

# Machine Learning

## Generalization theory and midterm review

DSC 240

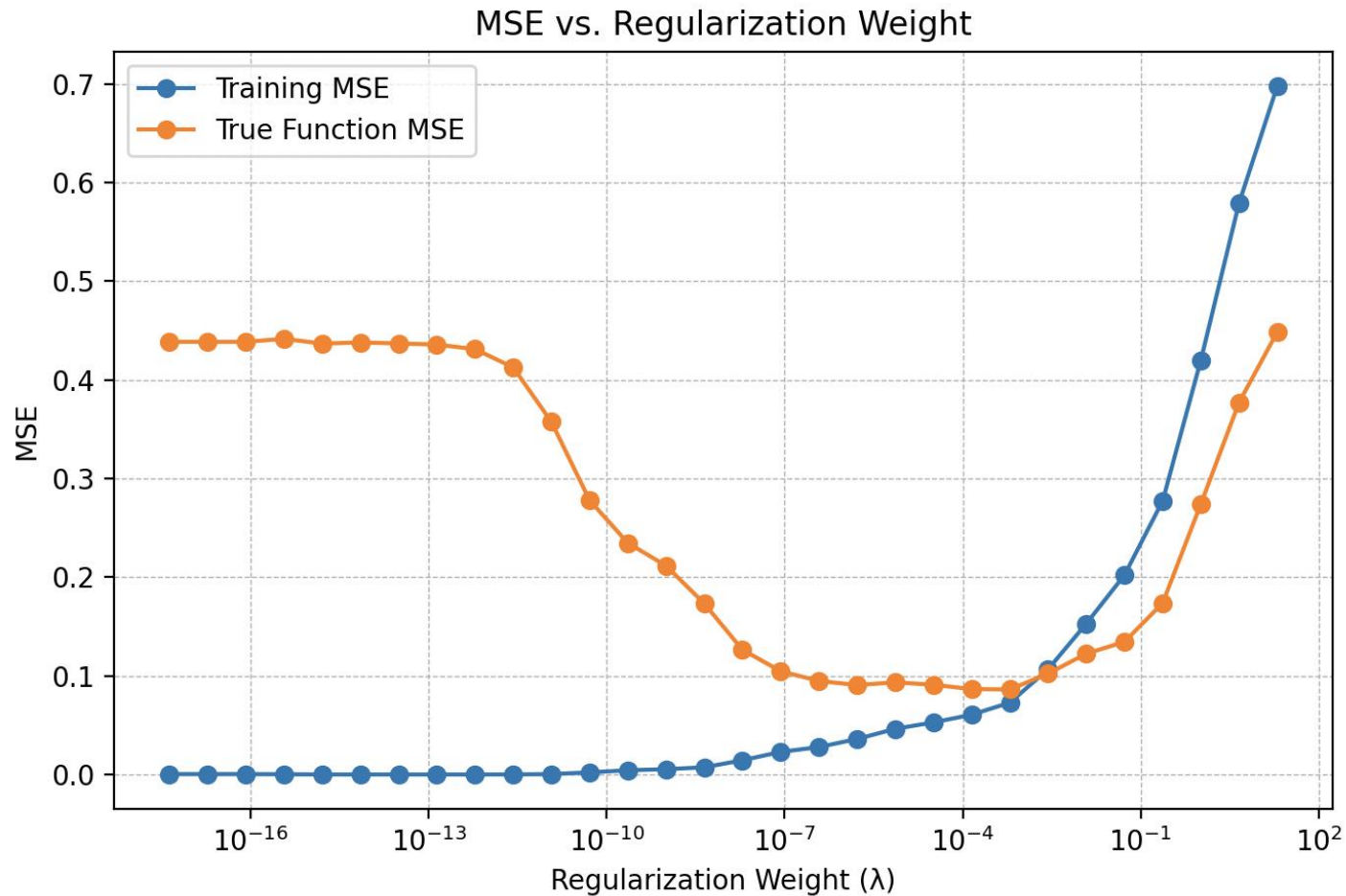
Feb 4, 2025

Instructor: Prof. Yu-Xiang Wang

# Last lecture

- Loss functions for regression problems
  - Solving non-linear "curve fitting" problem using linear regression
- Solving linear regression (with square loss)
  - SGD
  - GD
  - Direct solver:  $Ax = b$
- How to avoid overfitting?
  - Regularization
  - L1 and L2 regularization
- Case study: Predict California Housing Market
  - Bias-variance tradeoff on a real dataset
  - Effect of regularization and Regularization path

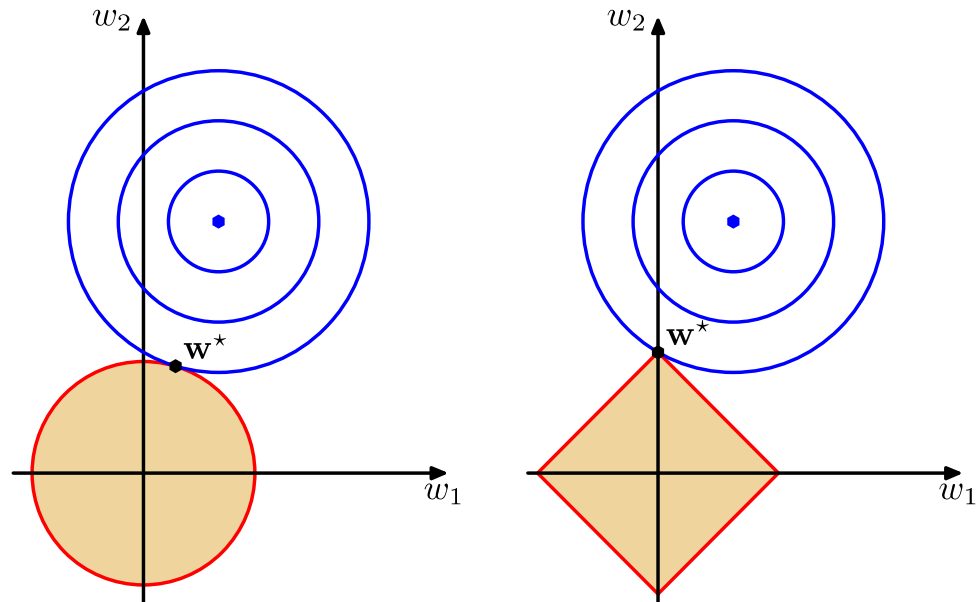
# Recap: Appropriate choice of regularization parameter prevents overfitting and underfitting



Recap: Regularization helps to **reduce overfitting** and induce **structures** in the solution.

- Ridge regression induces solutions that are small but dense.
- Lasso induces solutions that are “sparse”.

**Figure 3.4** Plot of the contours of the unregularized error function (blue) along with the constraint region (3.30) for the quadratic regularizer  $q = 2$  on the left and the lasso regularizer  $q = 1$  on the right, in which the optimum value for the parameter vector  $w$  is denoted by  $w^*$ . The lasso gives a sparse solution in which  $w_1^* = 0$ .



Regularization can be applied to classification problems too, especially when the number of features is very large.

- L2-regularization: maximize margin
  - Why?
  - Read up on “support vector machine”: Hinge loss + L2-regularization
  - Very interesting geometric insight. It also tells you why we shift “Perceptron loss” to “Hinge loss”
  - Will be covered after the midterm (next Tuesday).
- L1-regularization: feature selection
  - Promotes a sparse linear combination of the features
  - What’s the benefit?
    - Interpretability
    - Good generalization despite a larger number of irrelevant features.

# Recap: Mathematically defining the supervised learning problem

- Feature space:  $\mathcal{X}$
- Label space:  $\mathcal{Y}$
- A classifier (hypothesis):  $h : \mathcal{X} \rightarrow \mathcal{Y}$
- A hypothesis class:  $\mathcal{H}$
- Data:  $(x_1, y_1), \dots, (x_n, y_n) \in \mathcal{X} \times \mathcal{Y}$
- Learning task: Find  $h \in \mathcal{H}$  that “works well”.

# Recap: Examples of supervised learning problems

	Binary classification	Multi-class classification	Regression
Feature space	$\mathbb{R}^d$	$\mathbb{R}^d$	$\mathbb{R}^d$
Label space	$\{-1, 1\}$	$\{1, 2, 3, \dots, K\}$	$\mathbb{R}$
Popular Performance metric	Classification error (0-1 loss) for test data	Classification error (0-1 loss) for test data	Mean Square Error (MSE) vs ground truth
Popular surrogate loss (for training)	Logistic loss	Multiclass logistic loss aka. Cross-Entropy loss	Square loss

# How do we talk about generalization to “future data”?

- We need assumptions that relate training data to future data.
  - Assumption candidate 1: “training data”  $\supseteq$  “future data”
  - Assumption candidate 2: Both “training data” and “future data” satisfies that  $E[y | x] = f_0(x)$  for some  $f_0$
- Typical assumption in machine learning
  - iid assumption

# Today

- Theory for Linear Regression
  - Prediction error bound
  - Estimation error bound
- Theory for binary classification
- Midterm review

# Recap: Notations from probability

$\mathbb{E}_{\mathcal{D}}$  [Function of an r.v.  $X$ ]

$\mathbb{P}_{\mathcal{D}}$  [Event]

$f_{X \sim \mathcal{D}}(x)$

$F_{X \sim \mathcal{D}}(x)$

Conditional expectation / conditional probability / density

$\mathbb{E}[\text{Func}(X, Y) | Y]$

$\mathbb{P}[\text{Event\_of}(X, Y) | Y]$

$f(x|y)$

# Problem setup for linear regression

(A1) Assume input points (feature vectors) *span* the whole space

(A2) Assume the labels given by a true linear model + *independent* noise.

$$y_i = x_i \cdot \theta^* + \epsilon_i, \quad \mathbb{E}[\epsilon_i] = 0 \quad \text{Var}[\epsilon_i] = \sigma^2$$

- Two natural questions:
  - (prediction) How well can the learned hypothesis predict new observations?
  - (estimation) How close is the learned coefficients vector from the ground truth?

# Some useful linear algebraic notations

- Data matrix
- Label vector
- Ordinary Least Square fit (the minimizer of training loss)

$$\hat{\theta} = \arg \min_{\theta} \frac{1}{n} \sum_{i \in [n]} (x_i^T \theta - y_i)^2$$

How do we analyze the quality of the solution?

# Theorem for (fixed design) linear regression

**Theorem:** Assume (A1) and (A2), the ordinary least square estimator for linear regression satisfies:

For prediction: 
$$\mathbb{E} \left[ \frac{1}{n} \|X\hat{\theta} - X\theta^*\|^2 \right] \leq \frac{d\sigma^2}{n}$$

For estimation: 
$$\mathbb{E} \left[ \|\hat{\theta} - \theta^*\|^2 \right] \leq \text{tr}[(X^T X)^{-1}] \sigma^2$$

## Remarks:

1. For prediction tasks, it doesn't matter how the data is distributed. But for accurate estimation of coefficients,  $X$  should be well-conditioned.
2. Large number of features lead to overfitting. (e.g., use an  $n$ th order polynomial to fit  $n$  points)

# Today

- Theory for Linear Regression
  - Prediction error bound
  - Estimation error bound
- Theory for binary classification
- Midterm review

# The result relies on strong assumptions on how the data is generated

- e.g., it does NOT apply to the case for fitting a polynomial to a noisy sine function! (Why? Which assumption is the culprit?)
- The **statistical learning problem**:
  - Assumption B1: iid samples
  - Assumption B2: Bounded loss function
  - Assumption B3: Finite hypothesis class

# Loss, Risk, Empirical Risk: What do we mean by working well?

- Loss function

$$\ell(h, (x, y))$$

- Risk function

$$R(h, \mathcal{D}) = \mathbb{E}_{\mathcal{D}}[\ell(h, (x_i, y_i))]$$

- Empirical risk

$$\hat{R}(h, \text{Data}) = \frac{1}{n} \sum_{i=1}^n \ell(h, (x_i, y_i))$$

Learning is often achieved by solving the **Empirical Risk Minimization** problem

$$\hat{h} = \arg \min_{h \in \mathcal{H}} \hat{R}(h, \{(x_i, y_i) | i \in [n]\})$$

- Example 1 (Perceptron – smallest number of mistakes)
  
- Example 2 (Ordinary Least Square )

The goal is to bound the **excess risk** . This time we want a high probability bound.

- With probability at least  $1 - \delta$

$$R(\hat{h}) - R(h^*) \leq \epsilon$$

- Parameterize  $\epsilon$  as a function of
  - Number of data points
  - Size of the hypothesis class
  - Boundedness of the loss
  - Failure probability

# Introducing two powerful “hammers”: Hammer 1. Hoeffding’s inequality

**Theorem D.2 (Hoeffding’s inequality)** *Let  $X_1, \dots, X_m$  be independent random variables with  $X_i$  taking values in  $[a_i, b_i]$  for all  $i \in [m]$ . Then, for any  $\epsilon > 0$ , the following inequalities hold for  $S_m = \sum_{i=1}^m X_i$ :*

$$\mathbb{P}[S_m - \mathbb{E}[S_m] \geq \epsilon] \leq e^{-2\epsilon^2 / \sum_{i=1}^m (b_i - a_i)^2} \quad (\text{D.4})$$

$$\mathbb{P}[S_m - \mathbb{E}[S_m] \leq -\epsilon] \leq e^{-2\epsilon^2 / \sum_{i=1}^m (b_i - a_i)^2} . \quad (\text{D.5})$$

Roughly saying that the **empirical averages** of independent random variable converges to the **mean** at a  $O(1/\sqrt{n})$  rate, with high probability.

# Introducing two powerful “hammers”: Hammer 2. Union bound

**Lemma (Union bound):** For any probability distribution and any event  $E_1, E_2$ :

$$\mathbb{P}[E_1 \cup E_2] \leq \mathbb{P}[E_1] + \mathbb{P}[E_2]$$

Now let's apply these two hammers to solve statistical learning

1. For each hypothesis  $h$ , apply Hoeffding's inequality
2. Union bound over all hypothesis

Now let's apply these two hammers to solve statistical learning

**Theorem:** Assume (B1),(B2) and (B3), with probability at least  $1 - \delta$  (over the distribution of the data), ERM satisfies

$$R(\hat{h}) - R(h^*) = O\left(\sqrt{\frac{\log |\mathcal{H}| + \log(1/\delta)}{n}}\right)$$

# Example 1: Application to decision tree classifier

- $d$ -dimensional discrete feature (  $L$ -levels for each)
- $H$ -layer decision tree, binary decision in one layer
- $K$  Labels
- *Upper bound of the size of hypothesis class?*

## Example 2: Application to generic classification (no restriction on the hypothesis class)

- **d**-dimensional discrete feature ( **L**-levels for each)
- **K** labels
- *Total number of unique classifiers?*

## Example 3: Application to linear regression on finite computers (32 bit floats)

- **d**-dimensional “continuous” feature  $x$
- **d**-dimensional “continuous” weights / coefficients  $\theta$
- **1**-continuous label  $y$
- ***Total number of unique linear regression functions?***

(Assume bounded  $x, y, \theta$  )

# Checkpoint: Generalization theory

- For linear regression
  - Assume label is linear function + noise
  - Allows us to precisely quantify the prediction error and estimation error
  
- For learning with bounded loss
  - Concept of data distribution, risk and empirical risk
  - The ERM learning rule
  - Simple excess risk bound via a “Uniform convergence” argument.
  
- Generalization theory only provides high-level guideline
  - “ballpark” estimate of when overfitting occurs

# Today

- Theory for Linear Regression
  - Prediction error bound
  - Estimation error bound
- Theory for binary classification
- Midterm review

# What have we learned so far?

- Lecture 1: Basic concept of machine learning / example applications
- Lecture 2-3: Supervised learning
  - Binary classification (Spam filtering)
  - What is a classifier or hypothesis? How is a classifier described by its parameters
  - What is a decision boundary? How does the decision boundary of a classifier look like (e.g., in linear classifier or for a decision tree)
  - What is a hypothesis class?
  - Performances metrics: FP, FN, precision, recall, F1-score, ROC curve....
- Lecture 3: How to evaluate a classifier and prevent overfitting?
  - Data splitting methods: Holdout and Cross Validation
  - The issue of distribution shift

# What have we learned so far?

- Lecture 4: Linear algebra
  - Dot product, transpose, matrix-matrix, matrix-vector multiplication
  - Length, angle, distance to a hyperplane
  - L1-norm, L2-norm, L-inf norm
  - Representing (hyper)plane in its standard form
- Lecture 5-7: Learning a linear classifier.
  - Perceptron, GD, SGD.
  - What are their pros and cons? What are there differences?
  - Learning curves / convergence plots
- Lecture 6-7: Taking gradient of multivariate functions
  - Best way: partial derivative one-coordinate at a time
  - Fast way: remember some of the basic rules so you can “vectorize” your operations

# What have we learned so far?

- Lecture 7 - 8: Curve fitting, linear regression, regularization
  - How to evaluate a regression fit?
  - Optimizing linear regression objective using SGD, GD and closed-form solution
  - Prevent overfitting by L2 or L1 regularization
- Lecture 9 (today): Generalization theory
  - Theoretical bounds on learning and overfitting

Midterm will cover all the way until Lecture 7  
(including continuous optimization but not including linear regression and thereafter)

# What we can do in the remainder of the lecture

1. Ask me anything about any topic we covered so far
2. Go over problems in the practice midterm

Good luck with the midterm!