Learning From Data
Lecture 8
Linear Classification and Regression

Linear Classification
Linear Regression

M. Magdon-Ismail
CSCI 4100/6100
RECAP: Approximation Versus Generalization

VC Analysis

\[ E_{\text{out}} \leq E_{\text{in}} + \Omega(d_{\text{vc}}) \]

1. Did you fit your data well enough (\(E_{\text{in}}\))? 
2. Are you confident your \(E_{\text{in}}\) will generalize to \(E_{\text{out}}\)

Bias-Variance Analysis

\[ E_{\text{out}} = \text{bias} + \text{var} \]

1. How well can you fit your data (\(\text{bias}\))? 
2. How close to that best fit can you get (\(\text{var}\))?

The VC Insuarance Co.

The VC warranty had conditions for becoming void:

You can’t look at your data before choosing \(\mathcal{H}\). 
Data must be generated i.i.d from \(P(x)\). 
Data and test case from same \(P(x)\) (same bin).

\[ \mathcal{H}_0 \]
\[ \text{bias} = 0.50; \]
\[ \text{var} = 0.25. \]
\[ E_{\text{out}} = 0.75 \quad \checkmark \]

\[ \mathcal{H}_1 \]
\[ \text{bias} = 0.21; \]
\[ \text{var} = 1.69. \]
\[ E_{\text{out}} = 1.90 \]
RECAP: Decomposing The Learning Curve

VC Analysis

Bias-Variance Analysis

Pick $\mathcal{H}$ that can generalize and has a good chance to fit the data

Pick $(\mathcal{H}, \mathcal{A})$ to approximate $f$ and not behave wildly after seeing the data
• Linear models are perhaps \textit{the} fundamental model.

• The linear model is the first model to try.
The Linear Signal

linear in $\mathbf{x}$: gives the line/hyperplane separator

\[ S = \mathbf{w}^T \mathbf{x} \]

linear in $\mathbf{w}$: makes the algorithms work

$\mathbf{x}$ is the augmented vector: $\mathbf{x} \in \{1\} \times \mathbb{R}^d$
The Linear Signal

\[ s = \mathbf{w}^T \mathbf{x} \rightarrow \begin{cases} \text{sign}(\mathbf{w}^T \mathbf{x}) & \{ -1, +1 \} \\ \mathbf{w}^T \mathbf{x} & \mathbb{R} \\ \theta(\mathbf{w}^T \mathbf{x}) & [0, 1] \end{cases} \]

\[ y = \theta(s) \]
Linear Classification

\[ \mathcal{H}_{\text{lin}} = \{ h(x) = \text{sign}(\mathbf{w}^T \mathbf{x}) \} \]

1. \( E_{\text{in}} \approx E_{\text{out}} \) because \( d_{\text{vc}} = d + 1 \),

\[ E_{\text{out}}(h) \leq E_{\text{in}}(h) + O\left(\sqrt{\frac{d}{N} \log N}\right). \]

2. If the data is linearly separable, PLA will find a separator \( \implies E_{\text{in}} = 0 \).

\[ w(t+1) = w(t) + \mathbf{x}_i y_i \]

\[ \uparrow \] misclassified data point

\[ E_{\text{in}} = 0 \implies E_{\text{out}} \approx 0 \]

(\( f \) is well approximated by a linear fit).

What if the data is not separable (\( E_{\text{in}} = 0 \) is not possible)?

How to ensure \( E_{\text{in}} \approx 0 \) is possible?

pocket algorithm

select good features
Non-Separable Data
Minimizing $E_{in}$ is a hard combinatorial problem.

The Pocket Algorithm

− Run PLA
− At each step keep the best $E_{in}$ (and $w$) so far.

(Its not rocket science, but it works.)

(Other approaches: linear regression, logistic regression, linear programming . . . )
Each digit is a $16 \times 16$ image.
Digits Data

Each digit is a $16 \times 16$ image.

\[
\begin{bmatrix}
-1 & -1 & -1 & -1 & -1 & -1 & -0.63 & 0.86 & -0.17 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -0.99 & 0.3 & 1 & 0.31 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
-1 & -1 & -0.41 & 1 & 0.99 & -0.57 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -0.68 & 0.83 & 1 & 0.56 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -0.94 & 0.54 \\
1 & 0.78 & -0.72 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -0.26 & 0.95 & 1 & -0.16 & -1 & -1 & -1 & -1 & -1 & -0.99 & -0.71 & -0.83 & -1 & -1 & -1 & -1 & -1 & -0.8 & 0.91 & 1 & 0.3 & -0.96 & -1 & -1 & -0.55 & 0.49 & 1 & 0.88 & 0.09 & -1 & -1 & -1 & -1 & 0.28 & 1 & 0.88 & -0.8 & -1 & -0.9 & 0.14 & 0.97 & 1 & 1 & 1 & 1 & 1 & 0.99 & -0.74 & -1 & -1 & -0.95 & 0.84 & 1 & 0.32 & -1 & -1 & 0.35 & 1 & 0.65 & -0.10 & -0.18 & 1 & 0.98 & -0.72 & -1 & -1 & -0.63 & 1 & 1 & 0.07 & -0.92 & 0.11 & 0.96 & 0.30 & -0.88 & -1 & -0.07 & 1 & 0.64 & -0.99 & -1 & -1 & -0.67 & 1 & 1 & 0.75 & 0.34 & 1 & 0.70 & -0.94 & -1 & -1 & 0.54 & 1 & 0.02 & -1 & -1 & -1 & -0.90 & 0.79 & 1 & 1 & 1 & 1 & 0.53 & 0.18 & 0.81 & 0.83 & 0.97 & 0.86 & -0.63 & -1 & -1 & -1 & -1 & -0.45 & 0.82 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0.13 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -0.48 & 0.81 & 1 & 1 & 1 & 1 & 1 & 1 & 0.21 & -0.94 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -0.97 & -0.42 & 0.30 & 0.82 & 1 & 0.48 & -0.47 & -0.99 & -1 & -1 & -1 & -1 & -1 & -1 \\
\end{bmatrix}
\]

\[
\mathbf{x} = (1, x_1, \cdots, x_{256}) \quad \leftarrow \text{input} \\
\mathbf{w} = (w_0, w_1, \cdots, w_{256}) \quad \leftarrow \text{linear model} \\
\}
\]

\[d_{\text{VC}} = 257\]
**Intensity and Symmetry Features**

*feature*: an important property of the input that you think is useful for classification.

(dictionary.com: a prominent or conspicuous part or characteristic)

\[
\begin{align*}
\mathbf{x} &= (1, x_1, x_2) \quad \leftarrow \text{input} \\
\mathbf{w} &= (w_0, w_1, w_2) \quad \leftarrow \text{linear model}
\end{align*}
\]

\[
d_{\text{VC}} = 3
\]
PLA on Digits Data

PLA

Error (log scale)

$E_{out}$

$E_{in}$

Iteration Number, $t$

0 250 500 750 1000
Linear Regression

<table>
<thead>
<tr>
<th>attribute</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>age</td>
<td>32 years</td>
</tr>
<tr>
<td>gender</td>
<td>male</td>
</tr>
<tr>
<td>salary</td>
<td>40,000</td>
</tr>
<tr>
<td>debt</td>
<td>26,000</td>
</tr>
<tr>
<td>years in job</td>
<td>1 year</td>
</tr>
<tr>
<td>years at home</td>
<td>3 years</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Classification: Approve/Deny
Linear Regression

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>age</td>
<td>32 years</td>
</tr>
<tr>
<td>gender</td>
<td>male</td>
</tr>
<tr>
<td>salary</td>
<td>40,000</td>
</tr>
<tr>
<td>debt</td>
<td>26,000</td>
</tr>
<tr>
<td>years in job</td>
<td>1 year</td>
</tr>
<tr>
<td>years at home</td>
<td>3 years</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

**Classification:** Approve/Deny

**Regression:** Credit Line (dollar amount)

\[ h(x) = \sum_{i=0}^{d} w_i x_i = w^T x \]
Least Squares Linear Regression

Squared error →
Least Squares Linear Regression

\[ y = f(x) + \epsilon \]

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

\[ E_{\text{out}}(h) = \mathbb{E}_x[(h(x) - y)^2] \]

\[ h(x) = w^T x \]
Using Matrices for Linear Regression

\[
X = \begin{bmatrix}
-x_1 \\
-x_2 \\
\vdots \\
-x_N
\end{bmatrix}
\]

\text{data matrix, } N \times (d + 1)

\[
y = \begin{bmatrix}
y_1 \\
y_2 \\
\vdots \\
y_N
\end{bmatrix}
\]

\text{target vector}

\[
\hat{y} = \begin{bmatrix}
\hat{y}_1 \\
\hat{y}_2 \\
\vdots \\
\hat{y}_N
\end{bmatrix} = \begin{bmatrix}
w^T x_1 \\
w^T x_2 \\
\vdots \\
w^T x_N
\end{bmatrix} = Xw
\]

\text{in-sample predictions}

\[
E_{\text{in}}(w) = \frac{1}{N} \sum_{n=1}^{N} (\hat{y}_n - y_n)^2
\]

\[
= \frac{1}{N} \| \hat{y} - y \|_2^2
\]

\[
= \frac{1}{N} \| Xw - y \|_2^2
\]

\[
= \frac{1}{N} (w^T X^T Xw - 2w^T X^T y + y^T y)
\]
Linear Regression Solution

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

**Vector Calculus:** To minimize \( E_{\text{in}}(w) \), set \( \nabla_w E_{\text{in}}(w) = 0 \).

\[ \nabla_w (w^T A w) = (A + A^T) w, \quad \nabla_w (w^T b) = b. \]

\( A = X^T X \) and \( b = X^T y \):

\[ \nabla_w E_{\text{in}}(w) = \frac{2}{N} (X^T X w - X^T y) \]

Setting \( \nabla E_{\text{in}}(w) = 0 \):

\[ X^T X w = X^T y \quad \leftarrow \text{normal equations} \]

\[ w_{\text{lin}} = (X^T X)^{-1} X^T y \quad \leftarrow \text{when } X^T X \text{ is invertible} \]
Linear Regression Algorithm:

1. Construct the matrix $X$ and the vector $y$ from the data set $(x_1, y_1), \cdots, (x_N, y_N)$, where each $x$ includes the $x_0 = 1$ coordinate,

$$
X = \begin{bmatrix}
  x_1 \\
  x_2 \\
  \vdots \\
  x_N \\
\end{bmatrix}, \quad y = \begin{bmatrix}
  y_1 \\
  y_2 \\
  \vdots \\
  y_N \\
\end{bmatrix}.
$$

2. Compute the pseudo inverse $X^\dagger$ of the matrix $X$. If $X^TX$ is invertible,

$$
X^\dagger = (X^TX)^{-1}X^T
$$

3. Return $w_{\text{lin}} = X^\dagger y$. 
Generalization

The linear regression algorithm gets the smallest possible $E_{in}$ in one step.

Generalization is also good.
One can obtain a regression version of $d_{vc}$.

There are other bounds, for example:

$$
\mathbb{E}[E_{out}(h)] = \mathbb{E}[E_{in}(h)] + O\left(\frac{d}{N}\right)
$$
Linear Regression for Classification

Linear regression can learn *any* real valued target function.

For example $y_n = \pm 1$.  

Use linear regression to get $w$ with $w^T x_n \approx y_n = \pm 1$  
Then $\text{sign}(w^T x_n)$ will likely agree with $y_n = \pm 1$.  
These can be good initial weights for classification.

**Example.**  
Classifying 1 from not 1  
(multiclass $\to$ 2 class)