CSE 417T: Introduction to Machine Learning

Lecture 9: Nonlinear Models

Henry Chai
09/25/18
• Training data **does not** consist of probabilities

• Observations are still **binary**: $y_i = \pm 1$

• Goal is to learn $f(\bar{x}) = P(y = +1|\bar{x})$

• $h(\bar{x}) = \theta(\theta^T \bar{x}) = \frac{1}{1 + e^{-\theta^T \bar{x}}} = \left(1 + e^{-\theta^T \bar{x}}\right)^{-1} \in [0,1]$

• Note that $1 - \theta(\theta^T \bar{x}) = \theta(-\theta^T \bar{x})$
The cross-entropy error looks weird but actually arises naturally from our choice of hypothesis set and is also easier to minimize.

- Some hypothesis $h$ is good if:
  - the probability of the training data $\mathcal{D}$ given $h$ is high

$$E_{\text{in}}(\overline{w}) = \frac{1}{n} \sum_{i=1}^{n} \ln \left( 1 + e^{-y_i \overline{w}^T x_i} \right)$$
<table>
<thead>
<tr>
<th>( w_1 = (1, -1) )</th>
<th>( w_2 = (-1, 1) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>((3, +1))</td>
<td>( P(y = +1</td>
</tr>
<tr>
<td>((2, +1))</td>
<td>( P(y = +1</td>
</tr>
<tr>
<td>((5, -1))</td>
<td>( P(y = -1</td>
</tr>
</tbody>
</table>

\( P(\mathcal{D} | \mathcal{W}) \)

- \( 0.98 \times 0.12 \times 0.27 \approx 0.0318 \)
- \( 0.02 \times 0.88 \times 0.73 \approx 0.0128 \)

\( E_{in}(\mathcal{W}) \)

- \( \frac{\ln(1 + e^{-4}) + \ln(1 + e^2) + \ln(1 + e^4)}{3} \approx 1.15 \)
- \( \frac{\ln(1 + e^4) + \ln(1 + e^{-2}) + \ln(1 + e^{-1})}{3} \approx 1.49 \)
<table>
<thead>
<tr>
<th>$\bar{w}_1 = (1, -1)$</th>
<th>$\bar{w}^* \approx (4, -1)$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>(3, +1)</strong></td>
<td></td>
</tr>
<tr>
<td>$P{y = +1</td>
<td>3, \bar{w}_1} = \frac{1}{1 + e^{-[1 \quad -1]}} \approx 0.12$</td>
</tr>
<tr>
<td><strong>(2, +1)</strong></td>
<td></td>
</tr>
<tr>
<td>$P{y = +1</td>
<td>2, \bar{w}_1} = \frac{1}{1 + e^{-[1 \quad -1]}} \approx 0.27$</td>
</tr>
<tr>
<td><strong>(5, -1)</strong></td>
<td></td>
</tr>
<tr>
<td>$P{y = -1</td>
<td>5, \bar{w}_1} = 1 - P{y = +1</td>
</tr>
<tr>
<td>**$P(\bar{D}</td>
<td>\bar{w})$**</td>
</tr>
<tr>
<td>$0.98 \times 0.12 \times 0.27 \approx 0.0318$</td>
<td>$0.72 \times 0.73 \times 0.88 \approx 0.4625$</td>
</tr>
<tr>
<td><strong>$E_{\text{in}}(\bar{w})$</strong></td>
<td></td>
</tr>
<tr>
<td>$\frac{\ln(1 + e^{-4}) + \ln(1 + e^2) + \ln(1 + e^1)}{3} \approx 1.15$</td>
<td>$\frac{\ln(1 + e^{-1}) + \ln(1 + e^{-1}) + \ln(1 + e^{-2})}{3} \approx 0.25$</td>
</tr>
</tbody>
</table>
So unfortunately, even though this is the “right” error metric to use; setting the gradient w.r.t. \( \bar{w} \) equal to 0 and solving for the optimal \( \bar{w} \) is impossible.
Functions where there is only 1 global minimum are so-called convex. In 1d, such functions are characterized by having non-negative second-order derivatives (think $x^2$). The corollary in higher dimensions is the Hessian matrix being positive semidefinite.
Cross-entropy Error is Convex

\[ E_{\text{in}}(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^{n} \ln \left( 1 + e^{-y_i \mathbf{w}^T \mathbf{x}_i} \right) \]

\[ \nabla_{\mathbf{w}} E_{\text{in}}(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^{n} \frac{-y_i \mathbf{x}_i}{e^{y_i \mathbf{w}^T \mathbf{x}_i} + 1} \]

\[ H_{\mathbf{w}} E_{\text{in}}(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^{n} \frac{\left(y_i^2 e^{y_i \mathbf{w}^T \mathbf{x}_i} \right) \left( \mathbf{x}_i \mathbf{x}_i^T \right)}{\left(e^{y_i \mathbf{w}^T \mathbf{x}_i} + 1\right)^2} \]

\[ H_{\mathbf{w}} E_{\text{in}}(\mathbf{w}) \text{ is positive semidefinite } \forall \mathbf{w} \to E_{\text{in}}(\mathbf{w}) \text{ is convex} \]
• Suppose the current location is $\vec{w}(t)$

• Move some distance, $\eta$, in the “most downhill” direction possible, $\hat{v}$

• $\vec{w}(t+1) = \vec{w}(t) + \eta \hat{v}$
Multivariate first-order Taylor series expansion: \( f(\vec{x}) \approx f(\vec{a}) + (\vec{x} - \vec{a})^T \nabla f(\vec{a}) \)

The fourth line follows because \( \hat{v} \) is a unit vector and the inner product of a unit vector and any arbitrary vector is at most the L2 norm of the non-unit vector and is at least \(-1\) times the L2 norm of the non-unit vector (depending on the direction the unit vector points in).

Taylor expansion is a reasonable estimate if our step size \( \eta \) is not too big

\[ \vec{a}^T \vec{b} = ||\vec{a}|| ||\vec{b}|| \cos(\theta) \] where \( \theta \) is the angle between the two vectors.
• Fix $\eta$ and choose $\hat{v}$ to minimize $\Delta E_{in}$
  after making the update $\hat{w}_{(t+1)} = \hat{w}_{(t)} + \eta \hat{v}$

• $\Delta E_{in}(\hat{v}) = E_{in}(\hat{w}_{(t)} + \eta \hat{v}) - E_{in}(\hat{w}_{(t)})$
  $\approx \left( E_{in}(\hat{w}_{(t)}) + \eta \hat{v}^T \nabla_{w} E_{in}(\hat{w}_{(t)}) \right) - E_{in}(\hat{w}_{(t)})$
  $\approx \eta \hat{v}^T \nabla_{w} E_{in}(\hat{w}_{(t)})$

• $\Delta E_{in}(\hat{v}^*) = -\eta \| \nabla_{w} E_{in}(\hat{w}_{(t)}) \|$

• $\hat{v}^* = -\frac{\nabla_{w} E_{in}(\hat{w}_{(t)})}{\| \nabla_{w} E_{in}(\hat{w}_{(t)}) \|}$
• Fix $\eta$ and choose $\hat{\vartheta}$ to minimize $\Delta E_{in}$ after making the update $\hat{w}_{(t+1)} = \hat{w}_{(t)} + \eta \hat{\vartheta}$

• $\Delta E_{in}(\hat{\vartheta}) = E_{in}(\hat{w}_{(t)} + \eta \hat{\vartheta}) - E_{in}(\hat{w}_{(t)})$
  $\approx (E_{in}(\hat{w}_{(t)}) + \eta \hat{\vartheta}^T \nabla_w E_{in}(\hat{w}_{(t)})) - E_{in}(\hat{w}_{(t)})$
  $\approx \eta \hat{\vartheta}^T \nabla_w E_{in}(\hat{w}_{(t)})$

• $\eta \hat{\vartheta}^* \nabla_w E_{in}(\hat{w}_{(t)}) = -\eta \|\nabla_w E_{in}(\hat{w}_{(t)})\|$

• $\hat{\vartheta}^* = -\frac{\nabla_w E_{in}(\hat{w}_{(t)})}{\|\nabla_w E_{in}(\hat{w}_{(t)})\|}$
\[ \hat{v} \]

- Fix \( \eta \) and choose \( \hat{v} \) to minimize \( \Delta E_{in} \)
  after making the update \( \hat{w}_{(t+1)} = \hat{w}_{(t)} + \eta \hat{v} \)

- \( \Delta E_{in}(\hat{v}) = E_{in}(\hat{w}_{(t)} + \eta \hat{v}) - E_{in}(\hat{w}_{(t)}) \)
  \[ \approx \left( E_{in}(\hat{w}_{(t)}) + \eta \hat{v}^T \nabla_w E_{in}(\hat{w}_{(t)}) \right) - E_{in}(\hat{w}_{(t)}) \]
  \[ \approx \eta \hat{v}^T \nabla_w E_{in}(\hat{w}_{(t)}) \]

- \( \eta \hat{v}^T \nabla_w E_{in}(\hat{w}_{(t)}) = -\eta \| \nabla_w E_{in}(\hat{w}_{(t)}) \| \)

- \( -\eta \frac{\nabla_w E_{in}(\hat{w}_{(t)}) \cdot \nabla_w E_{in}(\hat{w}_{(t)})}{\| \nabla_w E_{in}(\hat{w}_{(t)}) \|^2} = -\eta \| \nabla_w E_{in}(\hat{w}_{(t)}) \|^2 \)
\[ \eta_t \]

- Set \( \eta_t = \eta_0 \| \nabla \overline{w} E_{\text{lin}}(\overline{w}_t) \| \)

- \( \eta_t \) decreases as \( t \) increases, because \( \| \nabla \overline{w} E_{\text{lin}}(\overline{w}_t) \| \) decreases as \( E_{\text{lin}}(\overline{w}_t) \) approaches its minimum
\[ \hat{v}_t^* = -\frac{\nabla \bar{W}^t(\bar{w}_t(\ell))}{\|\nabla \bar{W}^t(\bar{w}_t(\ell))\|} \]

\[ \eta_t = \eta_0 \|\nabla \bar{W}^t(\bar{w}_t(\ell))\| \]

\[ \bar{w}_{(t+1)} = \bar{w}_t(\ell) + \eta_t \hat{v}_t^* \]
\[ = \bar{w}_t(\ell) + (\eta_0 \|\nabla \bar{W}^t(\bar{w}_t(\ell))\|) \left( -\frac{\nabla \bar{W}^t(\bar{w}_t(\ell))}{\|\nabla \bar{W}^t(\bar{w}_t(\ell))\|} \right) \]
\[ = \bar{w}_t(\ell) - \eta_0 \nabla \bar{W}^t(\bar{w}_t(\ell)) \]
Gradient Descent

- Input: $D = \{(x_1^1, y_1), \ldots, (x_n^m, y_n)\}, \eta_0$
- Initialize $\bar{w}_0$ to all zeros and set $t = 0$
- While termination condition is not satisfied
  - Compute $\nabla_{\bar{w}} E_{in}(\bar{w}_t)$
  - Update $\bar{w}: \bar{w}_{(t+1)} = \bar{w}_t - \eta_0 \nabla_{\bar{w}} E_{in}(\bar{w}_t)$
  - Increment $t: t = t + 1$
- Output: $\bar{w}_t \rightarrow g(\bar{x}) = P\{y = +1|x\} = \frac{1}{1+e^{-\bar{w}_t^T \bar{x}}}$
**Gradient Descent**

- Input: \( \mathcal{D} = \{(x_1^1, y_1), \ldots, (x_n^n, y_n)\}, \eta_0, \tau \)
- Initialize \( \bar{w}_0 \) to all zeros and set \( t = 0 \)
- While \( t \leq \tau \)
  - Compute \( \nabla_{\bar{w}} E_{in}(\bar{w}_t) \)
  - Update \( \bar{w} \): \( \bar{w}_{t+1} = \bar{w}_t - \eta_t \nabla_{\bar{w}} E_{in}(\bar{w}_t) \)
  - Increment \( t \): \( t = t + 1 \)
- Output: \( \bar{w}_t \rightarrow g(\bar{x}) = P\{y = +1|\bar{x}\} = \frac{1}{1 + e^{-\bar{w}_t^T \bar{x}}} \)
Gradient Descent

- Input: $D = \{(x_1, y_1), ..., (x_n, y_n)\}, \eta_0, \epsilon$
- Initialize $\vec{w}_0$ to all zeros and set $t = 0$
- While $\|\nabla_{\vec{w}} E_{in}(\vec{w}_t)\| \geq \epsilon$
  - Compute $\nabla_{\vec{w}} E_{in}(\vec{w}_t)$
  - Update $\vec{w}$: $\vec{w}_{(t+1)} = \vec{w}_{(t)} - \eta_0 \nabla_{\vec{w}} E_{in}(\vec{w}_{(t)})$
  - Increment $t$: $t = t + 1$
- Output: $\vec{w}_t \rightarrow g(\vec{x}) = P\{y = +1|x\} = \frac{1}{1+e^{-\vec{w}_t^T \vec{x}}}$
Gradient Descent

- Input: $\mathcal{D} = \{(x_1, y_1), \ldots, (x_n, y_n)\}, \eta_0$
- Initialize $\overline{w}_0$ to all zeros and set $t = 0$
- While termination condition is not satisfied
  - Compute $\nabla_{\overline{w}} E_{ln}(\overline{w}_{(t)}) = \frac{1}{n} \sum_{i=1}^{n} \frac{-y_i x_i}{e^{y_i \overline{w}_{(t)}^T x_i} + 1}$
  - Update $\overline{w}_{(t+1)} = \overline{w}_{(t)} - \eta_0 \nabla_{\overline{w}} E_{ln}(\overline{w}_{(t)})$
  - Increment $t: t = t + 1$
- Output: $\overline{w}_t \rightarrow g(\overline{x}) = \frac{1}{1 + e^{-\overline{w}_t^T \overline{x}}}$
Stochastic Gradient Descent (SGD)

• Input: $\mathcal{D} = \{(x_1, y_1), \ldots, (x_n, y_n)\}, \eta_0$

• Initialize $\overline{w}_0$ to all zeros and set $t = 0$

• While termination condition is not satisfied
  • Pick a random data point in $\mathcal{D}$, $(\bar{x}, y)$
  • Compute $\nabla_{\overline{w}} e(\overline{w}(t), \bar{x}, y) = \frac{-y \bar{x}}{e^{\bar{w}}(t) \bar{x}^2 + 1}$
  • Update $\overline{w}: \overline{w}(t+1) = \overline{w}(t) - \eta_0 \nabla_{\overline{w}} e(\overline{w}(t), \bar{x}, y)$
  • Increment $t: t = t + 1$

• Output: $\overline{w}_t \rightarrow g(\bar{x}) = P\{y = +1 | \bar{x}\} = \frac{1}{1 + e^{-\overline{w}_t \bar{x}}}$

Reminiscent of PLA, SGD “on average” makes the same update as normal gradient descent, is a lot faster than normal gradient descent and its element of randomness can help avoid local minima in the case of the function being non-convex and
So $\frac{1}{2}$ is a reasonable choice for $b$ for some applications but recall from the error lecture that sometimes we can be more concerned about false positives than false negatives or vice versa.

- Use logistic regression to find $\mathbf{w}_t$

- Use $\mathbf{w}_t$ for classification: if $P\{y = +1|\mathbf{x}\} = \phi(\mathbf{w}_t^T\mathbf{x}) \geq \frac{1}{2}$ then classify $\mathbf{x}$ as $+1$; otherwise, classify $\mathbf{x}$ as $-1$

- $g(\mathbf{x}) = \text{sign}\left(\frac{1}{1+e^{-\mathbf{w}_t^T\mathbf{x}}} - b\right)$
Linear Models
Linear Models
Linear Models
Linear Models?
Linear Models?
Nonlinear Models
Here the function $\Phi$ is called the feature transformation.
Here the function $\Phi$ is called the feature transformation
Nonlinear Models

• Decide on a transformation $\Phi: \mathcal{X} \rightarrow \mathcal{Z}$

• Convert $\mathcal{D} = \{(x_1, y_1), ..., (x_n, y_n)\}$ to $\tilde{\mathcal{D}} = \{(\Phi(x_i) = \tilde{z}_i, y_i), ..., (\Phi(x_n) = \tilde{z}_n, y_n)\}$

• Fit a linear model using $\tilde{\mathcal{D}}$, $\tilde{g}(\tilde{z})$

• Return the corresponding predictor in the original space: $g(\tilde{x}) = \tilde{g}(\Phi(\tilde{x}))$
Nonlinear Models
Nonlinear Models
Nonlinear Models
Nonlinear Models
At what cost?

- VC dimension of linear separators: $d_{VC} = d + 1$ where $d = \text{the dimensionality of the input space } \mathcal{X}$

- Let $\tilde{d} = \text{the dimensionality of the transformed space } \mathcal{Z}$

- If $\tilde{d} > d$, then the learned hypothesis will not generalize as well:
  
  $E_{out}(g) \leq E_{in}(g) + O\left(\sqrt{d_{VC} \frac{\log(n)}{n}}\right)$
When you look at the training data and then pick a transformation to use, you have implicitly explored a large hypothesis set and reduced it to some smaller one; thus, when you go bound your generalization error, you will have to use the VC-dimension of the larger hypothesis set that you searched in your mind and not the VC-dimension associated with the transformation that you picked.

- But what if $\tilde{d} = d$?
- From the example: $\tilde{x} = [x_1, x_2]$ so $d = 2$ and $\tilde{z} = [(x_1 - 0.5)^2, (x_2 - 0.5)^2]$ so $\tilde{d} = 2$
- Data snooping: looking at the training data to decide what transformation to use
- **Pick a transformation before looking at training data**
These general $k^{th}$-order transforms can slow down computation e.g. recall that linear regression requires inverting a $(d+1)$-by-$(d+1)$ matrix which is an $O(d^3)$ operation
Linear Models
This is a 6\textsuperscript{th} order polynomial that perfectly fits the training data. Common sense might tell you that exactly fitting these training data points is excessive; we might prefer the linear model which has non-zero in-sample error but will likely generalize better.
## Tradeoffs

<table>
<thead>
<tr>
<th></th>
<th>Low-Dimensional Transformations</th>
<th>High-Dimensional Transformations</th>
</tr>
</thead>
<tbody>
<tr>
<td>$E_{in}$</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Generalization</td>
<td>Good</td>
<td>Bad</td>
</tr>
</tbody>
</table>