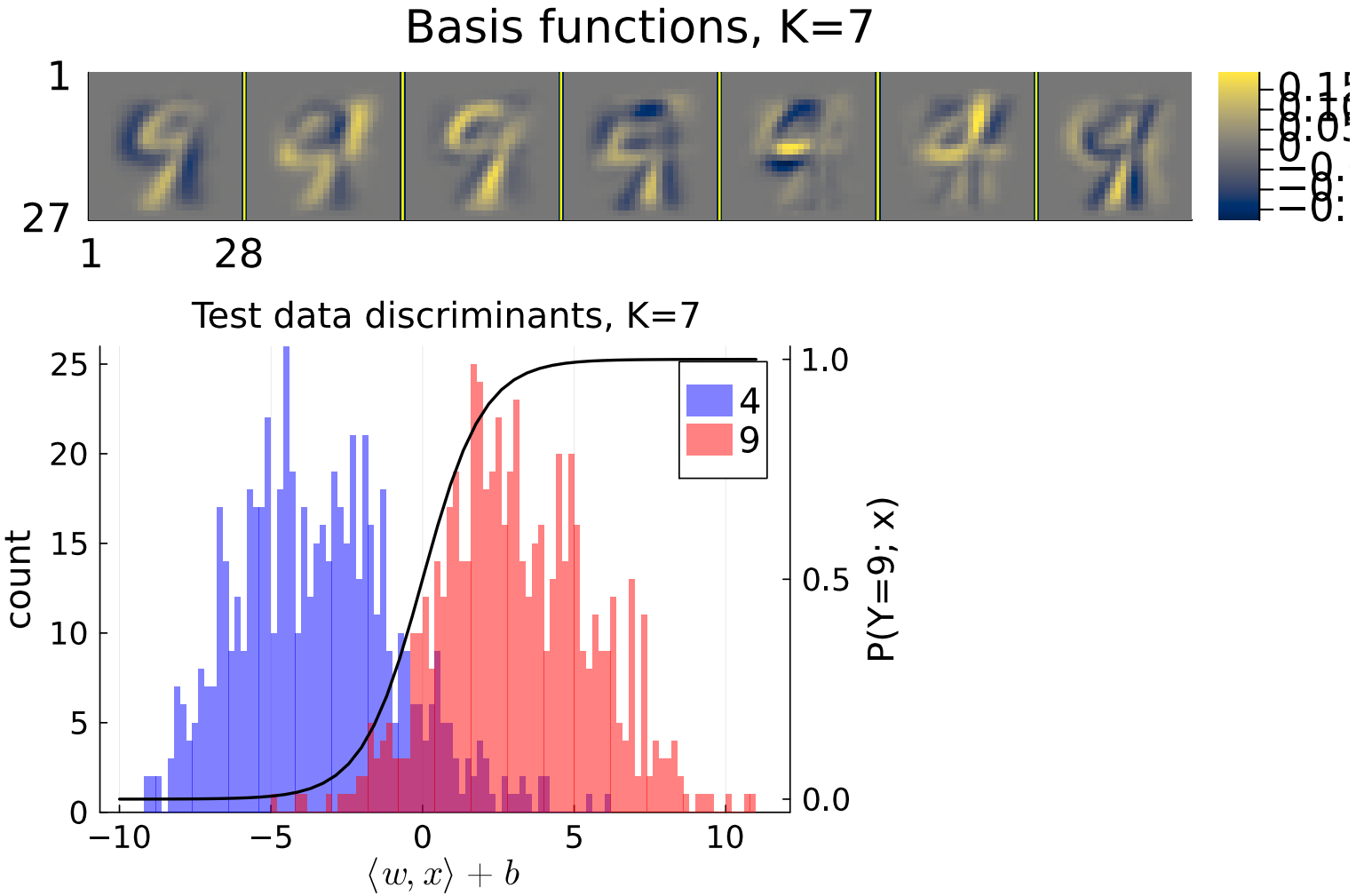






Harder case: Digits 4,9, $K = 7$

© 2026 Fessler



Working with original pixel-space images in HW5.

Hessian bound: binary case

© 2026 Fessler

- Fact (HW4).
  The Hessian of the negative log-likelihood is bounded:

$$\nabla^2(-\ell(\boldsymbol{\theta})) = \sum_{i=1}^{n} d_i \bar{\boldsymbol{x}}_i \bar{\boldsymbol{x}}_i^\top = \bar{\boldsymbol{X}}^\top \mathrm{Diag}(d_i) \bar{\boldsymbol{X}} \preceq d_{\max} \bar{\boldsymbol{X}}^\top \bar{\boldsymbol{X}}$$

  - d_i that depends on $\boldsymbol{x}_i$, $\boldsymbol{\theta}$
  - upper bound $d_{\max}$ is a small constant independent of $\boldsymbol{x}$ and $\boldsymbol{\theta}$.
- Precompute inverse of Hessian upper bound (accounting for $1/n$ and regularizer too) for a MM method

- Recall the negative **log-likelihood** of $\boldsymbol{\theta}$:

$$-\ell(\boldsymbol{\theta}) = -\sum_{i=1}^{n} \left[y_i \log\Big(\sigma(\tilde{f}(\boldsymbol{x}_i;\boldsymbol{\theta}))\Big) + (1-y_i) \log\Big(1 - \sigma(\tilde{f}(\boldsymbol{x}_i;\boldsymbol{\theta}))\Big)\right]$$
$$= \sum_{i=1}^{n} H_{\text{ce}}\Big(y_i, \sigma(\tilde{f}(\boldsymbol{x}_i;\boldsymbol{\theta}))\Big).$$

- **Cross-entropy** loss between two probability distributions:

$$0 \le H_{\text{ce}}(p,q) = \underbrace{-\mathbb{E}_p\{\log q\}}_{\text{general}} = \underbrace{-\sum_{x \in \mathcal{X}} p(x) \log q(x)}_{\text{discrete}}$$
$$= \underbrace{-[p \log q + (1-p)\log(1-q)]}_{\text{binary } \{0,1\}}$$

- Treat $0 \log 0 = 0$.
- This formulation readily extends to $K > 2$ classification.

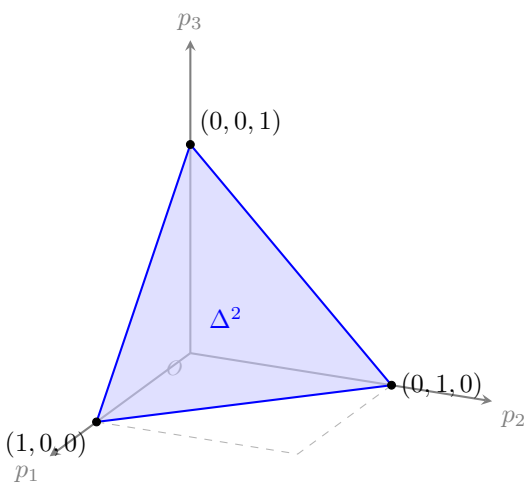
Softmax function (for multiclass)

© 2026 Fessler

- The **softmax function** transforms any vector $\boldsymbol{a} \in \mathbb{R}^K$ to a probability vector on the **unit simplex** Δ^{K-1} in $\mathbb{R}^K$ via:

$$\text{softmax}(\boldsymbol{a}) \triangleq \frac{1}{\sum_k \mathrm{e}^{a_k}} \begin{bmatrix} \mathrm{e}^{a_1} & \dots & \mathrm{e}^{a_K} \end{bmatrix}^\top \in \Delta^{K-1} \subset [0,1]^K \subset \mathbb{R}^K.$$

- Standard 2-simplex for $K = 3$: (TixZ code via Gemini)



- $\Delta^2 = \{\boldsymbol{p} \in \mathbb{R}^3 : p_1 + p_2 + p_3 = 1,\ p_k \geq 0\}$
- Akin to a "smooth argmax"
- Example: $\boldsymbol{a} = [-0.3, 6.1, 2.7, 0.8] \implies$ $\text{softmax}(\boldsymbol{a}) \approx [0.002, 0.96, 0.03, 0.005]$.
- One-hot argmax would be [0, 1, 0, 0].

Q09.4 The softmax function works element-wise?
A: True B: False **??**

- Use the numerically **safe version** of **softmax**:

$$\text{softmax}(\boldsymbol{a}) = \frac{1}{\sum_k \mathrm{e}^{a_k - m}} \begin{bmatrix} \mathrm{e}^{a_1 - m} & \dots & \mathrm{e}^{a_K - m} \end{bmatrix}^\top, \quad m = \max_j a_j$$

- Differentiable approximation to one-hot argmax
(hard threshold in binary case)
- Used in many (most?) classification networks, including the **attention** mechanism of **transformer** architectures (more later)
- Used for **multiclass logistic regression**.

- **Multinomial logistic regression** [1, §10.3] uses **categorical distribution** and softmax of affine functions to model posteriors:

$$\eta_k(\boldsymbol{x}) = \mathsf{P}\{Y = k \mid \boldsymbol{X} = \boldsymbol{x}; \boldsymbol{\theta}\}$$

$$= \mathsf{Cat}(Y = k \mid \mathsf{softmax}(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b})) = \frac{\mathrm{e}^{\boldsymbol{w}_k^\top \boldsymbol{x} + b_k}}{\sum_j \mathrm{e}^{\boldsymbol{w}_j^\top \boldsymbol{x} + b_j}}.$$

- Here $\boldsymbol{W} \in \mathbb{R}^{K \times d}$ and $\boldsymbol{b} \in \mathbb{R}^K$ so $\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b} \in \mathbb{R}^K$.
- The scaled **negative log-likelihood** has the **ERM** form with **cross-entropy** as the surrogate loss:

$$-\frac{1}{n}\ell(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n} H_{\mathsf{ce}}(\boldsymbol{y}_i, \mathsf{softmax}(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b})).$$

- $\boldsymbol{y}_i \in \{0,1\}^K$ is the **one-hot** encoding of the true label y_i, where $[\boldsymbol{y}_i]_k = \mathbb{I}_{\{y_i = k\}}$ for $k = 1, \ldots, K$.

- When d is large, it can be more **interpretable** to select a subset of the features, instead of using all d features.
- Ideally we might solve a **feature selection** problem such as

$$\arg\min_{\boldsymbol{\theta} \in \mathbb{R}^{d+1}} -\ell(\boldsymbol{\theta}) \text{ s.t. } \|\boldsymbol{\theta}\|_0 \leq N$$

  for some user-selected (or cross-validation selected) $N \in \mathbb{N}$, where $\|\boldsymbol{x}\|_0 = \sum_i \mathbb{I}_{\{x_i \neq 0\}}$.
- $\|\cdot\|_0$ is discontinuous and nonconvex, making optimization very difficult
- A practical alternative is to use a 1-norm regularizer:

$$J(\boldsymbol{\theta}) = -\frac{1}{n}\ell(\boldsymbol{\theta}) + \lambda \|\boldsymbol{\theta}\|_1 .$$

  This cost function is **convex**, though it is non-smooth, which motivates advanced optimization approaches like accelerated **proximal gradient** methods [2].

Logistic regression: Summary

© 2026 Fessler

- More than a classifier – it predicts the probability of each class
- LR assumption is less restrictive than LDA assumption (but if data are Gaussian, LDA is more powerful)
- Widely used in application domains including health sciences
- Example: predict whether a patient will develop a disease (*e.g.*, diabetes, coronary disease) based on various attributes (age, sex, weight, BMI, etc.)
- For imbalanced classes, may want to use weighted ERM.
- Multiclass extension "easy" using one-hot encoding and cross-entropy loss
- Nonlinear extension using nonlinear discriminant h:

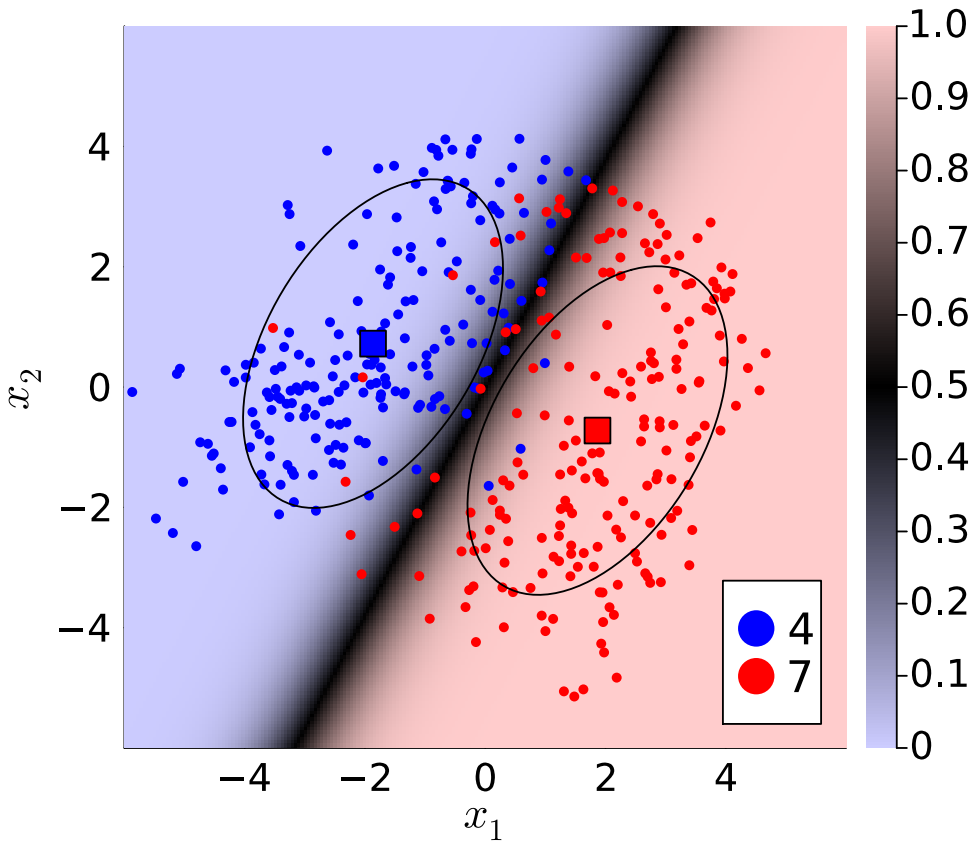
$$\eta(\boldsymbol{x} \mid \boldsymbol{\theta}) = \frac{1}{1 + \exp(-h(\boldsymbol{x}; \boldsymbol{\theta}))}$$

We will see examples later (kernel methods, neural networks)

For LDA we can also visualize the plug-in posterior prob's:

$$\hat{\eta}_k(\boldsymbol{x}) = \mathsf{P}\left\{ Y = k \mid \boldsymbol{X} = \boldsymbol{x}; \hat{\boldsymbol{\theta}} \right\} = \frac{\hat{\pi}_k \hat{g}_k(\boldsymbol{x}; \hat{\boldsymbol{\theta}})}{\sum_j \hat{\pi}_j \hat{g}_j(\boldsymbol{x}; \hat{\boldsymbol{\theta}})}.$$

LDA train error=6.0 %, test error = 5.75 %



CQ: SGD

© 2026 Fessler

For $J(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^{n} J_i(\boldsymbol{\theta})$, the SGD algorithm is as follows:

- Initialize $\boldsymbol{\theta}_0$; set $t \leftarrow 0$
- For $j = 1, \ldots,$ `max_iter`
  For $i = 1, \ldots, n$ *in random order*
  $\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} -$ _____
  $t \leftarrow t + 1$
  End
  If convergence condition satisfied, exit
  End

Q09.5 Fill in the blank:

A: $\alpha \nabla J(\boldsymbol{\theta}_{t-1})$ B: $\frac{1}{n} \nabla J_i(\boldsymbol{\theta}_{t-1})$ C: $\alpha_j \nabla J_k(\boldsymbol{\theta}_t)$

D: $\alpha_j \nabla J_i(\boldsymbol{\theta}_{t-1})$ E: none

Error probability via total probability:

$$\mathrm{P}\{f(\boldsymbol{X}) \neq Y\} = \sum_k \mathrm{P}\{f(\boldsymbol{X}) \neq k \mid Y = k\} \underbrace{\mathrm{P}\{Y = k\}}_{\pi_k}.$$

Example.
10 patients with cancer (+1)
990 patients without cancer (-1)
Suppose defective cancer test always says $f(\boldsymbol{X}) = -1$ ("no cancer"). It's risk (error probability) is:

$$\begin{aligned}\mathrm{P}\{f(\boldsymbol{X}) \neq Y\} &= \mathrm{P}\{f(\boldsymbol{X}) \neq +1 \mid Y = +1\} \mathrm{P}\{Y = +1\} \\ &\quad + \mathrm{P}\{f(\boldsymbol{X}) \neq -1 \mid Y = -1\} \mathrm{P}\{Y = -1\} \\ &= 0 \cdot \frac{10}{1000} + 1 \cdot \frac{990}{1000} = \frac{1}{100}.\end{aligned}$$

- One way to avoid this problem is **balanced accuracy** (or error).
- Balanced error is average of errors for each class:

$$\text{BE} = \frac{1}{K}\left(\sum_k \text{P}\{f(\boldsymbol{X}) \neq k \mid Y = k\}\right)$$

- For preceding example:

$$\frac{1}{2}(\text{P}\{f(\boldsymbol{X}) \neq +1 \mid Y = +1\} + \text{P}\{f(\boldsymbol{X}) \neq -1 \mid Y = -1\})$$
$$= \frac{1}{2}(1 + 0) = \frac{1}{2}.$$

- Exercise. Define a loss $L(y, \hat{y})$ corresponding to balanced error. Hint. It is not the 0-1 loss.

Classifier roadmap

© 2026 Fessler

- Full Bayesian methods
  - generative models $g_k(\boldsymbol{x}) = \mathrm{p}(\boldsymbol{x} \mid Y = k)$
  - use Bayes rule to determine the posterior $\eta_k(\boldsymbol{x}) = \mathrm{P}\{Y = k \mid \boldsymbol{X} = \boldsymbol{x}\} \propto \pi_k g_k(\boldsymbol{x})$
  - decision regions based on posterior $\arg\max_k \eta_k(\boldsymbol{x}) = \arg\max_k \pi_k g_k(\boldsymbol{x})$
- Logistic regression (and some other discriminative methods)
  - directly model the posterior $\eta_k(\boldsymbol{x}; \boldsymbol{\theta}) = \mathrm{P}\{Y = k \mid \boldsymbol{X} = \boldsymbol{x}; \boldsymbol{\theta}\}$
  - decision regions based on posterior $\arg\max_k \eta_k(\boldsymbol{x})$
- Could we **cut to the chase** and directly learn the decision regions from training data?

Drawback of plug-in classifiers

© 2026 Fessler

Plug-in methods require estimation of distributions or conditional distributions, which can be more difficult than estimating a

CQ: Linear classifiers

© 2026 Fessler

- Binary classification
- Training data $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)$.
- Labels are ± 1

Q09.6 In this setting, a "linear" classifier has the form

A: $\text{sign}(\boldsymbol{w}^\top \boldsymbol{x} + b)$

B: $\boldsymbol{w}^\top \boldsymbol{x} + b$

C: $\boldsymbol{x}^\top \boldsymbol{H} \boldsymbol{x} + \boldsymbol{x}^\top \boldsymbol{b} + c$

D: $\arg\max_k \boldsymbol{w}_k^\top \boldsymbol{x}$

E: none of these

Hyperplanes: Definition

© 2026 Fessler

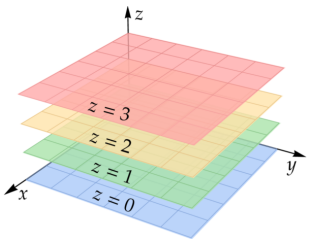


Figure 1: The xy-plane and several other horizontal planes.

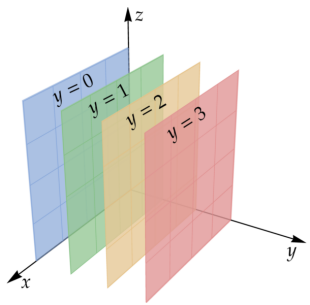


Figure 2: The xz-plane and several parallel planes.

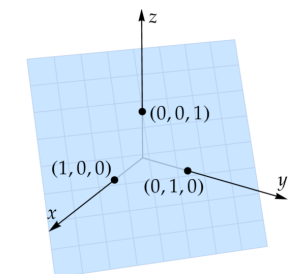


Figure 3: The plane $x + y + z = 1$.

Decision boundaries are often **hyperplanes** so we focus on those first.

- A **hyperplane** is a subset of $\mathbb{R}^d$ of the form

$$\mathcal{H} =$$

for some $\boldsymbol{w} \in \mathbb{R}^d$, $b \in \mathbb{R}$.

- In general, a hyperplane in $\mathbb{R}^d$ is

- The vector $\boldsymbol{w}$ is **orthogonal** to the hyperplane, and for this reason is called a

Figure courtesy [3].

A **hyperplane** is a subset of $\mathbb{R}^d$ of the form

$$\mathcal{H} = \{\boldsymbol{x} \in \mathbb{R}^d : \boldsymbol{w}^\top \boldsymbol{x} + b = 0\}, \quad \text{for some given } \boldsymbol{w} \in \mathbb{R}^d,\ b \in \mathbb{R} \ .$$

Hyperplanes: Unique representation?

© 2026 Fessler

For some given $\boldsymbol{w} \in \mathbb{R}^d$, $b \in \mathbb{R}$, define the hyperplane $\mathcal{H} \triangleq \{\boldsymbol{x} \in \mathbb{R}^d : \boldsymbol{w}^\top \boldsymbol{x} + b = 0\}$.

Q09.7 Is the representation unique? If any $\tilde{\boldsymbol{w}} \in \mathbb{R}^d$ and $\tilde{b} \in \mathbb{R}$ represent the same hyperplane as $(\boldsymbol{w}, b)$, then $\tilde{\boldsymbol{w}} = \boldsymbol{w}$ and $\tilde{b} = b$.

A: True B: False **??**

- Distances to hyperplanes are crucial for classification problems.
- Find the distance from $\boldsymbol{z} \in \mathbb{R}^d$ to $\mathcal{H} \triangleq \{\boldsymbol{x} \in \mathbb{R}^d : \boldsymbol{w}^\top \boldsymbol{x} + b = 0\}$.
- The point closest to $\boldsymbol{z}$ on the **hyperplane** $\mathcal{H}$ can be written as

- Thus the distance is

Separating hyperplanes: Definition

© 2026 Fessler

- Let $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)$ be training data with $\boldsymbol{x}_i \in \mathbb{R}^d$ for a **binary classification** problem.
- Assume $y_i \in \{-1, 1\}$.
- We say the training data are
  iff there exist $\boldsymbol{w} \in \mathbb{R}^d$, $b \in \mathbb{R}$ such that

- In this case we call the following set a

$$\mathcal{H} \triangleq \{\boldsymbol{x} \in \mathbb{R}^d : \boldsymbol{w}^\top \boldsymbol{x} + b = 0\}.$$

Separating hyperplanes: Choices

© 2026 Fessler

Are all **separating hyperplanes** equally good?

Max-margin hyperplanes: Definition

© 2026 Fessler

- The **margin** ρ of a **separating hyperplane** is the distance from the hyperplane to the nearest training point:

$$\rho(\boldsymbol{w}, b) \triangleq$$

- An **optimal separating hyperplane** or **maximum-margin hyperplane** uses the solution of

$$\hat{\boldsymbol{w}}, \hat{b} =$$

Max-margin hyperplane: Example

© 2026 Fessler

Q09.8 For $n > d$, the max-margin hyperplane $\mathcal{H}$ is always unique.

A: True B: False **??**

Separating hyperplane: Canonical form

© 2026 Fessler

- A **separating hyperplane** is said to be in **canonical form** iff $\boldsymbol{w}$ and b are such that

$$\begin{cases} \geq 1, & \forall i \\ = 1, & \text{for some } i. \end{cases}$$

- Every separating hyperplane can be expressed in canonical form.
- Suppose $\mathcal{H} = \{\boldsymbol{x} : \boldsymbol{w}_1^\top \boldsymbol{x} + b_1 = 0\}$ is a separating hyperplane (not necessarily in canonical form). Let

$$m \triangleq$$

and define

$$\boldsymbol{w}_2 = \frac{\boldsymbol{w}_1}{m}, \quad b_2 = \frac{b_1}{m}.$$

Then $\boldsymbol{w}_2$, b_2 express $\mathcal{H}$ in canonical form.

Canonical form hyperplanes

© 2026 Fessler

- Thus we can define the max-margin hyperplane using

$$\begin{aligned} \arg\max_{\boldsymbol{w},b} \quad & \min_{i=1,\dots,n} \frac{|\boldsymbol{w}^\top \boldsymbol{x}_i + b|}{\|\boldsymbol{w}\|_2} \\ \text{s.t.} \ \forall i \quad & y_i \left(\boldsymbol{w}^\top \boldsymbol{x}_i + b\right) \geq 1 \\ \exists i \quad & y_i \left(\boldsymbol{w}^\top \boldsymbol{x}_i + b\right) = 1. \end{aligned}$$

- Equivalently:

- This is an example of a **quadratic program.**
-

What if the data are not separable?

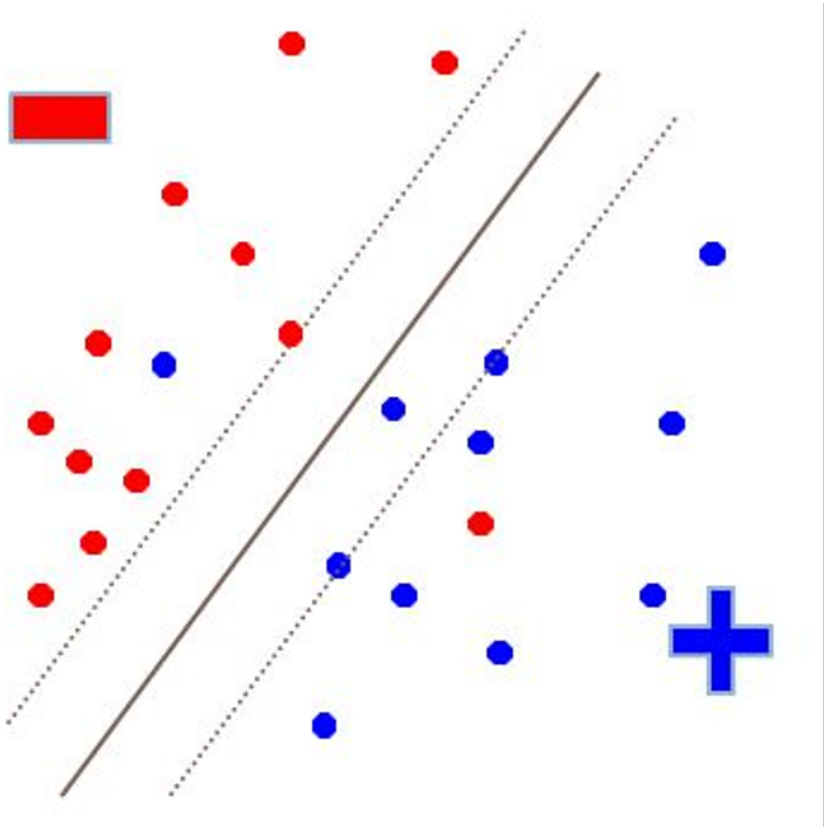


Figure courtesy Clay Scott.

Optimal soft-margin hyperplane: Prelude

© 2026 Fessler

- Introduce **slack variables** $\xi_1, \ldots, \xi_n \geq 0$.
  Greek letter ξ is pronounced "ksee."
- The **optimal soft-margin hyperplane** uses the solution of

$$\arg\min_{\boldsymbol{w}, b} \min_{\boldsymbol{\xi}} \frac{1}{2} \|\boldsymbol{w}\|_2^2 + \qquad$$

$$\text{s.t. } y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq \qquad \forall i = 1, \ldots, n$$

$$\forall i = 1, \ldots, n$$

- C is a user-defined parameter
- This is another **quadratic program**
- The OSM hyperplane is a special case of a
- How do we solve such *constrained* optimization problems?

References I

[1] K. P. Murphy.
Probabilistic machine learning: An introduction.
MIT Press, 2022.

[2] D. Kim and J. A. Fessler.
Adaptive restart of the optimized gradient method for convex optimization.
J. Optim. Theory Appl., 178(1):240–63, July 2018.

[3] Jim Belk.
8.2 planes and hyperplanes, 2021.
https://e.math.cornell.edu/people/belk/writing/LinearODEs8.pdf.

[4] C. Bishop.
Pattern recognition and machine learning.
Springer, 2006.

[5] T. Serafini, G. Zanghirati, and L. Zanni.
Gradient projection methods for quadratic programs and applications in training support vector machines.
Optimization Methods and Software, 20(2-3):353–78, 2005.

[6] R. O. Duda, P. E. Hart, and D. G. Stork.
Pattern classification.
Wiley, New York, 2001.

[7] Onmyphd.
Karush–kuhn–tucker conditions inequality constraint diagram, 2013.
http://www.onmyphd.com/?p=kkt.karush.kuhn.tucker, CC BY-SA 3.0,
https://commons.wikimedia.org/w/index.php?curid=27331272.

[8] Stephen Boyd and Lieven Vandenberghe.
Convex Optimization.
Cambridge University Press, 2004.

References II

[9] Emmanuel Candes and Benjamin Recht.
Exact matrix completion via convex optimization.
Communications of the ACM, 55(6):111–119, 2012.

[10] David Gross.
Recovering low-rank matrices from few coefficients in any basis.
IEEE Transactions on Information Theory, 57(3):1548–1566, 2011.

[11] Ali Jalali, Yudong Chen, Sujay Sanghavi, and Huan Xu.
Clustering partially observed graphs via convex optimization.
In *ICML*, 2011.

[12] Paul Hand.
Conditions for dual certificate existence in semidefinite rank-1 matrix recovery.
In *Proceedings of the 10th International Conference on Sampling Theory and Applications*, 2013.

[13] Xiaodong Li and Vladislav Voroninski.
Sparse signal recovery from quadratic measurements via convex programming.
SIAM Journal on Mathematical Analysis, 45(5):3019–3033, 2013.

[14] Qiuwei Li, Ashley Prater, Lixin Shen, and Gongguo Tang.
Overcomplete tensor decomposition via convex optimization.
In *2015 IEEE 6th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pages 53–56. IEEE, 2015.

[15] Stanley Osher and Wotao Yin.
Sparse recovery via ℓ_1 and l_1 optimization.
Notices of the International Congress of Chinese Mathematicians, 3(1):4–10, 2015.

[16] Carlos Fernandez-Granda.
Super-resolution of point sources via convex programming.
Information and Inference: A Journal of the IMA, 5(3):251–303, 2016.

References III

[17] Takayuki Iguchi, Dustin G Mixon, Jesse Peterson, and Soledad Villar.
Probably certifiably correct k-means clustering.
Mathematical Programming, 165(2):605–642, 2017.

[18] Chong You, Zhihui Zhu, Qing Qu, and Yi Ma.
Robust recovery via implicit bias of discrepant learning rates for double over-parameterization.
Advances in Neural Information Processing Systems, 33:17733–17744, 2020.

[19] Shunta Akiyama and Taiji Suzuki.
On learnability via gradient method for two-layer relu neural networks in teacher-student setting.
In *International Conference on Machine Learning*, pages 152–162. PMLR, 2021.

[20] Kyle Gilman, Sam Burer, and Laura Balzano.
A semidefinite relaxation for sums of heterogeneous quadratics on the stiefel manifold.
arXiv preprint arXiv:2205.13653, 2022.

[21] V. Piccialli, J. Schwiddessen, and A. M. Sudoso.
Optimization meets machine learning: an exact algorithm for semi-supervised support vector machines.
mathpro, 217(1):79–121, 2026.