Lecture 9: Linear Regression

Hsuan-Tien Lin (林軒田)
htlin@csie.ntu.edu.tw

Department of Computer Science
& Information Engineering
National Taiwan University
(國立台灣大學資訊工程系)
Roadmap

1. When Can Machines Learn?
2. Why Can Machines Learn?

Lecture 8: Noise and Error

Learning can happen with **target distribution** \( P(y|x) \) and **low** \( E_{in} \) w.r.t. \( err \)

3. How Can Machines Learn?

Lecture 9: Linear Regression

- Linear Regression Problem
- Linear Regression Algorithm
- Generalization Issue
- Linear Regression for Binary Classification

4. How Can Machines Learn Better?
Credit Limit Problem

<table>
<thead>
<tr>
<th>Feature</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>age</td>
<td>23 years</td>
</tr>
<tr>
<td>gender</td>
<td>female</td>
</tr>
<tr>
<td>annual salary</td>
<td>NTD 1,000,000</td>
</tr>
<tr>
<td>year in residence</td>
<td>1 year</td>
</tr>
<tr>
<td>year in job</td>
<td>0.5 year</td>
</tr>
<tr>
<td>current debt</td>
<td>200,000</td>
</tr>
</tbody>
</table>

credit limit? 100,000
Credit Limit Problem

**unknown target function**
\[ f : \mathcal{X} \to \mathcal{Y} \]
(ideal credit limit formula)

**training examples**
\[ \mathcal{D} : (x_1, y_1), \ldots, (x_N, y_N) \]
(historical records in bank)

**hypothesis set**
\[ \mathcal{H} \]
(set of candidate formula)

**learning algorithm**
\[ \mathcal{A} \]

**final hypothesis**
\[ g \approx f \]
(‘learned’ formula to be used)

<table>
<thead>
<tr>
<th>attribute</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>age</td>
<td>23 years</td>
</tr>
<tr>
<td>gender</td>
<td>female</td>
</tr>
<tr>
<td>annual salary</td>
<td>NTD 1,000,000</td>
</tr>
<tr>
<td>year in residence</td>
<td>1 year</td>
</tr>
<tr>
<td>year in job</td>
<td>0.5 year</td>
</tr>
<tr>
<td>current debt</td>
<td>200,000</td>
</tr>
<tr>
<td>credit limit?</td>
<td>100,000</td>
</tr>
</tbody>
</table>
### Credit Limit Problem

**Unknown Target Function**

\[ f : \mathcal{X} \rightarrow \mathcal{Y} \]

*(ideal credit limit formula)*

**Training Examples**

\[ \mathcal{D} : (x_1, y_1), \cdots, (x_N, y_N) \]

*(historical records in bank)*

**Hypothesis Set**

\[ \mathcal{H} \]

*(set of candidate formula)*

**Y = \mathbb{R}: Regression**

<table>
<thead>
<tr>
<th>Age</th>
<th>23 years</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gender</td>
<td>Female</td>
</tr>
<tr>
<td>Annual Salary</td>
<td>NTD 1,000,000</td>
</tr>
<tr>
<td>Year in Residence</td>
<td>1 year</td>
</tr>
<tr>
<td>Year in Job</td>
<td>0.5 year</td>
</tr>
<tr>
<td>Current Debt</td>
<td>200,000</td>
</tr>
</tbody>
</table>

Credit Limit? **100,000**

**Final Hypothesis**

\[ g \approx f \]

*(‘learned’ formula to be used)*

---

Hsuan-Tien Lin  (NTU CSIE)
Linear Regression Hypothesis

<table>
<thead>
<tr>
<th>Feature</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>age</td>
<td>23 years</td>
</tr>
<tr>
<td>annual salary</td>
<td>NTD 1,000,000</td>
</tr>
<tr>
<td>year in job</td>
<td>0.5 year</td>
</tr>
<tr>
<td>current debt</td>
<td>200,000</td>
</tr>
</tbody>
</table>

- For $\mathbf{x} = (x_0, x_1, x_2, \cdots, x_d)$ ‘features of customer’,
Linear Regression Hypothesis

For \( \mathbf{x} = (x_0, x_1, x_2, \cdots, x_d) \) ‘features of customer’, approximate the desired credit limit with a weighted sum:

\[
y \approx \sum_{i=0}^{d} w_i x_i
\]
Linear Regression Hypothesis

- For $\mathbf{x} = (x_0, x_1, x_2, \cdots, x_d)$ ‘features of customer’, approximate the desired credit limit with a weighted sum:

$$y \approx \sum_{i=0}^{d} w_i x_i$$

- linear regression hypothesis: $h(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$
Linear Regression Hypothesis

- For \( \mathbf{x} = (x_0, x_1, x_2, \cdots, x_d) \) ‘features of customer’, approximate the desired credit limit with a weighted sum:

\[
y \approx \sum_{i=0}^{d} w_i x_i
\]

- linear regression hypothesis: \( h(\mathbf{x}) = \mathbf{w}^T \mathbf{x} \)

\( h(\mathbf{x}) \): like perceptron, but without the sign
Illustration of Linear Regression

\[ \mathbf{x} = (\mathbf{x}) \in \mathbb{R} \]
Linear Regression

Linear Regression Problem

Illustration of Linear Regression

\[ \mathbf{x} = (\mathbf{x}) \in \mathbb{R} \]

\[ \mathbf{x} = (x_1, x_2) \in \mathbb{R}^2 \]
Linear Regression

Linear Regression Problem

Illustration of Linear Regression

\[ x = (x) \in \mathbb{R} \]

\[ x = (x_1, x_2) \in \mathbb{R}^2 \]

linear regression: find lines/hyperplanes with small residuals
The Error Measure

popular/historical error measure:

squared error $\text{err}(\hat{y}, y) = (\hat{y} - y)^2$
The Error Measure

popular/historical error measure:
squared error \( \text{err}(\hat{y}, y) = (\hat{y} - y)^2 \)

in-sample

\[
E_{\text{in}}(h) = \frac{1}{N} \sum_{n=1}^{N} (h(x_n) - y_n)^2
\]
The Error Measure

popular/historical error measure:

squared error \( \text{err}(\hat{y}, y) = (\hat{y} - y)^2 \)

in-sample

\[
E_{\text{in}}(w) = \frac{1}{N} \sum_{n=1}^{N} (h(x_n) - y_n)^2
\]
The Error Measure

- **popular/historical error measure:**
  - squared error $\text{err}(\hat{y}, y) = (\hat{y} - y)^2$

**in-sample**

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^{N} (h(\mathbf{x}_n) - y_n)^2$$

**out-of-sample**

$$E_{\text{out}}(\mathbf{w}) = \mathbb{E}_{(\mathbf{x}, y) \sim P} (\mathbf{w}^T \mathbf{x} - y)^2$$
The Error Measure

popular/historical error measure:
squared error \( \text{err}(\hat{y}, y) = (\hat{y} - y)^2 \)

\[
E_{\text{in}}(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^{N} (h(x_n) - y_n)^2
\]

\[
E_{\text{out}}(\mathbf{w}) = \mathbb{E}_{(x,y) \sim P} (\mathbf{w}^T \mathbf{x} - y)^2
\]

next: how to minimize \( E_{\text{in}}(\mathbf{w}) \)?
Consider using linear regression hypothesis \( h(x) = \mathbf{w}^T \mathbf{x} \) to predict the credit limit of customers \( \mathbf{x} \). Which feature below shall have a positive weight in a **good hypothesis** for the task?

1. birth month
2. monthly income
3. current debt
4. number of credit cards owned

Reference Answer: 2

Customers with higher monthly income should naturally be given a higher credit limit, which is captured by the positive weight on the 'monthly income' feature.
Consider using linear regression hypothesis $h(x) = w^T x$ to predict the credit limit of customers $x$. Which feature below shall have a positive weight in a good hypothesis for the task?

1. birth month
2. monthly income
3. current debt
4. number of credit cards owned

**Reference Answer: 2**

Customers with higher monthly income should naturally be given a higher credit limit, which is captured by the positive weight on the ‘monthly income’ feature.
Matrix Form of $E_{\text{in}}(w)$

$$E_{\text{in}}(w) = \frac{1}{N} \sum_{n=1}^{N} (w^T x_n - y_n)^2$$
Matrix Form of $E_{\text{in}}(w)$

\[
E_{\text{in}}(w) = \frac{1}{N} \sum_{n=1}^{N} (w^T x_n - y_n)^2 = \frac{1}{N} \sum_{n=1}^{N} (x_n^T w - y_n)^2
\]
Matrix Form of $E_{\text{in}}(w)$

$$E_{\text{in}}(w) = \frac{1}{N} \sum_{n=1}^{N} (w^T x_n - y_n)^2 = \frac{1}{N} \sum_{n=1}^{N} (x_n^T w - y_n)^2$$
Matrix Form of $E_{in}(\mathbf{w})$

\[
E_{in}(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^{N} (\mathbf{w}^T \mathbf{x}_n - y_n)^2 = \frac{1}{N} \sum_{n=1}^{N} (\mathbf{x}_n^T \mathbf{w} - y_n)^2
\]

\[
= \frac{1}{N} \left\| \begin{array}{c} \mathbf{x}_1^T \mathbf{w} - y_1 \\ \mathbf{x}_2^T \mathbf{w} - y_2 \\ \vdots \\ \mathbf{x}_N^T \mathbf{w} - y_N \\ \end{array} \right\|_2^2
\]
Matrix Form of $E_{in}(\mathbf{w})$

\[
E_{in}(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^{N} (\mathbf{w}^T \mathbf{x}_n - y_n)^2 = \frac{1}{N} \sum_{n=1}^{N} (\mathbf{x}_n^T \mathbf{w} - y_n)^2
\]

\[
= \frac{1}{N} \begin{bmatrix}
\mathbf{x}_1^T \mathbf{w} - y_1 \\
\mathbf{x}_2^T \mathbf{w} - y_2 \\
\vdots \\
\mathbf{x}_N^T \mathbf{w} - y_N
\end{bmatrix}^2
\]
Matrix Form of $E_{in}(\mathbf{w})$

\[
E_{in}(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^{N} (\mathbf{w}^T \mathbf{x}_n - y_n)^2 = \frac{1}{N} \sum_{n=1}^{N} (\mathbf{x}_n^T \mathbf{w} - y_n)^2
\]

\[
= \frac{1}{N} \left\| \begin{bmatrix} \mathbf{x}_1^T \mathbf{w} - y_1 \\ \mathbf{x}_2^T \mathbf{w} - y_2 \\ \vdots \\ \mathbf{x}_N^T \mathbf{w} - y_N \end{bmatrix} \right\|^2 = \frac{1}{N} \left\| \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_N^T \end{bmatrix} \mathbf{w} - \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \right\|^2
\]
Matrix Form of $E_{in}(w)$

$$E_{in}(w) = \frac{1}{N} \sum_{n=1}^{N} (w^T x_n - y_n)^2 = \frac{1}{N} \sum_{n=1}^{N} (x_n^T w - y_n)^2$$

$$= \frac{1}{N} \left\| \begin{bmatrix} x_1^T w - y_1 \\
 x_2^T w - y_2 \\
 \vdots \\
 x_N^T w - y_N \end{bmatrix} \right\|_2^2$$

$$= \frac{1}{N} \left\| \begin{bmatrix} -x_1^T & - & - \\
 -x_2^T & - & - \\
 \vdots & & \vdots \\
 -x_N^T & - & - \end{bmatrix} w - \begin{bmatrix} y_1 \\
 y_2 \\
 \vdots \\
 y_N \end{bmatrix} \right\|_2^2$$
Matrix Form of $E_{in}(\mathbf{w})$

$$E_{in}(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^{N} (\mathbf{w}^T \mathbf{x}_n - y_n)^2 = \frac{1}{N} \sum_{n=1}^{N} (\mathbf{x}_n^T \mathbf{w} - y_n)^2$$

$$= \frac{1}{N} \begin{bmatrix} \mathbf{x}_1^T \mathbf{w} - y_1 \\ \mathbf{x}_2^T \mathbf{w} - y_2 \\ \vdots \\ \mathbf{x}_N^T \mathbf{w} - y_N \end{bmatrix}^2$$

$$= \frac{1}{N} \begin{bmatrix} - \mathbf{x}_1^T \\ - \mathbf{x}_2^T \\ \vdots \\ - \mathbf{x}_N^T \end{bmatrix} \mathbf{w} - \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}^2$$

$$= \frac{1}{N} \| \begin{bmatrix} \mathbf{X} & \mathbf{w} \end{bmatrix} - \mathbf{y} \|_2^2$$
\[ \min_w E_{\text{in}}(w) = \frac{1}{N} \|Xw - y\|^2 \]
Linear Regression

Linear Regression Algorithm

\[ \min_w E_{in}(w) = \frac{1}{N} \|Xw - y\|^2 \]

- \( E_{in}(w) \): continuous, differentiable, **convex**
\[ \min_{\mathbf{w}} E_{\text{in}}(\mathbf{w}) = \frac{1}{N} \| \mathbf{Xw} - \mathbf{y} \|^2 \]

- \( E_{\text{in}}(\mathbf{w}) \): continuous, differentiable, **convex**
- necessary condition of ‘best’ \( \mathbf{w} \)

\[
\nabla E_{\text{in}}(\mathbf{w}) \equiv \begin{bmatrix}
\frac{\partial E_{\text{in}}}{\partial w_0}(\mathbf{w}) \\
\frac{\partial E_{\text{in}}}{\partial w_1}(\mathbf{w}) \\
\vdots \\
\frac{\partial E_{\text{in}}}{\partial w_d}(\mathbf{w})
\end{bmatrix} = \begin{bmatrix}
0 \\
0 \\
\cdots \\
0
\end{bmatrix}
\]

—not possible to ‘roll down’
\[
\min_w E_{\text{in}}(w) = \frac{1}{N} \|Xw - y\|^2
\]

- \(E_{\text{in}}(w)\): continuous, differentiable, **convex**
- necessary condition of ‘best’ \(w\)

\[
\nabla E_{\text{in}}(w) \equiv \begin{bmatrix}
\frac{\partial E_{\text{in}}}{\partial w_0}(w) \\
\frac{\partial E_{\text{in}}}{\partial w_1}(w) \\
\vdots \\
\frac{\partial E_{\text{in}}}{\partial w_d}(w)
\end{bmatrix} = \begin{bmatrix}
0 \\
0 \\
\cdots \\
0
\end{bmatrix}
\]

—*not possible* to ‘roll down’

**task:** find \(w_{\text{LIN}}\) such that \(\nabla E_{\text{in}}(w_{\text{LIN}}) = 0\)
The Gradient $\nabla E_{in}(w)$

$$E_{in}(w) = \frac{1}{N} \|Xw - y\|^2 = \frac{1}{N} \left( w^T w - 2w^T X^T y + y^T y \right)$$
The Gradient $\nabla E_{\text{in}}(w)$

$$E_{\text{in}}(w) = \frac{1}{N} \|Xw - y\|^2 = \frac{1}{N} \left( w^T X^T X w - 2w^T X^T y + y^T y \right)$$
The Gradient $\nabla E_{\text{in}}(w)$

$$E_{\text{in}}(w) = \frac{1}{N} \|Xw - y\|^2 = \frac{1}{N} \left( w^T X^T X w - 2w^T X^T y + y^T y \right)$$
The Gradient $\nabla E_{in}(\mathbf{w})$

\[
E_{in}(\mathbf{w}) = \frac{1}{N} \| \mathbf{Xw} - \mathbf{y} \|^2 = \frac{1}{N} \left( \mathbf{w}^T \mathbf{X}^T \mathbf{X} \mathbf{w} - 2\mathbf{w}^T \mathbf{X}^T \mathbf{y} + \mathbf{y}^T \mathbf{y} \right) \]

one $\mathbf{w}$ only

\[
E_{in}(\mathbf{w}) = \frac{1}{N} \left( aw^2 - 2bw + c \right) \]
The Gradient $\nabla E_{in}(w)$

$$E_{in}(w) = \frac{1}{N} \|Xw - y\|^2 = \frac{1}{N} \left( w^T X^T X w - 2w^T X^T y + y^T y \right)$$

**one $w$ only**

$$E_{in}(w) = \frac{1}{N} \left( aw^2 - 2bw + c \right)$$

$$\nabla E_{in}(w) = \frac{1}{N} (2aw - 2b)$$

**simple! :-)**
The Gradient $\nabla E_{in}(w)$

$$E_{in}(w) = \frac{1}{N} \|Xw - y\|^2 = \frac{1}{N} \left( w^T X^T X w - 2w^T X^T y + y^T y \right)$$

**one $w$ only**

$$E_{in}(w) = \frac{1}{N} \left( aw^2 - 2bw + c \right)$$

$$\nabla E_{in}(w) = \frac{1}{N} \left( 2aw - 2b \right)$$

**vector $w$**

$$E_{in}(w) = \frac{1}{N} \left( w^T A w - 2w^T b + c \right)$$

**simple! :-)**
The Gradient $\nabla E_{in}(w)$

$$E_{in}(w) = \frac{1}{N} \|Xw - y\|^2 = \frac{1}{N} \left( w^T X^T X w - 2w^T X^T y + y^T y \right)$$

**one $w$ only**

$$E_{in}(w) = \frac{1}{N} \left( aw^2 - 2bw + c \right)$$

$$\nabla E_{in}(w) = \frac{1}{N} (2aw - 2b)$$

simple! :-)

**vector $w$**

$$E_{in}(w) = \frac{1}{N} \left( w^T A w - 2w^T b + c \right)$$

$$\nabla E_{in}(w) = \frac{1}{N} (2Aw - 2b)$$

similar (derived by definition)
The Gradient $\nabla E_{\text{in}}(w)$

$$E_{\text{in}}(w) = \frac{1}{N} \|Xw - y\|^2 = \frac{1}{N} \left( w^T X^T X w - 2w^T X^T y + y^T y \right)$$

**one $w$ only**

$$E_{\text{in}}(w) = \frac{1}{N} \left( aw^2 - 2bw + c \right)$$

$$\nabla E_{\text{in}}(w) = \frac{1}{N} \left( 2aw - 2b \right)$$

**vector $w$**

$$E_{\text{in}}(w) = \frac{1}{N} \left( w^T A w - 2w^T b + c \right)$$

$$\nabla E_{\text{in}}(w) = \frac{1}{N} \left( 2Aw - 2b \right)$$

simple! :-)

similar (derived by definition)

$$\nabla E_{\text{in}}(w) = \frac{2}{N} \left( X^T Xw - X^T y \right)$$
Optimal Linear Regression Weights

task: find $w_{\text{LIN}}$ such that
\[
\frac{2}{N} (X^T X w - X^T y) = \nabla E_{\text{in}}(w) = 0
\]
Optimal Linear Regression Weights

task: find $w_{\text{LIN}}$ such that $\frac{2}{N} (X^T X w - X^T y) = \nabla E_{\text{in}}(w) = 0$

invertible $X^T X$

- easy! unique solution

$$w_{\text{LIN}} = \left( X^T X \right)^{-1} X^T y$$
Optimal Linear Regression Weights

Task: find $w_{\text{LIN}}$ such that

$$\frac{2}{N} (X^T Xw - X^T y) = \nabla E_{\text{in}}(w) = 0$$

invertible $X^T X$

- **easy!** unique solution

$$w_{\text{LIN}} = \left(X^T X\right)^{-1} X^T y$$

pseudo-inverse $X^\dagger$
Optimal Linear Regression Weights

Task: find $w_{LIN}$ such that
\[ \frac{2}{N} (X^T Xw - X^T y) = \nabla E_{in}(w) = 0 \]

Invertible $X^T X$

- Easy! Unique solution
  \[ w_{LIN} = \left( X^T X \right)^{-1} X^T y \]
  - Pseudo-inverse $x^+$
  - Often the case because $N \gg d + 1$
Optimal Linear Regression Weights

task: find $w_{\text{LIN}}$ such that $\frac{2}{N} (X^T X w - X^T y) = \nabla E_{\text{in}}(w) = 0$

invertible $X^T X$

- easy! unique solution

$$w_{\text{LIN}} = \left( X^T X \right)^{-1} X^T y$$

- pseudo-inverse $X^\dagger$

- often the case because $N \gg d + 1$

singular $X^T X$

- many optimal solutions

Hsuan-Tien Lin  (NTU CSIE)  Machine Learning Foundations  10/23
linear regression

**Linear Regression Algorithm**

**Optimal Linear Regression Weights**

(task: find $w_{\text{LIN}}$ such that $\frac{2}{N} (X^T X w - X^T y) = \nabla E_{\text{in}}(w) = 0$)

**Invertible $X^T X$**

- **easy!** unique solution
  
  $$w_{\text{LIN}} = \left( X^T X \right)^{-1} X^T y$$

  pseudo-inverse $X^\dagger$

- often the case because $N \gg d + 1$

**Singular $X^T X$**

- **many** optimal solutions
  
  - one of the solutions
    
    $$w_{\text{LIN}} = X^\dagger y$$

  by defining $X^\dagger$ in other ways

Hsuan-Tien Lin  (NTU CSIE)  
Machine Learning Foundations
Optimal Linear Regression Weights

Task: find $\mathbf{w}_{\text{LIN}}$ such that
\[
\frac{2}{N} (\mathbf{X}^T \mathbf{X} \mathbf{w} - \mathbf{X}^T \mathbf{y}) = \nabla E_{\text{in}}(\mathbf{w}) = 0
\]

invertible $\mathbf{X}^T \mathbf{X}$

- easy! unique solution
  \[
  \mathbf{w}_{\text{LIN}} = \left( \mathbf{X}^T \mathbf{X} \right)^{-1} \mathbf{X}^T \mathbf{y}
  \]
  pseudo-inverse $\mathbf{X}^\dagger$
- often the case because $N \gg d + 1$

singular $\mathbf{X}^T \mathbf{X}$

- many optimal solutions
- one of the solutions
  \[
  \mathbf{w}_{\text{LIN}} = \mathbf{X}^\dagger \mathbf{y}
  \]
  by defining $\mathbf{X}^\dagger$ in other ways

practical suggestion:

use well-implemented $\dagger$ routine
instead of $(\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T$
for numerical stability when almost-singular
Linear Regression Algorithm

1. from $\mathcal{D}$, construct input matrix $X$ and output vector $y$ by

$$X = \begin{bmatrix}
- & - & x_1^T & - & - \\
- & - & x_2^T & - & - \\
& & \ldots & & \\
- & - & x_N^T & - & - \\
\end{bmatrix}_{N \times (d+1)}$$

$$y = \begin{bmatrix}
y_1 \\
y_2 \\
\ldots \\
y_N \\
\end{bmatrix}_{N \times 1}$$
Linear Regression Algorithm

1. From $\mathcal{D}$, construct input matrix $X$ and output vector $y$ by

$$X = \begin{bmatrix} \cdots & x_1^T & \cdots \\ \cdots & x_2^T & \cdots \\ \cdots & \cdots & \cdots \\ \cdots & x_N^T \end{bmatrix}_{N \times (d+1)} \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \cdots \\ y_N \end{bmatrix}_{N \times 1}$$

2. Calculate pseudo-inverse $X^\dagger$
Linear Regression Algorithm

1. From $\mathcal{D}$, construct input matrix $X$ and output vector $y$ by

$$X = \begin{bmatrix} - & - & x_1^T & - & - \\ - & - & x_2^T & - & - \\ & & \ldots & & \\ - & - & x_N^T & - & - \end{bmatrix}_{N \times (d+1)} \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \ldots \\ y_N \end{bmatrix}_{N \times 1}$$

2. Calculate pseudo-inverse

$$X^\dagger = (d+1) \times N$$

3. Return

$$w_{\text{LIN}} = X^\dagger y = (d+1) \times 1$$
Linear Regression Algorithm

1. from $\mathcal{D}$, construct input matrix $X$ and output vector $y$ by

$$X = \begin{bmatrix}
\vdots & - & - & x_1^T & - & - \\
\vdots & - & - & x_2^T & - & - \\
\vdots & - & - & \cdots & - & - \\
\vdots & - & - & x_N^T & - & - \\
\end{bmatrix}_{N \times (d+1)}$$

$$y = \begin{bmatrix}
y_1 \\
y_2 \\
\vdots \\
y_N \\
\end{bmatrix}_{N \times 1}$$

2. calculate pseudo-inverse

$$X^\dagger \in (d+1) \times N$$

3. return $w_{\text{LIN}} = X^\dagger y \in (d+1) \times 1$

simple and efficient
with good $\dagger$ routine
Fun Time

After getting $\mathbf{w}_{\text{LIN}}$, we can calculate the predictions $\hat{y}_n = \mathbf{w}_{\text{LIN}}^T \mathbf{x}_n$. If all $\hat{y}_n$ are collected in a vector $\hat{\mathbf{y}}$ similar to how we form $\mathbf{y}$, what is the matrix formula of $\hat{\mathbf{y}}$?

1. $\mathbf{y}$
2. $\mathbf{X}^T \mathbf{y}$
3. $\mathbf{X}^\dagger \mathbf{y}$
4. $\mathbf{X}^\dagger \mathbf{X}^T \mathbf{y}$

Reference Answer:

Note that $\hat{\mathbf{y}} = \mathbf{X} \mathbf{w}_{\text{LIN}}$. Then, a simple substitution of $\mathbf{w}_{\text{LIN}}$ reveals the answer.
After getting $w_{\text{LIN}}$, we can calculate the predictions $\hat{y}_n = w_{\text{LIN}}^T x_n$. If all $\hat{y}_n$ are collected in a vector $\hat{y}$ similar to how we form $y$, what is the matrix formula of $\hat{y}$?

1. $y$
2. $XX^T y$
3. $XX^\dagger y$
4. $XX^\dagger XX^T y$

**Reference Answer:** 3

Note that $\hat{y} = Xw_{\text{LIN}}$. Then, a simple substitution of $w_{\text{LIN}}$ reveals the answer.
Is Linear Regression a ‘Learning Algorithm’?

\[ w_{\text{LIN}} = X^\dagger y \]
Is Linear Regression a ‘Learning Algorithm’?

\[ w_{\text{LIN}} = X^\dagger y \]

No!

- analytic (closed-form) solution, ‘instantaneous’
Is Linear Regression a ‘Learning Algorithm’?

\[ w_{LIN} = X^\dagger y \]

No!

- analytic (closed-form) solution, ‘instantaneous’
- not improving \( E_{in} \) nor \( E_{out} \) iteratively
Is Linear Regression a ‘Learning Algorithm’?

\[ \mathbf{w}_{\text{LIN}} = \mathbf{X}^\dagger \mathbf{y} \]

**No!**
- analytic (closed-form) solution, ‘instantaneous’
- not improving \( E_{\text{in}} \) nor \( E_{\text{out}} \) iteratively

**Yes!**
- good \( E_{\text{in}} \)?
  - yes, optimal!
Is Linear Regression a ‘Learning Algorithm’?

No!
- analytic (closed-form) solution, ‘instantaneous’
- not improving $E_{\text{in}}$ nor $E_{\text{out}}$ iteratively

Yes!
- good $E_{\text{in}}$? yes, optimal!
- good $E_{\text{out}}$? yes, finite $d_{\text{VC}}$ like perceptrons
Is Linear Regression a ‘Learning Algorithm’?

\[ w_{\text{LIN}} = X^\dagger y \]

No!
- analytic (\text{closed-form}) solution, ‘instantaneous’
- not improving \( E_{\text{in}} \) nor \( E_{\text{out}} \) iteratively

Yes!
- good \( E_{\text{in}} \)?
  yes, optimal!
- good \( E_{\text{out}} \)?
  yes, finite \( d_{\text{VC}} \) like perceptrons
- improving iteratively?
  somewhat, within an iterative pseudo-inverse routine
Is Linear Regression a ‘Learning Algorithm’?

\[ \mathbf{w}_{\text{LIN}} = \mathbf{X}^\dagger \mathbf{y} \]

No!
- analytic (closed-form) solution, ‘instantaneous’
- not improving \( E_{\text{in}} \) nor \( E_{\text{out}} \) iteratively

Yes!
- good \( E_{\text{in}} \)?
  yes, optimal!
- good \( E_{\text{out}} \)?
  yes, finite \( d_{\text{VC}} \) like perceptrons
- improving iteratively?
  somewhat, within an iterative pseudo-inverse routine

if \( E_{\text{out}}(\mathbf{w}_{\text{LIN}}) \) is good, learning ‘happened’!
Benefit of Analytic Solution: ‘Simpler-than-VC’ Guarantee

$$\overline{E_{in}} = \mathcal{E}_{D \sim P^N} \left\{ E_{in}(w_{\text{LIN}} \text{ w.r.t. } D) \right\}$$
Benefit of Analytic Solution:
‘Simpler-than-VC’ Guarantee

\[
\bar{E}_{\text{in}} = \mathcal{E}_{D \sim P^N}\left\{E_{\text{in}}(\mathbf{w}_{\text{LIN}} \, \text{w.r.t.} \, D)\right\} \overset{\text{to be shown}}{=} \text{noise level} \cdot \left(1 - \frac{d+1}{N}\right)
\]
Benefit of Analytic Solution: ‘Simpler-than-VC’ Guarantee

\[ E_{\text{in}} = \mathbb{E}_{D \sim \mathcal{P}^N} \left\{ E_{\text{in}}(\mathbf{w}_{\text{LIN}} \text{ w.r.t. } D) \right\} \text{ to be shown} = \text{noise level} \cdot \left( 1 - \frac{d+1}{N} \right) \]

\[ E_{\text{in}}(\mathbf{w}_{\text{LIN}}) = \frac{1}{N} \| \mathbf{y} - \hat{\mathbf{y}} \|^2 \]
 predictions
Benefit of Analytic Solution:
‘Simpler-than-VC’ Guarantee

\[
\bar{E}_{\text{in}} = \mathcal{E}_{D \sim P_N} \left\{ E_{\text{in}}(w_{\text{LIN}} \text{ w.r.t. } D) \right\} \text{ to be shown}
\]

\[
= \text{noise level} \cdot \left(1 - \frac{d+1}{N}\right)
\]

\[
E_{\text{in}}(w_{\text{LIN}}) = \frac{1}{N} \| y - \hat{y} \|^2 = \frac{1}{N} \| y - X X^\dagger y \|_2^2
\]

predictions
Benefit of Analytic Solution: ‘Simpler-than-VC’ Guarantee

\[
E_{in} = \mathcal{E}_{D \sim P^N} \left\{ E_{in}(w_{\text{LIN}} \text{ w.r.t. } D) \right\} \quad \text{to be shown}
\]

\[
\overset{\sim}{E} = \text{noise level} \cdot \left(1 - \frac{d+1}{N}\right)
\]

\[
E_{in}(w_{\text{LIN}}) = \frac{1}{N} \| y - \hat{y} \|^2 = \frac{1}{N} \| y - XX^\dagger y \|^2_{w_{\text{LIN}}}
\]

\[
= \frac{1}{N} \| (I - XX^\dagger)y \|^2_{\text{identity}}
\]
Benefit of Analytic Solution:
‘Simpler-than-VC’ Guarantee

\[ \bar{E}_{\text{in}} = \mathcal{E}_{D \sim P^N} \left\{ E_{\text{in}}(w_{\text{LIN}} \text{ w.r.t. } D) \right\} \]  
\[ \text{to be shown} \quad \text{noise level} \cdot \left( 1 - \frac{d+1}{N} \right) \]

\[ E_{\text{in}}(w_{\text{LIN}}) = \frac{1}{N} \| y - \hat{y} \|^2 \]
\[ = \frac{1}{N} \| y - XX^\dagger y \|_2^2 \]
\[ = \frac{1}{N} \| (I - XX^\dagger) y \|_2^2 \]

call \( XX^\dagger \) the \text{hat matrix} \( H \)
because it \text{puts } \wedge \text{ on } y
Geometric View of Hat Matrix

\( y \) in \( \mathbb{R}^N \)

- \( \hat{y} = Xw_{\text{LIN}} \) within the span of \( X \) columns

\( y - \hat{y} \) smallest:

\( y - \hat{y} \perp \text{span of } X \)

\( \mathbb{H} \): project \( y \) to \( \hat{y} \) in \( \text{span } X \)

\( I - \mathbb{H} \): transform \( y \) to \( y - \hat{y} \perp \text{span } X \)

claim: \( \text{trace}\left(I - \mathbb{H}\right) = N - (d + 1) \).

Why? :-)

Hsuan-Tien Lin (NTU CSIE)
Linear Regression

Generalization Issue

**Geometric View of Hat Matrix**

\[ y = \hat{y} = Xw_{\text{LIN}} \text{ within the span of } X \text{ columns} \]

\[ y - \hat{y} \text{ smallest: } y - \hat{y} \perp \text{span} \]
Geometric View of Hat Matrix

\[ \hat{y} = Xw_{\text{LIN}} \text{ within the span of } X \text{ columns} \]
\[ y - \hat{y} \text{ smallest: } y - \hat{y} \perp \text{span} \]
\[ H: \text{project } y \text{ to } \hat{y} \in \text{span} \]
\( \hat{y} = Xw_{\text{LIN}} \) within the span of \( X \) columns

\( y - \hat{y} \) smallest: \( y - \hat{y} \perp \text{span} \)

\( H \): project \( y \) to \( \hat{y} \in \text{span} \)

\( I - H \): transform \( y \) to \( y - \hat{y} \perp \text{span} \)

in \( \mathbb{R}^N \)
Geometric View of Hat Matrix

\[
\hat{y} = X w_{\text{LIN}} \quad \text{within the span of } X \text{ columns}
\]

\[
y - \hat{y} \quad \text{smallest: } y - \hat{y} \perp \text{span}
\]

\[
H: \text{project } y \text{ to } \hat{y} \in \text{span}
\]

\[
I - H: \text{transform } y \text{ to } y - \hat{y} \perp \text{span}
\]

Claim: \[\text{trace}(I - H) = N - (d + 1). \text{ Why? :-)}\]
An Illustrative ‘Proof’

- if \( y \) comes from some ideal \( f(X) \in \text{span} \) plus noise
An Illustrative ‘Proof’

- if **y** comes from some ideal $f(X) \in \text{span}$ plus noise
- noise transformed by $I - H$ to be $y - \hat{y}$
An Illustrative ‘Proof’

- If \( y \) comes from some ideal \( f(X) \in \text{span} \) plus noise
- **noise** transformed by \( I - H \) to be \( y - \hat{y} \)

\[
E_{\text{in}}(w_{\text{LIN}}) = \frac{1}{N} \| y - \hat{y} \|^2 = \frac{1}{N} \| (I - H)\text{noise} \|^2
\]
An Illustrative ‘Proof’

- If $y$ comes from some ideal $f(X) \in \text{span}$ plus noise
- Noise transformed by $I - H$ to be $y - \hat{y}$

\[
E_{\text{in}}(w_{\text{LIN}}) = \frac{1}{N} \| y - \hat{y} \|^2 = \frac{1}{N} \| (I - H)\text{noise} \|^2 \\
= \frac{1}{N} (N - (d + 1)) \| \text{noise} \|^2
\]
An Illustrative ‘Proof’

- if \( y \) comes from some ideal \( f(X) \in \text{span} \) plus noise
- noise transformed by \( I - H \) to be \( y - \hat{y} \)

\[
E_{\text{in}}(w_{\text{LIN}}) = \frac{1}{N} \| y - \hat{y} \|^2 = \frac{1}{N} \| (I - H)\text{noise} \|^2 \\
= \frac{1}{N} (N - (d + 1)) \| \text{noise} \|^2
\]

\[
\overline{E_{\text{in}}} = \text{noise level} \cdot \left(1 - \frac{d+1}{N}\right)
\]
Linear Regression  Generalization Issue

An Illustrative ‘Proof’

- if \( y \) comes from some ideal \( f(X) \in \text{span} \) plus noise
- noise transformed by \( I - H \) to be \( y - \hat{y} \)

\[
E_{\text{in}}(w_{\text{LIN}}) = \frac{1}{N} \| y - \hat{y} \|^2 = \frac{1}{N} \| (I - H)\text{noise} \|^2 \\
= \frac{1}{N} (N - (d + 1)) \| \text{noise} \|^2
\]

\[
E_{\text{in}} = \text{noise level} \cdot (1 - \frac{d+1}{N}) \\
E_{\text{out}} = \text{noise level} \cdot (1 + \frac{d+1}{N}) \text{ (complicated!)}
\]
\[
\begin{align*}
\overline{E_{\text{out}}} &= \text{noise level} \cdot \left(1 + \frac{d+1}{N}\right) \\
\overline{E_{\text{in}}} &= \text{noise level} \cdot \left(1 - \frac{d+1}{N}\right)
\end{align*}
\]
The Learning Curve

\[
\overline{E_{\text{out}}} = \text{noise level} \cdot \left(1 + \frac{d+1}{N}\right)
\]

\[
\overline{E_{\text{in}}} = \text{noise level} \cdot \left(1 - \frac{d+1}{N}\right)
\]

- both converge to \(\sigma^2\) (noise level) for \(N \to \infty\)
The Learning Curve

\[
\overline{E_{\text{out}}} = \text{noise level} \cdot (1 + \frac{d+1}{N})
\]
\[
\overline{E_{\text{in}}} = \text{noise level} \cdot (1 - \frac{d+1}{N})
\]

- both converge to \( \sigma^2 \) (noise level) for \( N \to \infty \)
- expected generalization error: \( \frac{2(d+1)}{N} \)
  — similar to worst-case guarantee from VC
The Learning Curve

\[ E_{\text{out}} = \text{noise level} \cdot \left(1 + \frac{d+1}{N}\right) \]

\[ E_{\text{in}} = \text{noise level} \cdot \left(1 - \frac{d+1}{N}\right) \]

- both converge to \( \sigma^2 \text{ (noise level)} \) for \( N \to \infty \)
- expected generalization error: \( \frac{2(d+1)}{N} \)
  —similar to worst-case guarantee from VC

linear regression (LinReg): learning ‘happened’!
Which of the following property about $H$ is not true?

1. $H$ is symmetric
2. $H^2 = H$ (double projection = single one)
3. $(I - H)^2 = I - H$ (double residual transform = single one)
4. none of the above

You can conclude that 2 and 3 are true by their physical meanings! :)
Which of the following property about $H$ is not true?

1. $H$ is symmetric
2. $H^2 = H$ (double projection = single one)
3. $(I - H)^2 = I - H$ (double residual transform = single one)
4. none of the above

Reference Answer: 4

You can conclude that 2 and 3 are true by their physical meanings! :-(
Linear Classification vs. Linear Regression

**Linear Classification**

\[ \mathcal{Y} = \{-1, +1\} \]
\[ h(x) = \text{sign}(w^T x) \]
\[ \text{err}(\hat{y}, y) = \mathbb{I}[\hat{y} \neq y] \]

**NP-hard** to solve in general
Linear Classification vs. Linear Regression

**Linear Classification**

\[ Y = \{-1, +1\} \]
\[ h(x) = \text{sign}(w^T x) \]
\[ \text{err}(\hat{y}, y) = \mathbb{I}[\hat{y} \neq y] \]

*NP-hard* to solve in general

**Linear Regression**

\[ Y = \mathbb{R} \]
\[ h(x) = w^T x \]
\[ \text{err}(\hat{y}, y) = (\hat{y} - y)^2 \]

*efficient analytic solution*
Linear Classification vs. Linear Regression

**Linear Classification**

\[
\mathcal{Y} = \{-1, +1\} \\
h(x) = \text{sign}(w^T x) \\
\text{err}(\hat{y}, y) = [\hat{y} \neq y]
\]

NP-hard to solve in general

**Linear Regression**

\[
\mathcal{Y} = \mathbb{R} \\
h(x) = w^T x \\
\text{err}(\hat{y}, y) = (\hat{y} - y)^2
\]

efficient analytic solution

\[
\{-1, +1\} \subseteq \mathbb{R}
\]
## Linear Classification vs. Linear Regression

### Linear Classification

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{Y}$</td>
<td>${-1, +1}$</td>
</tr>
<tr>
<td>$h(x)$</td>
<td>$\text{sign}(w^T x)$</td>
</tr>
<tr>
<td>$\text{err}(\hat{y}, y)$</td>
<td>$[\hat{y} \neq y]$</td>
</tr>
</tbody>
</table>

NP-hard to solve in general

### Linear Regression

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{Y}$</td>
<td>$\mathbb{R}$</td>
</tr>
<tr>
<td>$h(x)$</td>
<td>$w^T x$</td>
</tr>
<tr>
<td>$\text{err}(\hat{y}, y)$</td>
<td>$(\hat{y} - y)^2$</td>
</tr>
</tbody>
</table>

efficient analytic solution

\{-1, +1\} $\subset \mathbb{R}$: linear regression for classification?
## Linear Classification vs. Linear Regression

<table>
<thead>
<tr>
<th>Linear Classification</th>
<th>Linear Regression</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{Y} = {-1, +1}$</td>
<td>$\mathcal{Y} = \mathbb{R}$</td>
</tr>
<tr>
<td>$h(x) = \text{sign}(w^T x)$</td>
<td>$h(x) = w^T x$</td>
</tr>
<tr>
<td>$\text{err}(\hat{y}, y) = \left| \hat{y} \neq y \right|$</td>
<td>$\text{err}(\hat{y}, y) = (\hat{y} - y)^2$</td>
</tr>
</tbody>
</table>

**NP-hard** to solve in general  

**Efficient analytic solution**

\{−1, +1\} ⊂ \mathbb{R}: linear regression for classification?

1. run LinReg on binary classification data $\mathcal{D}$ (**efficient**)
Linear Classification vs. Linear Regression

**Linear Classification**
- $\mathcal{Y} = \{-1, +1\}$
- $h(x) = \text{sign}(w^T x)$
- $\text{err}(\hat{y}, y) = [\hat{y} \neq y]$

**Linear Regression**
- $\mathcal{Y} = \mathbb{R}$
- $h(x) = w^T x$
- $\text{err}(\hat{y}, y) = (\hat{y} - y)^2$

NP-hard to solve in general

efficient analytic solution

$\{-1, +1\} \subset \mathbb{R}$: linear regression for classification?

1. run LinReg on binary classification data $D$ (efficient)
2. return $g(x) = \text{sign}(w_{\text{LIN}}^T x)$

Hsuan-Tien Lin (NTU CSIE)
Linear Classification vs. Linear Regression

### Linear Classification

\[ Y = \{-1, +1\} \]
\[ h(x) = \text{sign}(w^T x) \]
\[ \text{err}(\hat{y}, y) = \mathbb{I}[\hat{y} \neq y] \]

**NP-hard** to solve in general

### Linear Regression

\[ Y = \mathbb{R} \]
\[ h(x) = w^T x \]
\[ \text{err}(\hat{y}, y) = (\hat{y} - y)^2 \]

**efficient analytic solution**

\( \{-1, +1\} \subset \mathbb{R} \): linear regression for classification?

1. run LinReg on binary classification data \( \mathcal{D} \) *(efficient)*
2. return \( g(x) = \text{sign}(w_{\text{LIN}}^T x) \)

but explanation of this **heuristic**?
Relation of Two Errors

\[
\text{err}_{0/1} = \left[ \text{sign}(w^T x) \neq y \right] \quad \text{err}_{\text{sqr}} = (w^T x - y)^2
\]
Relation of Two Errors

\[ \text{err}_{0/1} = \left[ \text{sign}(\mathbf{w}^T \mathbf{x}) \neq y \right] \quad \text{err}_{sqr} = (\mathbf{w}^T \mathbf{x} - y)^2 \]
Relation of Two Errors

\[ \text{err}_{0/1} = \left[ \text{sign}(w^T x) \neq y \right] \]

\[ \text{err}_{\text{sqr}} = (w^T x - y)^2 \]

**desired \( y = 1 \)**

**desired \( y = -1 \)**
Relation of Two Errors

\[
\text{err}_{0/1} = \left[ \text{sign}(w^T x) \neq y \right] \quad \text{err}_{\text{sqr}} = (w^T x - y)^2
\]

desired \( y = 1 \)

\[
\text{err} \quad \text{squared} \quad 0/1
\]

desired \( y = -1 \)

\[
\text{err} \quad w^T x
\]

\[
\text{err}_{0/1} \leq \text{err}_{\text{sqr}}
\]
Linear Regression for Binary Classification

\[ \text{err}_{0/1} \leq \text{err}_{\text{sqr}} \]
Linear Regression for Binary Classification

\[ \text{err}_{0/1} \leq \text{err}_{\text{sqr}} \]

classification \[ E_{\text{out}}(w) \leq \text{classification } E_{\text{in}}(w) + \sqrt{\ldots} \]
Linear Regression for Binary Classification

\[
\text{classification } E_{\text{out}}(w) \leq \text{classification } E_{\text{in}}(w) + \sqrt{\ldots}
\]

\[
\leq \text{regression } E_{\text{in}}(w) + \sqrt{\ldots}
\]

\[
\operatorname{err}_{0/1} \leq \operatorname{err}_{\text{sqr}}
\]
Linear Regression for Binary Classification

\[ \text{err}_{0/1} \leq \text{err}_{\text{sqr}} \]

\[ \text{classification } E_{\text{out}}(\mathbf{w}) \leq \text{classification } E_{\text{in}}(\mathbf{w}) + \sqrt{\ldots} \]

\[ \leq \text{regression } E_{\text{in}}(\mathbf{w}) + \sqrt{\ldots} \]

- (loose) upper bound \( \text{err}_{\text{sqr}} \) as \( \hat{\text{err}} \) to approximate \( \text{err}_{0/1} \)
Linear Regression for Binary Classification

\[ \text{err}_{0/1} \leq \text{err}_{\text{sqr}} \]

classification \( E_{\text{out}}(\mathbf{w}) \) \( \leq \) classification \( E_{\text{in}}(\mathbf{w}) + \sqrt{\cdots} \)

\leq \) regression \( E_{\text{in}}(\mathbf{w}) + \sqrt{\cdots} \)

- (loose) upper bound \( \text{err}_{\text{sqr}} \) as \( \widehat{\text{err}} \) to approximate \( \text{err}_{0/1} \)
- trade bound tightness for efficiency
Linear Regression for Binary Classification

\[ \text{err}_{0/1} \leq \text{err}_{\text{sqr}} \]

\[
\text{classification } E_{\text{out}}(w) \leq \text{classification } E_{\text{in}}(w) + \sqrt{\ldots}
\]
\[
\leq \text{regression } E_{\text{in}}(w) + \sqrt{\ldots}
\]

- (loose) upper bound \( \text{err}_{\text{sqr}} \) as \( \hat{\text{err}} \) to approximate \( \text{err}_{0/1} \)
- trade \textbf{bound tightness} for \textbf{efficiency}

\( w_{\text{LIN}} \): useful baseline classifier, or as \textit{initial PLA/pocket vector}
Which of the following functions are upper bounds of the pointwise 0/1 error $[\text{sign}(w^T x) \neq y]$ for $y \in \{-1, +1\}$?

1. $\exp(-yw^T x)$
2. $\max(0, 1 - yw^T x)$
3. $\log_2(1 + \exp(-yw^T x))$
4. all of the above

Reference Answer: 4

Plot the curves and you'll see. Thus, all three can be used for binary classification. In fact, all three functions connect to very important algorithms in machine learning and we will discuss one of them soon in the next lecture. Stay tuned. :-)

Hsuan-Tien Lin (NTU CSIE)

Machine Learning Foundations
Which of the following functions are upper bounds of the pointwise 0/1 error \[\text{sign}(w^T x) \neq y\] for \(y \in \{-1, +1\}\)?

1. \(\exp(-yw^T x)\)
2. \(\max(0, 1 - yw^T x)\)
3. \(\log_2(1 + \exp(-yw^T x))\)
4. all of the above

Reference Answer: 4

Plot the curves and you’ll see. Thus, all three can be used for binary classification. In fact, all three functions connect to very important algorithms in machine learning and we will discuss one of them soon in the next lecture.

Stay tuned. :-)
Summary

1. When Can Machines Learn?
2. Why Can Machines Learn?

Lecture 8: Noise and Error

3. How Can Machines Learn?

Lecture 9: Linear Regression

- Linear Regression Problem
  - use hyperplanes to approximate real values
- Linear Regression Algorithm
  - analytic solution with pseudo-inverse
- Generalization Issue
  \[ E_{out} - E_{in} \approx \frac{2(d+1)}{N} \] on average
- Linear Regression for Binary Classification
  - 0/1 error \leq squared error

• next: binary classification, regression, and then?

4. How Can Machines Learn Better?