Deep Learning

Logistic Regression Classifier

Dr. Mehran Safayani

safayani@iut.ac.ir

safayani.iut.ac.ir

https://www.aparat.com/mehran.safayani

https://github.com/safayani/deep_learning_course

Department of Electrical and computer engineering, Isfahan university of technology, Isfahan, Iran

Supervised Learning

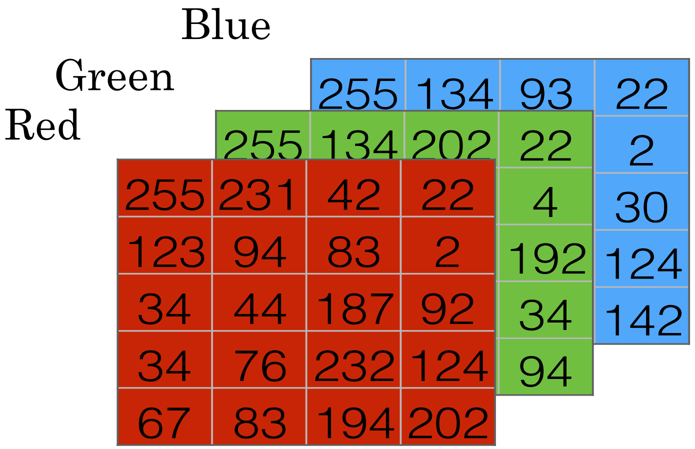
Input(x)	Output (y)	Application
Home features	Price	Real Estate
Ad, user info	Click on ad? (0/1)	Online Advertising
Image	Object (1,…,1000)	Photo tagging
Audio	Text transcript	Speech recognition
English	Chinese	Machine translation
Image, Radar info	Position of other cars	Autonomous driving

Basics of Neural Network Programming

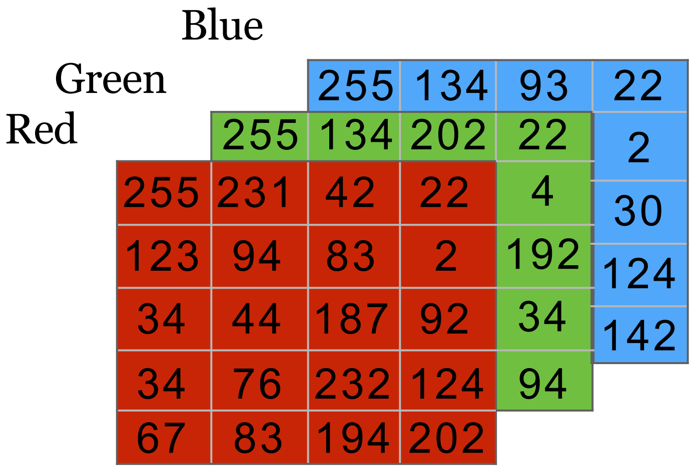
Binary Classification

Binary Classification

1 (cat) vs 0 (non cat)



Binary classification



$$\vec{x} = \begin{bmatrix} 255 \\ 231 \\ \vdots \\ 254 \\ 253 \\ 250 \\ 220 \end{bmatrix} \qquad 64 \times 64 \times 3 = \underbrace{12288}_{n_x}$$

$$\vec{x} \rightarrow \boxed{\text{model}} \rightarrow \hat{y}$$

- Notation

$(\vec{x}, y) \qquad x \in R^{n_x}, \ y \in \{0,1\}$

Binary classification

- m training example: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}, \ldots, (x^{(m)}, y^{(m)})\}$
- $X = \underbrace{\begin{bmatrix} | & | & & | \\ x^{(1)} & x^{(2)} & \ldots\ldots\ldots & x^{(\mathrm{m})} \\ | & | & & | \end{bmatrix}}_{\mathrm{m}} \; n_x$

$X \in \mathbb{R}^{n_x \times m}$

$X.\mathrm{shape} = (n_x, m)$

$Y = [y^{(1)}, y^{(2)}, \ldots, y^{(m)}]$

$Y \in \mathbb{R}^{1 \times m}$

$Y.\mathrm{shape} = (1, m)$

Logistic Regression

- Given x , output $\hat{y} = P(y=1|x)$ $0 \le \hat{y} \le 1$

  $x \in \mathbb{R}^{n_x}$ parameters: $w \in \mathbb{R}^{n_x}$, $b \in \mathbb{R}$

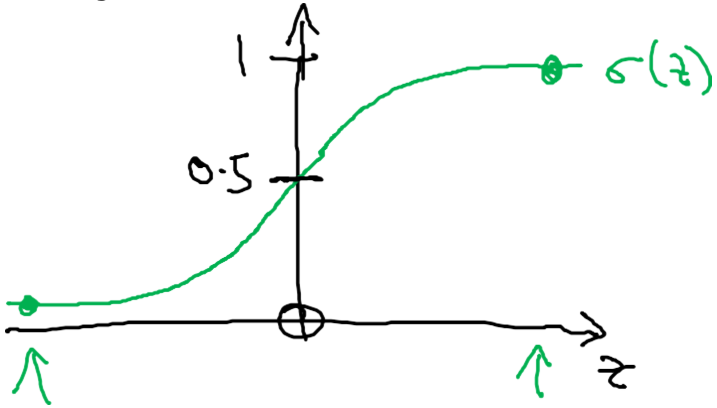
$$\hat{y} = w^T x + b$$

$$\hat{y} = \sigma(\overbrace{w^T x + b}^{z})$$

Sigmoid function $\sigma(z) = \frac{1}{1+e^{-z}}$

if z large $\sigma(z) \approx 1$

if z large negative $\sigma(z) \approx 0$



Logistic Regression

- $\hat{y} = \sigma(\underbrace{w^T x + b}_{z})$

$$\hat{y} = w^T x$$

$$x_0 = 1\,, x \in \mathbb{R}^{n_x+1}$$

$$W = \begin{array}{l} \left.\begin{bmatrix} b = w_0 \\ w_1 \\ w_2 \\ w_3 \\ . \\ . \\ . \\ w_{n_x} \end{bmatrix}\right. \end{array} \begin{array}{l} \}\ \mathrm{b} \\ \left.\begin{array}{l} \\ \\ \\ \\ \\ \\ \\ \end{array}\right\}\ \mathrm{w} \end{array}$$

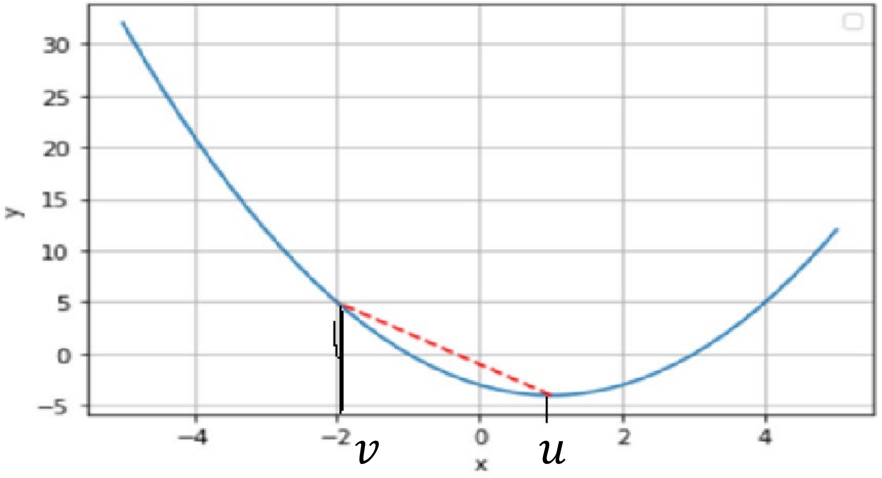
Logistic Regression cost function

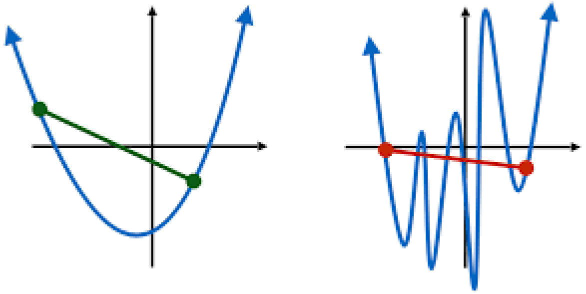
- Loss (error) function: $L(\hat{y}, y) = \frac{1}{2}(\hat{y} - y)^2 = \frac{1}{2}\left(\sigma\left(w^T x + b\right) - y\right)^2$ SE: Square Error

- What is the problem?

- .

Convexity





convex nonconvex

https://mlstory.org/optimization.html

Function h(u) with u∈ X is convex if for any u, v ∈ X and for any $0 \leq \lambda \leq 1$ we have:

$$\mathbf{h(\lambda u + (1 - \lambda) v) \leq \lambda\, h(u) + (1 - \lambda)\, h(v)}$$

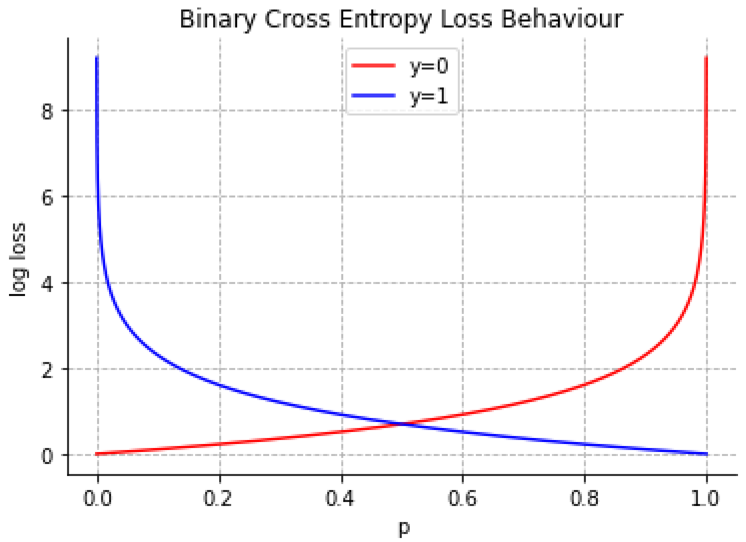
برای توابع محدب هر بهینه محلی یک بهینه سراسری است.

Cross Entropy

- $L(\hat{y}, y) = -(y \log \hat{y} + (1-y)\log(1-\hat{y}))$

if $y=1$: $L(\hat{y}, y) = -\log \hat{y}$

if $y=0$: $L(\hat{y}, y) = -\log(1-\hat{y})$



https://datamonje.com/classification-loss-functions/

- Cost function: $J(w,b) = \frac{1}{m}\sum_{i=1}^{m} L\left(y^{(i)}, \hat{y}^{(i)}\right)$

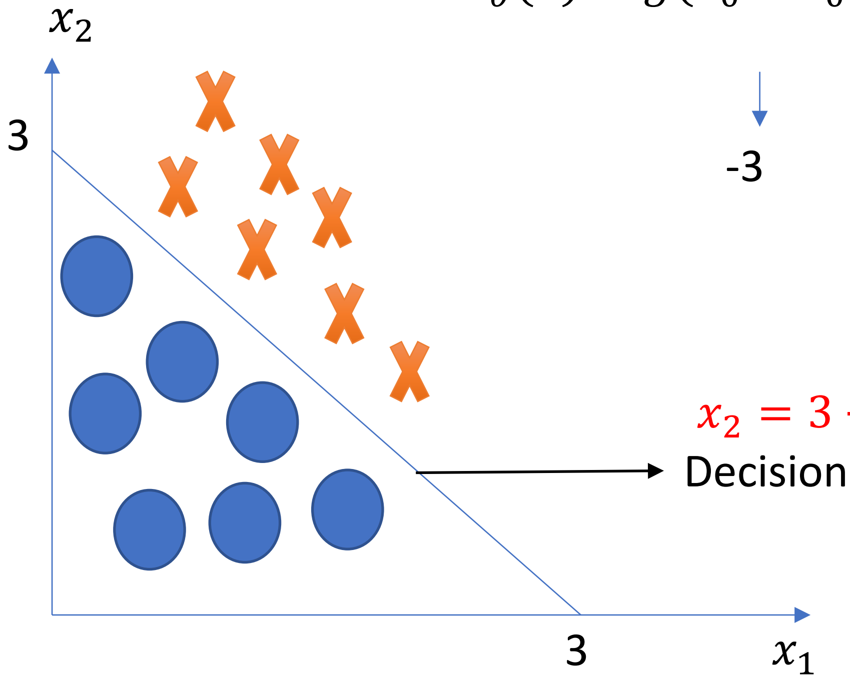
$$= -\frac{1}{m}\sum_{i=1}^{m} [y^{(i)}\log \hat{y}^{(i)} + (1-y^{(i)})\log(1-\hat{y}^{(i)})]$$

Decision Boundary

$$\theta = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$$

$$h_\theta(x) = g(\theta_0 + \theta_0 x_1 + \theta_2 x_2)$$

-3 1 1



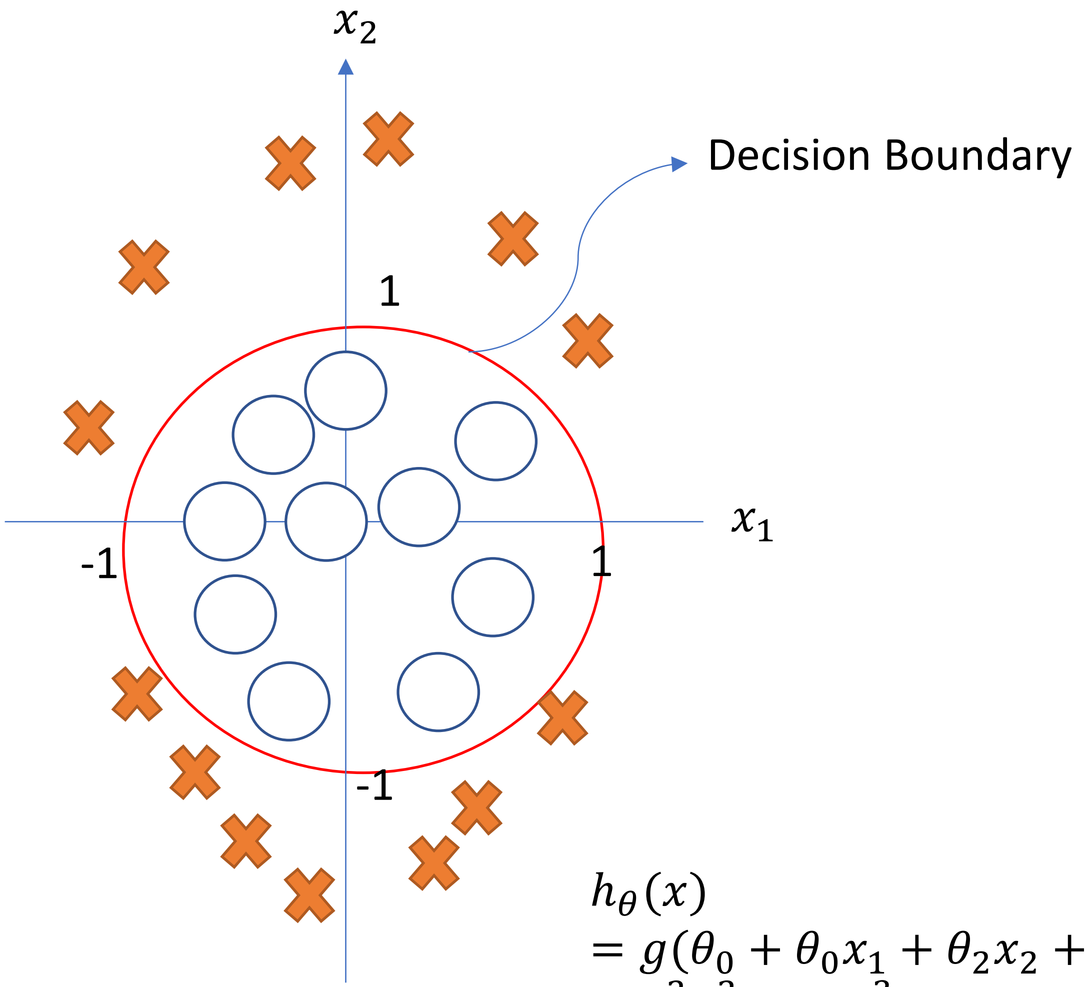
$x_2 = 3 - x_1$

Decision Boundary

Predict $y = 1, if \underbrace{-3 + x_1 + x_2 \geq 0}_{\theta^T X}$

$x_1 + x_2 \geq 3$

Non-Linear Decision Boundaries



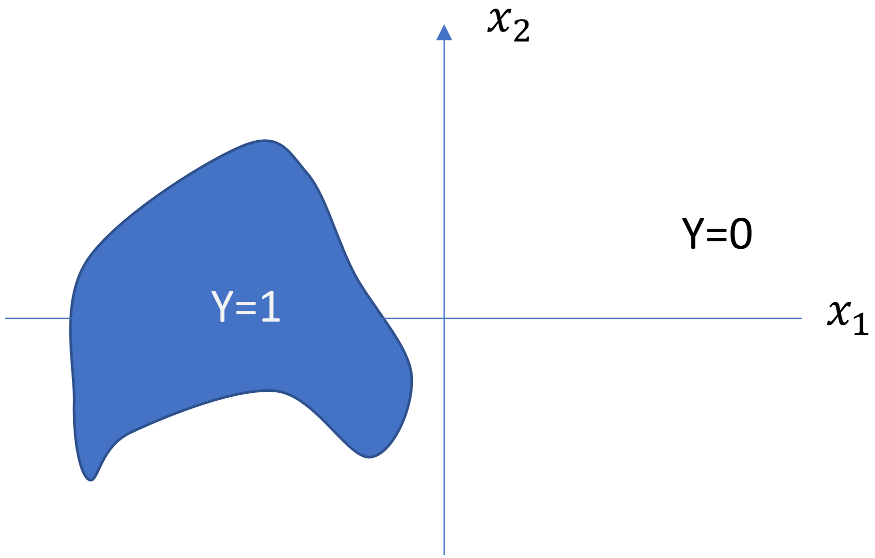
$$h_\theta(x) = g(\overset{-1}{\theta_0} + \overset{0}{\theta_0} x_1 + \overset{0}{\theta_2} x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$$

$$\theta = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

Predict $y = 1, if \underbrace{-3 + x_1^2 + x_2^2 \geq 0}_{\theta^T X}$

$$x_1{}^2 + x_2{}^2 \geq 1$$

$$h_\theta(x) = g(\theta_0 + \theta_0 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_1^2 x_2 + \theta_5 x_1^2 x_2^2 + \theta_6 x_1^3 x_2 + \ldots)$$



Gradient Descent

Cost Function

Minimize $\boldsymbol{J(b, w_1)}$

$\boldsymbol{b, w_1}$

If $J(w_1) = (w_1 - 2)^2$

$$\frac{dJ(w_1)}{dw_1} = 0 \quad \Rightarrow \quad \frac{dJ(w_1)}{dw_1} = 2(w_1 - 2) = 0 \quad \Rightarrow \quad w_1 = 2$$

Gradient Descent

Minimize $\boldsymbol{J(b, w_1)}$

$\boldsymbol{b, w_1}$

Minimize $\boldsymbol{J(b, w_1, \dots, w_n)}$

$\boldsymbol{b, w_1, \dots, w_n}$

Repeat until convergence: {

b= $\boldsymbol{w_0}$

For j=0,…,n

$$\boldsymbol{w_j} = \boldsymbol{w_j} - \alpha \frac{d\boldsymbol{J(b, w_1, \dots, w_n)}}{d\boldsymbol{w_j}}$$

}

α **is learning rate**

Updating all $\boldsymbol{w_j}$ $\boldsymbol{Simultaneously}$

Convergence condition:

$$\|W^{t+1} - W^{t}\|_2 \leq \varepsilon$$

Gradient Descent

Correct form

$$\text{temp0} = b - \alpha \frac{dJ(b, w_1)}{db}$$

$$\text{temp1} = w_1 - \alpha \frac{dJ(b, w_1)}{dw_1}$$

بروزرسانی همزمان

b = temp0

w_1 = temp1

✓

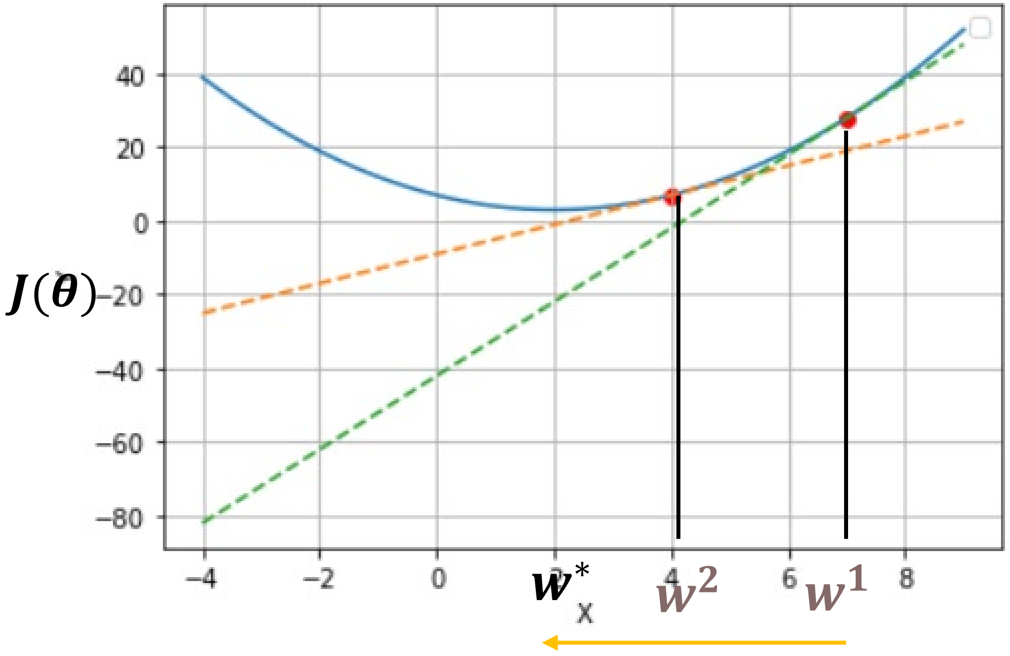
Incorrect form

$$b = b - \alpha \frac{dJ(b , w_1)}{db}$$

$$w_1 = w_1 - \alpha \frac{dJ(b, w_1)}{dw_1}$$

✗

Gradient Descent



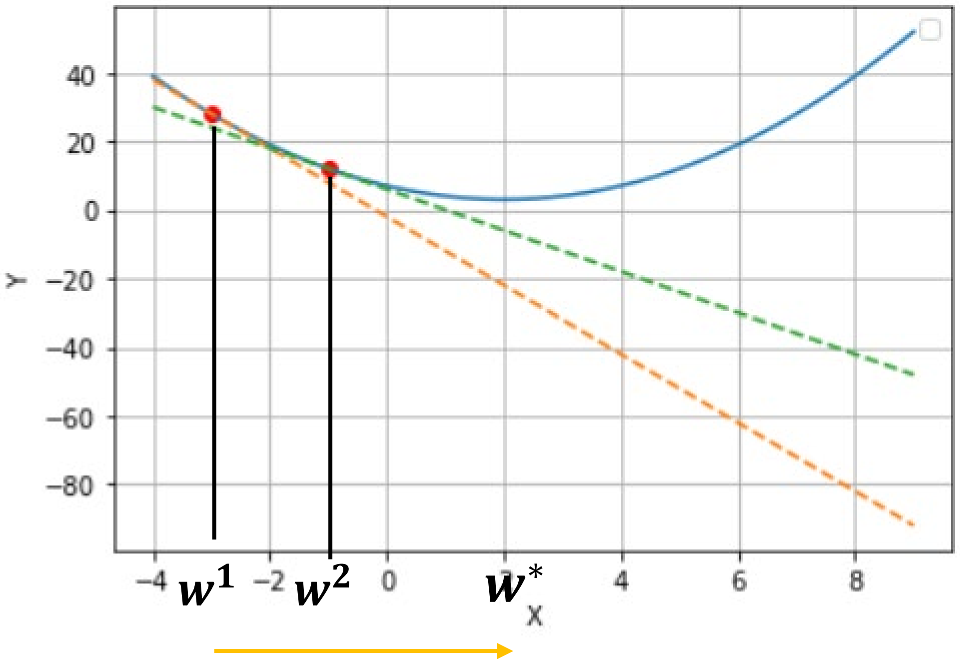
خطوط مماس نشان داده شده دارای شیب یا مشتق مثبت هستند.
درنتیجه:

$$\frac{dJ(w^1)}{dw^1} > 0\ ,\ \alpha > 0 \Rightarrow \alpha\frac{dJ(w^1)}{dw^1} > 0$$

$$\Rightarrow w^2 = w^1 - \alpha dw^1$$

w کوچکتر میشود و به سمت چپ حرکت میکنیم.

Gradient Descent



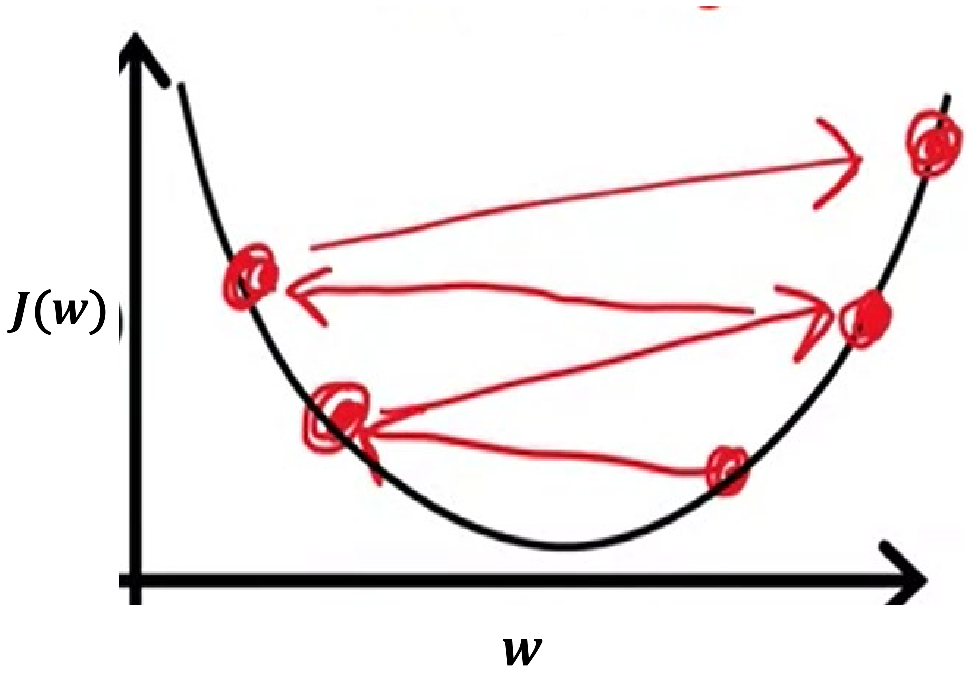
خطوط مماس نشان داده شده دارای شیب یا مشتق مثبت هستند.
درنتیجه:

$$\frac{dJ(w^1)}{dw^1} < 0\ ,\ \ \alpha > 0 \Rightarrow \alpha\frac{dJ(w^1)}{dw^1} < 0$$

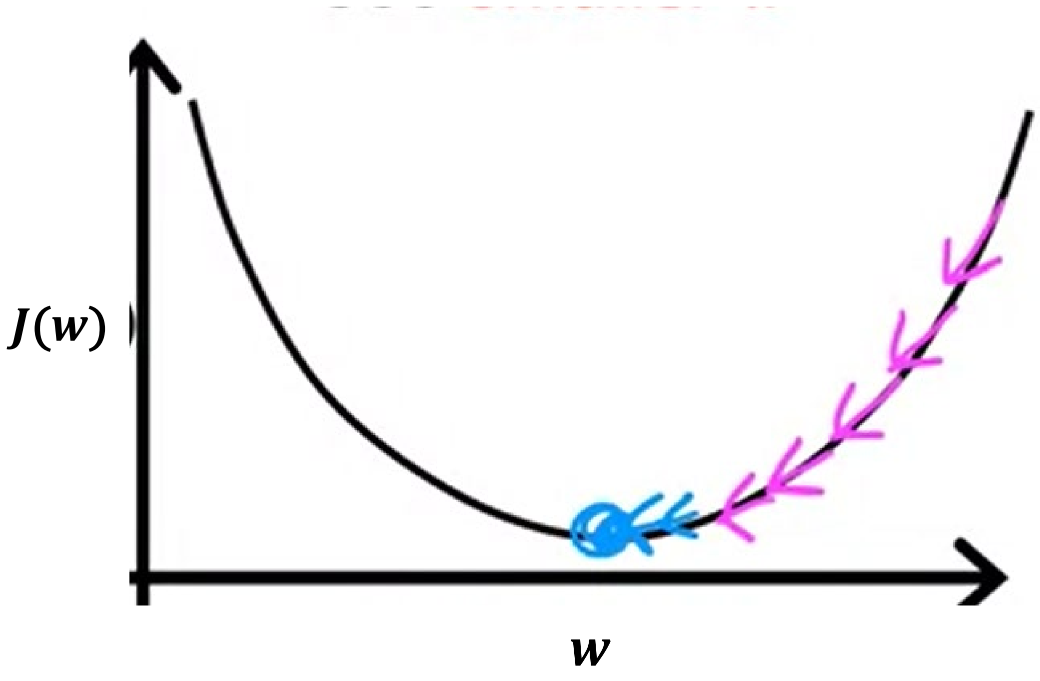
$$\Rightarrow w^2 = w^1 - \alpha d w^1$$

w بزرگتر میشود و به سمت راست حرکت میکنیم.

Choosing Learning Rate

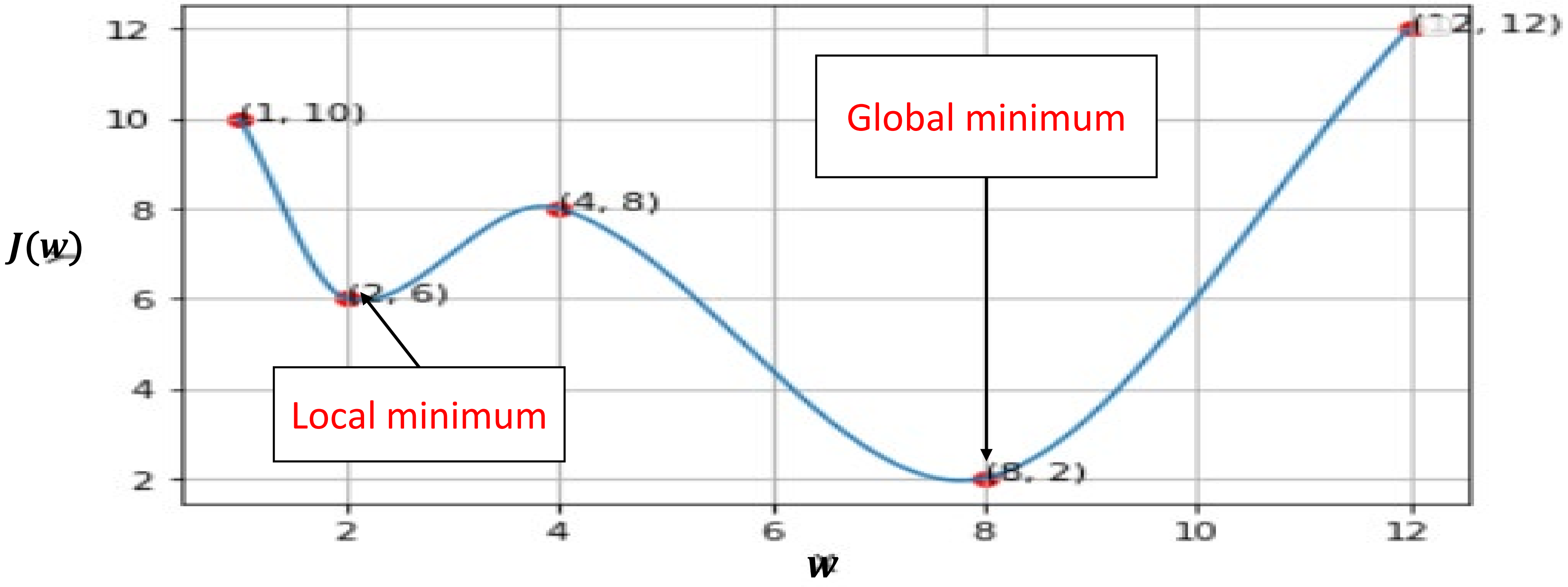


α is too large

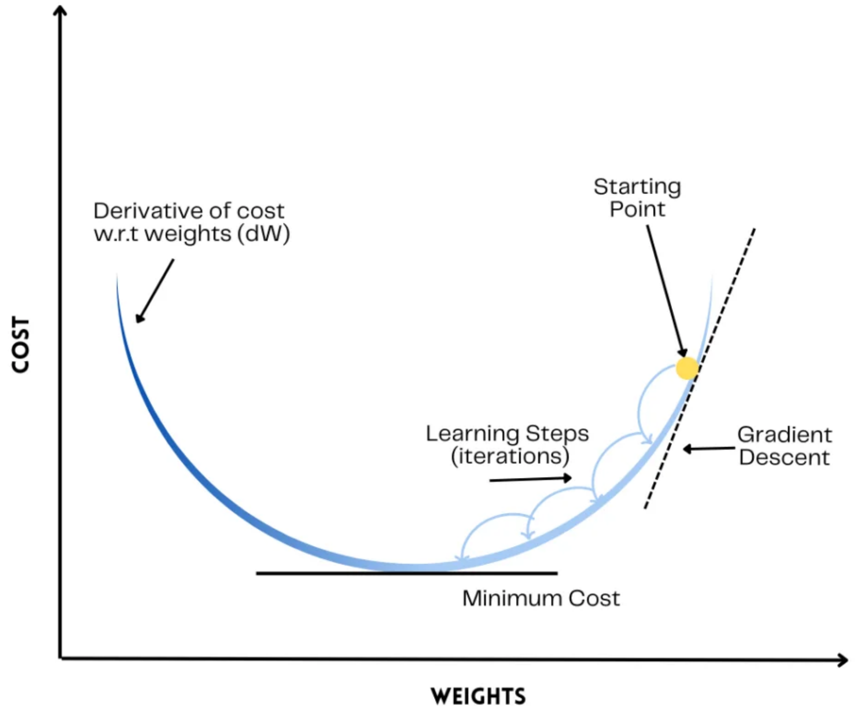


α is small

Gradient Descent Weakness



Gradient Descent



- 1) $\alpha > 0$

$$\text{Repeat}\{$$

$$w = w - \alpha \underbrace{\frac{dJ(w)}{d(w)}}_{dw}$$

$$\}\text{until convergence}$$

$$w = w - \alpha dw$$

$$z = w^T x + b$$

- $\hat{y} = a = \sigma(z)$
- $L(a, y) = -(y \log a + (1 - y) \log(1 - a))$

https://www.askpython.com/

Gradient Descent

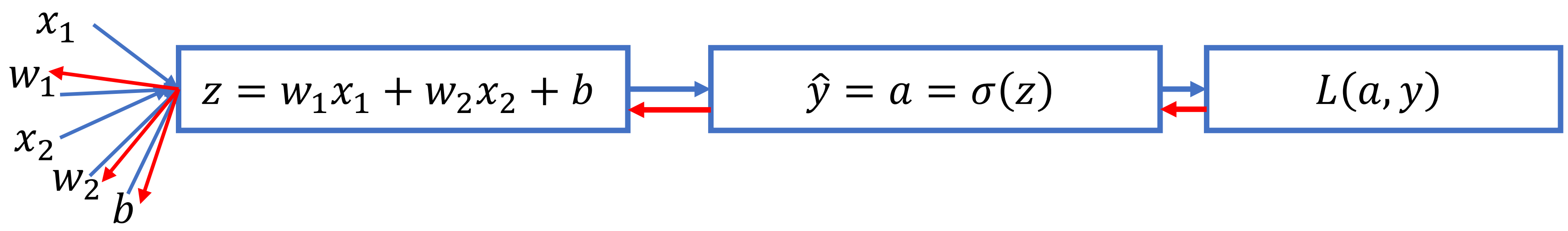
$$L(a, y) = -(y \log a + (1 - y) \log(1 - a))$$

Computational Graph

$$\frac{da}{dz} = \acute{\sigma}(z) = \underbrace{\sigma(z)}_{a}\,\underbrace{(1 - \sigma(z))}_{1-a}$$

$$da = \frac{dL}{da} = \frac{-y}{a} + \frac{1 - y}{1 - a} = \frac{a - y}{a(1 - a)}$$

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



$$dz = \frac{dL}{dz} = \frac{dL(a, y)}{dz} = \frac{dL}{da} \times \frac{da}{dz} = \frac{a - y}{a(1 - a)} \times a(1 - a) = a - y$$

$$dw_1 = \frac{dL}{dw_1} = \frac{dL}{\underbrace{dz}_{dz}} \times \frac{dz}{\underbrace{dw_1}_{x_1}} = x_1\, dz$$

$$dw_2 = x_2\, dz$$

$$db = \frac{dL}{db} = \frac{dL}{dz} \times \frac{\overbrace{dz}^{1}}{db} = dz$$

Gradient Descent

- $J(w,b) = \frac{1}{\mathrm{m}} \sum_{\mathrm{i}=1}^{\mathrm{m}} \mathrm{L}\left(a^{(\mathrm{i})}, \mathrm{y}^{(\mathrm{i})}\right)$
- $a^{(i)} = \hat{y}^{(i)} = \sigma\left(z^{(i)}\right) = \sigma\left(wx^{(i)} + b\right)$

- $dw_j = \frac{1}{\mathrm{m}} \sum_{\mathrm{i}=1}^{\mathrm{m}} \frac{d\mathrm{L}\left(a^{(\mathrm{i})}, \mathrm{y}^{(\mathrm{i})}\right)}{dw_j}$

Logistic regression on m examples

$J = 0$; $dw_1 = 0$; $dw_2 = 0$; $db = 0$;

$w_1 \leftarrow \text{random}$ $w_2 \leftarrow \text{random}$ $b \leftarrow \text{random}$

$$w^t = \begin{bmatrix} w_1^t \\ w_2^t \\ b^t \end{bmatrix} \quad w^{t+1} = \begin{bmatrix} w_1^{t+1} \\ w_2^{t+1} \\ b^{t+1} \end{bmatrix}$$

$$\|w^{t+1} - w^t\|_2 \leq \varepsilon$$

Repeat{

For i=1 to m

$$z^{(i)} = w^T x^{(i)} + b$$

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

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

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

$$dw_1 += x_1^{(i)} dz^{(i)} \qquad dw_2 += x_2^{(i)} dz^{(i)} \qquad db += dz^{(i)}$$

$$dw = \begin{bmatrix} dw_1 \\ dw_2 \\ db \end{bmatrix}$$

$$\|dw\| \leq \varepsilon = 10^{-4}$$

$J /= m$; $dw_1 /= m$; $dw_2 /= m$; $db /= m$;

$w_1 = w_1 - \alpha\, dw_1$ $w_2 = w_2 - \alpha\, dw_2$ $b = b - \alpha\, db$

What's wrong with the code?

} until convergence

Logistic regression on m examples

$w_1 \leftarrow \text{random} \quad w_2 \leftarrow \text{random} \quad b \leftarrow \text{random}$

$$w^t = \begin{bmatrix} w_1^t \\ w_2^t \\ b^t \end{bmatrix} \quad w^{t+1} = \begin{bmatrix} w_1^{t+1} \\ w_2^{t+1} \\ b^{t+1} \end{bmatrix}$$

Repeat{

$J = 0; \quad dw_1 = 0; \quad dw_2 = 0; \quad db = 0;$

$$\|w^{t+1} - w^t\|_2 \le \varepsilon$$

For i=1 to m

$$z^{(i)} = w^T x^{(i)} + b$$

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

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

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

$$dw_1 \mathrel{+}= x_1^{(i)} dz^{(i)} \quad dw_2 \mathrel{+}= x_2^{(i)} dz^{(i)} \quad db \mathrel{+}= dz^{(i)}$$

$$dw = \begin{bmatrix} dw_1 \\ dw_2 \\ db \end{bmatrix}$$

$$\|dw\| \le \varepsilon = 10^{-4}$$

$J/= m; \quad dw_1/= m; \quad dw_2/= m; \quad db/= m;$

$w_1 = w_1 - \alpha\, dw_1 \quad w_2 = w_2 - \alpha\, dw_2 \quad b = b - \alpha\, db$

} until convergence

Vectorizing Logistic Regression

- *z=0;*

  For i in range(n_x)

  $$z \mathrel{+}= w[i] * x[i]$$

  $$z \mathrel{+}= b$$

First method

- *z=0;*

  z =np.dot(w, x)+b

Second method

SIMD

GPU

Vectorizing Logistic Regression

$z^{(1)} = w^T x^{(1)} + b$ $\quad z^{(2)} = w^T x^{(2)} + b$ $\quad z^{(m)} = w^T x^{(m)} + b$

$a^{(1)} = \sigma(z^{(1)})$ $\quad a^{(2)} = \sigma(z^{(2)})$ $\quad a^{(m)} = \sigma(z^{(m)})$

- $\mathrm{X} = \begin{bmatrix} | & | & & | \\ x^{(1)} & x^{(2)} & \cdots\cdots\cdots & x^{(\mathrm{m})} \\ | & | & & | \end{bmatrix} \in R^{n_x \times m}$ $\quad \underbrace{[w_1 \ldots w_{n_x}]}_{w^T} \begin{bmatrix} | & | & & | \\ x^{(1)} & x^{(2)} & \cdots\cdots\cdots & x^{(\mathrm{m})} \\ | & | & & | \end{bmatrix}$
- $Z = \begin{bmatrix} \underbrace{z^{(1)}}_{w^T x^{(1)} + b} & \underbrace{z^{(2)}}_{w^T x^{(2)} + b} & \ldots & \underbrace{z^{(m)}}_{w^T x^{(m)} + b} \end{bmatrix} = w^T X + [b \; b \cdots b]_{1 \times m}$
- $Z = \underbrace{np \cdot dot(w \cdot T, X)}_{1_\times m} + \underbrace{b}_{(1{,}1)}$ *"broadcasting"* $[b, b \ldots, b]_{1 \times m}$

Vectorizing Logistic Regression

- $A = [a^{(1)}, a^{(2)}, \dots, a^{(m)}] = \sigma(\underbrace{Z}_{1\times m})$
- $dz^{(1)} = a^{(1)} - y^{(1)}$ $\quad dz^{(2)} = a^{(2)} - y^{(2)}$
- $dZ = [dz^{(1)}\ dz^{(2)} \dots dz^{(m)}]_{1\times m}$
- $A = [a^{(1)}\ a^{(2)} \dots a^{(m)}]$ $\quad Y = [y^{(1)}\ y^{(2)} \dots y^{(m)}]$
- $dZ = A - Y = [a^{(1)} - y^{(1)}\ \ a^{(2)} - y^{(2)}\ \dots\ a^{(m)} - y^{(m)}]$

$$1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ \vdots \\ \vdots \\ 1 \end{bmatrix}_{m\times 1}$$

- $dw=0$ $\quad db=0$

$$\begin{bmatrix} dw_1 \\ dw_2 \\ dw_3 \\ \vdots \\ dw_{nx} \end{bmatrix}_{nx\times 1}$$

$$\begin{cases} dw \mathrel{+}= \underbrace{x^1}_{\text{بردار}}\ \underbrace{dz^1}_{\overset{\text{عدد}}{a^1-y^1}} \\ dw \mathrel{+}= x^2 dz^2 \\ dw \mathrel{+}= x^m dz^m \end{cases}$$

$dw /= m$

$$\begin{cases} db \mathrel{+}= \underbrace{dz1}_{\overset{\text{عدد}}{a1-y1}} \\ db \mathrel{+}= dz2 \\ db \mathrel{+}= dz^m \end{cases}$$

$db /= m$

$$db = \frac{1}{m}\sum_{i=1}^{m} dz^{(i)} = \frac{1}{m} np.sum(dz)$$

$$db = \frac{1}{m} dz \times 1$$

$$dz = [dz^1\ \ dz^2 \dots dz^m]_{1\times m}$$

$$dw = \frac{1}{m}\left[x^{(1)}dz^1 + x^{(2)}dz^2 + \cdots + x^{(m)}dz^m\right]$$

$$\boxed{dw = \frac{1}{m} X\, dz^T} = \frac{1}{m}\begin{bmatrix} x^{(1)} x^{(2)} & \dots\dots\dots & x^{(m)} \end{bmatrix} \times \begin{bmatrix} dz^1 \\ dz^2 \\ dz^3 \\ \vdots \\ \vdots \\ \vdots \\ dz^m \end{bmatrix}$$

Vectorizing Logistic Regression

- $w_1, w_2, b \leftarrow random$

For iter in range(1000)

$$\textbf{1 epoch}\left\{\begin{aligned}
&Z = W^T X + b = np \cdot dot(w.T, X) + b \\
&A = \sigma(Z) \\
&dZ = A - Y \\
&dw = \frac{1}{m} X\, dz^T \\
&db = \frac{1}{m} np.sum(dz) \\
&w = w - \alpha\, dw \\
&b = b - \alpha\, db
\end{aligned}\right.$$

$$w = np \cdot random \cdot \mathrm{randn}(n_x, 1)$$

Logistic Regression Cost function

- $\hat{y} = \sigma(w^T x + b) \qquad 0 < \sigma(z) = \frac{1}{1+e^{-z}} < 1$
- $\hat{y} = p(y = 1|x)$
- $\left.\begin{array}{l} if \quad y = 1 \;\; : \quad p(y|x) = \hat{y} \\ if \quad y = 0 \;\; : \quad p(y|x) = 1 - \hat{y} \end{array}\right\} \boldsymbol{p(y|x)}$

- $\boxed{p(y|x) = \hat{y}^y (1 - \hat{y})^{1-y}} \qquad distribution?\ Bernoulli$
- $if \quad y = 1 \;\; : \quad p(y|x) = \hat{y}$
- $if \quad y = 0 \;\; : \quad p(y|x) = 1 - \hat{y}$
- $\mathrm{logp(y|x)} = \log[\hat{y}^y (1 - \hat{y})^{1-y}] = y \log \hat{y} + (1 - y) \log(1 - \hat{y})$

  $= -L(\hat{y}, y) \;\; Max\ Likelihood$

Logistic Regression Cost function

- $\log P(\, labels\ in\ trainingset\,) = \log \prod_{i=1}^{m} P\left(y^{i} \mid x^{i}\right)$
- $\log P(\cdots) = \sum_{i=1}^{m} \log P\left(y^{(i)} \mid x^{(i)}\right) = -\sum_{i=1}^{m} L\left(\hat{y}^{(i)}, y^{(i)}\right)$

- Cost function $\underbrace{J(w,b)}_{\boldsymbol{minimize}} = \frac{1}{m} \sum_{i=1}^{m} L\left(\hat{y}^{(i)}, y^{(i)}\right)$

Neural Networks