Backup plans:

If this Zoom meeting ends prematurely, five-minute break, check Piazza
Karma Lectures

• Tuesday, April 21, 11:40am: Lily Weng, MIT
  “Evaluating Robustness of Neural Networks”
  • Recorded: https://vod.video.cornell.edu/media/CS+Colloquium+-+Lily+Weng/1_1q6zajy6

• Tuesday, April 28, 11:40am: Ashia Wilson, MIT
  • Live: https://cornell.zoom.us/j/276979194
Abstract Representation of a Logistic Neuron

where

\[
    g(z) = \frac{1}{1 + e^{-z}}
\]

"activation"

"bias" \( w_0 \)

\[ a(x) = g(w \cdot x) \]

where \( g(z) = \frac{1}{1 + e^{-z}} \)
Gradient Descent (Logistic Regression)

Current hypothesis: \( h_\mathbf{w}(\mathbf{x}) \)

Initialize \( w_0, w_1, w_2, \ldots, w_n \) \[\text{[commonly: small random weights]}\]

Repeat

For \( i = 1 \) to \( N \) \[\text{[for each example]}\]

\[
\begin{align*}
    h &\leftarrow h_\mathbf{w}(\mathbf{x}_i) = g(\mathbf{w} \cdot \mathbf{x}_i) \\
    \text{For } j = 0 \text{ to } n &\quad \text{[for each feature]} \\
    w_j &\leftarrow w_j + \alpha g(\mathbf{w} \cdot \mathbf{x}_i)(1 - g(\mathbf{w} \cdot \mathbf{x}_i))(y_i - g(\mathbf{w} \cdot \mathbf{x}_i))x_{ij}
\end{align*}
\]

Until <stopping condition>
Comparing Update Rules

\[ w_j \leftarrow w_j + \alpha x_{ij}(y_i - h) \]

\[ w_j \leftarrow w_j + \alpha g(\bar{w} \cdot \bar{x}_i)(1 - g(\bar{w} \cdot \bar{x}_i))(y_i - g(\bar{w} \cdot \bar{x}_i))x_{ij} \]

Bigger updates when activation is closer to 0.5
Smaller updates the farther it gets from 0.5
= Tries to keep classifier from being wishy-washy
Gradient Descent (Logistic Regression)

Current hypothesis: $h_w(\bar{x})$

Initialize $w_0, w_1, w_2, \ldots, w_n$ [commonly: small random weights]

Repeat

For $i = 1$ to $N$ [for each example]

$$h \leftarrow h_w(\bar{x}_i) = g(\bar{w} \cdot \bar{x}_i)$$

For $j = 0$ to $n$ [for each feature]

$$w_j \leftarrow w_j + \alpha g(\bar{w} \cdot \bar{x}_i)(1 - g(\bar{w} \cdot \bar{x}_i))(y_i - g(\bar{w} \cdot \bar{x}_i))x_{ij}$$

Until <stopping condition>
Gradient Descent (Logistic Regression)

Current hypothesis: $h_w(\bar{x})$

Initialize $w_0, w_1, w_2, ..., w_n$ [commonly: small random weights]

Repeat

For $i = 1$ to $N$ [for each example]

$h \leftarrow h_w(\bar{x}_i) = g(\bar{w} \cdot \bar{x}_i)$

For $j = 0$ to $n$ [for each feature]

$w_j \leftarrow w_j + \alpha h(1 - h)(y_i - h)x_{ij}$

Until <stopping condition>
General Case: Abstract Representation of a Neuron

\[ a(\bar{x}) = g(\overline{w} \cdot \bar{x}) \]

- **“activation”**
- **“bias”** $w_0$
- $w_1$
- $w_2$
- $x_1$
- $x_2$
Gradient Descent (Logistic Regression)

Current hypothesis: $h_{\overline{w}}(\overline{x})$

Initialize $w_0, w_1, w_2, ..., w_n$ [commonly: small random weights]

Repeat

For $i = 1$ to $N$ [for each example]

$h \leftarrow h_{\overline{w}}(\overline{x}_i) = g(\overline{w} \cdot \overline{x}_i)$

For $j = 0$ to $n$ [for each feature]

$w_j \leftarrow w_j + \alpha h(1 - h)(y_i - h)x_{ij}$

Until <stopping condition>
Gradient Descent: General Case

Current hypothesis: $h_{\vec{w}}(\vec{x})$

Initialize $w_0, w_1, w_2, ..., w_n$ [commonly: small random weights]

Repeat

For $i = 1$ to $N$ [for each example]

$h \leftarrow h_{\vec{w}}(\vec{x}_i) = g(\vec{w} \cdot \vec{x}_i)$

For $j = 0$ to $n$ [for each feature]

$w_j \leftarrow w_j + \alpha h(1 - h)(y_i - h)x_{ij}$

Until <stopping condition>
Gradient Descent: General Case

Current hypothesis: \( h_\overline{w}(\overline{x}) \)

Initialize \( w_0, w_1, w_2, \ldots, w_n \) [commonly: small random weights]

Repeat

For \( i = 1 \) to \( N \) [for each example]

\[
\begin{align*}
    h &\leftarrow h_\overline{w}(\overline{x}_i) = g(\overline{w} \cdot \overline{x}_i) \\
    h' &\leftarrow h'_\overline{w}(\overline{x}_i) = g' (\overline{w} \cdot \overline{x}_i)
\end{align*}
\]

For \( j = 0 \) to \( n \) [for each feature]

\[
    w_j \leftarrow w_j + \alpha h' (y_i - h) x_{ij}
\]

Until <stopping condition>
Abstract Representation of a tanh Neuron

\[
a(\bar{x}) = g(\bar{w} \cdot \bar{x})
\]

where \( g(z) = \tanh(\bar{w} \cdot \bar{x}) \)

Range (-1, +1)
Abstract Representation of a tanh Neuron

\[
a(\bar{x}) = g(\bar{w} \cdot \bar{x})
\]

where \( g(z) = \tanh(\bar{w} \cdot \bar{x}) \)

\[
g'(z) = 1 - g^2(z)
\]
Gradient Descent: General Case

Current hypothesis: $h_{\overline{w}}(\overline{x})$

Initialize $w_0, w_1, w_2, \ldots, w_n$ [commonly: small random weights]

Repeat
  
  For $i = 1$ to $N$ [for each example]
    
    $h \leftarrow h_{\overline{w}}(\overline{x}_i) = g(\overline{w} \cdot \overline{x}_i)$
    
    $h' \leftarrow h'_{\overline{w}}(\overline{x}_i) = g'((\overline{w} \cdot \overline{x}_i)$
    
  For $j = 0$ to $n$ [for each feature]
    
    $w_j \leftarrow w_j + \alpha h'(y_i - h)x_{ij}$

Until <stopping condition>
Gradient Descent: tanh

Current hypothesis: $h_w(\bar{x})$

Initialize $w_0, w_1, w_2, ..., w_n$  
[commonly: small random weights]

Repeat
  For $i = 1$ to $N$  
  [for each example]
  $h \leftarrow h_w(\bar{x}_i) = g(\bar{w} \cdot \bar{x}_i) = \tanh(\bar{w} \cdot \bar{x})$
  $h' \leftarrow h'_w(\bar{x}_i) = g'(\bar{w} \cdot \bar{x}_i) = 1 - g^2(z) = 1 - h^2$
  For $j = 0$ to $n$  
  [for each feature]
  $w_j \leftarrow w_j + \alpha h'(y_i - h)x_{ij}$

Until <stopping condition>
Gradient Descent: tanh

Current hypothesis: \( h_\overline{w}(\overline{x}) \)

Initialize \( w_0, w_1, w_2, \ldots, w_n \)  
[commonly: small random weights]

Repeat

For \( i = 1 \) to \( N \)  
[for each example]

\[
h \leftarrow h_\overline{w}(\overline{x}_i) = g(\overline{w} \cdot \overline{x}_i) = \tanh(\overline{w} \cdot \overline{x})
\]

For \( j = 0 \) to \( n \)  
[for each feature]

\[
w_j \leftarrow w_j + \alpha(1 - h^2)(y_i - h)x_{ij}
\]

Until <stopping condition>
Gradient Descent: General Case

Current hypothesis: $h_w(\bar{x})$

Initialize $w_0, w_1, w_2, \ldots, w_n$ [commonly: small random weights]

Repeat

For $i = 1$ to $N$ [for each example]

$h \leftarrow h_w(\bar{x}_i) = g(\bar{w} \cdot \bar{x}_i)$

$h' \leftarrow h'_w(\bar{x}_i) = g'(\bar{w} \cdot \bar{x}_i)$

For $j = 0$ to $n$ [for each feature]

$w_j \leftarrow w_j + \alpha h'(y_i - h)x_{ij}$

Until <stopping condition>
Gradient Descent: General Case

Current hypothesis: $h_w(\bar{x})$

Initialize $w_0, w_1, w_2, \ldots, w_n$  
[commonly: small random weights]

Repeat

For $i = 1$ to $N$  
[for each example]

$h \leftarrow h_w(x_i) = g(\bar{w} \cdot \bar{x}_i)$

$h' \leftarrow h'_w(x_i) = g'(\bar{w} \cdot \bar{x}_i)$

For $j = 0$ to $n$  
[for each feature]

$w_j \leftarrow w_j + \alpha h'(y_i - h)x_{ij}$

Until <stopping condition>
Logistic Regression Error $D(\bar{w})$ and Its Gradient

$$\frac{\partial}{\partial w_j} \text{Error}_D(\bar{w}) = -2 \sum_{i=1}^{N} g'(\bar{w} \cdot \bar{x}_i)(y_i - g(\bar{w} \cdot \bar{x}_i))x_{ij}$$

Focusing on a single example $(\bar{x}_i, y_i)$

$$\frac{\partial}{\partial w_j} \text{Error}_{\{\{\bar{x}_i, y_i\}\}}(\bar{w}) = -2 g'(\bar{w} \cdot \bar{x}_i)(y_i - g(\bar{w} \cdot \bar{x}_i))x_{ij}$$

$$w_j \leftarrow w_j + \alpha g'(\bar{w} \cdot \bar{x}_i) (y_i - g(\bar{w} \cdot \bar{x}_i))x_{ij}$$
Logistic Regression Error $D(\bar{w})$ and Its Gradient

$$\frac{\partial}{\partial w_j} \text{Error}_D(\bar{w}) = -2 \sum_{i=1}^{N} g'(\bar{w} \cdot \bar{x}_i)(y_i - g(\bar{w} \cdot \bar{x}_i))x_{ij}$$

Focusing on a single example $(\bar{x}_i, y_i)$

$$\frac{\partial}{\partial w_j} \text{Error}_{\{(\bar{x}_i, y_i)\}}(\bar{w}) = -2 g'(\bar{w} \cdot \bar{x}_i)(y_i - g(\bar{w} \cdot \bar{x}_i))x_{ij}$$

$$w_j \leftarrow w_j + \alpha g'(\bar{w} \cdot \bar{x}_i)(y_i - g(\bar{w} \cdot \bar{x}_i))x_{ij}$$

This is not the gradient, it’s a single example
Gradient Descent: General Case

Current hypothesis: $h_{\bar{w}}(\bar{x})$

Initialize $w_0, w_1, w_2, \ldots, w_n$ [commonly: small random weights]

Repeat

For $i = 1$ to $N$ [for each example]

$h \leftarrow h_{\bar{w}}(\bar{x}_i) = g(\bar{w} \cdot \bar{x}_i)$

$h' \leftarrow h'_{\bar{w}}(\bar{x}_i) = g'(\bar{w} \cdot \bar{x}_i)$

For $j = 0$ to $n$ [for each feature]

$w_j \leftarrow w_j + \alpha h'(y_i - h)x_{ij}$

Until <stopping condition>
Gradient Descent: General Case

Current hypothesis: \( h_\overline{w}(\overline{x}) \)

Initialize \( w_0, w_1, w_2, ..., w_n \) [commonly: small random weights]

Repeat

For \( i = 1 \) to \( N \) [for each example]

\[ h \leftarrow h_\overline{w}(\overline{x}_i) = g(\overline{w} \cdot \overline{x}_i) \]
\[ h' \leftarrow h'_\overline{w}(\overline{x}_i) = g'(\overline{w} \cdot \overline{x}_i) \]

For \( j = 0 \) to \( n \) [for each feature]

\[ w_j \leftarrow w_j + \alpha h'(y_i - h)x_{ij} \]

[Randomly reorder the data]

Until <stopping condition>
Logistic Regression Error $D(\mathbf{w})$ and Its Gradient

$$\frac{\partial}{\partial w_j} \text{Error}_D(\mathbf{w}) = -2 \sum_{i=1}^{N} g'(\mathbf{w} \cdot \mathbf{x}_i)(y_i - g(\mathbf{w} \cdot \mathbf{x}_i))x_{ij}$$

Focusing on a single example $(\mathbf{x}_i, y_i)$

$$\frac{\partial}{\partial w_j} \text{Error}_{\{(\mathbf{x}_i, y_i)\}}(\mathbf{w}) = -2 g'(\mathbf{w} \cdot \mathbf{x}_i)(y_i - g(\mathbf{w} \cdot \mathbf{x}_i))x_{ij}$$

Stochastic gradient descent: Updates based on smaller subsets of D
Multi-Layer Neural Network
Multi-Layer Neural Network

\[ a_5(x_i) = g\left( w_{05} + w_{35}a_3(x_i) + w_{45}a_4(x_i) \right) \]

\[ a_3(x_i) = g\left( w_{03} + w_{13}a_1(x_i) + w_{23}a_2(x_i) \right) \]

\[ a_4(x_i) = g\left( w_{04} + w_{14}a_1(x_i) + w_{24}a_2(x_i) \right) \]

\[ a_1(x_i) = x_1 \]

\[ a_2(x_i) = x_2 \]
Multi-Layer Neural Network

\[ a_5(x_i) = g(w_{05} + w_{35}a_3(x_i) + w_{45}a_4(x_i)) \]

\[ a_3(x_i) = g(w_{03} + w_{13}a_1(x_i) + w_{23}a_2(x_i)) \]

\[ a_4(x_i) = g(w_{04} + w_{14}a_1(x_i) + w_{24}a_2(x_i)) \]

\[ a_1(x_i) = x_1 \]

\[ a_2(x_i) = x_2 \]
Multi-Layer Neural Network

• Can have more than a single output
  • Example: one output per move in Go – merit of each move

• Can have more than one hidden layer
  • Typically each layer feeds into next layer

• Design choices: # of layers, “width” of layers

• (Many generalizations beyond this)
Multi-Layer Neural Network

• Can have more than a single output
  • Example: one output per move in Go – merit of each move

• Can have more than one hidden layer
  • Typically each layer feeds into next layer

• Design choices: # of layers, “width” of layers

• (Many generalizations beyond this)
Multi-Layer Neural Network

• Increases our “representational power”

• Learning the weights requires something different
Multi-Layer Neural Network

• Increases our “representational power”

• Learning the weights requires something different
  Can compute the derivative of $\frac{\partial}{\partial w_{ij}} \text{Error}_D(\bar{w})$ for each $w_{ij}$
  Computing $\frac{\partial}{\partial w_{ij}} \text{Error}_D(\bar{w})$ is more complex for lower levels