EECS 553 Lecture 05
Machine Learning (ECE)
[class version]

Jeff Fessler

University of Michigan
©Laura Balzano and Jeff Fessler, UM, September 12, 2026, 20:12

F26

Logistics

© 2026 Fessler

- Groups and first paper ideas due Friday, 1/30
  - Canvas →People →Project Groups (self sign-in)
  - Must create group to submit (1 per group) first doc as described in l04 (no template)
- HW 2 due Friday 9/18

Recap / Outline

© 2026 Fessler

- Recap
  - Quadratic discriminant analysis (QDA)
  - Naïve Bayes plug-in classifier
- Today
  - Regression
    - Minimum MSE estimator
    - Generative approach
    - Plug-in regression
    - Discriminative approach (OLS)
    - Regularized least squares
  - Empirical risk minimization (ERM)
    - Surrogate losses
  - Optimization preview
    - Classifier design
    - Large-scale regression

- Regression and classification are the two classical **supervised learning** problems.
- Every feature vector $\boldsymbol{x}$ is associated to a response variable Y, and the goal is to predict Y from $\boldsymbol{x}$, *i.e.*, $\hat{y} = f(\boldsymbol{x}) \approx Y$.
- As in classification, this prediction function f must be learned from training data.

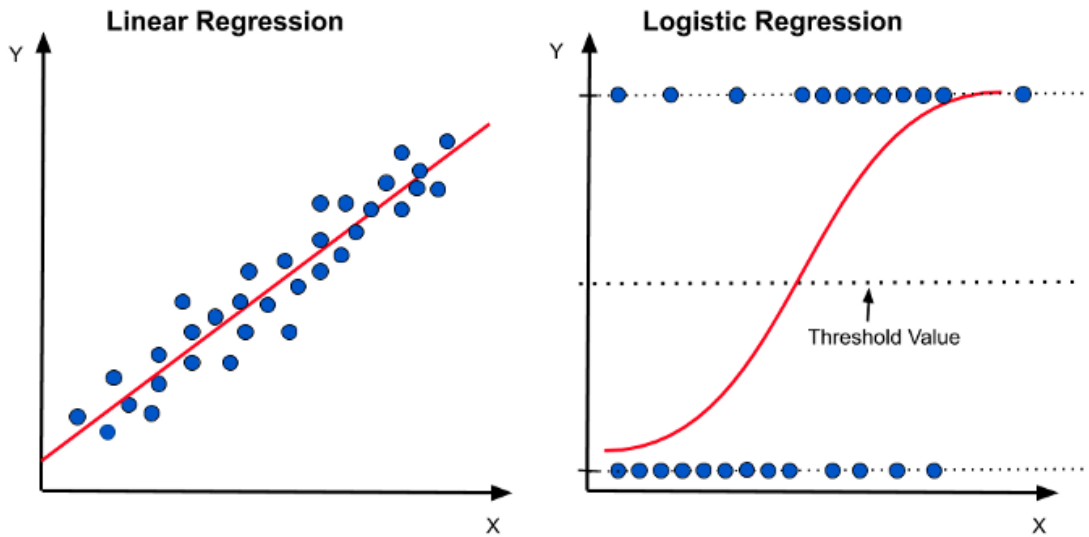


Figure courtesy ai.plainenglish.io

Regression setting

© 2026 Fessler

- The main difference between **classification** and **regression** is

$$Y \in$$

  - Classification involves **categorical** variables (labels).
  - Regression typically involves **continuous** variables (real numbers).
  - Another important type in data science is **ordinal data**.
- Probabilistic formulations assume $(\boldsymbol{X}, Y)$ are jointly distributed random variables with some $\mathrm{p}(\boldsymbol{x}, y)$ [1, §11.2.3.5].
- The goal is to predict the response or "label" $Y \in \mathbb{R}$ from features $\boldsymbol{X} \in \mathbb{R}^d$ using a **regression function**

$$f : \mathbb{R}^d \mapsto \mathbb{R}.$$

- We use different performance measures for regression because the response Y is "continuous."
- The 0-1 loss would be
  Typically $\mathrm{P}\{f(\boldsymbol{X}) = Y\} = 0$.
- In regression problems, the most common **loss** function is **squared error**, also called **quadratic loss**:

$$L(y, \hat{y}) = (\hat{y} - y)^2.$$

- The corresponding **risk** is the **mean squared error** (**MSE**):

$$R(f) \triangleq$$

- Here "mean" refers to the expectation over $\mathrm{p}(\boldsymbol{x}, y)$, *i.e.*, the

Conditional mean predictor

© 2026 Fessler

- Just like classification, there is a regression function f_* that achieves the minimum value R_* of the MSE.
- **Theorem:** The function that minimizes the

$$f_*(\boldsymbol{x}) \triangleq$$

  Proof in Prof. Scott's lecture notes on Canvas.
- This function is called the **conditional mean** predictor or minimum MSE (**MMSE**) estimator.
- Awesome theorem in theory.
  - MMSE perspective is useful for modern diffusion generative models, *e.g.*, [2, 3].
- Challenges in practice:
  - The expectation/integral can be difficult except in special cases.
  - Often we do not know $\mathrm{p}(y \mid \boldsymbol{x})$.

- One approach is to assume $(\boldsymbol{X}, Y)$ have a joint multivariate Gaussian distribution:

$$\begin{bmatrix} \boldsymbol{X} \\ Y \end{bmatrix} \sim$$

where the **cross covariance** is $\boldsymbol{\Sigma}_{xy} = \mathbb{E}\{\boldsymbol{X} Y\} \in \mathbb{R}^d$.

- Under this (strong) Gaussian assumption, the MMSE estimator is analytically tractable [1, §11.2.3.5]:

$$f_*(\boldsymbol{x}) = \mathbb{E}\{Y \mid \boldsymbol{X} = \boldsymbol{x}\} = \mu_y + \boldsymbol{\Sigma}_{xy}^\top \boldsymbol{\Sigma}_{xx}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_x)$$

$$= \quad$$

$$\boldsymbol{w} \triangleq \boldsymbol{\Sigma}_{xx}^{-1}\boldsymbol{\Sigma}_{xy}, \quad b \triangleq \mu_y - \boldsymbol{\Sigma}_{xy}^\top \boldsymbol{\Sigma}_{xx}^{-1}\boldsymbol{\mu}_x$$

- The estimator is

- For a practical plug-in method, we estimate the means, the covariances, and the cross-covariances from training data:

$$\hat{\boldsymbol{\mu}}_x = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i, \qquad \hat{\mu}_y = \frac{1}{n}\sum_{i=1}^{n} y_i$$

$$\widehat{\boldsymbol{\Sigma}}_{xx} = \frac{1}{n}\sum_{i=1}^{n} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_x)(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_x)^\top$$

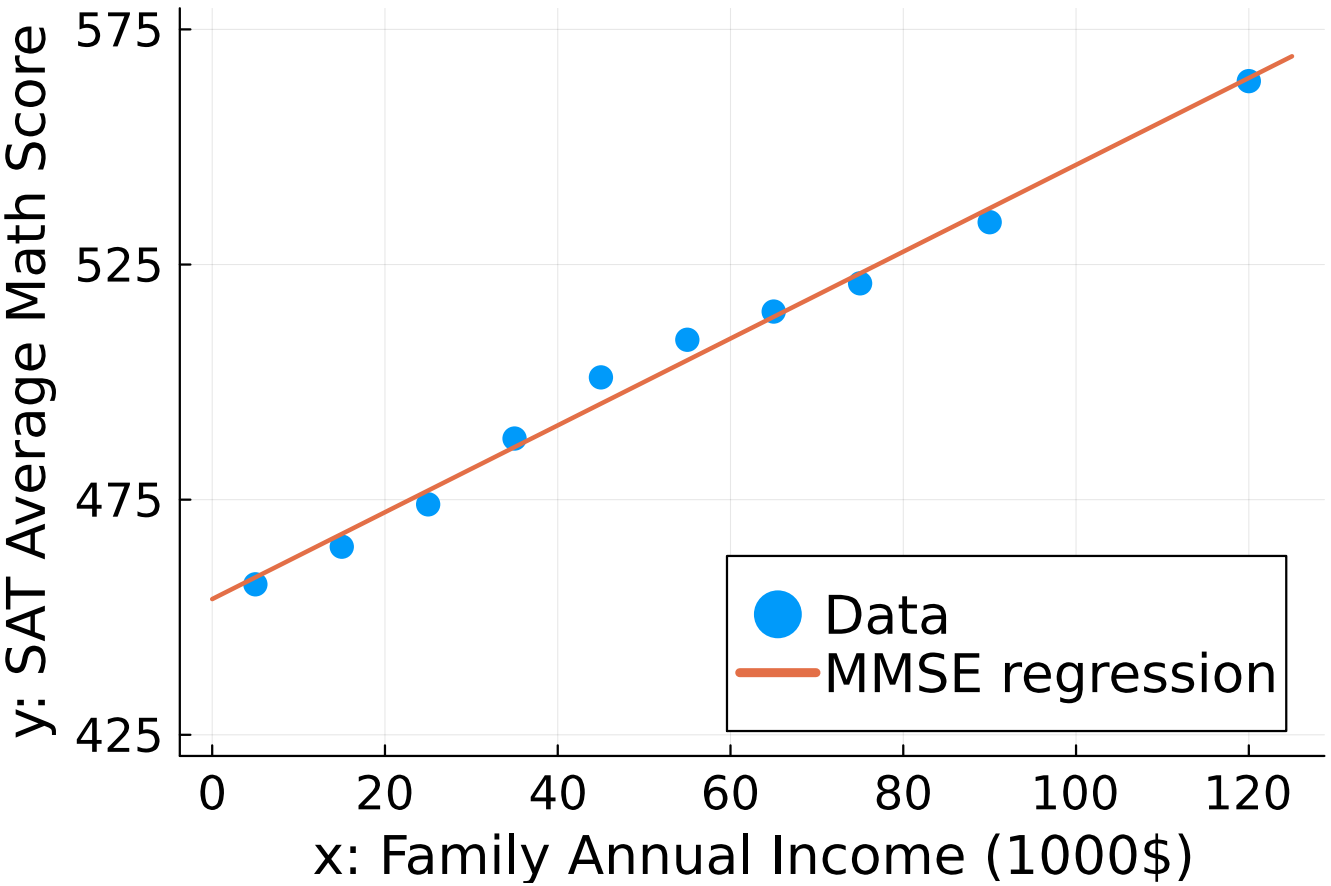
$$\widehat{\boldsymbol{\Sigma}}_{xy} = \frac{1}{n}\sum_{i=1}^{n}$$

- Then the plug-in estimator is $f(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + b$, $\boldsymbol{w} = \hat{\boldsymbol{\Sigma}}_{xx}^{-1} \hat{\boldsymbol{\Sigma}}_{xy}, \quad b = \hat{\mu}_y - \hat{\boldsymbol{\Sigma}}_{xy}^\top \hat{\boldsymbol{\Sigma}}_{xx}^{-1} \hat{\boldsymbol{\mu}}_x$
- Again need $n \geq d$ (at least) for invertibility of $\widehat{\boldsymbol{\Sigma}}_{xx}$.
- Can work well *if* joint Gaussian distribution is reasonable and if n is large enough.
- Does not scale well to large d.

MMSE regression example

© 2026 Fessler

MMSE regression demo using data from [4]



- In practice the joint distribution of $(\boldsymbol{X}, Y)$ is unknown, so we must estimate f_* using training data $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)$.
- Choose f to minimize the

$$\underbrace{R(f) = \mathbb{E}_{\boldsymbol{X},Y}\left\{(f(\boldsymbol{X}) - Y)^2\right\}}_{\text{ideal}} \overset{?}{\approx} \underbrace{\widehat{R}(f) =}_{\text{practical}}$$

- Here: "mean" in MSE refers to the
- To avoid overfitting, we must restrict f to belong to some **regression model**, *i.e.*,
- For now, we focus on the **linear model** (actually **affine**):

$$f(\boldsymbol{x}) = \qquad \text{where } \boldsymbol{w} \in \mathbb{R}^d, \ b \in \mathbb{R}.$$

- This is a (typical) **discriminative** approach to regression.

Define vectors and matrix:

$$\boldsymbol{y} \triangleq \qquad \bar{\boldsymbol{X}} \triangleq \qquad \boldsymbol{\theta} \triangleq$$

- The bar above $\bar{\boldsymbol{X}}$ reminds us that the first column is all ones.
- Minimizing MSE involves the following optimization problem known as **least-squares linear regression** or **ordinary least squares**:

$$\underset{b \in \mathbb{R},\, \boldsymbol{w} \in \mathbb{R}^d}{\arg\min} \; \frac{1}{n} \sum_{i=1}^{n} (\underbrace{\boldsymbol{w}^\top \boldsymbol{x}_i + b}_{} - y_i)^2 =$$

CQ: regression for tall

© 2026 Fessler

Q05.1 Assume $d+1 \leq n$ and $\bar{\boldsymbol{X}} \in \mathbb{R}^{n \times (d+1)}$ has full **rank**.
Then

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta} \in \mathbb{R}^{d+1}}{\arg\min} \frac{1}{n} \left\| \bar{\boldsymbol{X}} \boldsymbol{\theta} - \boldsymbol{y} \right\|_2^2 = \quad ?$$

A: $\bar{\boldsymbol{X}}^\top (\bar{\boldsymbol{X}} \bar{\boldsymbol{X}}^\top)^{-1} \boldsymbol{y}$

B: $\bar{\boldsymbol{X}}^{-1} \boldsymbol{y}$

C: $(\bar{\boldsymbol{X}}^\top \bar{\boldsymbol{X}})^{-1} \bar{\boldsymbol{X}}^\top \boldsymbol{y}$

D: $\bar{\boldsymbol{X}} (\bar{\boldsymbol{X}}^\top \bar{\boldsymbol{X}})^{-1} \bar{\boldsymbol{X}}^\top \boldsymbol{y}$

E: minimizer need not be unique

For $\bar{\boldsymbol{X}} \in \mathbb{R}^{n \times (d+1)}$ and $\boldsymbol{y} \in \mathbb{R}^n$, **ordinary least squares** (**OLS**) seeks the minimizer

$$\underset{\boldsymbol{\theta} \in \mathbb{R}^{d+1}}{\arg\min} \frac{1}{n} \left\| \bar{\boldsymbol{X}} \boldsymbol{\theta} - \boldsymbol{y} \right\|_2^2 .$$

Q05.2 What happens when $d \geq n$?

A: $\bar{\boldsymbol{X}} \bar{\boldsymbol{X}}^\top$ is never invertible, so we cannot compute any OLS solution

B: $\bar{\boldsymbol{X}}^\top \bar{\boldsymbol{X}}$ is never invertible, so the OLS solution is not unique

C: No OLS solution exists

D: The unique solution is $\bar{\boldsymbol{X}}^+ \boldsymbol{y}$

E: None of these

Regularized least squares linear regression I

- For greater generality, especially to handle cases where $\bar{\boldsymbol{X}}$ is rank deficient or when $\bar{\boldsymbol{X}}^\top \bar{\boldsymbol{X}}$ is poorly conditined, we often add a

$$\hat{\boldsymbol{w}}, \hat{b} = \underset{\boldsymbol{w}\in\mathbb{R}^d,\ b\in\mathbb{R}}{\arg\min} \frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{w}^\top \boldsymbol{x}_i + b - y_i)^2 +$$

- $\lambda \geq 0$ is the **regularization parameter** or **hyperparameter**.
- If we choose this method is called
  - The term $\lambda \|\boldsymbol{w}\|_2^2$ is called the
  - In deep learning it's called **weight decay**.
- Prof. Scott's course notes on Canvas derive $\hat{\boldsymbol{w}}, \hat{b}$ two different ways. Both are essentially "551 review" ala [5, Ch. 5].

Regularized least squares linear regression II

- To avoid a systematic bias, often we choose [6, p. 10]

$$\Omega(\boldsymbol{w}, b) = \|\boldsymbol{w}\|_2^2 .$$

- If we choose
  this method is known as [7] the
- The "selection" here is important: The resulting $\hat{\boldsymbol{\theta}}$ is typically sparse, so the method *selects* a subset of the features for prediction, improving interpetability.
- If we choose

$$\lambda\Omega(\boldsymbol{\theta}) =$$

  this method is known as [8] the

Regression summary

© 2026 Fessler

- Squared loss (because $\mathcal{Y} = \mathbb{R}$) leads to
- MSE risk $\mathbb{E}\{(f(\boldsymbol{X}) - Y)^2\}$
- Conditional mean estimator has MMSE:

$$\hat{y} = f_*(\boldsymbol{x}) = \mathbb{E}\{Y \mid \boldsymbol{X} = \boldsymbol{x}\}$$

- Gaussian assumption (generative model) led to affine estimator

$$f_*(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + b, \quad \boldsymbol{w}, b \text{ from Gaussian model}$$

- $\boldsymbol{w} = \boldsymbol{\Sigma}_{xx}^{-1} \boldsymbol{\Sigma}_{xy}$ not scalable to large d
- Discriminative approach with
  $f(\boldsymbol{x}; \boldsymbol{\theta}) = \boldsymbol{w}^\top \boldsymbol{x} + b, \ \boldsymbol{\theta} = (\boldsymbol{w}, b) \in \mathbb{R}^{d+1}$
- Fitting using _____ (MSE) becomes ordinary LS
- Added regularizers to avoid overfitting
- Examples? See [5, Ch. 5] and associated demos.

Big picture: Toward ERM

© 2026 Fessler

- More generally, regularized **empirical risk minimization** (**ERM**) learns a prediction function f by solving

$$\arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(\boldsymbol{x}_i)) + \lambda \Omega(f),$$

- Different methods result from different choices for
- We will see several examples hereafter beyond regularized least squares, including neural network methods.
- Advantages of such a unified framework:
  - Helps to compare and contrast different methods.
  - Some (forthcoming) optimization strategies can solve large classes of ERM methods.
  - ERM framework facilitates general theoretical analysis and understanding, which in turn facilitates progress.

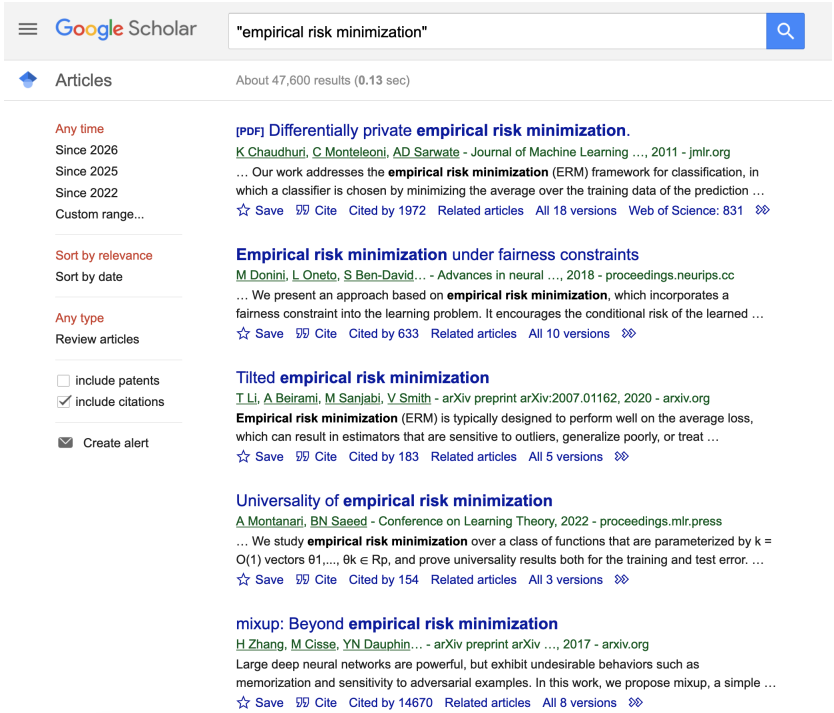
Empirical risk minimization: Examples

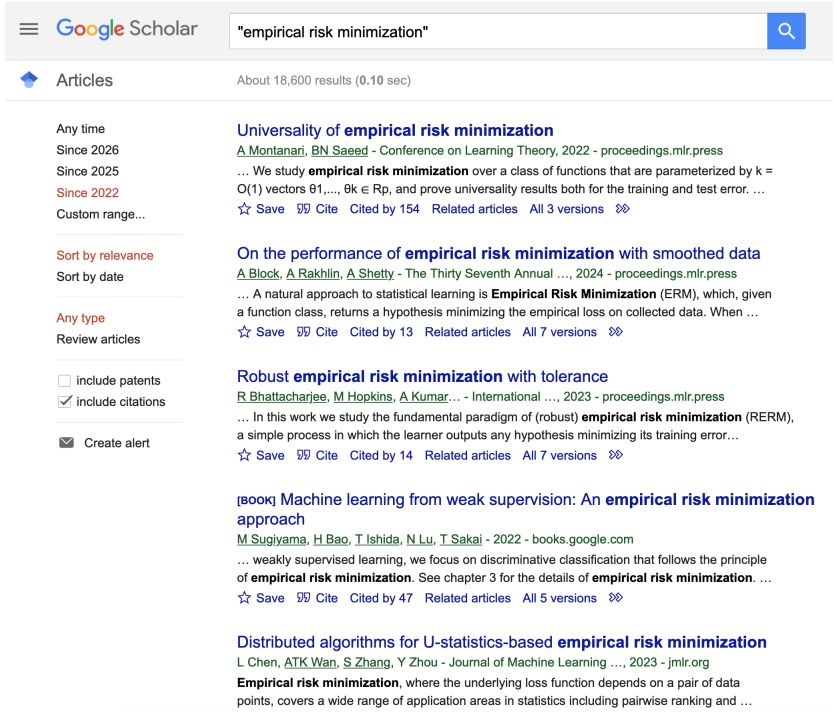
© 2026 Fessler

- Examples of methods in this common ERM framework:
  - Linear regression
  - Logistic regression
  - Support vector machines
  - Boosting
  - Decision trees
  - Neural networks
  - Sparse linear regression
  - ...
- Key ingredients of **empirical risk minimization** ERM:
  - Loss and risk
  - Surrogate losses
  - Regularization
  - Optimization methods

Empirical risk minimization: Citations

© 2026 Fessler





Empirical risk minimization framework

© 2026 Fessler

- Consider a supervised learning problem with jointly distributed features / labels $(\boldsymbol{X}, Y)$.
- Let $\mathcal{Y}$ denote the output space
  - Regression:
  - Binary classification:
  - Classification:
- A **loss** is a function $L : \mathcal{Y} \times \mathcal{Y} \mapsto [0, \infty)$.
- Let $f : \mathbb{R}^d \mapsto \mathcal{Y}$ be a function. The **risk** of f is

$$R_L(f) \triangleq \mathbb{E}_{\boldsymbol{X},Y}\{L(Y, f(\boldsymbol{X}))\}.$$

- For regression problems, f is a regression function.
- If L is the **squared error** loss

$$L(y, \hat{y}) = |y - \hat{y}|^2 ,$$

then the **risk** is the **mean squared error** (**MSE**):

$$R_L(f) = \mathbb{E}_{\boldsymbol{X},Y}\left\{ |Y - f(\boldsymbol{X})|^2 \right\}.$$

- MSE is not **robust** to data **outliers**.
  See robust regression demo [5, §6.2.5].
- If L is the **absolute deviation** loss

$$L(y, \hat{y}) =$$

then the **risk** is the (more robust) **mean absolute error**:

$$R_L(f) =$$

CQ: binary classification loss

© 2026 Fessler

For binary classification problems, f is called a **decision function** or **discriminant function**. The predicted label is

$$\hat{y} = f(\boldsymbol{x}) = \text{sign}(\tilde{f}(\boldsymbol{x})).$$

Q05.3 The function $L(y, \hat{y}) = \mathbb{I}_{\{y \neq \hat{y}\}}$ is called the

A: 0-1 loss

B: indicator loss

C: absolute error

D: exact loss

E: empirical risk

CQ: binary classification risk

© 2026 Fessler

For binary classification problems, f is called a **decision function** or **discriminant function**. The predicted label is

$$\hat{y} = f(\boldsymbol{x}) = \text{sign}(\tilde{f}(\boldsymbol{x})).$$

Q05.4 If $L(y, \hat{y}) = \mathbb{I}_{\{y \neq \hat{y}\}}$, then the **risk** is $R(f) = ?$

A: $\mathrm{P}\Big\{ Y \neq \text{sign}(\tilde{f}(\boldsymbol{X})) \Big\}$

B: $\mathbb{E}\Big\{ |Y - \text{sign}(\tilde{f}(\boldsymbol{X}))| \Big\}$

C: $\mathbb{E}\{|Y - f(\boldsymbol{X})|^2\}$

D: $\mathbb{E}\{|Y - f(\boldsymbol{X})|\}$

E: none of these

Empirical risk minimization (ERM): Framework © 2026 Fessler

- Given: training data $(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_n, y_n)$ for regression *or* binary classification.
- Given: selected loss function $L(y, \hat{y})$
- For any (classifier or regression) function $f : \mathbb{R}^d \mapsto \mathcal{Y}$, the

$$\hat{R}(f) = \hat{R}_n(f) \triangleq$$

- If $\{(\boldsymbol{x}_i, y_i)\}$ are IID from the joint distribution $\mathrm{p}(\boldsymbol{x}, y)$, then the **empirical risk** is an **unbiased estimate** of the **risk**:

$$\mathbb{E}_{\mathcal{D}}\left\{\hat{R}_n(f)\right\} = R(f),$$

and the **strong law of large numbers** guarantees that $\hat{R}_n(f)$ **converges almost surely** to the risk $R(f)$ as $n \to \infty$.

- Trying to find f by minimizing $\hat{R}$ often would lead to over-fitting.
- (Regularized) **empirical risk minimization** learns f by solving

$$\arg\min_{f \in \mathcal{F}} J(f), \quad J(f) \triangleq$$

$$=$$

where

- $\mathcal{F}$ is the set of candidate f functions.
  Examples:
- $\Omega(f)$ is the regularizer. Examples:
- $\lambda \geq 0$, user-specified (or **cross validated**) regularization parameter
- Modern ML practitioners often employ methods that create

Surrogate losses for ERM: Why

© 2026 Fessler

Choice of loss function L is important for ERM.

- Regression
  - Squared error loss and many robust losses for regression lead to ERM problems that often are computationally tractable.
  - Even the non-differentiable absolute deviation loss (aka "ℓ_1 loss") is at least convex, so good optimizers exist.
- Classification
  - ERM based on the **0-1 loss** for classification is *not* tractable, even for "linear" classifiers!
  - These challenges motivate using **surrogate loss** functions, perhaps especially for classification problems.

Surrogate losses: What

© 2026 Fessler

- A **surrogate loss** is a loss that takes the place of another, usually because of nicer computational properties (*e.g.*, convex, differentiable).
- Whereas a **loss** function is $L : \mathcal{Y} \times \mathcal{Y} \mapsto [0, \infty)$, *i.e.*, $L(y, \hat{y})$, often a **surrogate loss** for binary classification is
- Common surrogate losses for 0-1 loss in binary classification:
  - Logistic loss: $L(y, \tilde{f}(\boldsymbol{x})) =$
  - Hinge loss $L(y, \tilde{f}(\boldsymbol{x})) =$
    This is a convex upper-bound for 0-1 loss [1, 4.3.2].
- Both surrogate losses can be expressed

Surrogate losses: Examples

© 2026 Fessler

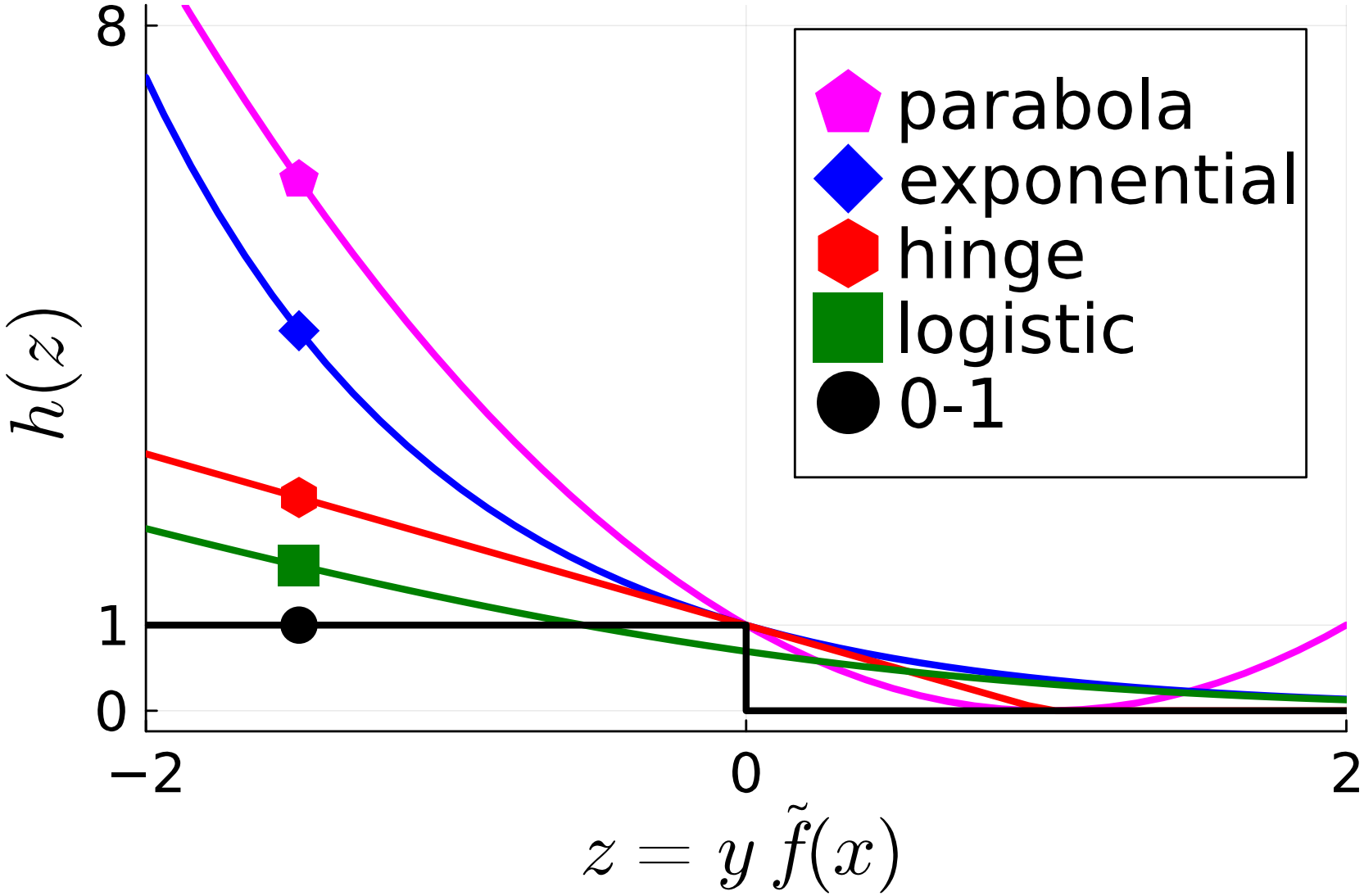


Figure from [5, Ch. 9].

Surrogate losses: Rationale

What makes it a "surrogate" specifically, and not just the loss?

- Usually there is some notion of the loss we would desire in the absence of computational complexity, in which case a surrogate tries to approximate it but with good complexity characteristics.
- In fact we may minimize a surrogate loss to design a classifier, but then report the performance on the original loss, *e.g.*, minimize hinge loss, but report the

- Fisher Consistency: A surrogate loss is classification-calibrated (or Fisher consistent) if minimizing the surrogate loss also minimizes the 0–1 risk in the limit of infinite data [9].
- A surrogate loss is "classification calibrated" if minimizing it leads to a classifier that gives the same label as the Bayes classifier [10].
- There are other notions of calibration that can relate surrogate losses to their original counterparts [11].
- Typical surrogate losses are convex functions, so we can use computationally efficient, convex optimization methods to solve for a classifier that minimizes the (surrogate) risk.
- TL;DR: The goal when choosing a surrogate loss is to

Binary classifier design via ERM: Preview

© 2026 Fessler

- Pick family of classifiers $\mathcal{F}$ to be "linear" discriminants:

$$f(\boldsymbol{x}) = f(\boldsymbol{x}; \boldsymbol{\theta}) = \operatorname{sign}(\tilde{f}(\boldsymbol{x}; \boldsymbol{\theta}))$$
$$\tilde{f}(\boldsymbol{x}; \boldsymbol{\theta}) = \boldsymbol{w}^\top \boldsymbol{x} + b, \quad \boldsymbol{\theta} = (\boldsymbol{w}, b) \in \mathbb{R}^{d+1}$$

- Pick a surrogate loss function

$$L(y, \tilde{f}(\boldsymbol{x})) = h(y \; \tilde{f}(\boldsymbol{x}))$$

- Regularized empirical risk minimization $\arg\min_{f \in \mathcal{F}}$ involves

$$J(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^{n} h(y_i \; \tilde{f}(\boldsymbol{x}_i; \boldsymbol{\theta})) + \lambda \Omega(\boldsymbol{\theta})$$

- Classifier design becomes a nontrivial

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta} \in \mathbb{R}^{d+1}} J(\boldsymbol{\theta}).$$

- This approach to **logistic regression** covered in 551 [5, §9.5].

Ridge regression: Compute complexity

© 2026 Fessler

- Consider the computational complexity of computing the regularized LS estimate in the ERM framework:

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta} \in \mathbb{R}^{d+1}}{\arg\min}\, J(\boldsymbol{\theta}), \quad J(\boldsymbol{\theta}) = \frac{1}{n}\frac{1}{2}\left\|\bar{\boldsymbol{X}}\boldsymbol{\theta} - \boldsymbol{y}\right\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{\theta}\|_2^2$$
$$= (\bar{\boldsymbol{X}}^\top \bar{\boldsymbol{X}} + \lambda n \boldsymbol{I})^{-1} \bar{\boldsymbol{X}}^\top \boldsymbol{y}$$

- Recall $\bar{\boldsymbol{X}} \in \mathbb{R}^{n \times (d+1)}$, $\boldsymbol{y} \in \mathbb{R}^n$, and $\boldsymbol{w} \in \mathbb{R}^d$.
- Computing $\bar{\boldsymbol{X}}^\top \bar{\boldsymbol{X}}$ is
- Matrix inversion is
- Overall:
- Even using more efficient "backslash" via **QR decomposition** is $O(d^2)$, and becomes compute expensive for large d.
- Thus, even in this "closed-form" $\hat{\boldsymbol{\theta}}$ situation, it can be preferable to use

References I

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

[2] X. Xu, Y. Sun, J. Liu, B. Wohlberg, and U. S. Kamilov.
Provable convergence of plug-and-play priors with MMSE denoisers.
IEEE Signal Proc. Letters, 27:1280–4, 2020.

[3] J. Liu, R. Anirudh, J. J. Thiagarajan, S. He, K. A. Mohan, U. S. Kamilov, and H. Kim.
DOLCE: A model-based probabilistic diffusion framework for limited-angle CT reconstruction.
In *Proc. Intl. Conf. Comp. Vision*, pages 10464–74, 2023.

[4] L. M. Lesser.
Critical values and transforming data: teaching statistics with social justice.
J. of Statistics Education, 15(1), August 2017.

[5] J. A. Fessler and R. R. Nadakuditi.
Linear algebra for data science, machine learning, and signal processing.
Cambridge, 2024.

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

[7] R. Tibshirani.
Regression shrinkage and selection via the LASSO.
J. Royal Stat. Soc. Ser. B, 58(1):267–88, 1996.

[8] H. Zou and T. Hastie.
Regularization and variable selection via the elastic net.
J. Royal Stat. Soc. Ser. B, 67(2):301–20, 2005.

References II

[9] Y. Lin.
A note on margin-based loss functions in classification.
Statist. Probab. Letters, 68(1):73–82, 2004.

[10] Peter L Bartlett, Michael I Jordan, and Jon D McAuliffe.
Convexity, classification, and risk bounds.
Journal of the American Statistical Association, 101(473):138–156, 2006.

[11] Alekh Agarwal, Sham Kakade, Nikos Karampatziakis, Le Song, and Gregory Valiant.
Least squares revisited: Scalable approaches for multi-class prediction.
In *International Conference on Machine Learning*, pages 541–549. PMLR, 2014.