Recap
Linear Models

- \( H \) contains hypothesis \( h(\tilde{x}) \) as some function of \( \tilde{w}^T \tilde{x} \)

- Algorithm:
  - Focus on \( g = \arg\min_{h \in H} E_{in}(h) \)
  - Gradient descent is one of the common optimization algorithms
Gradient Descent

• Gradient descent algorithm
  • Initialize $\vec{w}(0)$
  • For $t = 0, ...$
    • $\vec{w}(t + 1) \leftarrow \vec{w}(t) - \eta \nabla \vec{w} E_{in}(\vec{w}(t))$
    • Terminate if the stop conditions are met
  • Return the final weights

• Stochastic gradient decent
  • Replace the update step:
    • Randomly choose $n$ from $\{1, ..., N\}$
    • $\vec{w}(t + 1) \leftarrow \vec{w}(t) - \eta \nabla \vec{w} e_n(\vec{w}(t))$

Works for functions where gradient exists everywhere
Nonlinear Transformation

\[ \mathbf{\hat{z}} = \Phi(\mathbf{\hat{x}}) \]

\[ g(\mathbf{\hat{x}}) = g^{(z)}(\Phi(\mathbf{\hat{x}})) = \text{sign}(\mathbf{\hat{w}}^{(z)} T \Phi(\mathbf{\hat{x}})) \]

\[ g^{(z)}(\mathbf{\hat{z}}) = \text{sign}(\mathbf{\hat{w}}^{(z)} T \mathbf{\hat{z}}) \]
MUST Choose \( \Phi \) BEFORE Looking at the Data

- Rely on domain knowledge (feature engineering)
  - Handwriting digit recognition example

- Use common sets of feature transformation
  - Polynomial transformation
  - E.g., 2nd order Polynomial transformation
    \[ \tilde{x} = (1, x_1, x_2), \quad \Phi_2(\tilde{x}) = (1, x_1, x_2, x_1x_2, x_1^2, x_2^2) \]
  - Plus: more powerful (contains circle, ellipse, hyperbola, etc)
  - Minus:
    - More computation/storage
    - Worse generalization error

The VC dimension of d-dim perceptron is \( d+1 \)
Q-th Order Polynomial Transform

• \( \tilde{x} = (1, x_1, ..., x_d) \)
• \( \Phi_1(\tilde{x}) = \tilde{x} \)
• \( \Phi_Q(\tilde{x}) = (\Phi_{Q-1}(\tilde{x}), x_1^Q, x_1^{Q-1}x_2, ..., x_d^Q) \)

• Each element in \( \Phi_Q(\tilde{x}) \) is in the form of \( \sum_{i=1}^{d} x_i^{a_i} \)
  • where \( \sum_{i=1}^{d} a_i \leq Q \), and \( a_i \) is a non-negative integer

• Number of elements in \( \Phi_Q(\tilde{x}) \): \( \binom{Q + d}{Q} \) (including the initial 1)
Structural Hypothesis Sets

• Let $H_Q$ be the linear model for the $\Phi_Q(\vec{x})$ space

• Let $g_Q = \arg\min_{h \in H_Q} E_{in}(h)$
  - $H_1 \subseteq H_2 \subseteq H_3 \subseteq \cdots$
  - $d_{vc}(H_1) \leq d_{vc}(H_2) \leq \cdots$
  - $E_{in}(g_1) \geq E_{in}(g_2) \geq \cdots$
The notes are not intended to be comprehensive. They should be accompanied by lectures and/or textbook. Let me know if you spot errors.
Overfitting

[Adapted from the slides by Malik Magdon-Ismail]
Setup of the Discussion

• Regression with polynomial transform
  • Input: 1-dimensional \( x \)
  • \( \Phi_Q(x) = (1, x, x^2, x^3, \ldots, x^Q) \)
  • \( H_Q = \{h(x) = w_0 + w_1x + w_2x^2 + \cdots + w_Qx^Q\} \)

• \( Q \)-th-order polynomial fit
  • Solve linear regression on the \( \Phi_Q(\vec{x}) \) space using \( H_Q \)

• Looking to minimize \( E_{in}: g_Q = \text{argmin}_{h \in H_Q} E_{in}(h) \)
A Simple Example

• Target $f$: $4^{\text{th}}$ order function
• # data points: $N = 5$
• Small noise:
  • $y = f(x) + \epsilon$ with small $\epsilon$
• $4^{\text{th}}$ order polynomial fit
  • $h(x) = w_0 + w_1 x + w_2 x^2 + w_3 x^3 + w_4 x^4$
  • Find $g_4 = \arg\min_h E_{\text{in}}(h)$
A Simple Example

• Target $f$: 4\textsuperscript{th} order function
• # data points: $N = 5$
• Small noise:
  • $y = f(x) + \epsilon$ with small $\epsilon$
• 4\textsuperscript{th} order polynomial fit
  • $h(x) = w_0 + w_1 x + w_2 x^2 + w_3 x^3 + w_4 x^4$
  • Find $g_4 = \arg\min_h E_{in}(h)$

Classical overfitting: $E_{in} = 0$, but lead to a large $E_{out}$

Fitting the **noise** instead of the data
What is Overfitting?

Fitting the data more than is warranted
Overfitting is Not Just Bad Generalization
Overfitting is Not Just Bad Generalization

Overfitting
Going for lower and lower $E_{in}$ results in higher and higher $E_{out}$
Case Study: 2\textsuperscript{nd} vs 10\textsuperscript{th} Order Polynomial Fit
Which model do you choose for the left problem and why?

$H_2$: 2\textsuperscript{nd} order polynomial fit

$H_{10}$: 10\textsuperscript{th} order polynomial fit
Target Function: $10^{th}$ Order $f$ with Noise

- Irony of two learners $O$ and $R$
- Both **know** the target is $10^{th}$ order
- $O$ chooses $H_1$
- $R$ chooses $H_2$
- $R$ outperforms $O$
Which model do you choose for the right problem and why?

$H_2$: 2nd order polynomial fit

$H_{10}$: 10th order polynomial fit
Simpler $H$ is better even for complex target with no noise
Is There Really “No Noise”?  

Simple $f$ with noise.  

Complex $f$ with no noise.
Is There Really “No Noise”?

$H$ should match **quantity and quality of data**, not (unknown) $f$
Why is $H_2$ Better than $H_{10}$?

When $N$ is small, $E_{out}(g_{10}) \geq E_{out}(g_2)$
A Detailed Experiment

Study the **level of noise** and **target complexity**, and \# data points \( N \)

\[
y = f(x) + \epsilon(x) = \sum_{q=0}^{Q_f} \alpha_q x^q + \epsilon(x)
\]

Noise level: variance \( \sigma^2 \) of \( \epsilon(x) \)

Target complexity: \( Q_f \)

Data set size: \( N \)
The Overfit Measure

• Fit the data set using $H_2$ and $H_{10}$
  • Let $g_2$ and $g_{10}$ be the learned hypothesis

• Overfit measure
  • $E_{out}(g_{10}) - E_{out}(g_2)$
  • This value is large if overfit happens
Overfit Measure: $E_{out}(g_{10}) - E_{out}(g_{2})$
Noise: The part of $y$ we cannot model
Stochastic Noise

We would like to learn from $\bigcirc$:

$$y_n = f(x_n)$$

Unfortunately, we only observe $\bigcirc$:

$$y_n = f(x_n) + \text{‘stochastic noise’}$$

no one can model this

Stochastic Noise: fluctuations/measurement errors we cannot model.
Deterministic Noise

We would like to learn from \( h^*(x_n) \):

\[ y_n = h^*(x_n) \]

Unfortunately, we only observe \( \hat{y}_n \):

\[ y_n = f(x_n) = h^*(x_n) + \text{‘deterministic noise’} \]

\( \mathcal{H} \) cannot model this

**Deterministic Noise:** the part of \( f \) we cannot model.
Deterministic Noise

We would like to learn from $\exists$:
$$y_n = h^*(x_n)$$

Unfortunately, we only observe $\forall$:
$$y_n = f(x_n) = h^*(x_n) + \text{`deterministic noise'}$$

Deterministic Noise: the part of $f$ we cannot model.
Both sources of noises hurt learning

**Stochastic Noise**

- **Source:** random measurement errors
- Re-measure $y_n$
- Stochastic noise changes.
- Change $\mathcal{H}$
- Stochastic noise the same.

**Deterministic Noise**

- **Source:** learner’s $\mathcal{H}$ cannot model $f$
- Re-measure $y_n$
- Deterministic noise the same.
- Change $\mathcal{H}$
- Deterministic noise changes.

We have single $\mathcal{D}$ and fixed $\mathcal{H}$ so we cannot distinguish
Noise and Bias-Variance Decomposition

\[ y = f(\hat{x}) + \epsilon \]

\[ \mathbb{E}[E_{out}(\hat{x})] = \sigma^2 + \text{bias} + \text{variance} \]

- Stochastic Noise
- Deterministic noise
How to Fight Overfitting

• Regularization

• Validation

• (The focus of the next two lectures)