Computer Vision:
Logistic regression

Siheng Chen 陈思衡

Logistic regression

Supervised learning: Classification

Application: Image classification

Method: Logistic regression

Optimization

Image classification

ImageNet LSVRC-2010 contest:

- Dataset: 1.2 million labeled images, 1000 classes
- Task: Given a new image, label it with the correct class
- Multiclass classification problem

http://image-net.org/

Image classification

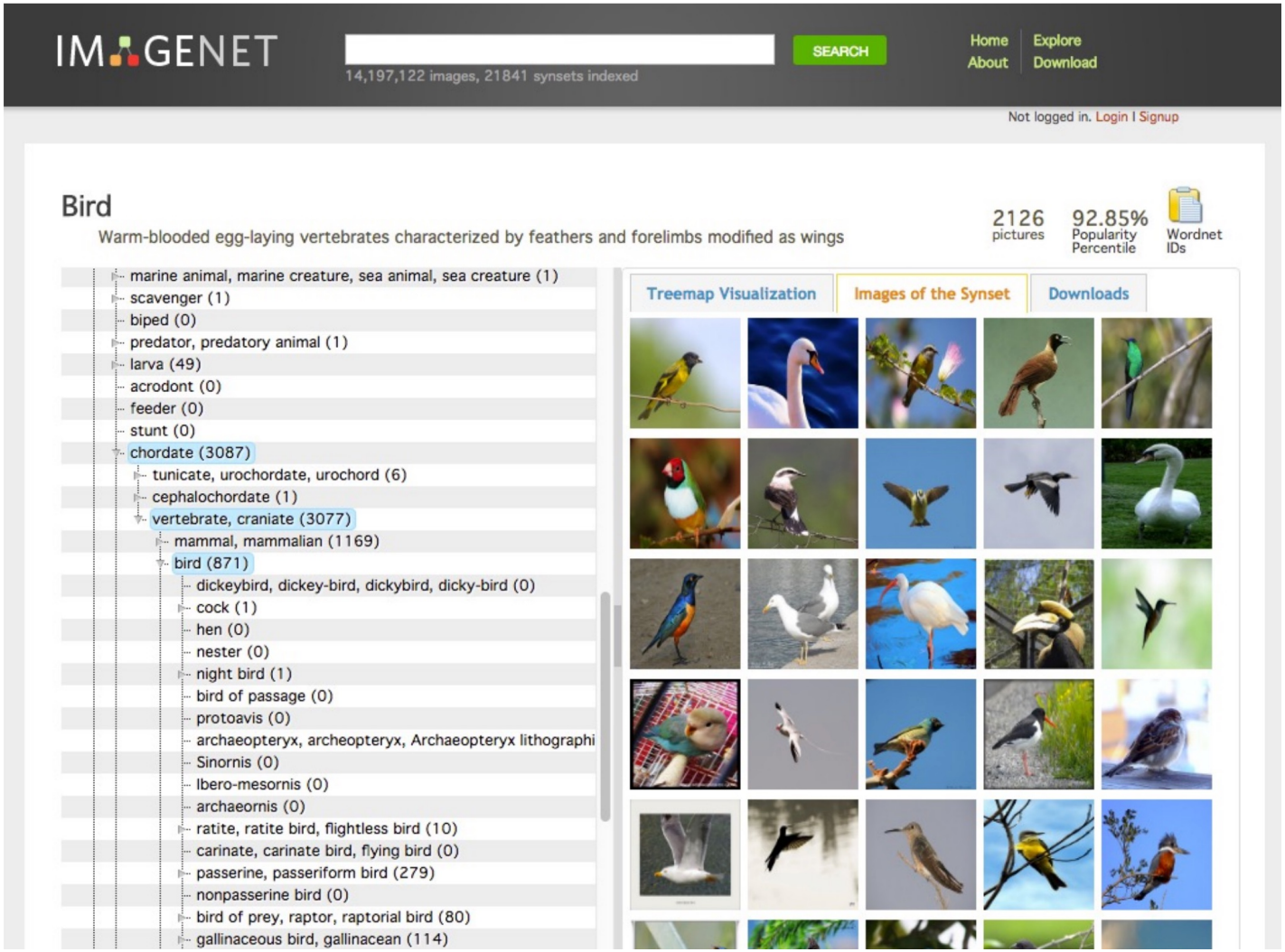


Image classification

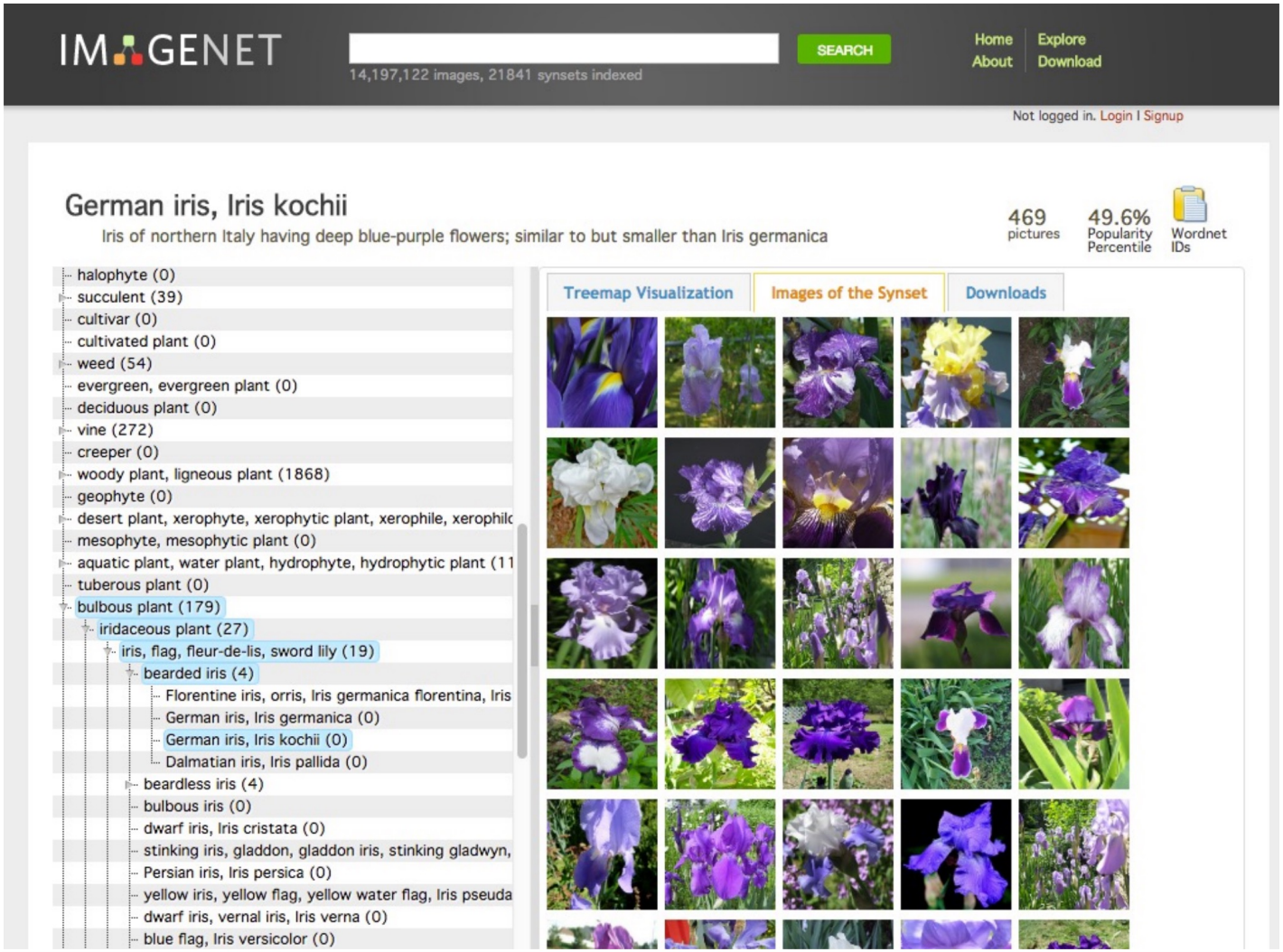


Image classification

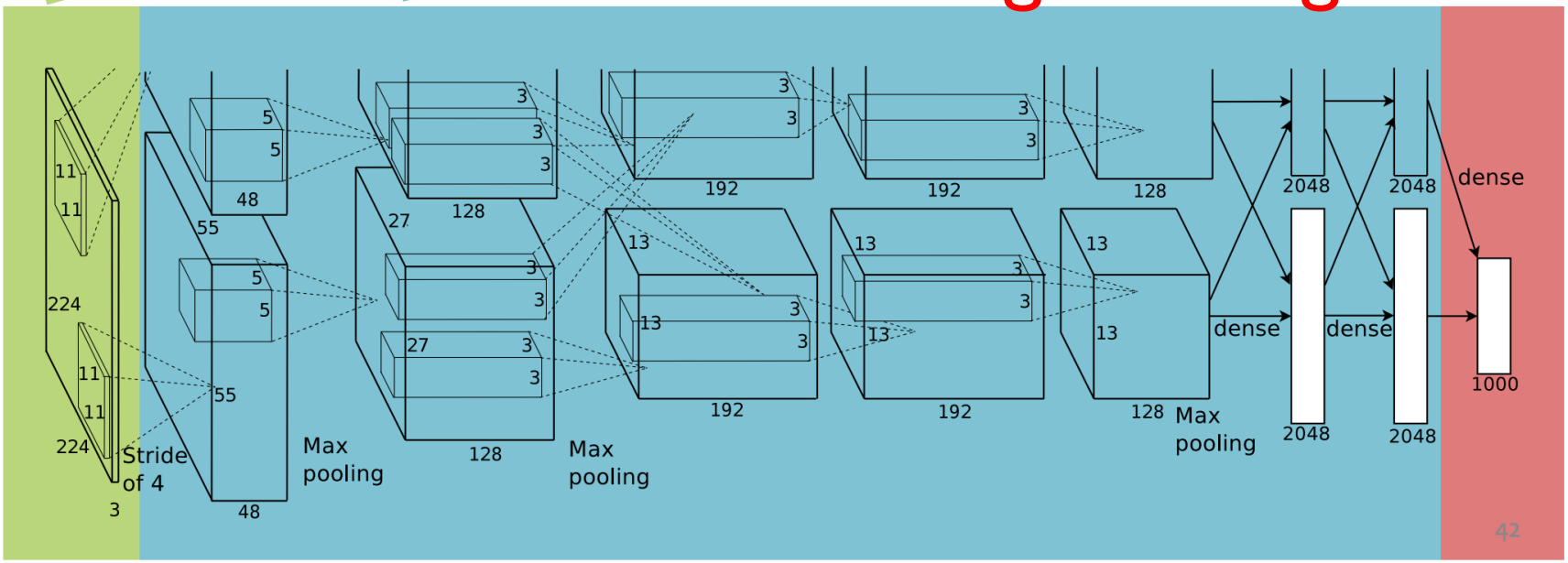
CNN for Image Classification
(Krizhevsky, Sutskever & Hinton, 2011)
17.5% error on ImageNet LSVRC-2010 contest

Input image (pixels)

- Five convolutional layers (w/max-pooling)
- Three fully connected layers

1000-way softmax

Logistic Regression!



http://www.cs.cmu.edu/~mgormley/courses/10601/slides/lecture9-logreg.pdf

Binary classification

PCA

classifier

1 (human face) vs.
0 (non human face)

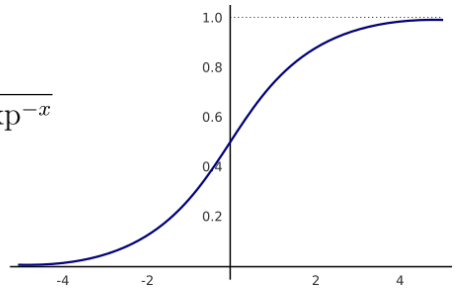
$\mathbf{I} \in \mathbb{R}^{H \times W}$ $\quad \mathbf{x} \in \mathbb{R}^{d}$ $\quad y \in \{0, 1\}$

classifier: $\widehat{y} = P(y = 1|\mathbf{x})$

logistic regression: $\widehat{y} = \sigma(\mathbf{w}^T \mathbf{x} + b)$

trainable parameters

$$\sigma(x) = \frac{1}{1 + \exp^{-x}}$$



Logistic regression

m training examples: $\{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \cdots, (\mathbf{x}^{(m)}, y^{(m)})\}$

model: $\hat{y} = \sigma(\mathbf{w}^T \mathbf{x} + b)$

loss function (cross entropy): $L(\hat{y}, y) = -(y \log \hat{y} + (1 - y) \log(1 - \hat{y}))$

cost function:

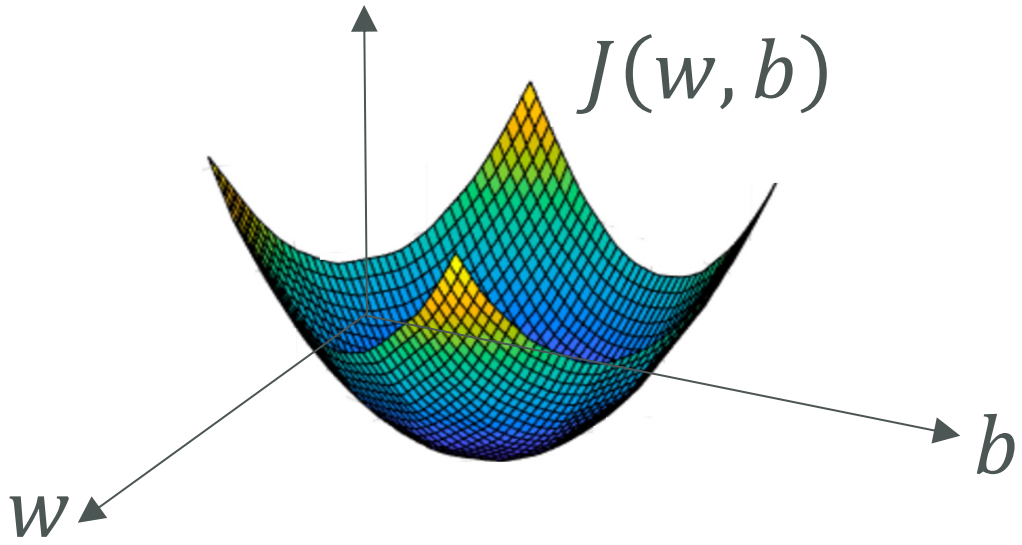
$$J(\mathbf{w}, b) = \frac{1}{m} \sum_{i=1}^{m} L(\hat{y}^{(i)}, y^{(i)})$$

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

Logistic regression

want to find w, b that minimize $J(w, b)$

$$J(\mathbf{w}, b) = \frac{1}{m}\sum_{i=1}^{m} L(\hat{y}^{(i)}, y^{(i)}) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)} \log \hat{y}^{(i)} + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)})\right]$$

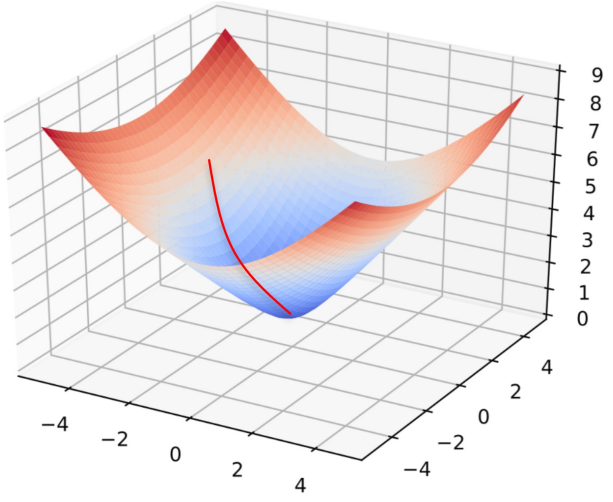


Logistic regression

gradient descent

Algorithm 1 Gradient Descent

1: **procedure** GD($\mathcal{D}$, $\boldsymbol{\theta}^{(0)}$)
2: $\quad\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^{(0)}$
3: $\quad$**while** not converged **do**
4: $\quad\quad\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \gamma \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$
5: $\quad$**return** $\boldsymbol{\theta}$

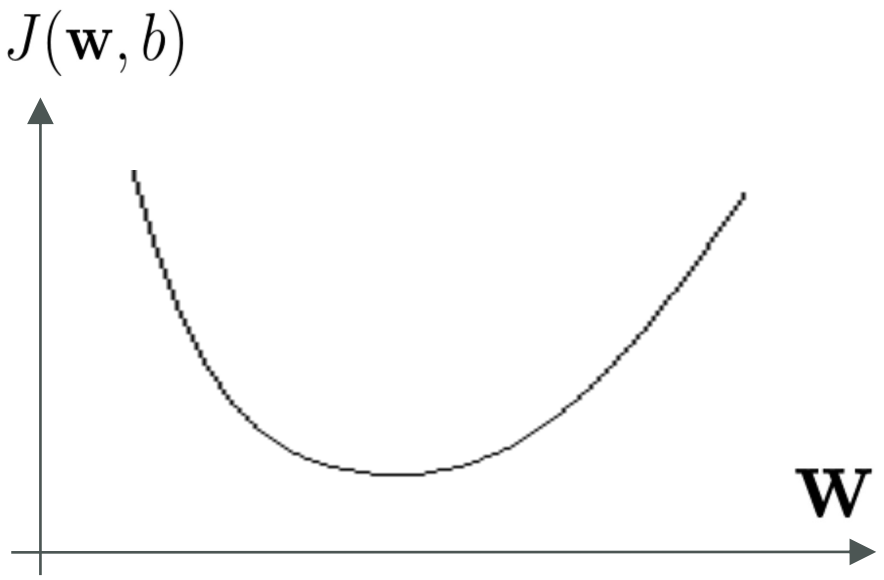


https://arxiv.org/pdf/1609.04747.pdf

Logistic regression

want to find w, b that minimize $J(w, b)$

$$J(\mathbf{w}, b) = \frac{1}{m}\sum_{i=1}^{m} L(\hat{y}^{(i)}, y^{(i)}) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)} \log \hat{y}^{(i)} + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)})\right]$$



learning rate

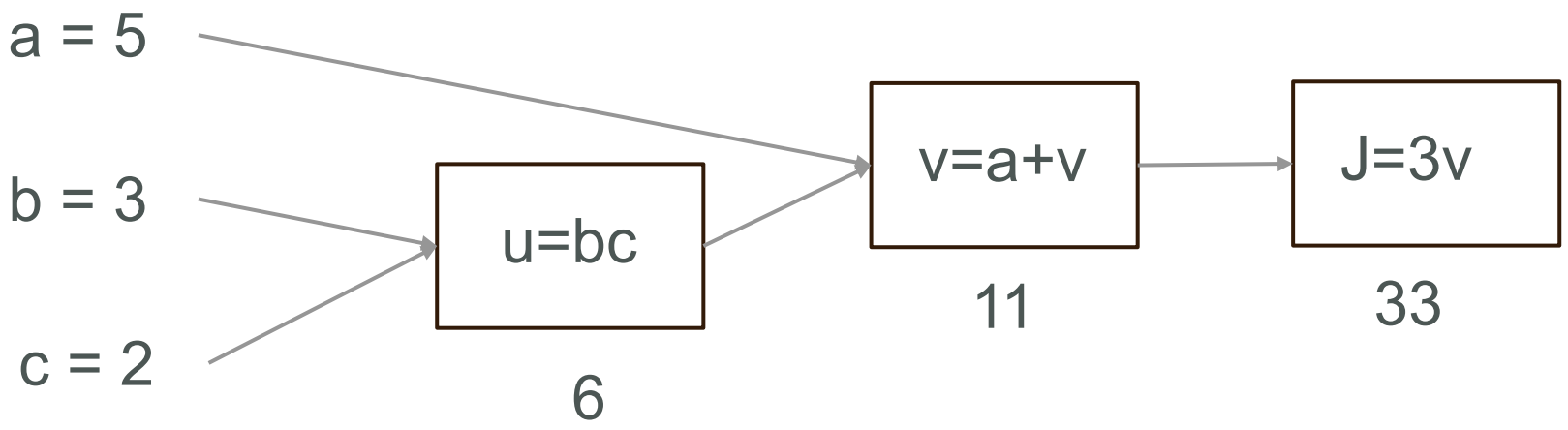
$$\mathbf{w} \leftarrow \mathbf{w} - \alpha \frac{\partial J}{\partial \mathbf{w}}$$

$$b \leftarrow b - \alpha \frac{\partial J}{\partial b}$$

Logistic regression

Computational graph

$$J(a,b,c) = 3(a+bc) = 3*(5+3*2) = 33$$

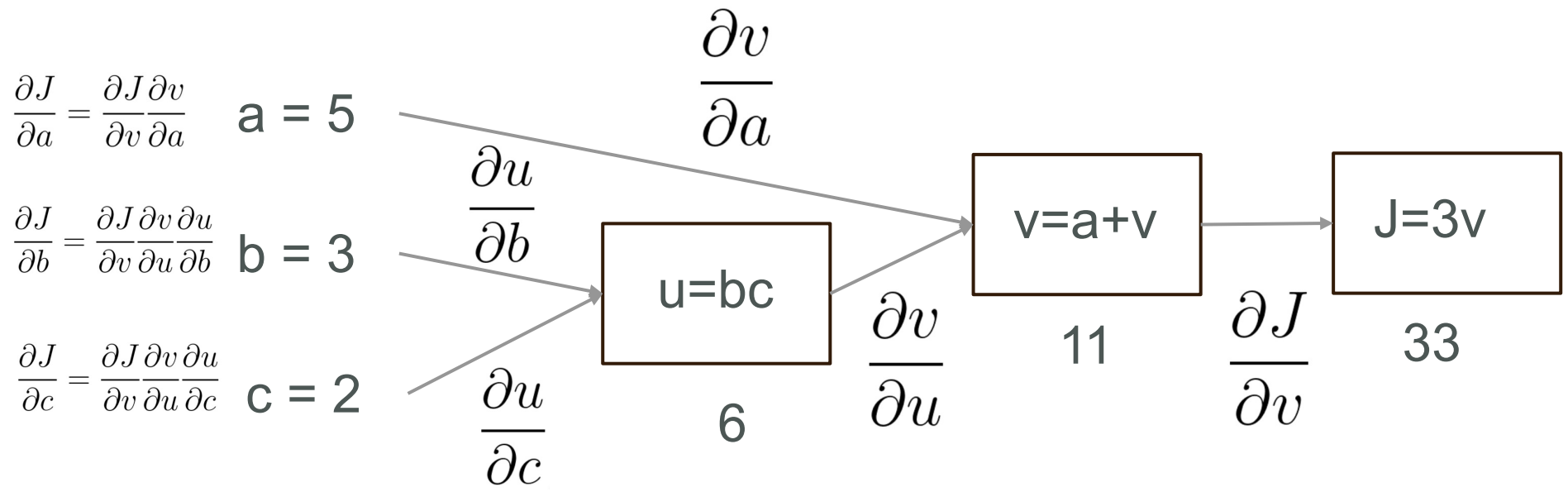


Logistic regression

Computing **derivatives**

$$J(a,b,c) = 3(a+bc) = 3*(5+3*2) = 33$$

$\frac{\partial J}{\partial a}$ How would a's change effect J?



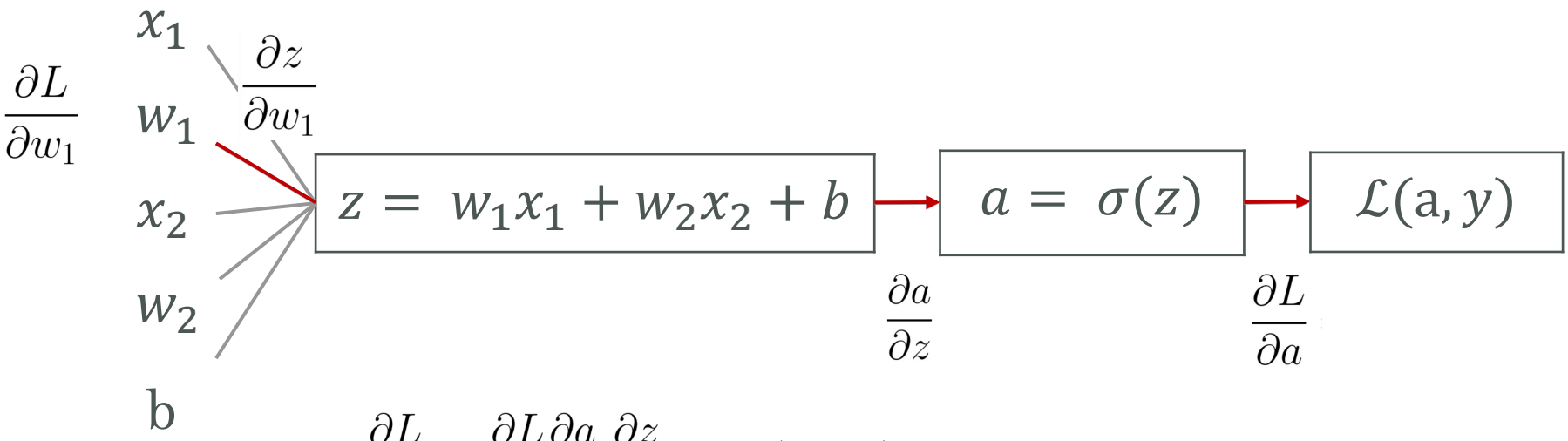
Logistic regression

A single sample

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

$$\hat{y} = a = \sigma(z)$$

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



$$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial a}\frac{\partial a}{\partial z}\frac{\partial z}{\partial w_1} = x_1(a - y)$$

Logistic regression

m training samples

$$J(\mathbf{w}, b) = \frac{1}{m}\sum_{i=1}^{m} L(a^{(i)}, y^{(i)}) \quad a^{(i)} = \hat{y}^{(i)} = \sigma(z^{(i)}) = \sigma(\mathbf{w}^T\mathbf{x}^{(i)} + b)$$

Partial derivative computation

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

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

update

$$\mathbf{w} \leftarrow \mathbf{w} - \alpha\frac{\partial J}{\partial \mathbf{w}} \qquad b \leftarrow b - \alpha\frac{\partial J}{\partial b}$$

Logistic regression

Vectorization

$$z^{(1)} = w^T x^{(1)} + b \qquad z^{(2)} = w^T x^{(2)} + b \qquad z^{(3)} = w^T x^{(3)} + b$$

$$a^{(1)} = \sigma(z^{(1)}) \qquad a^{(2)} = \sigma(z^{(2)}) \qquad a^{(3)} = \sigma(z^{(3)})$$

$$\mathbf{z} = \begin{bmatrix} z^{(1)} \\ z^{(2)} \\ \vdots \\ z^{(m)} \end{bmatrix} = \begin{bmatrix} \mathbf{w}^T\mathbf{x}^{(1)} + b \\ \mathbf{w}^T\mathbf{x}^{(2)} + b \\ \vdots \\ \mathbf{w}^T\mathbf{x}^{(m)} + b \end{bmatrix}$$

data matrix: $\mathbf{X} = [\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \cdots, \mathbf{x}^{(m)}] \in \mathbb{R}^{d \times m}$

```
z = np.dot(X.T, w) + b
a = 1 / (1 + np.exp(-x))
```

Logistic regression

Partial derivative computation

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

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

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

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

vectorization

$$\frac{1}{m}\sum_{i=1}^{m}(a^{(i)} - y^{(i)})\mathbf{x}^{(i)} = \frac{1}{m}\left[\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \cdots \mathbf{x}^{(m)}\right]\begin{bmatrix} a^{(1)} - y^{(1)} \\ a^{(2)} - y^{(2)} \\ \vdots \\ a^{(m)} - y^{(m)} \end{bmatrix}$$

Logistic regression

update

$$\mathbf{w} \leftarrow \mathbf{w} - \alpha \frac{\partial J}{\partial \mathbf{w}} \qquad b \leftarrow b - \alpha \frac{\partial J}{\partial b}$$

vectorization

$$\mathbf{w} \leftarrow \mathbf{w} - \frac{\alpha}{m}\mathbf{X}(\mathbf{a} - \mathbf{y}) \qquad b \leftarrow b - \frac{\alpha}{m}\mathbf{1}_m^T(\mathbf{a} - \mathbf{y})$$

Logistic regression

data matrix: $\mathbf{X} = [\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \cdots, \mathbf{x}^{(m)}] \in \mathbb{R}^{d \times m}$

label matrix: $\mathbf{Y} = [y^{(1)}, y^{(2)}, \cdots, y^{(m)}] \in \mathbb{R}^{1 \times m}$

model: $\widehat{y} = \sigma(\mathbf{w}^T \mathbf{x} + b)$

loss function (cross entropy): $L(\widehat{y}, y) = -(y \log \widehat{y} + (1 - y) \log(1 - \widehat{y}))$

minimize cost function: $J(\mathbf{w}, b) = \frac{1}{m} \sum_{i=1}^{m} L(\widehat{y}^{(i)}, y^{(i)})$

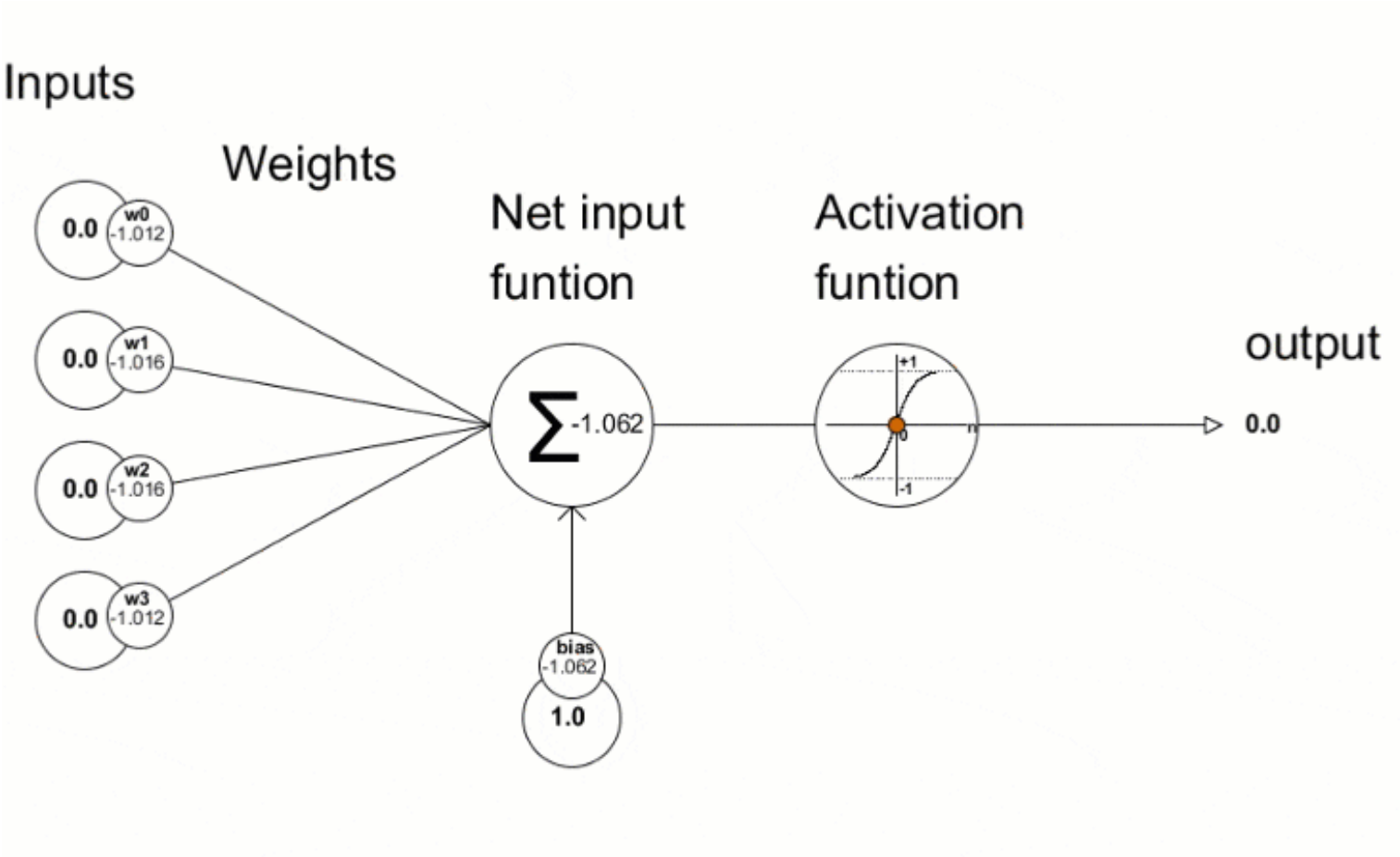
iterative solver:

$$\mathbf{w} \leftarrow \mathbf{w} - \frac{\alpha}{m} \mathbf{X}(\mathbf{a} - \mathbf{y})$$

$$b \leftarrow b - \frac{\alpha}{m} \mathbf{1}_m^T (\mathbf{a} - \mathbf{y})$$

Logistic regression

Training



Logistic regression

data matrix: $\mathbf{X} = [\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \cdots, \mathbf{x}^{(m)}] \in \mathbb{R}^{d \times m}$

label matrix: $\mathbf{Y} = [y^{(1)}, y^{(2)}, \cdots, y^{(m)}] \in \mathbb{R}^{1 \times m}$

model: $\hat{y} = \sigma(\mathbf{w}^T \mathbf{x} + b)$

$\frac{e^{z_k}}{\sum_j e^{z_j}}$

loss function (cross entropy): $L(\hat{y}, y) = -(y \log \hat{y} + (1 - y) \log(1 - \hat{y}))$

$-\sum_{c=1}^{C} y_c \log \hat{y}_c$

minimize cost function: $J(\mathbf{w}, b) = \frac{1}{m} \sum_{i=1}^{m} L(\hat{y}^{(i)}, y^{(i)})$

iterative solver:

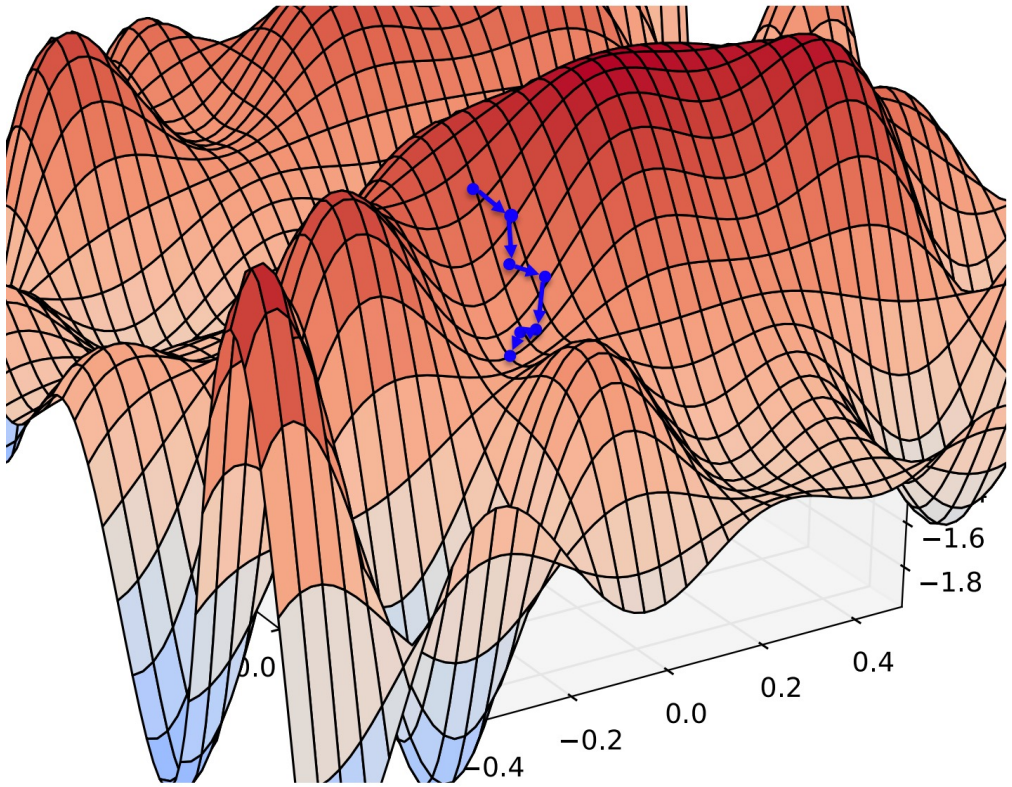
$$\mathbf{w} \leftarrow \mathbf{w} - \frac{\alpha}{m} \mathbf{X}(\mathbf{a} - \mathbf{y})$$

$$b \leftarrow b - \frac{\alpha}{m} \mathbf{1}_m^T (\mathbf{a} - \mathbf{y})$$

Binary classification → multi-class classification

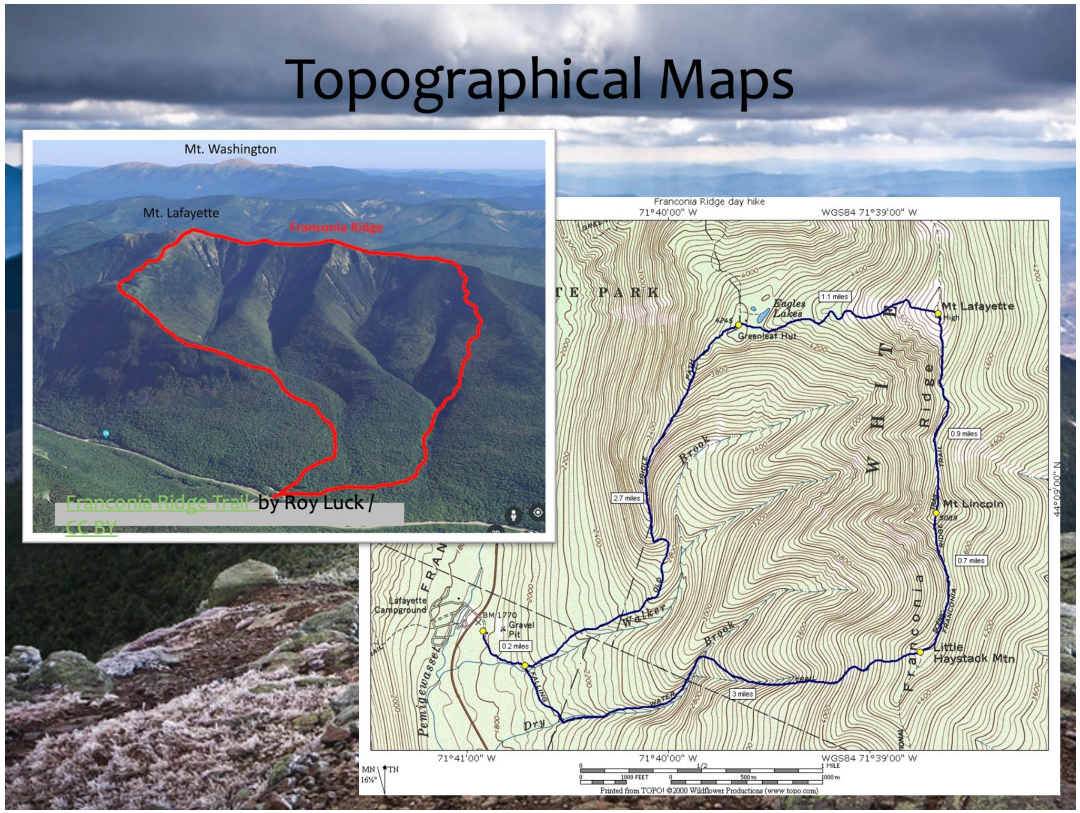
More about optimization

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



More about optimization

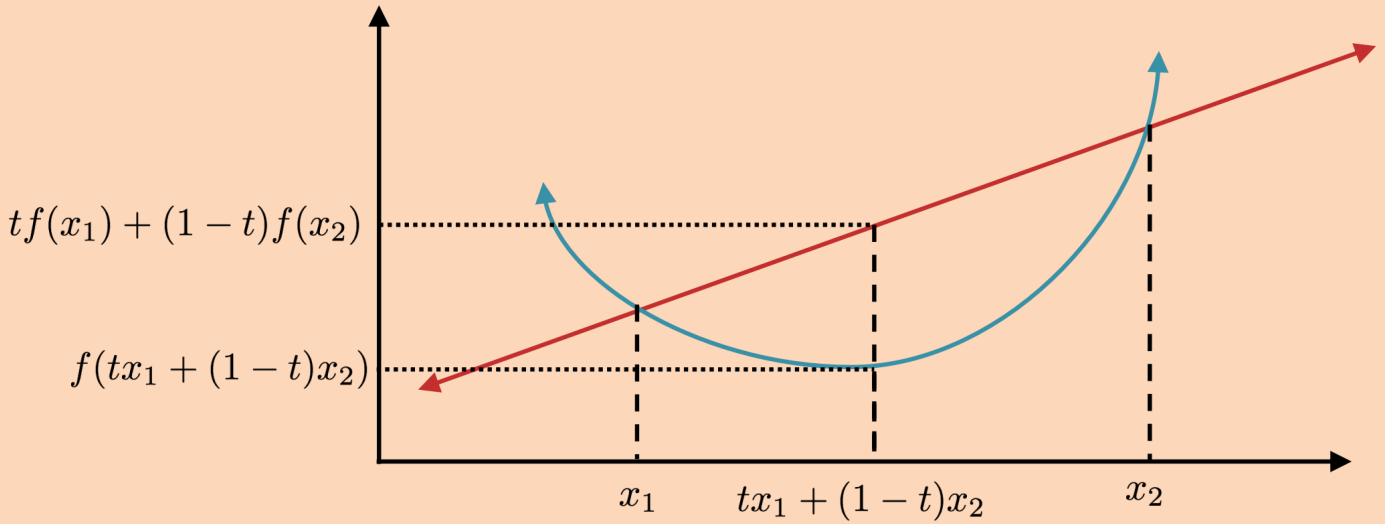
$$\text{minimize} \quad J(\mathbf{w}, b) = \frac{1}{m} \sum_{i=1}^{m} L(\hat{y}^{(i)}, y^{(i)})$$



More about optimization

Function $f : \mathbb{R}^M \rightarrow \mathbb{R}$ is **convex**
if $\forall\ \mathbf{x}_1 \in \mathbb{R}^M, \mathbf{x}_2 \in \mathbb{R}^M, 0 \leq t \leq 1$:

$$f(t\mathbf{x}_1 + (1-t)\mathbf{x}_2) \leq tf(\mathbf{x}_1) + (1-t)f(\mathbf{x}_2)$$



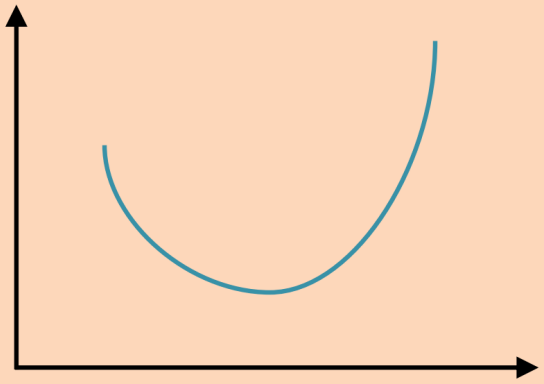
http://www.cs.cmu.edu/~mgormley/courses/10601/slides/lecture8-opt.pdf

More about optimization

Suppose we have a function $f(x) : \mathcal{X} \to \mathcal{Y}$.

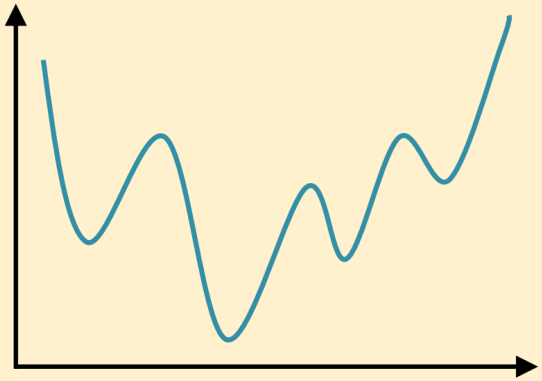
- The value x^* is a **global minimum** of f iff $f(x^*) \leq f(x), \forall x \in \mathcal{X}$.
- The value x^* is a **local minimum** of f iff $\exists \epsilon$ s.t. $f(x^*) \leq f(x), \forall x \in [x^* - \epsilon, x^* + \epsilon]$.

Convex Function



- Each **local minimum** is a **global minimum**

Nonconvex Function

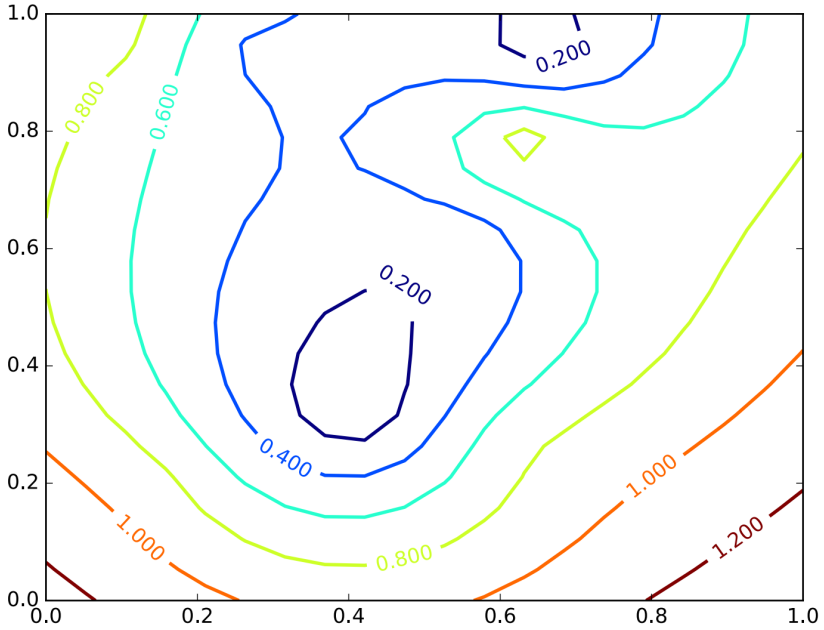


- A *nonconvex* function is **not convex**
- Each **local minimum** is **not** necessarily a **global minimum**

More about optimization

To solve an optimization problem

- Closed-form solution (PCA)
- Iterative solver (Logistic regression)
  - Gradient descent

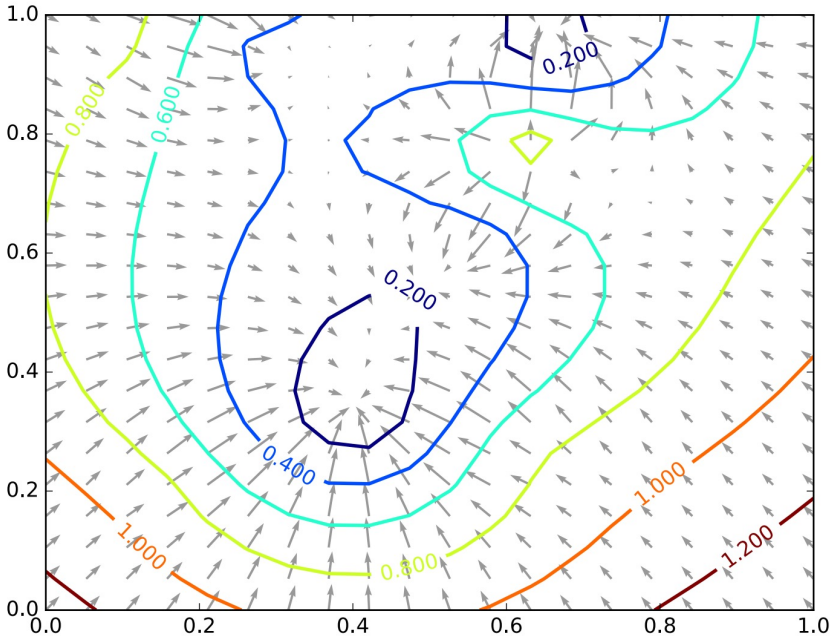


http://www.cs.cmu.edu/~mgormley/courses/10601/slides/lecture8-opt.pdf

More about optimization

To solve an optimization problem

- Closed-form solution (PCA)
- Iterative solver (Logistic regression)
  - Gradient descent

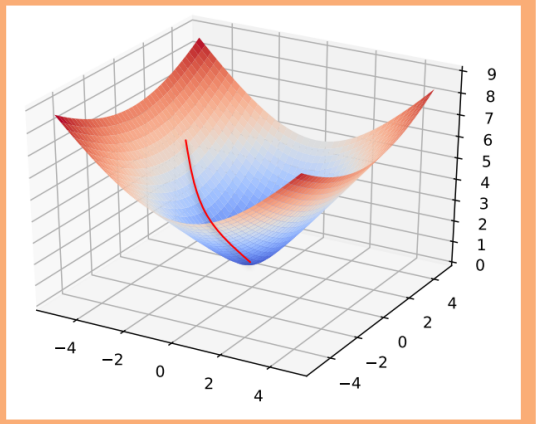


http://www.cs.cmu.edu/~mgormley/courses/10601/slides/lecture8-opt.pdf

More about optimization

Algorithm 1 Gradient Descent

1: **procedure** GD($\mathcal{D}$, $\boldsymbol{\theta}^{(0)}$)
2: $\quad\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^{(0)}$
3: $\quad$**while** not converged **do**
4: $\quad\quad\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \gamma \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$
5: $\quad$**return** $\boldsymbol{\theta}$



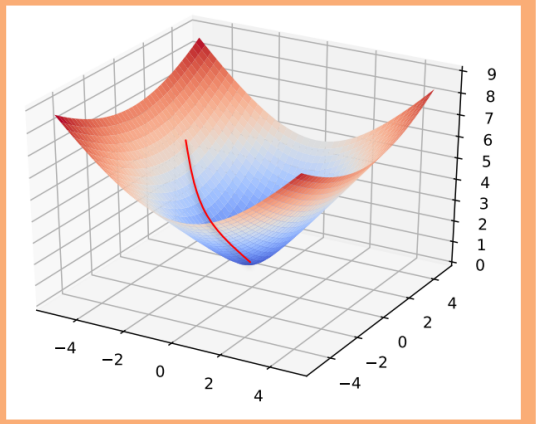
In order to apply GD to Linear Regression all we need is the **gradient** of the objective function (i.e. vector of partial derivatives).

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \begin{bmatrix} \frac{d}{d\theta_1} J(\boldsymbol{\theta}) \\ \frac{d}{d\theta_2} J(\boldsymbol{\theta}) \\ \vdots \\ \frac{d}{d\theta_M} J(\boldsymbol{\theta}) \end{bmatrix}$$

More about optimization

Algorithm 1 Gradient Descent

1: **procedure** GD($\mathcal{D}$, $\boldsymbol{\theta}^{(0)}$)
2: $\quad\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^{(0)}$
3: $\quad$**while** not converged **do**
4: $\quad\quad\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \gamma \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$
5: $\quad$**return** $\boldsymbol{\theta}$



There are many possible ways to detect **convergence**. For example, we could check whether the L2 norm of the gradient is below some small tolerance.

$$||\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})||_2 \leq \epsilon$$

Alternatively we could check that the reduction in the objective function from one iteration to the next is small.

More about optimization

Linear regression

1. Assume $\mathcal{D}$ generated as:

$$\mathbf{x}^{(i)} \sim p^*(\cdot)$$
$$y^{(i)} = h^*(\mathbf{x}^{(i)})$$

2. Choose hypothesis space, $\mathcal{H}$:
all linear functions in M-dimensional space

$$\mathcal{H} = \{h_{\boldsymbol{\theta}} : h_{\boldsymbol{\theta}}(\mathbf{x}) = \boldsymbol{\theta}^T\mathbf{x}, \boldsymbol{\theta} \in \mathbb{R}^M\}$$

3. Choose an objective function:
mean squared error (MSE)

$$\begin{aligned} J(\boldsymbol{\theta}) &= \frac{1}{N}\sum_{i=1}^{N} e_i^2 \\ &= \frac{1}{N}\sum_{i=1}^{N}\left(y^{(i)} - h_{\boldsymbol{\theta}}(\mathbf{x}^{(i)})\right)^2 \\ &= \frac{1}{N}\sum_{i=1}^{N}\left(y^{(i)} - \boldsymbol{\theta}^T\mathbf{x}^{(i)})\right)^2 \end{aligned}$$

4. Solve the unconstrained optimization problem via favorite method:

- *gradient descent*
- *closed form*
- *stochastic gradient descent*
- ...

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, J(\boldsymbol{\theta})$$

5. Test time: given a new $\mathbf{x}$, make prediction $\hat{y}$

$$\hat{y} = h_{\hat{\boldsymbol{\theta}}}(\mathbf{x}) = \hat{\boldsymbol{\theta}}^T\mathbf{x}$$

More about optimization

Linear regression

To solve the Ordinary Least Squares problem we compute:

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}(y^{(i)} - (\boldsymbol{\theta}^T\mathbf{x}^{(i)}))^2$$
$$= (\mathbf{X}^T\mathbf{X})^{-1}(\mathbf{X}^T\mathbf{Y})$$

Closed-form solution

Algorithm 1 GD for Linear Regression

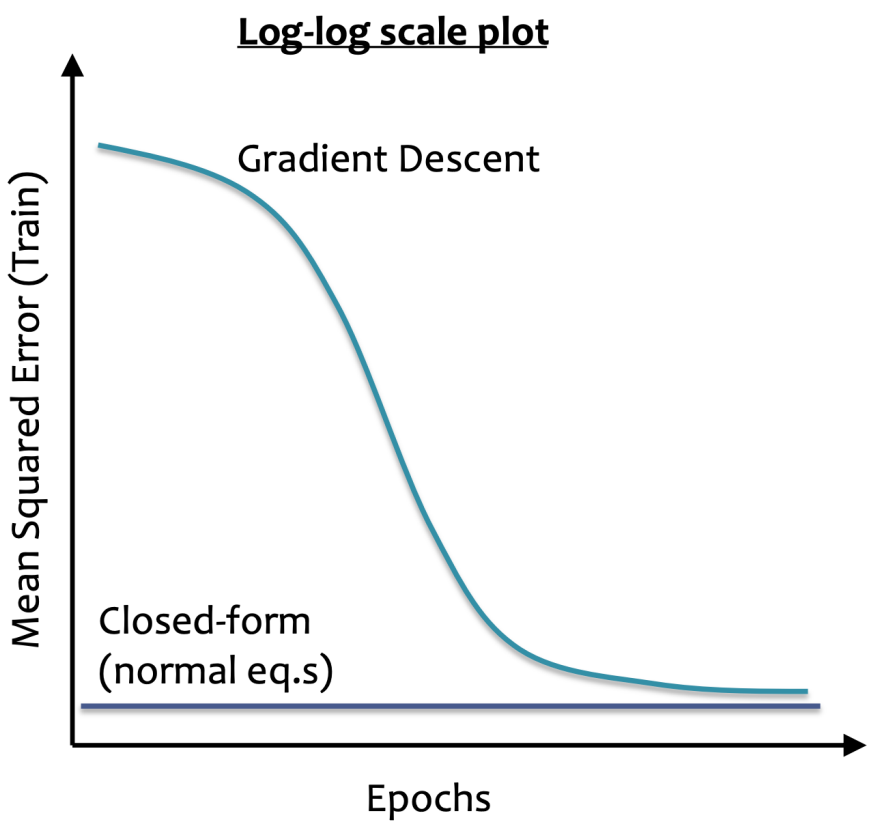
1: **procedure** GDLR($\mathcal{D}$, $\boldsymbol{\theta}^{(0)}$)
2: $\quad\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^{(0)}$ ▷ Initialize parameters
3: $\quad$**while** not converged **do**
4: $\quad\quad\mathbf{g} \leftarrow \sum_{i=1}^{N}(\boldsymbol{\theta}^T\mathbf{x}^{(i)} - y^{(i)})\mathbf{x}^{(i)}$ ▷ Compute gradient
5: $\quad\quad\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \gamma\mathbf{g}$ ▷ Update parameters
6: $\quad$**return** $\boldsymbol{\theta}$

Iterative solver by gradient descent

http://www.cs.cmu.edu/~mgormley/courses/10601/slides/lecture8-opt.pdf

More about optimization

Linear regression



- *Def*: an **epoch** is a single pass through the training data

1. For GD, only **one update** per epoch
2. For SGD, **N updates** per epoch
   N = (# train examples)

- SGD reduces MSE much more rapidly than GD
- For GD / SGD, training MSE is initially large due to uninformed initialization

More about optimization

Linear regression

To solve the Ordinary Least Squares problem we compute:

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}(y^{(i)} - (\boldsymbol{\theta}^T\mathbf{x}^{(i)}))^2$$
$$= (\mathbf{X}^T\mathbf{X})^{-1}(\mathbf{X}^T\mathbf{Y})$$

Closed-form solution

Algorithm 1 GD for Linear Regression

1: **procedure** GDLR($\mathcal{D}$, $\boldsymbol{\theta}^{(0)}$)
2: $\quad\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^{(0)}$ ▷ Initialize parameters
3: $\quad$**while** not converged **do**
4: $\quad\quad\mathbf{g} \leftarrow \sum_{i=1}^{N}(\boldsymbol{\theta}^T\mathbf{x}^{(i)} - y^{(i)})\mathbf{x}^{(i)}$ ▷ Compute gradient
5: $\quad\quad\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \gamma\mathbf{g}$ ▷ Update parameters
6: $\quad$**return** $\boldsymbol{\theta}$

Iterative solver by gradient descent

High computational cost!

determined the number of iterations

Computational Complexity of OLS:

$\mathbf{X}^T\mathbf{X}$	$O(M^2N)$
$(\quad)^{-1}$	$O(M^{2.373})$
$\mathbf{X}^T\mathbf{Y}$	$O(MN)$
$(\quad)^{-1}(\quad)$	$O(M^2)$
total	$O(M^2N + M^{2.373})$

Thank you very much!

sihengc@sjtu.edu.cn