Machine Learning (CE 40477)

Fall 2024

Ali Sharifi-Zarchi

CE Department
Sharif University of Technology

September 30, 2024

1. Generalization
2. Probabilistic regression
3. References

Generalization Overview

Main Idea: The ability of a model to perform well on unseen data

- **Training Set**: $D = \{(x_i, y_i)\}_{i=1}^{n}$
- **Test Set**: New data not seen during training
- **Cost Function**: Measures how well the model fits data

$$J(w) = \sum_{i=1}^{n} (y^{(i)} - h_w(x^{(i)}))^2$$

- **Objective**: Minimize the cost function on unseen data (generalization error)

Expected Test Error

Definition: Expected performance on unseen data

- Test data sampled from the same distribution $p(x, y)$

$$J(w) = \mathbb{E}_{p(x,y)}\left[(y - h_w(x))^2\right]$$

- Approximate using test set $\hat{J}(w)$
- Generalization error is the gap between training and test performance.

Training vs Test Error

Key Concept: Training error measures fit on known data, test error on unseen data

- **Training (empirical) error**:

$$J_{\text{train}}(w) = \frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} - h_w(x^{(i)})\right)^2$$

- **Test error**:

$$J_{\text{test}}(w) = \frac{1}{m}\sum_{i=1}^{m}\left(y_{\text{test}}^{(i)} - h_w(x_{\text{test}}^{(i)})\right)^2$$

- **Goal**: Minimize the test error (generalization).

Overfitting Definition

Concept: A model fits the training data well but performs poorly on the test set

$$J_{\text{train}}(w) \ll J_{\text{test}}(w)$$

- Causes: Model too complex, high variance
- Consequence: Captures noise in training data, fails on unseen data

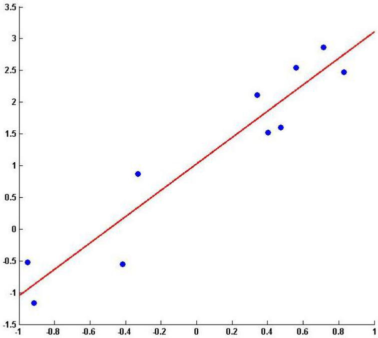
Underfitting Definition

Concept: The model is too simple and cannot capture the structure of the data

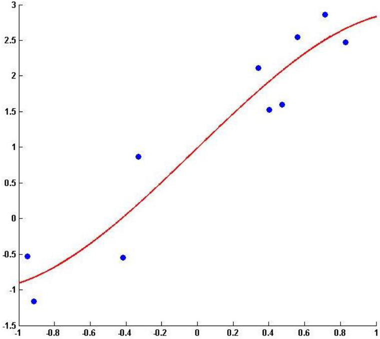
$$J_{\text{train}}(w) \approx J_{\text{test}}(w) \gg 0$$

- Causes: Model lacks complexity, high bias
- Consequence: Poor fit on both training and test data

Generalization: polynomial regression



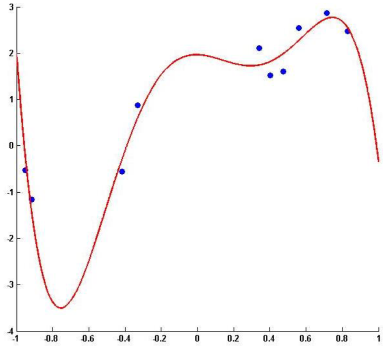
Degree of 1



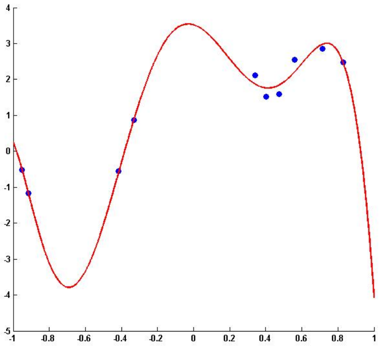
Degree of 3

Example adapted from slides of Dr. Soleymani, ML course, Sharif University of technology.

Overfitting: polynomial regression



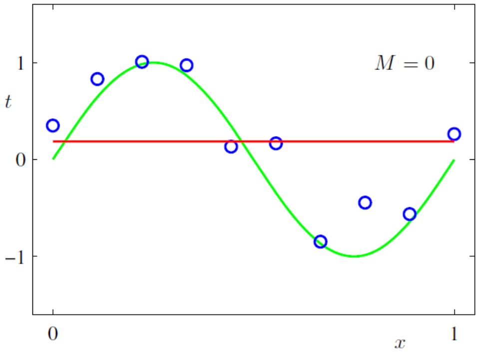
Degree of 5

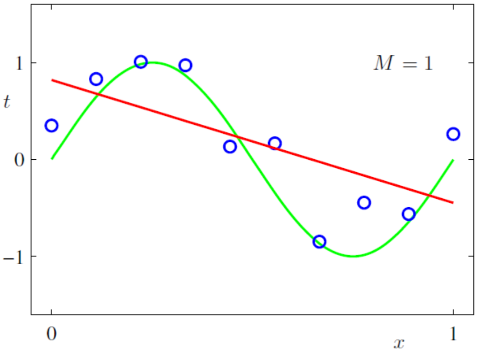


Degree of 7

Example adapted from slides of Dr. Soleymani, ML course, Sharif University of technology.

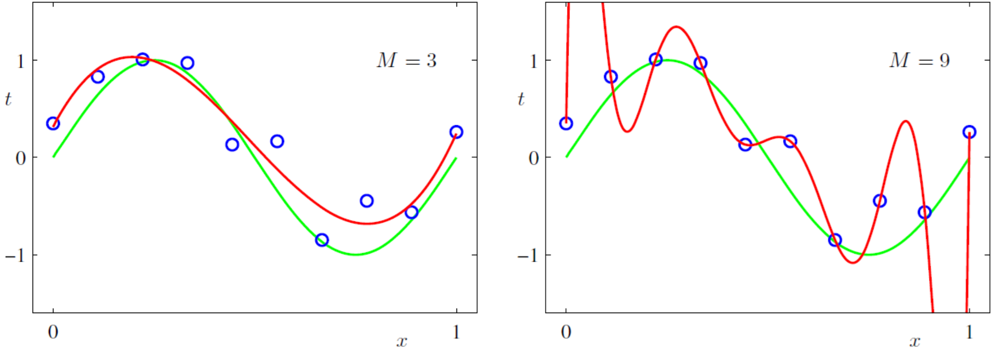
Polynomial regression with various degrees: example





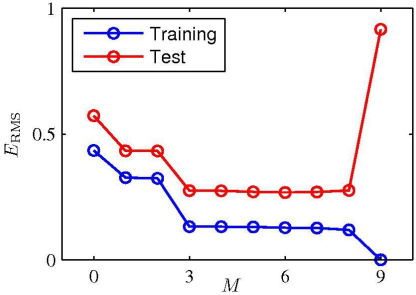
Figures adapted from Machine Learning and Pattern Recognition, Bishop

Polynomial regression with various degrees: example (cont.)



Figures adapted from Machine Learning and Pattern Recognition, Bishop

Root mean squared error



$$E_{RMS} = \sqrt{\frac{\sum_{i=1}^{n} \left(y^{(i)} - f(\mathbf{x}^{(i)}; \mathbf{w}\right)^2}{n}}$$

Figures adapted from Machine Learning and Pattern Recognition, Bishop

Bias-Variance Decomposition

Generalization error decomposition:

$$\mathbb{E}[(y - h_w(x))^2] = (\text{Bias})^2 + \text{Variance} + \text{Noise}$$

- **Bias**: Error due to simplifying assumptions in the model

$$\text{Bias}(x) = \mathbb{E}[h_w(x)] - f(x)$$

- **Variance**: Sensitivity of the model to training data

$$\text{Variance}(x) = \mathbb{E}[(h_w(x) - \mathbb{E}[h_w(x)])^2]$$

- **Noise**: Irreducible error from the inherent randomness in data

Bias-Variance Decomposition Proof

Assume $f(x)$ is the ground truth and observation y is a noisy observation $y = f(x) + \epsilon$ where $\epsilon \mathcal{N}(0, \sigma^2)$. We start with the definition of the expected squared error, which is:

$$\mathbb{E}_{data}\left[\left(\hat{f}(x) - y\right)^2\right] = \mathbb{E}_{data}\left[\left(\hat{f}(x) - f(x) + \epsilon\right)^2\right]$$

$$= \mathbb{E}\left[\left(\hat{f}(x) - f(x)\right)^2 - 2\epsilon\left(\hat{f}(x) - f(x)\right) + \epsilon^2\right]$$

Since we assume the noise ϵ has zero mean and variance σ^2, the term $\mathbb{E}[\epsilon] = 0$, and thus:

$$\mathbb{E}[\epsilon^2] = \sigma^2$$

Since $\mathbb{E}[\epsilon] = 0$ and ϵ is independent of the parenthesis, we can write:

$$\mathbb{E}\left[-2\epsilon\left(\hat{f}(x) - f(x)\right)\right] = 0$$

Bias-Variance Decomposition Proof (cont.)

Now, we decompose the squared difference $\left(\hat{f}(x)-f(x)\right)^2$ as follows:

$$\mathbb{E}\left[\left(\hat{f}(x)-f(x)\right)^2\right]=\mathbb{E}\left[\left(\hat{f}(x)-\mathbb{E}\left[\hat{f}(x)\right]+\mathbb{E}\left[\hat{f}(x)\right]-f(x)\right)^2\right]$$

Expanding this further:

$$=\mathbb{E}\left[\left(\hat{f}(x)-\mathbb{E}\left[\hat{f}(x)\right]\right)^2\right]+\mathbb{E}\left[\left(\mathbb{E}\left[\hat{f}(x)\right]-f(x)\right)^2\right]+2\mathbb{E}\left[\left(\hat{f}(x)-\mathbb{E}\left[\hat{f}(x)\right]\right)\left(\mathbb{E}\left[\hat{f}(x)\right]-f(x)\right)\right]$$

Since $\mathbb{E}[\epsilon A]=\mathbb{E}[\epsilon]\,\mathbb{E}[A]$, A and ϵ are independent and $\mathbb{E}[\epsilon]$ we have $\mathbb{E}[\epsilon A]=0$ thus:

$$\mathbb{E}[\mathbb{E}[\hat{f}(x)]-\hat{f}(x)]=\mathbb{E}\left[\hat{f}(x)\right]-\mathbb{E}\left[\hat{f}(x)\right]=0$$

Bias-Variance Decomposition Proof (cont.)

Thus, the expected squared error becomes:

$$\mathbb{E}_{data}\left[\left(\hat{f}(x_n) - y\right)^2\right] = \text{Variance} + \text{Bias}^2 + \sigma^2$$

where:

- **Variance** is $\mathbb{E}\left[\left(\hat{f}(x) - \mathbb{E}\left[\hat{f}(x)\right]\right)^2\right]$
- **Bias** is $\mathbb{E}\left[\left[\hat{f}(x)\right] - f(x)\right]$
- **Noise** is σ^2

High Bias in Simple Models

Explanation: Simple models, such as linear regression, often underfit

$$h_w(x) = w_0 + w_1 x$$

- Bias remains large even with infinite data

$$\text{Bias}^2 \gg \text{Variance}$$

- Leads to large generalization error

High Variance in Complex Models

Explanation: Complex models tend to overfit

$$h_w(x) = w_0 + w_1 x + w_2 x^2 + \cdots + w_m x^m$$

- Variance dominates when the model is too complex

$$\text{Variance} \gg \text{Bias}$$

- Fits noise, leading to high test error

Bias-Variance Tradeoff

Tradeoff: Balancing between bias and variance is key for optimal performance

- Low complexity: High bias, low variance
- High complexity: Low bias, high variance

Regularization

Purpose: Prevent overfitting by penalizing large weights

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

- Common regularizers: L1 and L2 norms
- λ controls the balance between fit and simplicity

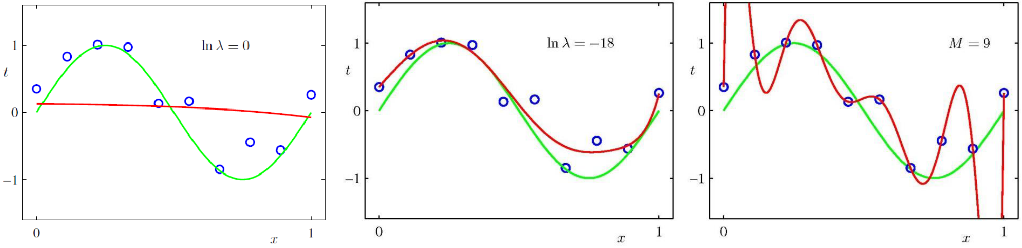
Effect of Regularization Parameter λ

Balancing Fit and Complexity:

$$J_\lambda(w) = J(w) + \lambda \sum_{j=1}^{m} w_j^2 = J(w) + \lambda \mathbf{w}^T \mathbf{w}$$

- Large λ: Forces smaller weights, reduces complexity, increases bias, decreases variance
- Small λ: Allows larger weights, increases complexity, reduces bias, increases variance

Effect of Regularization parameter λ



$$J_\lambda(w) = \sum_{i=1}^{n} \left(t^{(n)} - f(\mathbf{x}^{(n)}; \mathbf{w})\right)^2 + \lambda \mathbf{w}^T \mathbf{w}$$

$$f(\mathbf{x}^{(n)}; \mathbf{w}) = w_0 + w_1 x + \cdots + w_9 x^9$$

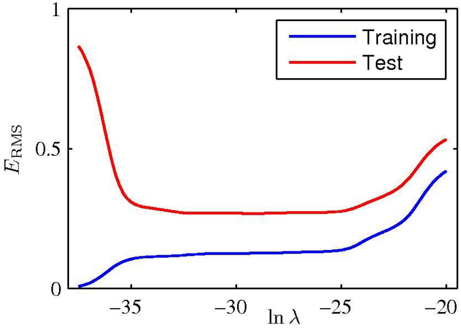
Figures adapted from Machine Learning and Pattern Recognition, Bishop

Effect of regularization on weights

	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
$w_0^\star$	0.35	0.35	0.13
$w_1^\star$	232.37	4.74	-0.05
$w_2^\star$	-5321.83	-0.77	-0.06
$w_3^\star$	48568.31	-31.97	-0.05
$w_4^\star$	-231639.30	-3.89	-0.03
$w_5^\star$	640042.26	55.28	-0.02
$w_6^\star$	-1061800.52	41.32	-0.01
$w_7^\star$	1042400.18	-45.95	-0.00
$w_8^\star$	-557682.99	-91.53	0.00
$w_9^\star$	125201.43	72.68	0.01

Table adapted from Machine Learning and Pattern Recognition, Bishop

Regularization parameter



- λ controls the effective complexity of the model
- hence the degree of overfitting

Figures adapted from Machine Learning and Pattern Recognition, Bishop

Introduction to Regression (Probabilistic Perspective)

- **Objective:** Model the relationship between input $\mathbf{x}$ and output y.
- **Uncertainty:** Output y has an associated uncertainty modeled by a probability distribution.
- **Example:**

$$y = f(\mathbf{x}; \mathbf{w}) + \epsilon \quad , \quad \epsilon \sim \mathcal{N}(0, \sigma^2)$$

- The goal is to learn $f(\mathbf{x}; \mathbf{w})$ to predict y.

Curve Fitting with Noise

- In real-world scenarios, observed output y is noisy.
- **Model: True output plus noise**

$$y = f(\mathbf{x}; \mathbf{w}) + \epsilon$$

- Noise represents unknown or unmodeled factors.
- **Example:** Predicting house prices based on features with inherent unpredictability.

Expected Value of Output

- Best Estimate: The conditional expectation of y given $\mathbf{x}$.

$$\mathbb{E}[y|\mathbf{x}] = f(\mathbf{x};\mathbf{w})$$

- Goal: Learn a function $f(\mathbf{x};\mathbf{w})$ that represents the average behavior of the data.
- Key Point: The model captures the mean of the target variable given input $\mathbf{x}$.

Maximum Likelihood Estimation (MLE)

- **MLE:** A method to estimate parameters that maximize the likelihood of the data.
- Given data $\mathscr{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$, MLE maximizes:

$$L(\mathscr{D}; \mathbf{w}, \sigma^2) = \prod_{i=1}^{n} p(y_i | \mathbf{x}_i, \mathbf{w}, \sigma^2)$$

- MLE finds parameters $\mathbf{w}$ and σ^2 that best explain the data.

Maximum Likelihood Estimation (cont.)

- Instead of maximizing the likelihood, it is often easier to maximize the log-likelihood:

$$\log L(\mathscr{D};\mathbf{w},\sigma^2) = \sum_{i=1}^{n} \log p(y_i|\mathbf{x}_i,\mathbf{w},\sigma^2)$$

- It is because $\log f(x)$ preserves the behaviour of $f(x)$.
- It is also easier to find derivative on summation of terms.

Univariate Linear Function Example

- Assuming Gaussian noise with parameters $(0, \sigma^2)$, probability of observing real output value y is:

$$p(y|\mathbf{x}, \mathbf{w}, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y - f(\mathbf{x}; \mathbf{w}))^2}{2\sigma^2}\right)$$

- For a simple linear model $f(\mathbf{x}; \mathbf{w}) = w_0 + w_1 x$ we have:

$$p(y|x, \mathbf{w}, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y - w_0 - w_1 x)^2}{2\sigma^2}\right)$$

- **Key Observation:** Points far from the fitted line will have a low likelihood value.

Log-Likelihood and Sum of Squares

- Using log-likelihood we have:

$$\log L(\mathcal{D}; \mathbf{w}, \sigma^2) = -n \log \sigma - \frac{n}{2} \log(2\pi) - \frac{1}{2\sigma^2} \sum_{i=1}^{n} (y^{(i)} - f(\mathbf{x}^{(i)}; \mathbf{w}))^2$$

- Since the objective of MLE is to optimize with regards to random variables, we can rule out the constants:

$$\log L(\mathcal{D}; \mathbf{w}, \sigma^2) \sim -\sum_{i=1}^{n} (y^{(i)} - f(\mathbf{x}^{(i)}; \mathbf{w}))^2$$

- **Equivalence:** Maximizing the log-likelihood is equivalent to minimizing the Sum of Squared Errors (SSE):

$$J(\mathbf{w}) = \sum_{i=1}^{n} (y^{(i)} - f(\mathbf{x}^{(i)}; \mathbf{w}))^2$$

Estimating σ^2

- The maximum likelihood estimate of the noise variance σ^2:

$$\hat{\sigma}^2 = \frac{1}{n} \sum_{i=1}^{n} \left(y^{(i)} - f(\mathbf{x}^{(i)}; \hat{\mathbf{w}}) \right)^2$$

- Interpretation: Mean squared error of the predictions.
- Note: σ^2 reflects the noise level in the observations.

1. Generalization
2. Probabilistic regression
3. **References**

Contributions

- **These slides are authored by:**
  - Arshia Gharooni
  - Mahan Bayhaghi

[1] C. M., *Pattern Recognition and Machine Learning.*
Information Science and Statistics, New York, NY: Springer, 1 ed., Aug. 2006.

[2] M. Soleymani Baghshah, “Machine learning.” Lecture slides.

[3] A. Ng and T. Ma, *CS229 Lecture Notes.*

[4] T. Mitchell, *Machine Learning.*
McGraw-Hill series in computer science, New York, NY: McGraw-Hill Professional, Mar. 1997.

[5] Y. S. Abu-Mostafa, M. Magdon-Ismail, and H.-T. Lin, *Learning From Data: A Short Course.*
New York, NY: AMLBook, 2012.

[6] S. Goel, H. Bansal, S. Bhatia, R. A. Rossi, V. Vinay, and A. Grover, “CyCLIP: Cyclic Contrastive Language-Image Pretraining,” *ArXiv*, vol. abs/2205.14459, May 2022.