Loss Functions and Regularization

David I. Inouye

Tuesday, September 15, 2020

Outline

- Loss functions
  - Regression losses
  - Classification losses
- Regularization
  - "Implicit regularization" by changing k in KNN
  - L2 regularization
  - L1 regularization and feature selection
- Caveat: Very brief introduction to these concepts
  - If you want to learn more, take ECE595 Machine Learning I (Prof. Stanley Chan)

Many machine learning methods minimize the average loss (a.k.a. risk minimization)

- Remember linear regression objective:

$$\theta^* = \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \left(y_i - f_\theta(\boldsymbol{x}_i)\right)^2$$

- We can rewrite this as:

$$\theta^* = \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \ell\left(y_i, f_\theta(\boldsymbol{x}_i)\right)$$

  - where $\ell(a, b) = (a - b)^2$ is the **loss function**
- Many supervised ML can be written as above

Many supervised ML can be written minimizing the average loss

- Ordinary least squares uses **squared loss**:

$$\ell(a, b) = (a - b)^2$$

- Logistic regression uses **logistic loss**

$$\ell(a, b) = a \log b + (1 - a) \log(1 - b)$$

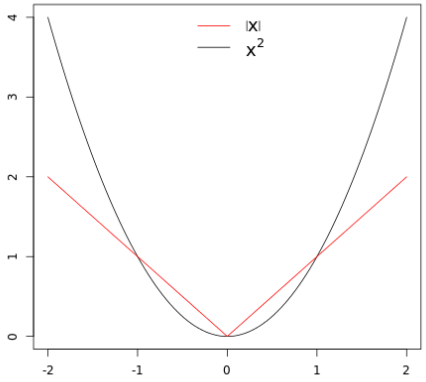
- Classification error is known as **0-1 loss**

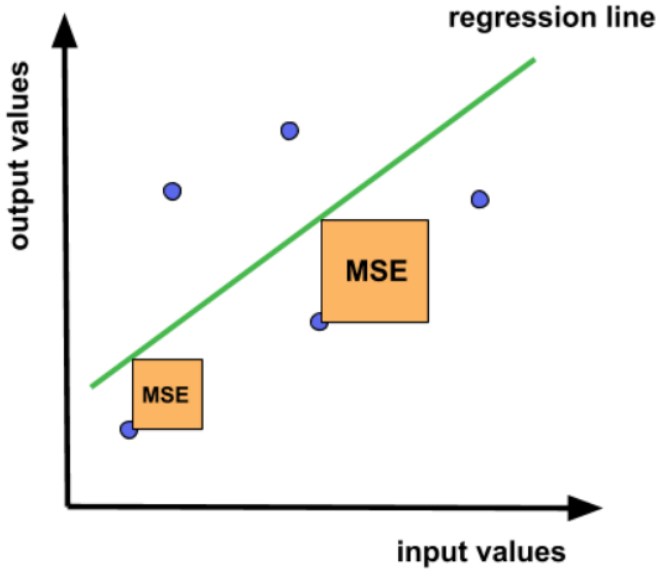
$$\ell(a, b) = \begin{cases} 0, & \text{if } a = b \\ 1, & \text{otherwise} \end{cases}$$

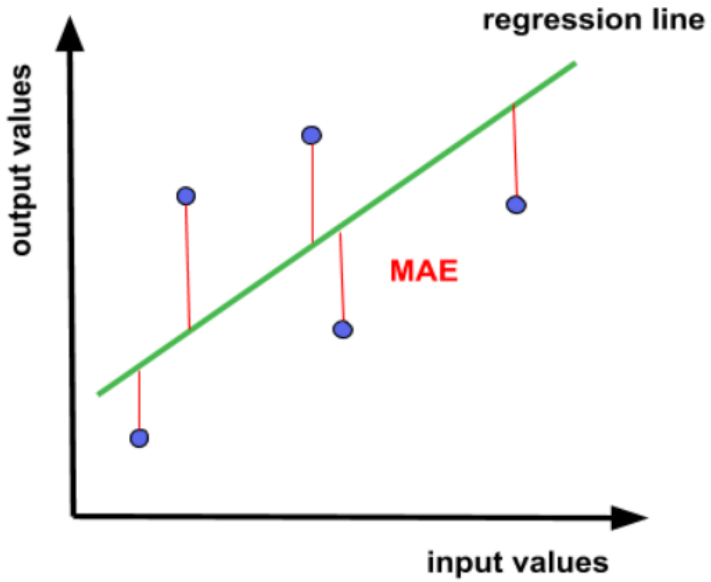
Example: Absolute error is less sensitive to outliers but is harder to optimize

- Absolute error loss is:

$$\ell(a, b) = |a - b|$$







https://www.datacourses.com/evaluation-of-regression-models-in-scikit-learn-846/

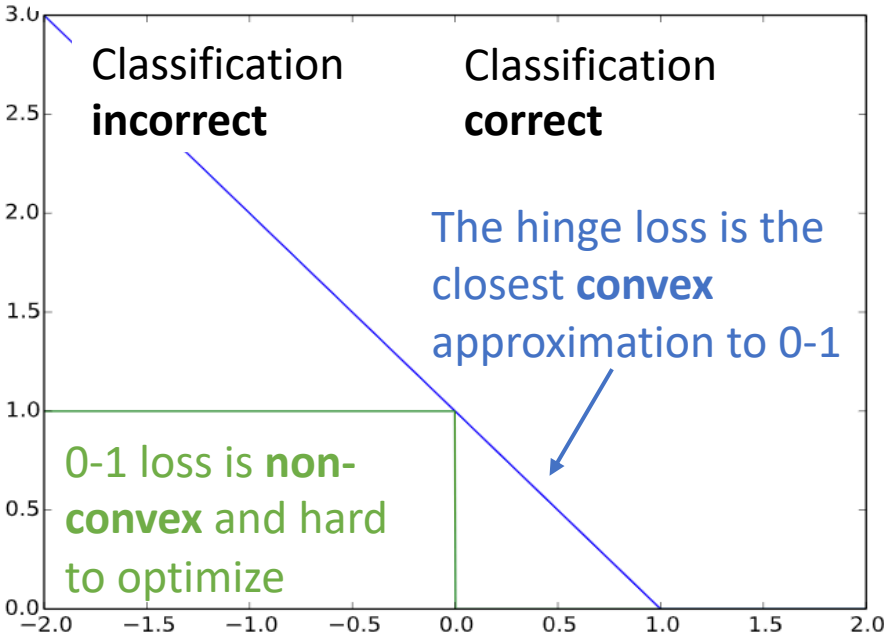
Example: The **hinge loss** is used for learning support vector machine (SVM) classifiers

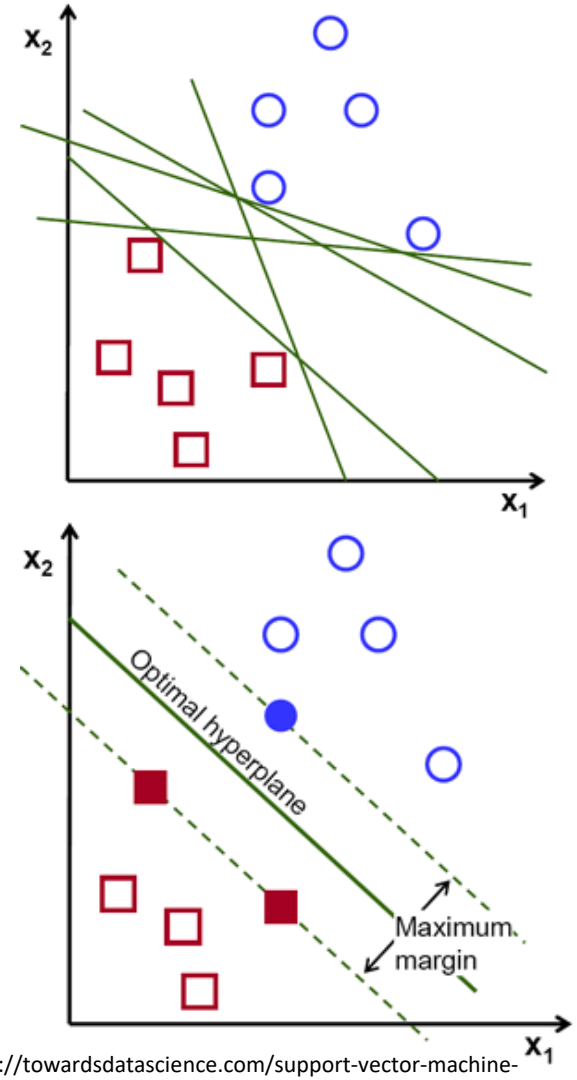
- **Hinge loss** is defined as:

$$\ell(a, b) = \max\{0, 1 - ab\}$$

(Note: $a \in \{-1, 1\}$)

(Assume $a = 1$ below)





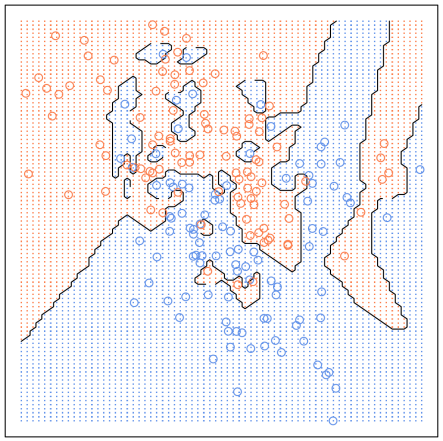
https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

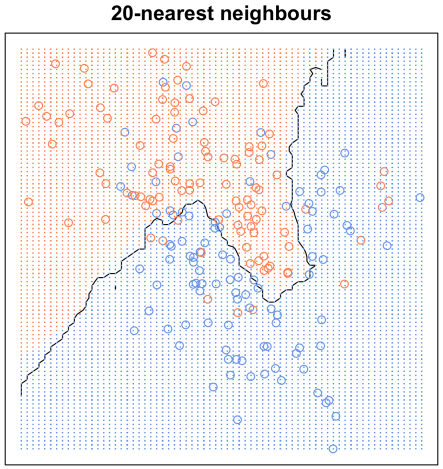
Regularization is a common method to improve *generalization* by **reducing the complexity of a model**

- k in KNN can be seen as an *implicit* regularization technique

- We can use *explicit* regularization for parametric models by adding a **regularizer** $R(\theta)$

$$\min_{\theta} \sum_{i} \ell\left(y_i, f_\theta(\boldsymbol{x}_i)\right) + \lambda\, R(\theta)$$





https://kevinzakka.github.io/2016/07/13/k-nearest-neighbor/

Brief aside: 1D polynomial regression can be computed by creating polynomial "pseudo" features

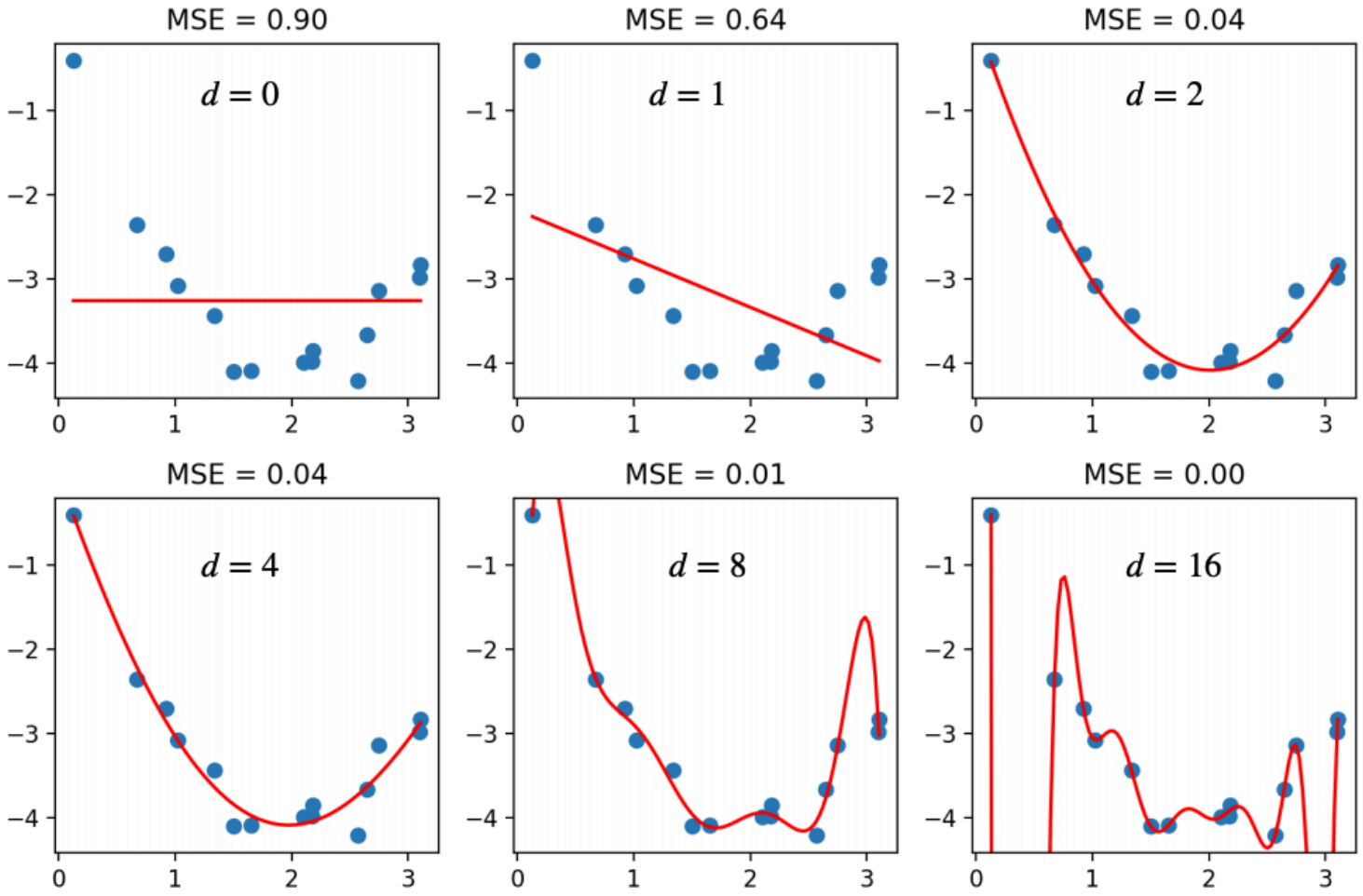
- Suppose we have 1D input data, i.e., $X \in \mathcal{R}^{n \times 1}$
- We can create pseudo polynomial features, e.g.

$$X' = \begin{bmatrix} x_1 & x_1^2 & x_1^3 \\ x_2 & x_2^2 & x_2^3 \\ x_3 & x_3^3 & x_3^3 \end{bmatrix} \in \mathcal{R}^{n \times 3}$$

- Linear regression can then be used to fit a polynomial model

$$y_i = \theta_1 x_i + \theta_2 \left(x_i^2\right) + \theta_3 \left(x_i^3\right) \ldots$$

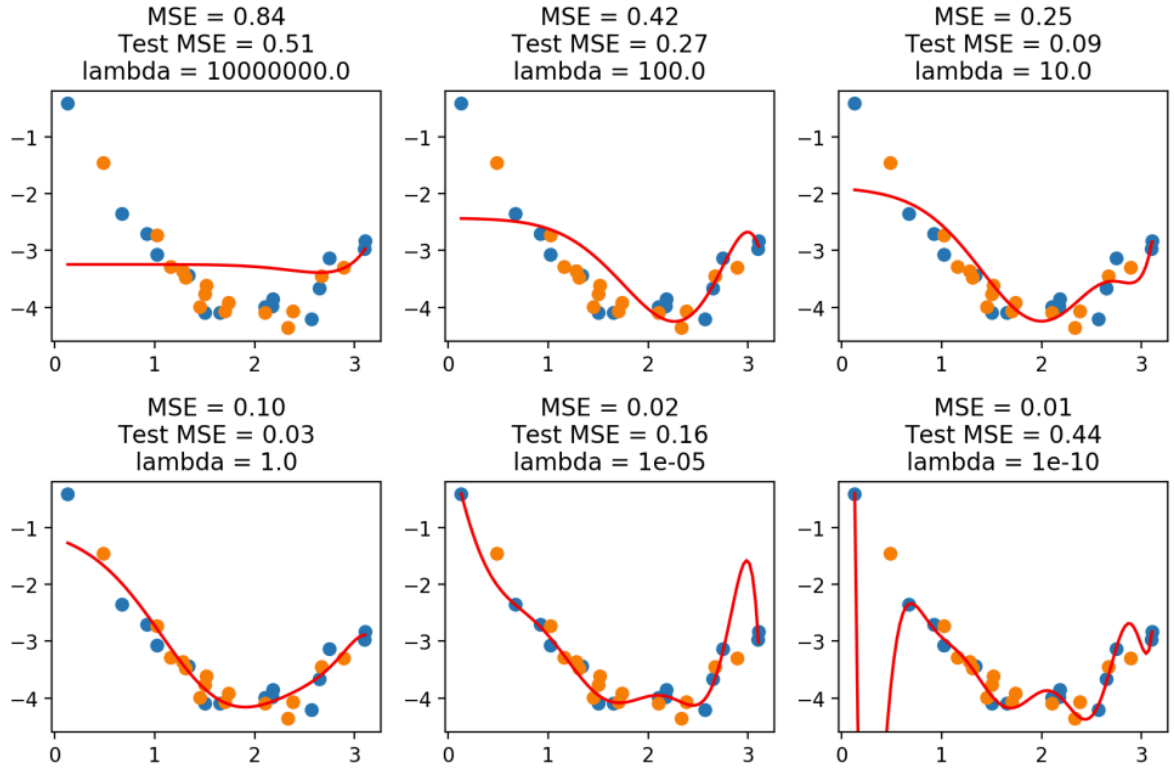
Brief aside: 1D polynomial regression can be computed by creating polynomial “pseudo” features



Ridge Regression: A squared norm regularizer encourages small parameter values

- Ridge regression is defined as:

$$\min_{\theta} \|\boldsymbol{y} - X\theta\|_2^2 + \lambda\|\theta\|_2^2$$

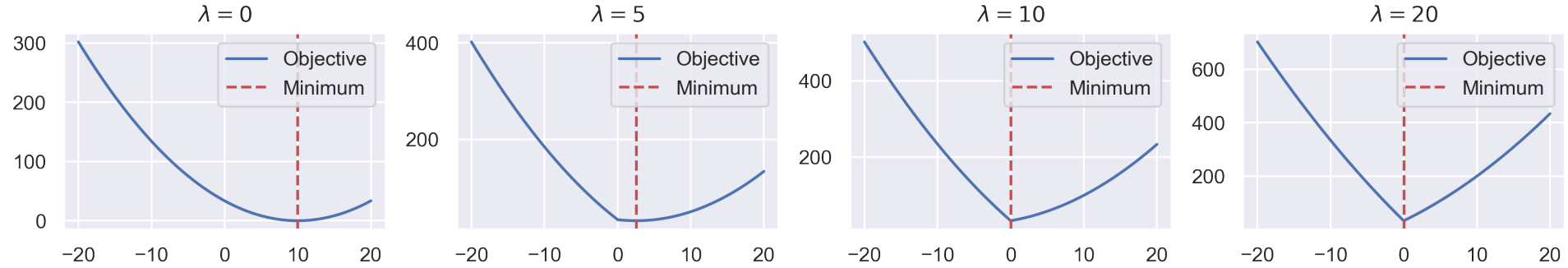


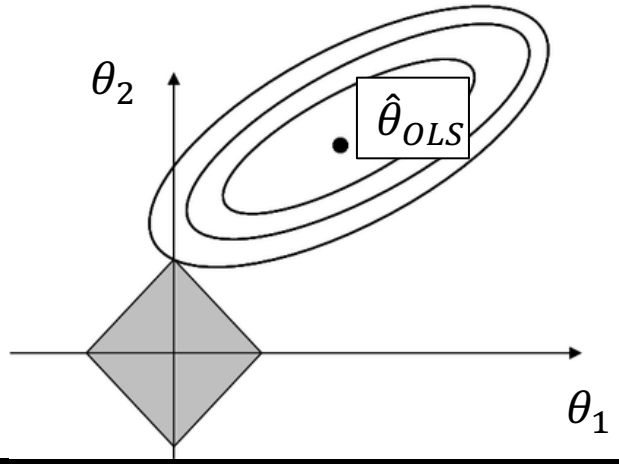
Regularizing the parameters of 1D polynomial regression helps to improve test MSE if chosen appropriately.

Lasso Regression: An L_1 norm regularizer encourages sparsity in the parameters (i.e., zeros)

- Lasso regression is defined as:

$$\min_{\theta} \|\boldsymbol{y} - X\theta\|_2^2 + \lambda\|\theta\|_1$$





Because lasso encourages **exact zeros**, lasso can be used for **feature selection**.

$$f_\theta(x) = \theta_1 x_1 + \theta_2 x_2$$
$$= (0)x_1 + \theta_2 x_2$$
$$= \theta_2 x_2$$