Data Science

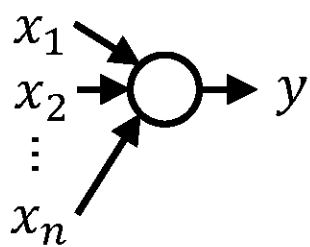
Week 9

Fitting a Model to Data (2)-
Classification:

Neural Networks

Basics of Neural Network

Binary Classification

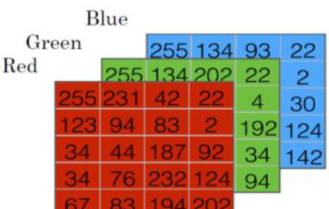


Binary Classification

64 pixel

64 pixel

1 (Cat) vs
0 (Non-Cat)



Feature

$$x = \begin{bmatrix} 255 \\ 231 \\ 42 \\ \vdots \\ 255 \\ 134 \\ 202 \\ \vdots \\ 255 \\ 134 \\ 93 \\ \vdots \end{bmatrix}$$

y

$$n_x = 64 \times 64 \times 3 = 12288$$

http://cs230.stanford.edu/files/C1M2.pdf

Basics of Neural Network

Logistic Regression

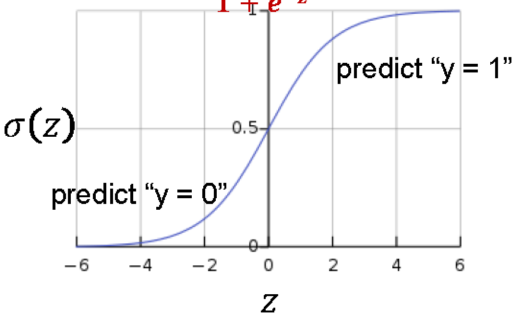
Logistic Regression

Given $\mathbf{x}, y$, want $\hat{y} = P(y = 1|\mathbf{x})$ infinitely close to y,

Where, $\mathbf{x} \in \mathbb{R}^{n_x}, 0 \le \hat{y} \le 1$, output predicted value $\hat{y} = \sigma(\mathbf{w}^T\mathbf{x} + b)$

Unknown Parameters: $\mathbf{w} \in \mathbb{R}^{n_x}, b \in \mathbb{R}$

Sigmoid: $\boldsymbol{\sigma(z)} = \dfrac{\mathbf{1}}{\mathbf{1} + \boldsymbol{e^{-z}}}$



$$z \to +\infty, \sigma(z) \approx \frac{1}{1 + e^{-\infty}} = 1$$

$$z \to -\infty, \sigma(z) \approx \frac{1}{1 + e^{+\infty}} = 0$$

Basics of Neural Network

Logistic Regression Loss Function

Logistic Regression Loss Function

Logistic Regression $\hat{y}^{(i)} = \sigma\left(\boldsymbol{\theta}^T \mathbf{x}^{(i)}\right), 0 \leq \hat{y}^{(i)} \leq 1,\quad \boldsymbol{\sigma}(\boldsymbol{z}) = \frac{\mathbf{1}}{\mathbf{1} + \boldsymbol{e}^{-\boldsymbol{z}}}$

Training set with m samples

$$\left(\mathbf{x}^{(1)}, y^{(1)}\right), \left(\mathbf{x}^{(2)}, y^{(2)}\right), \cdots, \left(\mathbf{x}^{(m)}, y^{(m)}\right)$$

$$\mathbf{x}^{(i)} = \begin{bmatrix} x_0^{(i)} \\ x_1^{(i)} \\ x_2^{(i)} \\ \vdots \\ x_n^{(i)} \end{bmatrix}, x_0^{(i)} = 1, y^{(i)} \in \{0, 1\}$$

How to choose parameters $\boldsymbol{\theta}$ to make $\hat{y}^{(i)}$ infinitely close to $y^{(i)}$?

Logistic Regression Loss Function

Loss (error) function:

$$L(\hat{y}^{(i)}, y^{(i)}) = -y^{(i)}\ln\left(\hat{y}^{(i)}\right) - (1 - y^{(i)})\ln\left(1 - \hat{y}^{(i)}\right)$$

$$L(\hat{y}^{(i)}, y^{(i)}) \geq 0$$

$$\text{if } y^{(i)} = 1, \left(1 - y^{(i)}\right) = 0$$

$$when\ L(\hat{y}^{(i)}, y^{(i)}) = -\ln\left(\hat{y}^{(i)}\right) \to 0, \hat{y}^{(i)} \to 1$$

$$\text{if } y^{(i)} = 0, y^{(i)} = 0$$

$$when\ L(\hat{y}^{(i)}, y^{(i)}) = -\ln\left(1 - \hat{y}^{(i)}\right) \to 0, \hat{y}^{(i)} \to 0$$

Logistic Regression

$$J(\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m} L(\hat{y}^{(i)}, y^{(i)})$$

$$= -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)}\ \ln\left(\hat{y}^{(i)}\right) + (1 - y^{(i)})\ \ln\left(1 - \hat{y}^{(i)}\right)\right]$$

Learning: find parameter $\boldsymbol{\theta}$ to $\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$

Prediction: given new x output $\hat{y} = \sigma(\boldsymbol{\theta}^T \mathbf{x}) = \frac{1}{1+e^{-\boldsymbol{\theta}^T \mathbf{x}}}$

Basics of Neural Network

Gradient Descent for Logistic Regression

Gradient Descent

$$J(\boldsymbol{\theta}) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)}\ \ln\left(h_{\boldsymbol{\theta}}\left(x^{(i)}\right)\right) + (1 - y^{(i)})\ \ln\left(1 - h_{\boldsymbol{\theta}}\left(x^{(i)}\right)\right)\right]$$

Goal: $\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$

Good news: Convex function!

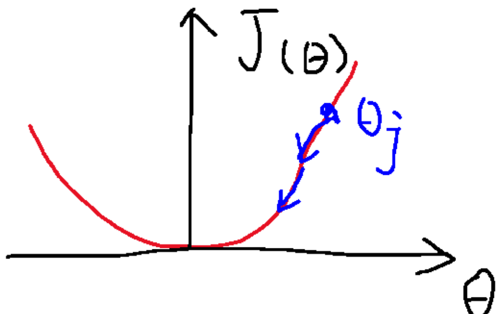
Bad news: No analytical solution

Repeat

{

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\boldsymbol{\theta})$$

}



http://cs230.stanford.edu/files/C1M2.pdf

Gradient Descent

$$J(\boldsymbol{\theta}) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)}\ \ln\left(h_{\boldsymbol{\theta}}\left(x^{(i)}\right)\right) + (1 - y^{(i)})\ \ln\left(1 - h_{\boldsymbol{\theta}}\left(x^{(i)}\right)\right)\right]$$

Goal: $\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$

Good news: Convex function!
Bad news: No analytical solution

Repeat

{

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\boldsymbol{\theta})$$

}

(Simultaneously update all θ_j)

$$\boxed{\frac{\partial}{\partial \theta_j} J(\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m}\left(h_{\boldsymbol{\theta}}\left(x^{(i)}\right) - y^{(i)}\right) x_j^{(i)}}$$

Basics of Neural Network

Implementation

of Logistic Regression

Implementing Logistic Regression

k is the number of iterations
m is the number of sample data

$w_1 = 0$, $w_2 = 0$, b = 0, $\alpha = 4$
for t = 1 to k:
J = 0, dw_1 = 0, dw_2 = 0, db = 0
for i = 1 to m:

$$z^{(i)} = w_1 x_1{}^{(i)} + w_2 x_2{}^{(i)} + b$$

$$a^{(i)} = \sigma(z^{(i)})$$

$$J \mathrel{+}= -\left[y^{(i)} \ln a^{(i)} + (1 - y^{(i)}) \ln(1 - a^{(i)})\right]$$

$$dz^{(i)} = a^{(i)} - y^{(i)}$$

$$dw_1 \mathrel{+}= x_1^{(i)} dz^{(i)}$$

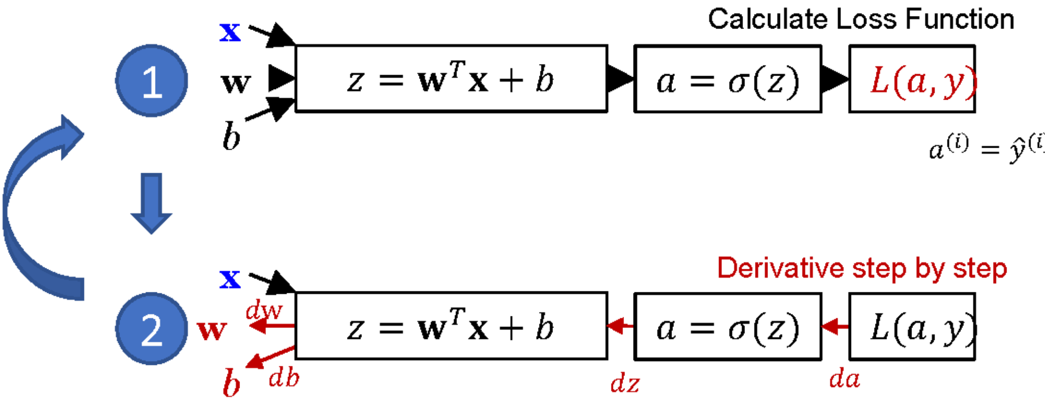
$$dw_2 \mathrel{+}= x_2^{(i)} dz^{(i)}$$

$$db \mathrel{+}= dz^{(i)}$$

J = J/m, dw_1 = dw_1/m, dw_2 = dw_2/m, db = db/m
$w_1 = w_1 - \alpha\, dw_1$, $w_2 = w_2 - \alpha\, dw_2$, b = b- α db

Training using Logistic Regression

Gradient Descent: process to estimate the parameter $(\mathbf{w}, b)$



$$dw_1 = \frac{1}{m}\sum_{i=1}^{m}(a^{(i)} - y^{(i)})\, x_1^{(i)}$$

$$dw_2 = \frac{1}{m}\sum_{i=1}^{m}(a^{(i)} - y^{(i)})\, x_2^{(i)}$$

$\cdots$

$$db = \frac{1}{m}\sum_{i=1}^{m}(a^{(i)} - y^{(i)})$$

Gradient Descent

$$J(\mathbf{w}, b) = \frac{1}{m}\sum_{i=1}^{m} L(a^{(i)}, y)$$

$$\boxed{-\frac{y^{(i)}}{a^{(i)}} + \frac{1-y^{(i)}}{1-a^{(i)}}} \qquad \boxed{a^{(i)}(1-a^{(i)})} \qquad \boxed{x_1^{(i)}}$$

$$dw_1 = \sum_{i=1}^{m} \frac{dL}{da^{(i)}} \times \frac{da^{(i)}}{dz^{(i)}} \times \frac{dz^{(i)}}{dw_1} = \sum_{i=1}^{m} (a^{(i)} - y^{(i)}) x_1^{(i)}$$

$$dw_2 = \sum_{i=1}^{m} \frac{dL}{da^{(i)}} \times \frac{da^{(i)}}{dz^{(i)}} \times \frac{dz^{(i)}}{dw_2} = \sum_{i=1}^{m} (a^{(i)} - y^{(i)}) x_2^{(i)}$$

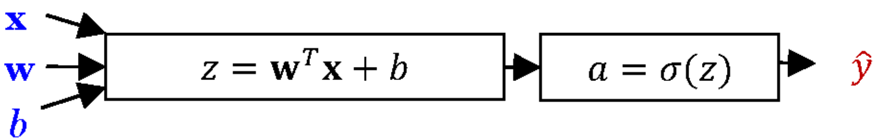
$$\vdots$$

$$db = \sum_{i=1}^{m} \frac{dL}{da^{(i)}} \times \frac{da^{(i)}}{dz^{(i)}} \times \frac{dz^{(i)}}{db} = \sum_{i=1}^{m} (a^{(i)} - y^{(i)})$$

$$dw_1 = \frac{1}{m} dw_1, \quad dw_2 = \frac{1}{m} dw_2, \ldots \quad db = \frac{1}{m} db$$

Test New Data using Logistic Model

Prediction (Test): given new $\mathbf{x}$ output $\hat{y} = \sigma(\mathbf{w}^T\mathbf{x} + b) = \frac{1}{1+e^{-\mathbf{w}^T\mathbf{x}+b}}$



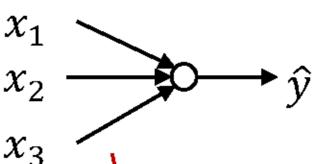
http://cs230.stanford.edu/files/C1M3.pdf

Shallow Neural Networks

Network Representation

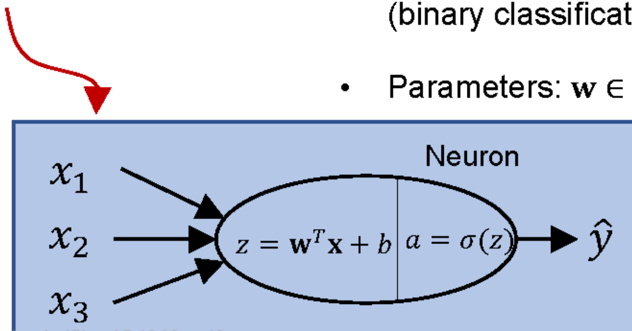
"Neuron" Representation of Logistic Regression

- Logistic Regression

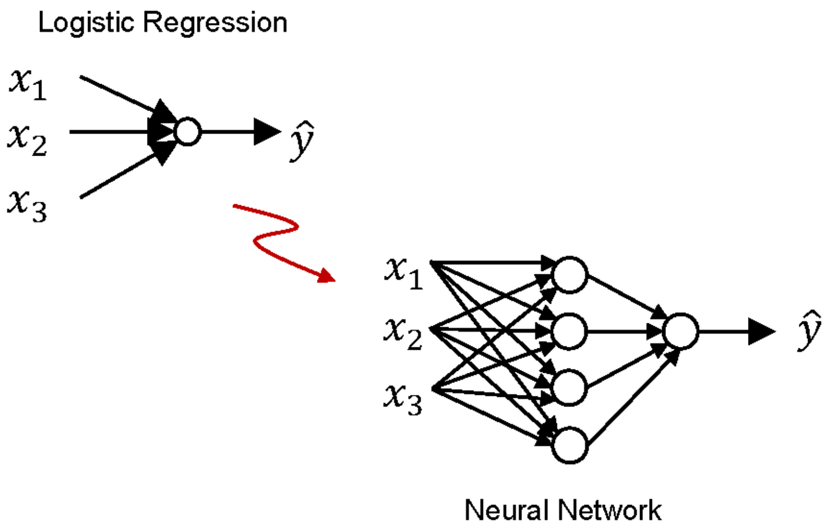


Given $\mathbf{x}, y$, want parameters $\mathbf{w}$ and b to make $\hat{y}$ infinitely close to y

- Where, $\mathbf{x} \in \mathbb{R}^{n_x}, 0 \leq \hat{y} \leq 1$, $\hat{y} = \sigma(\mathbf{w}^T\mathbf{x} + b)$ is the probability of the given $\mathbf{x}$ belongs to the class 1 (binary classification case
- Parameters: $\mathbf{w} \in \mathbb{R}^{n_x}, b \in \mathbb{R}$

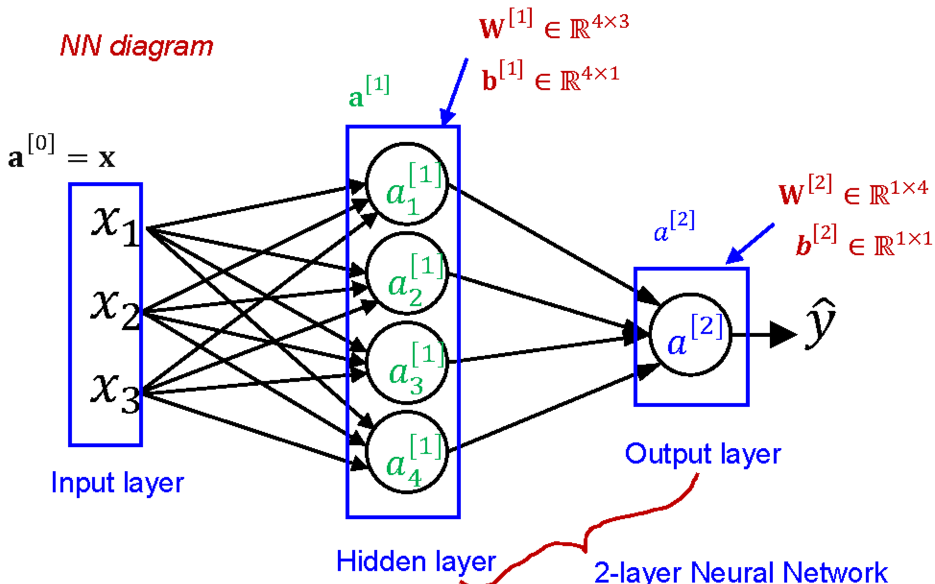


What is a Neural Network?



http://cs230.stanford.edu/files/C1M3.pdf

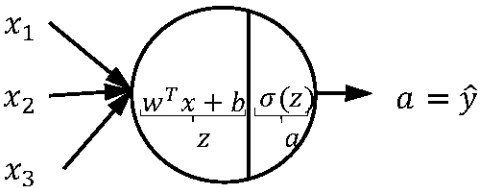
Neural Network (NN) Representation



Shallow Neural Networks

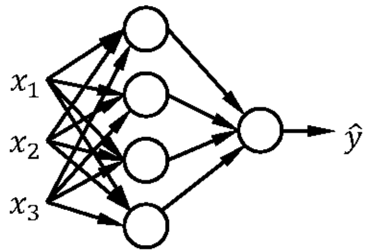
Computing Neural Network's Output

Neural Network Representation

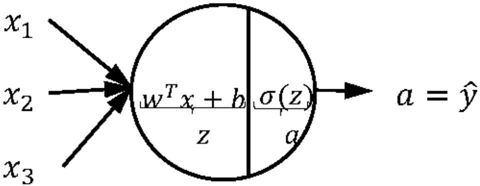


$$z = \mathbf{w}^T\mathbf{x} + b$$

$$a = \sigma(z)$$

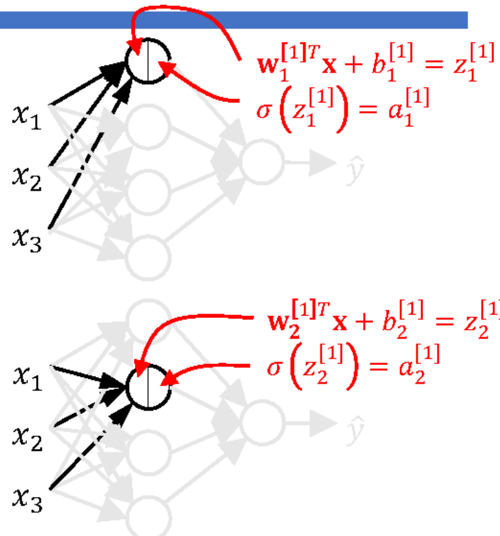


Calculation in Neural Unit

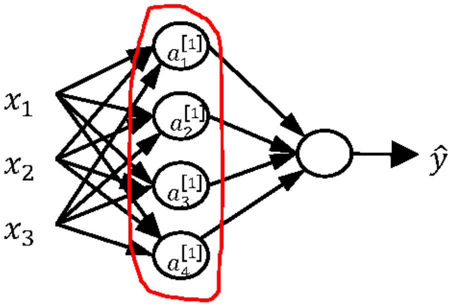


$z = \mathbf{w}^T\mathbf{x} + b$

$a = \sigma(z)$



Calculation in Neural Network



e.g. $\mathbf{w}_1^{[1]} = [\mathrm{w}_{1,1}^{[1]}, \mathrm{w}_{1,2}^{[1]}, \mathrm{w}_{1,3}^{[1]}]^T$

$$z_1^{[1]} = \mathbf{w}_1^{[1]T}\mathbf{x} + b_1^{[1]}, \quad a_1^{[1]} = \sigma(z_1^{[1]})$$

$$z_2^{[1]} = \mathbf{w}_2^{[1]T}\mathbf{x} + b_2^{[1]}, \quad a_2^{[1]} = \sigma(z_2^{[1]})$$

$$z_3^{[1]} = \mathbf{w}_3^{[1]T}\mathbf{x} + b_3^{[1]}, \quad a_3^{[1]} = \sigma(z_3^{[1]})$$

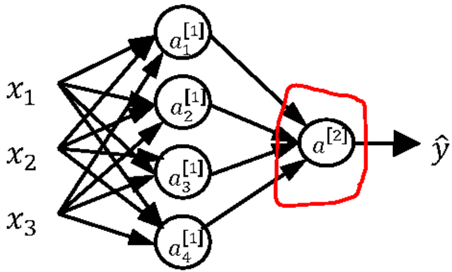
$$z_4^{[1]} = \mathbf{w}_4^{[1]T}\mathbf{x} + b_4^{[1]}, \quad a_4^{[1]} = \sigma(z_4^{[1]})$$

$$\mathbf{W}^{[1]} = \begin{bmatrix} \mathbf{w}_1^{[1]T} \\ \mathbf{w}_2^{[1]T} \\ \mathbf{w}_3^{[1]T} \\ \mathbf{w}_4^{[1]T} \end{bmatrix}$$

$$\mathbf{z}^{[1]} = \mathbf{W}^{[1]}\mathbf{x} + \mathbf{b}^{[1]}$$

$$\mathbf{a}^{[1]} = \sigma\left(\mathbf{z}^{[1]}\right)$$

Calculation in Neural Network



$$\mathbf{z}^{[2]} = \mathbf{w}^{[2]}\mathbf{a}^{[1]} + b^{[2]}$$

$$\sigma\left(\mathbf{w}^{[2]}\mathbf{a}^{[1]} + b^{[2]}\right) = a^{[2]} = \hat{y}$$

Given input $\mathbf{x}$:

$$\mathbf{z}^{[1]} = \mathbf{W}^{[1]}\underbrace{\mathbf{x}}_{\mathbf{a}^{[0]}} + \mathbf{b}^{[1]}$$

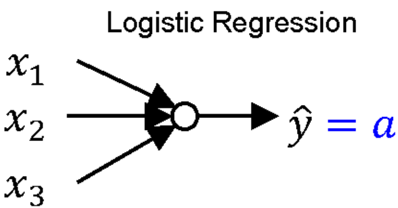
$$\mathbf{a}^{[1]} = \sigma(\mathbf{z}^{[1]})$$

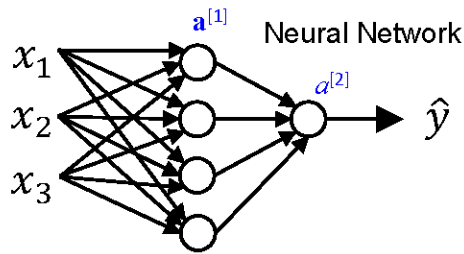
$$z^{[2]} = \mathbf{w}^{[2]}\mathbf{a}^{[1]} + b^{[2]}$$

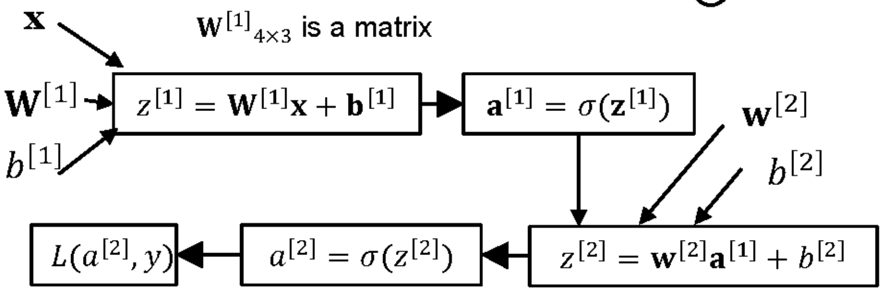
$$a^{[2]} = \sigma(z^{[2]})$$

http://cs230.stanford.edu/files/C1M3.pdf

Neural Network Representation and Output







$a^{[2]} = \hat{y}^{(i)}$

$\mathbf{w}^{[2]}$ is a vector in this case

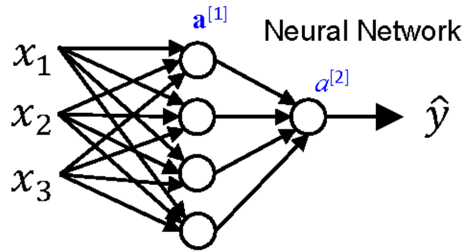
$$L(\hat{y}^{(i)}, y^{(i)}) = -y^{(i)}\ln\left(\hat{y}^{(i)}\right) - (1 - y^{(i)})\ln\left(1 - \hat{y}^{(i)}\right)$$

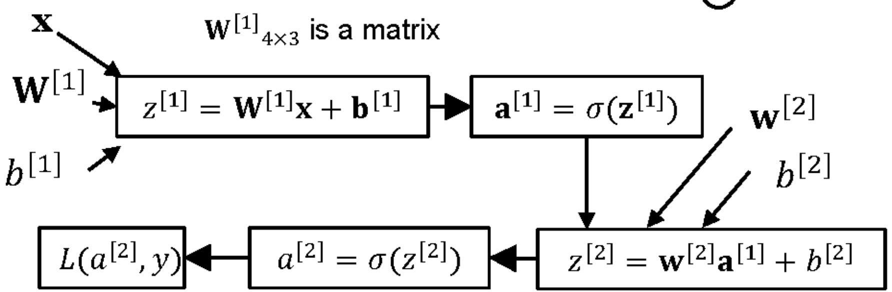
Shallow Neural Networks

Gradient descent for neural networks

Neural Network

To find parameters **W** ($\mathbf{W}^{[1]}$, $\mathbf{w}^{[2]}$ ...) and **b** ($\mathbf{b}^{[1]}$, $b^{[2]}$ ...) to make $\hat{y}^{(i)}$ infinitely close to $y^{(i)}$.





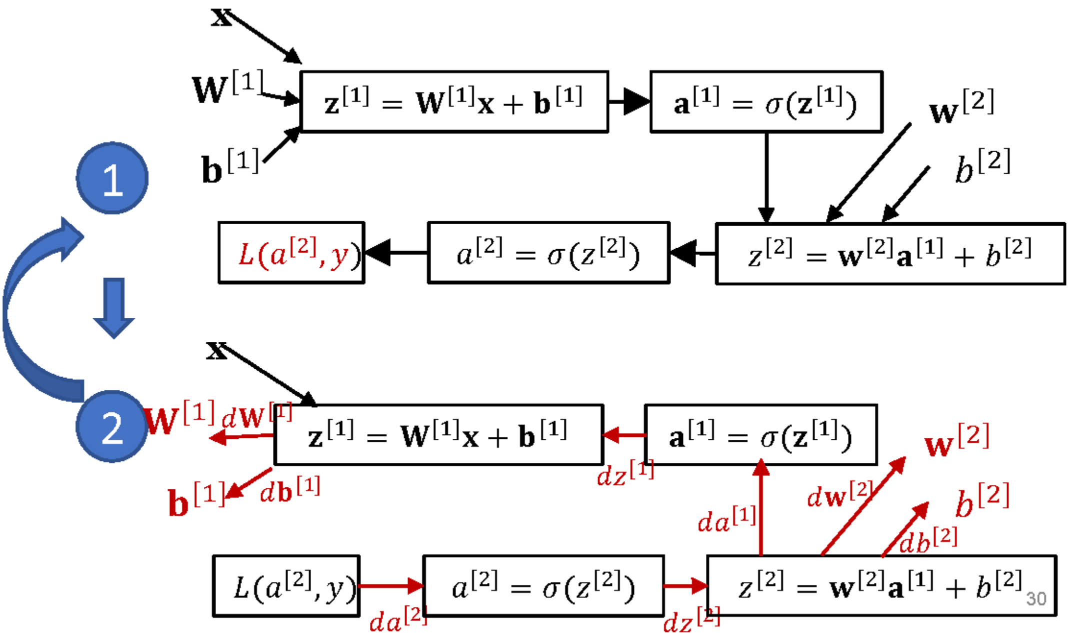
$a^{[2]} = \hat{y}^{(i)}$

$L(\hat{y}^{(i)}, y^{(i)}) = -y^{(i)}\ln\left(\hat{y}^{(i)}\right) - (1 - y^{(i)})\ln\left(1 - \hat{y}^{(i)}\right)$

$\mathbf{w}^{[2]}$ is a vector in this case

http://cs230.stanford.edu/files/C1M3.pdf

Training in Neural Network



Gradient Descent for Training Neural Networks

Parameters: $\mathbf{W}^{[1]}, \mathbf{b}^{[1]}, \mathbf{w}^{[2]}, b^{[2]}$

Cost function: $J\left(\mathbf{W}^{[1]}, \mathbf{b}^{[1]}, \mathbf{w}^{[2]}, b^{[2]}\right) = \frac{1}{m}\sum_{i=1}^{m} L(a^{[2]}, y)$

Gradient descent:

Repeat

{

Compute predicts $(\hat{y}^{(i)})$ ← Forward Propagation

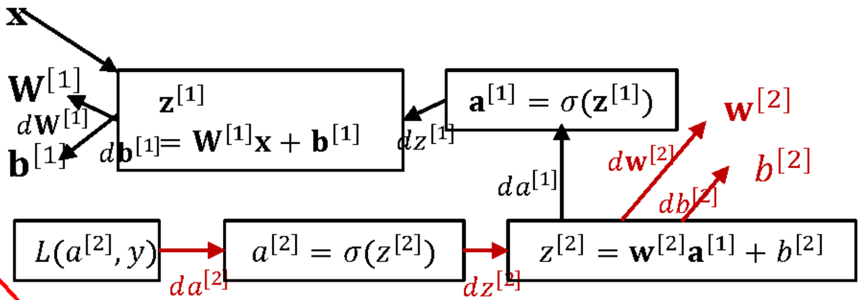
$\mathrm{d}\mathbf{W}^{[1]} = \frac{\partial J}{\partial \mathbf{W}^{[1]}}, \mathrm{d}\mathbf{b}^{[1]} = \frac{\partial J}{\partial \mathbf{b}^{[1]}}$

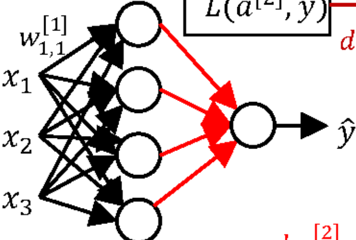
$\mathrm{d}\mathbf{W}^{[2]} = \frac{\partial J}{\partial \mathbf{W}^{[2]}}, \mathrm{d}\mathbf{b}^{[2]} = \frac{\partial J}{\partial \mathbf{b}^{[2]}}$ ← Backpropagation

$\mathbf{W}^{[1]} = \mathbf{W}^{[1]} - \alpha \cdot \mathrm{d}\mathbf{W}^{[1]}, \mathbf{b}^{[1]} = \mathbf{b}^{[1]} - \alpha \cdot \mathrm{d}\mathbf{b}^{[1]}$ ← Updating

$\mathbf{W}^{[2]} = \mathbf{W}^{[2]} - \alpha \cdot \mathrm{d}\mathbf{W}^{[2]}, \mathbf{b}^{[2]} = \mathbf{b}^{[2]} - \alpha \cdot \mathrm{d}\mathbf{b}^{[2]}$

}





$$J(\mathbf{W}^{[1]}, \mathbf{b}^{[1]}, \mathbf{w}^{[2]}, b^{[2]}) = \frac{1}{m}\sum_{i=1}^{m} L(a^{[2]}, y)$$

$$dw_1^{[2]} = \sum_{i=1}^{m} \frac{dL}{da^{[2](i)}} \times \frac{da^{[2](i)}}{dz^{[2](i)}} \times \frac{dz^{[2](i)}}{dw_1^{[2]}} = \sum_{i=1}^{m} (a^{[2](i)} - y^{[2](i)})\, a_1^{[1](i)}$$

$$dw_1^{[2]} = \frac{1}{m} dw_1^{[2]},$$

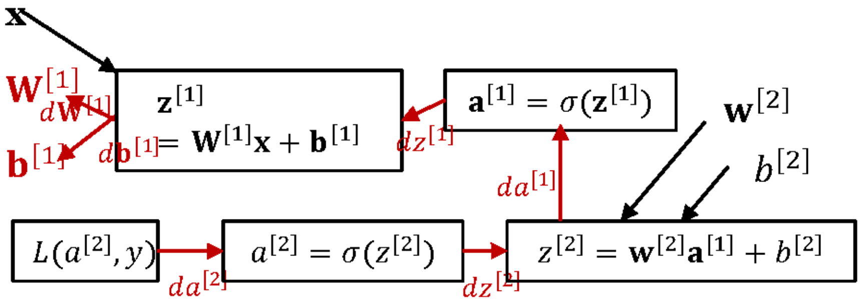
$$dw_2^{[2]} = \sum_{i=1}^{m} \frac{dL}{da^{[2](i)}} \times \frac{da^{[2](i)}}{dz^{[2](i)}} \times \frac{dz^{[2](i)}}{dw_2^{[2]}} = \sum_{i=1}^{m} (a^{[2](i)} - y^{[2](i)})\, a_2^{[1](i)}$$

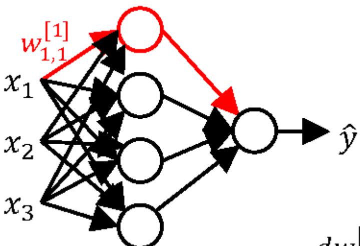
$$dw_2^{[2]} = \frac{1}{m} dw_2^{[2]}, \ldots$$

$$\vdots$$

$$db^{[2]} = \sum_{i=1}^{m} \frac{dL}{da^{[2](i)}} \times \frac{da^{[2](i)}}{dz^{[2](i)}} \times \frac{dz^{[2](i)}}{db^{[2]}} = \sum_{i=1}^{m} (a^{[2](i)} - y^{[2](i)})$$

$$db^{[2]} = \frac{1}{m} db^{[2]}$$



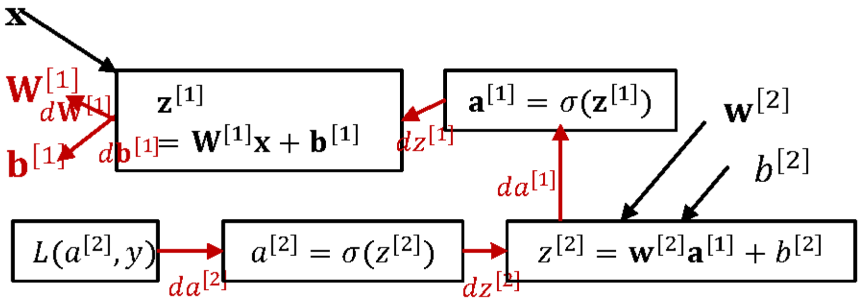


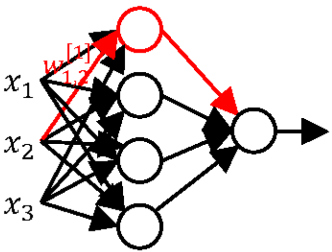
$$dw^{[1]}_{1,1} = \sum_{i=1}^{m} \frac{dL}{da^{[2](i)}} \times \frac{da^{[2](i)}}{dz^{[2](i)}} \times \frac{dz^{[2](i)}}{da_1^{[1](i)}} \times \frac{da_1^{[1](i)}}{dz_1^{[1](i)}} \times \frac{dz_1^{[1](i)}}{dw^{[1]}_{1,1}}$$

$$= \sum_{i=1}^{m} (a^{[2](i)} - y^{[2](i)}) \times w_1^{[2](i)} \times a_1^{[1](i)} (1 - a_1^{[1](i)}) \times x_1^{(i)}$$

$$dw^{[1]}_{1,1} = \frac{1}{m} dw^{[1]}_{1,1},$$

http://cs230.stanford.edu/files/C1M3.pdf





$$dw^{[1]}_{1,2} = \sum_{i=1}^{m} \frac{dL}{da^{[2](i)}} \times \frac{da^{[2](i)}}{dz^{[2](i)}} \times \frac{dz^{[2](i)}}{da_1^{[1](i)}} \times \frac{da_1^{[1](i)}}{dz_1^{[1](i)}} \times \frac{dz_1^{[1](i)}}{dw^{[1]}_{1,2}}$$

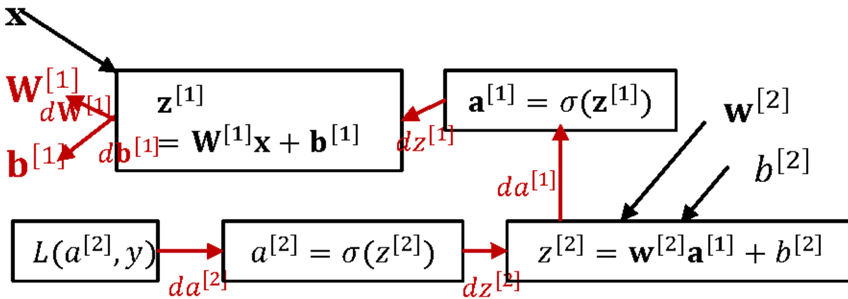
$$= \sum_{i=1}^{m} (a^{[2](i)} - y^{[2](i)}) \times w_1^{[2](i)} \times a_1^{[1](i)}(1 - a_1^{[1](i)}) \times x_2^{(i)}$$

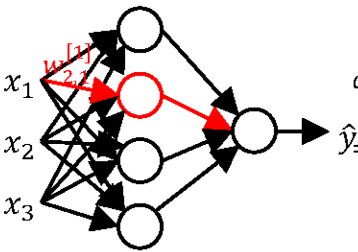
$$db^{[1]}_1 = \sum_{i=1}^{m} \frac{dL}{da^{[2](i)}} \times \frac{da^{[2](i)}}{dz^{[2](i)}} \times \frac{dz^{[2](i)}}{da_1^{[1](i)}} \times \frac{da_1^{[1](i)}}{dz_1^{[1](i)}} \times \frac{dz_1^{[1](i)}}{db^{[1]}_1}$$

$$= \sum_{i=1}^{m} (a^{[2](i)} - y^{[2](i)}) \times w_1^{[2](i)} \times a_1^{[1](i)}(1 - a_1^{[1](i)})$$

$$dw^{[1]}_{1,2} = \frac{1}{m} dw^{[1]}_{1,2},$$

$$db^{[1]}_1 = \frac{1}{m} db^{[1]}_1,$$





$$dw_{2,1}^{[1]} = \sum_{i=1}^{m} \frac{dL}{da^{[2](i)}} \times \frac{da^{[2](i)}}{dz^{[2](i)}} \times \frac{dz^{[2](i)}}{da_2^{[1](i)}} \times \frac{da_2^{[1](i)}}{dz_2^{[1](i)}} \times \frac{dz_2^{[1](i)}}{dw_{2,1}^{[1]}}$$

$$= \sum_{i=1}^{m} (a^{[2](i)} - y^{[2](i)}) \times w_2^{[2](i)} \times a_2^{[1](i)}(1 - a_2^{[1](i)}) \times x_1^{(i)}$$

$$dw_{2,1}^{[1]} = \frac{1}{m} dw_{2,1}^{[1]},$$

$$db_2^{[1]} = \sum_{i=1}^{m} \frac{dL}{da^{[2](i)}} \times \frac{da^{[2](i)}}{dz^{[2](i)}} \times \frac{dz^{[2](i)}}{da_2^{[1](i)}} \times \frac{da_2^{[1](i)}}{dz_2^{[1](i)}} \times \frac{dz_2^{[1](i)}}{db_2^{[1]}}$$

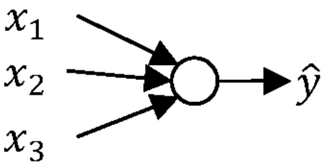
$$db_2^{[1]} = \frac{1}{m} db_2^{[1]},$$

$$= \sum_{i=1}^{m} (a^{[2](i)} - y^{[2](i)}) \times w_2^{[2](i)} \times a_2^{[1](i)}(1 - a_2^{[1](i)})$$

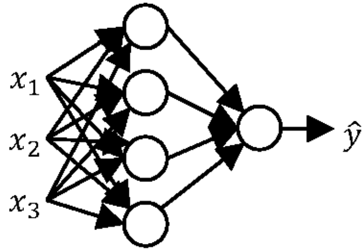
Deep Neural Networks

Deep L-layer Neural network

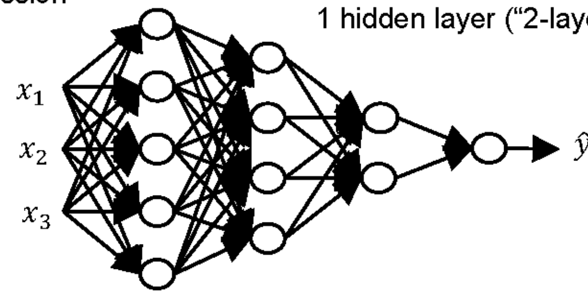
What is a deep neural network?



logistic regression



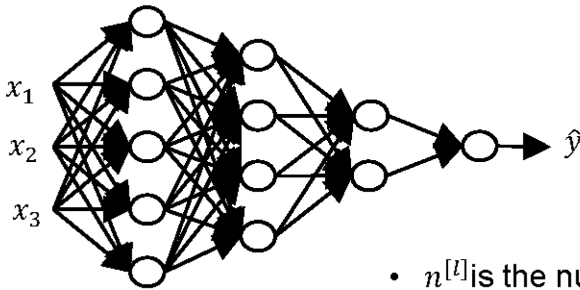
1 hidden layer (“2-layer NN”)



3 hidden layers (“4-layer NN”)

http://cs230.stanford.edu/files/C1M4.pdf

Deep neural network notation



- $n^{[l]}$is the number of the neural unit in layer l
- $\mathbf{a}^{[l]} = activations\ in\ layer\ l$
- $\mathbf{a}^{[l]} = g(\mathbf{z}^{[l]})$
- $\mathbf{W}^{[l]}, \mathbf{b}^{[l]}$ are weights for $\mathbf{z}^{[l]}$

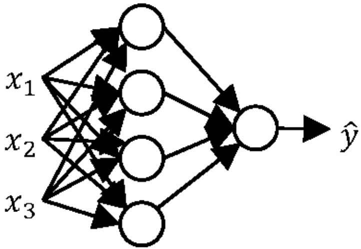
- 3 hidden layers (“4-layer NN”)
- 0-layer: input x_1, x_2, x_3 ($n^{[0]} = 3$)
- 1-layer: 5 neural units ($n^{[1]} = 5$)
- 2-layer: 4 neural units ($n^{[2]} = 4$)
- 3-layer: 2 neural units ($n^{[3]} = 2$)
- 4-layer (output layer): 1 neural units ($n^{[4]} = 1$)

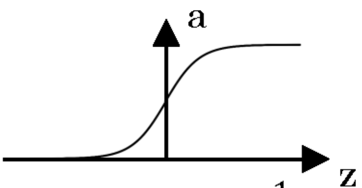
http://cs230.stanford.edu/files/C1M4.pdf

Deep Neural Networks

Activation functions

Activation Functions





sigmoid: $a = \dfrac{1}{1+e^{-z}}$

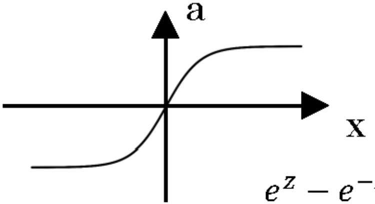
Given input $\mathbf{x}$:

$$\mathbf{z}^{[1]} = \mathbf{W}^{[1]}\mathbf{x} + \mathbf{b}^{[1]}$$

$$\mathbf{a}^{[1]} = \sigma(\mathbf{z}^{[1]}) \qquad g(\mathbf{z}^{[1]})$$

$$z^{[2]} = \mathbf{w}^{[2]}\mathbf{a}^{[1]} + b^{[2]}$$

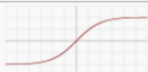
$$a^{[2]} = \sigma(z^{[2]}) \qquad g(z^{[2]})$$



tanh: $a = \dfrac{e^{z}-e^{-z}}{e^{z}+e^{-z}}$

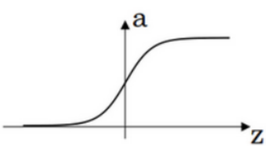
http://cs230.stanford.edu/files/C1M4.pdf

Other activation functions

Name	Plot	Equation	Derivative
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1+e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
TanH		$f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x \geq 0 \end{cases}$
Parameteric Rectified Linear Unit (PReLU)[2]		$f(x) = \begin{cases} \alpha x & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x \geq 0 \end{cases}$

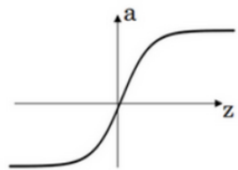
Other activation functions

1) **sigmoid activation function**



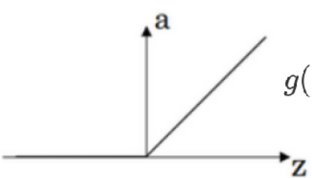
$$g(z) = \frac{1}{1+e^{-z}}$$

2) **Tanh activation function**



$$g(z) = \tanh(z)$$

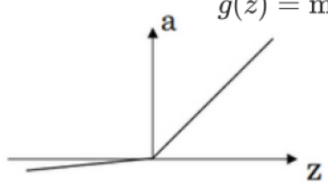
3) **Rectified Linear Unit (ReLU)**



$$g(z) = max(0, z)$$

4) **Leaky linear unit (Leaky ReLU)**

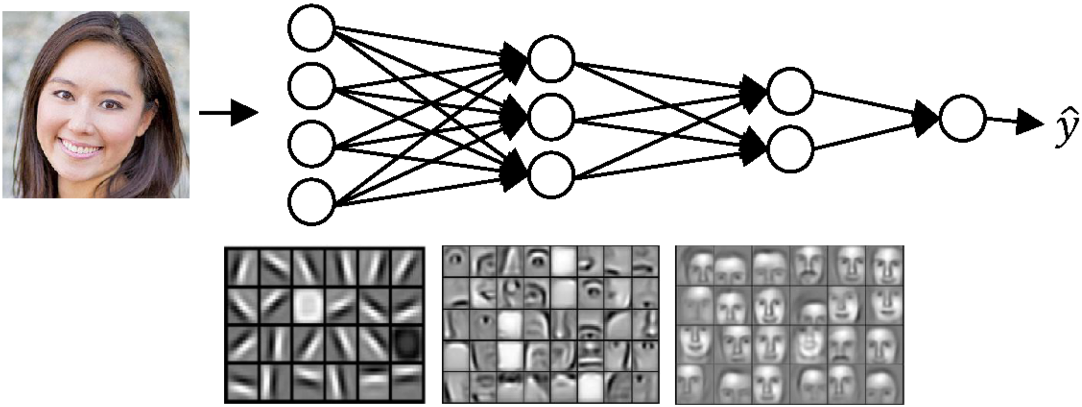
$$g(z) = \max(0.01z, z)$$



Deep Neural Networks

Why deep representations?

Intuition about deep representation



http://cs230.stanford.edu/files/C1M4.pdf

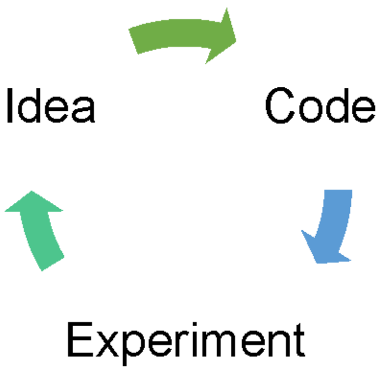
What are hyperparameters?

Parameters: $\mathbf{W}^{[1]}, \mathbf{b}^{[1]}, \mathbf{W}^{[2]}, \mathbf{b}^{[2]}, \mathbf{W}^{[3]}, \mathbf{b}^{[3]}$ ...

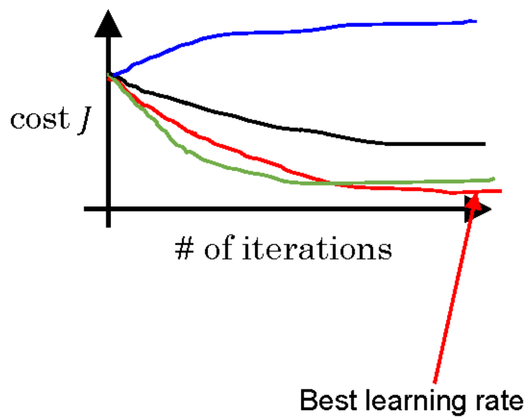
- HyperParameters:
  Learning rate
  The number of iterations
  The number of hidden layers
  The number of neural units in each layer
  Choice of activation function
  mini batch size...

http://cs230.stanford.edu/files/C1M4.pdf

Empirical process in training

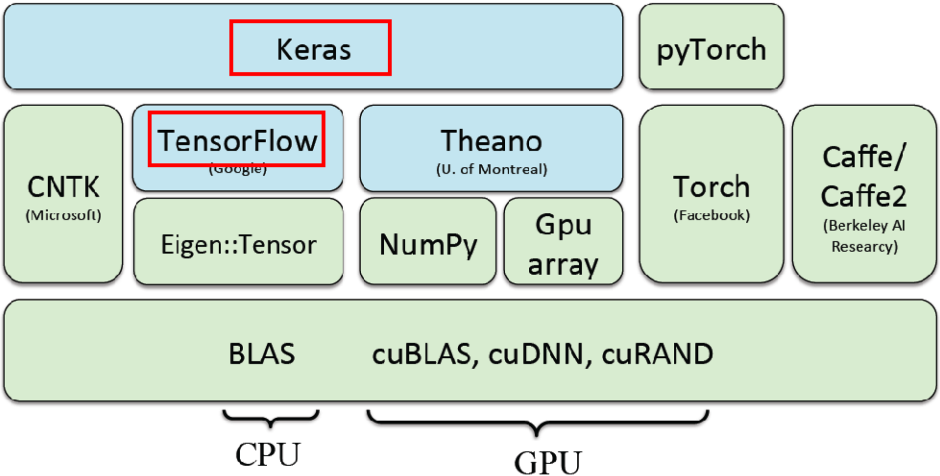


Training with different learning rate



http://cs230.stanford.edu/files/C1M4.pdf

Deep Learning Software



Data Science

Week 10

Fitting a Model to Data (3)-
Regression