# Artificial Intelligence II (CS4442B & CS9542B)

## Overfitting, Cross-Validation, and Regularization

Boyu Wang
Department of Computer Science
University of Western Ontario

# Motivation examples: polynomial regression

- As the degree of the polynomial increases, there is more degrees of freedom, and the (training) error approaches to zero.

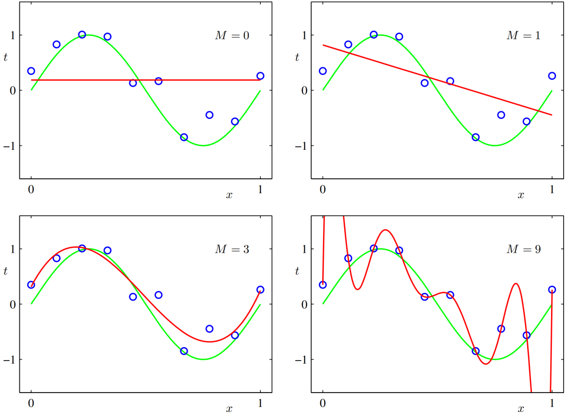
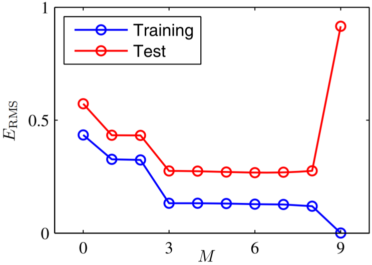



Figure credit: Christopher Bishop

# Motivation examples: polynomial regression

- Minimizing the training/empirical loss does NOT indicate a good test/generalization performance.
- Overfitting: Very low training error, very high test error.




| | $M = 0$ | $M = 1$ | $M = 6$ 3 | $M = 9$ |
|---|---|---|---|---|
| $w_0^\star$ | 0.19 | 0.82 | 0.31 | 0.35 |
| $w_1^\star$ | | -1.27 | 7.99 | 232.37 |
| $w_2^\star$ | | | -25.43 | -5321.83 |
| $w_3^\star$ | | | 17.37 | 48568.31 |
| $w_4^\star$ | | | | -231639.30 |
| $w_5^\star$ | | | | 640042.26 |
| $w_6^\star$ | | | | -1061800.52 |
| $w_7^\star$ | | | | 1042400.18 |
| $w_8^\star$ | | | | -557682.99 |
| $w_9^\star$ | | | | 125201.43 |

Figure credit: Christopher Bishop

# Overfitting – general phenomenon

- Too simple (e.g., small $M$) $\rightarrow$ underfitting
- Too complex (e.g., large $M$) $\rightarrow$ overfitting

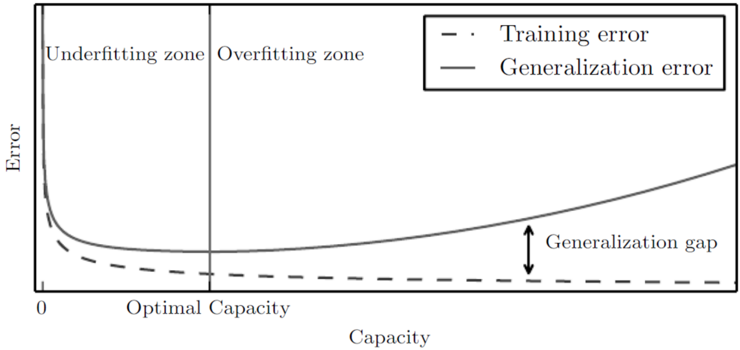



Figure credit: Ian Goodfellow

# Overfitting

- Training loss and test loss are different
- Larger the hypothesis class, easier to find a hypothesis that fits the training data
  - but may have large test error (overfitting)
- Prevent overfitting:
  - Large data set
  - Throw away useless hypothesis class (model selection)
  - Control model complexity (regularization)

# Larger data set

- Overfitting is mostly due to sparseness of data.
- Same model complexity: more data ⇒ less overfitting. With more data, more complex (i.e. more flexible) models can be used.

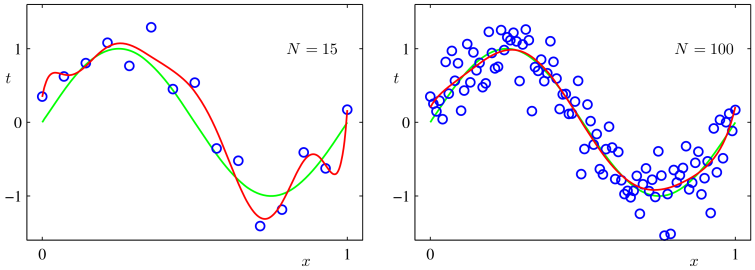



Figure credit: Christopher Bishop

# Model selection

- How to choose the optimal model complexity/hyper-parameter (e.g., choose the best degree for polynomial regression)
- Cannot be done by training data alone

# Model selection

- How to choose the optimal model complexity/hyper-parameter (e.g., choose the best degree for polynomial regression)
- Cannot be done by training data alone
- We can use our prior knowledge or expertise (e.g., somehow we know that the degree should not exceed 4)
- Create held-out data to approximate the test error (i.e., mimic the test data)
  - called validation data set

# Model selection: cross-validation

For each order of polynomial $M$

1. Randomly split the training data into $K$ groups, and following procedure $K$ times:
   i. Leave out the $k$-th group from the training set as a validation set
   ii. Use the other other $K-1$ to find best parameter vector $w_k$
   iii. Measure the error of $w_k$ on the validation set; call this $J_k$
2. Compute the average errors: $J = \frac{1}{K}\sum_{k=1}^{K} J_k$

Choose the order of polynomial $M$ with the lowest error $J$.

# Model selection: cross-validation

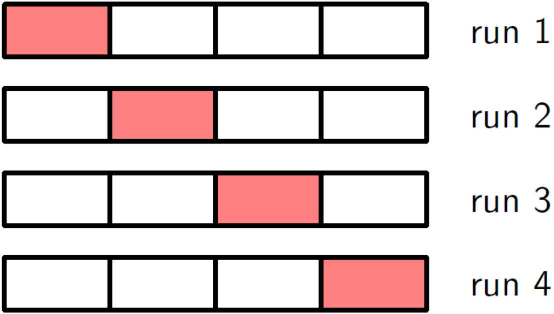



Figure: $K$-fold cross-validation for the case of $K = 4$

Figure credit: Christopher Bishop

# General learning procedure

Given a training set and a test set

1. Use cross-validation to choose the hyper-parameter/hypothesis class.
2. Once the hyper-parameter is selected, use the entire training set to find the best model parameters $w$.
3. Evaluate the performance of $w$ on the test set.

These sets must be disjoint! – you should never touch the test data before you evaluate your model.

# Summary of cross-validation

- Can also used for selecting other hyper-parameters for model/algorithm (e.g., number of hidden layers of neural networks, learning rate of gradient descent, or even different machine learning models)
- Very straightforward to implement algorithm
- Provides a great estimate of the true error of a model
- Leave-one-out cross-validation: number of groups = number of training instances
- Computationally expensive; even worse when there are more hyper-parameters

# Regularization

- Intuition: complicated hypotheses lead to overfitting
- Idea: penalize the model complexity (e.g., large values of $w_j$):

$$L(w) = J(w) + \lambda R(w)$$

  where $J(w)$: training loss, $R(w)$: regularization function/regularizer, and $\lambda \geq 0$: regularization parameter to control the tradeoff between data fitting and model complexity.

# $\ell_2$-norm regularization for linear regression

Objective function:

$$L(w) = \frac{1}{2}\sum_{i=1}^{m}\Big(\sum_{j=1}^{n} w_0 + w_j \cdot x_{i,j} - y_i\Big)^2 + \frac{\lambda}{2}\sum_{j=1}^{n} w_j^2$$

- No regularization on $w_0$!

Equivalently, we have

$$L(w) = \frac{1}{2}||Xw - y||_2^2 + \frac{\lambda}{2} w^\top \hat{I} w$$

where $w = [w_0, w_1, \ldots, w_n]^\top$

$$\hat{I} = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

# $\ell_2$-norm regularization for linear regression

Objective function:

$$
\begin{aligned}
L(w) &= \frac{1}{2}||Xw - y||_2^2 + \frac{\lambda}{2} w^\top \hat{I} w \\
&= \frac{1}{2}\Big(w^\top (X^\top X + \lambda \hat{I}) w - w^\top X^\top y - y^\top X w + y^\top y\Big)
\end{aligned}
$$

Optimal solution (by solving $\nabla L(w) = 0$):

$$
w = (X^\top X + \lambda \hat{I})^{-1} X^\top y
$$

# More on $\ell_2$-norm regularization

$$\arg\min_{w} \frac{1}{2}||Xw - y||_2^2 + \frac{\lambda}{2} w^\top \hat{I} w = (X^\top X + \lambda \hat{I})^{-1} X^\top y$$

- $\ell_2$-norm regularization pushes the parameters towards to 0.
- $\lambda = 0 \Rightarrow$ same as in the regular linear regression
- $\lambda \to \infty \Rightarrow w \to 0$
- $0 < \lambda < \infty \Rightarrow$ magnitude of the weights will be smaller than in the regular linear regression

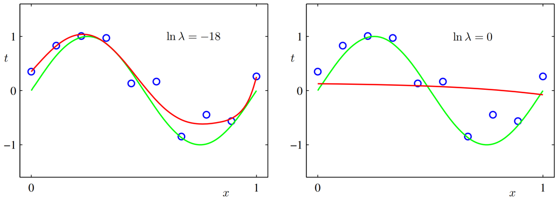



Figure credit: Christopher Bishop

# Another view of $\ell_2$-norm regularization

- From the optimization theory[1], we know that

$$\min_{w} \quad J(w) + \lambda R(w)$$

  is equivalent to

$$\begin{aligned} \min_{w} \quad & J(w) \\ \text{such that} \quad & R(w) \leq \eta \end{aligned}$$

  for some $\eta \geq 0$.

- Hence, $\ell_2$-regularized linear regression can be re-formulated as (we only consider $w_j, j > 0$ here)

$$\begin{aligned} \min_{w} \quad & ||Xw - y||_2^2 \\ \text{such that} \quad & ||w||_2^2 \leq \eta \end{aligned}$$

[1] e.g., Boyd and Lieven. Convex Optimization. 2004.

# Visualizing $\ell_2$-norm regularization (2 features)

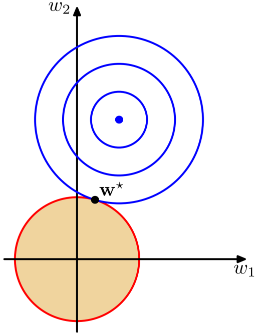



Figure: $w^* = (X^\top X + \lambda I)^{-1} X y$

Figure credit: Christopher Bishop

# $\ell_1$-norm regularization

- Instead of using $\ell_2$-norm, we use $\ell_1$-norm to control the model complexity:

$$\min_w \frac{1}{2}\sum_{i=1}^{m}\Big(\sum_{j=1}^{n} w_0 + w_j \cdot x_{i,j} - y_i\Big)^2 + \lambda \sum_{j=1}^{n}|w_j|$$

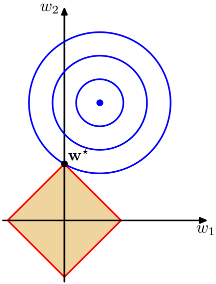
which is equivalent to

$$\begin{aligned} \min_w \quad & \frac{1}{2}\sum_{i=1}^{m}\Big(\sum_{j=1}^{n} w_0 + w_j \cdot x_{i,j} - y_i\Big)^2 \\ \text{such that} \quad & \sum_{j=1}^{n}|w_j| \leq \eta \end{aligned}$$

- Also called LASSO (least absolute shrinkage and selection operator).
- No analytical solution anymore!

# Visualizing $\ell_1$-norm regularization (2 features)

- If $\lambda$ is large enough , the circle is very likely to intersect the diamond at one of the corners.
- This makes $\ell_1$-norm regularization much more likely to make some weights exactly 0.
- In other words, we essentially perform feature selection!



# Comparison of $\ell_2$ and $\ell_1$

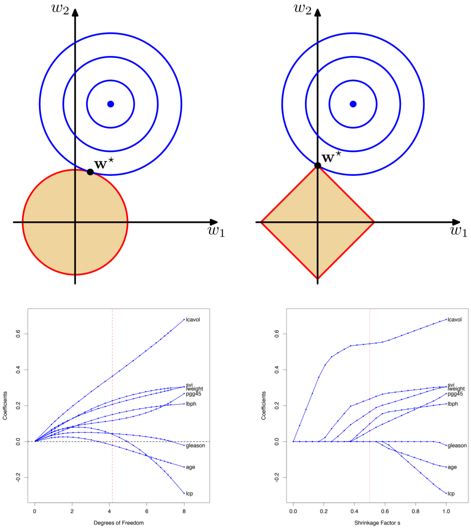



Figure credit: Bishop; Hastie, Tibshirani & Friedman

# Summary of regularization

- Both are commonly used approaches to avoid overfitting.
- Both push the weights towards 0.
- $\ell_2$ produces small, but non-zero weights, while $\ell_1$ is likely to make some weights exactly 0.
- $\ell_1$ optimization is computationally more expensive than $\ell_2$.
- Choose appropriate $\lambda$: cross-validation is often used.