tubi

Understand math behind Deep Learning

Given by Qiang Chen

2018-07-05 Tubi Talent Time(Beijing Office)

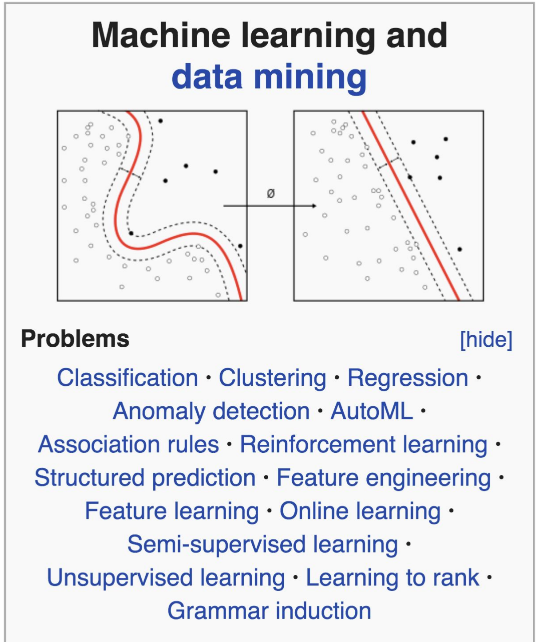
Outline

- Goals, understand and implementation
- What’s deep learning
- ML: Four key parts
- ML: Examples
- ML: Optimization, Derivative and gradient descent
- DL: Chain Rule of Calculus and Back-Propagation
- Learn more about deep learning
- References

Goal

- Know how deep learning works
- Running some code to verify the theory

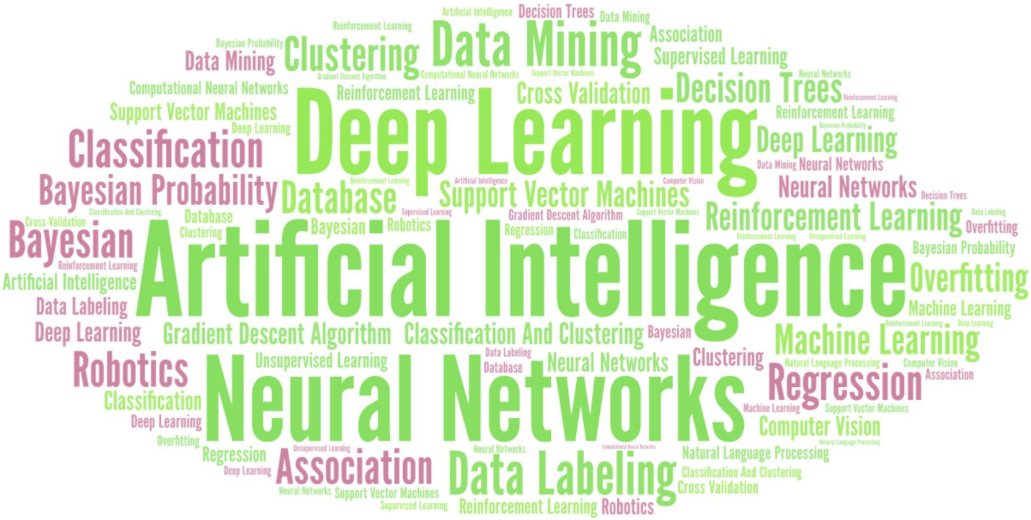
What’s deep learning (classification, regression)



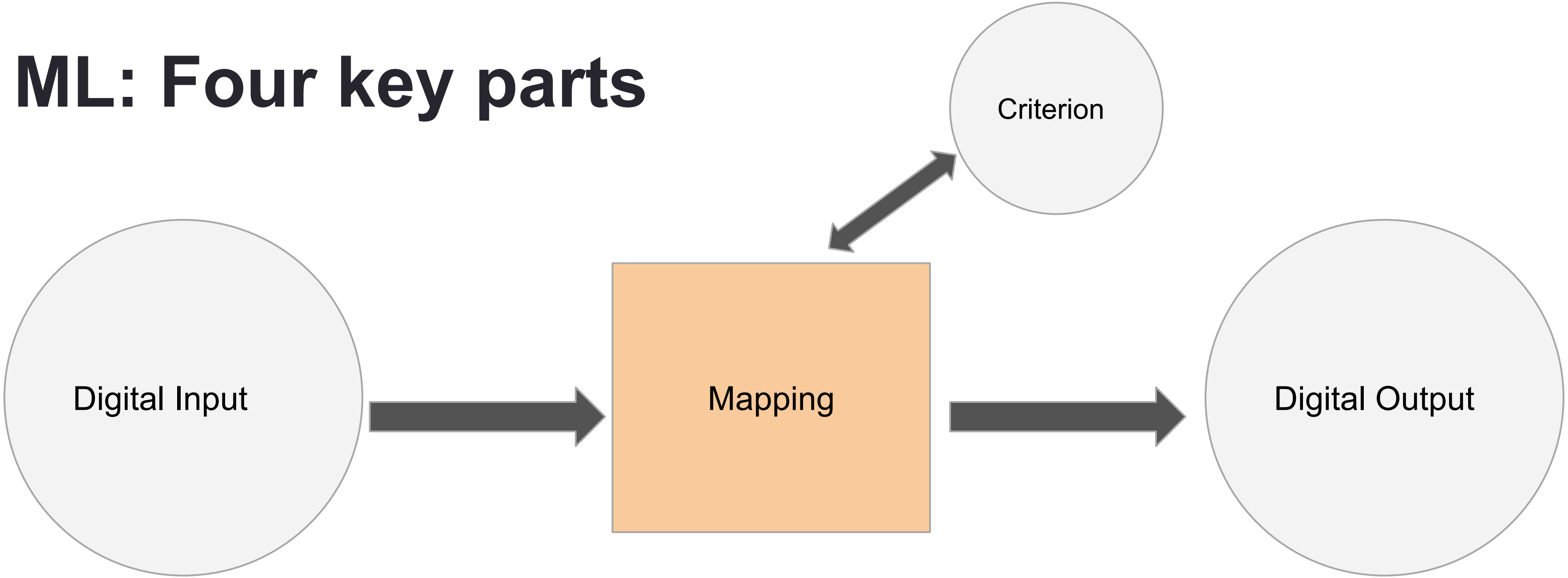
Machine learning and data mining

Problems [hide]

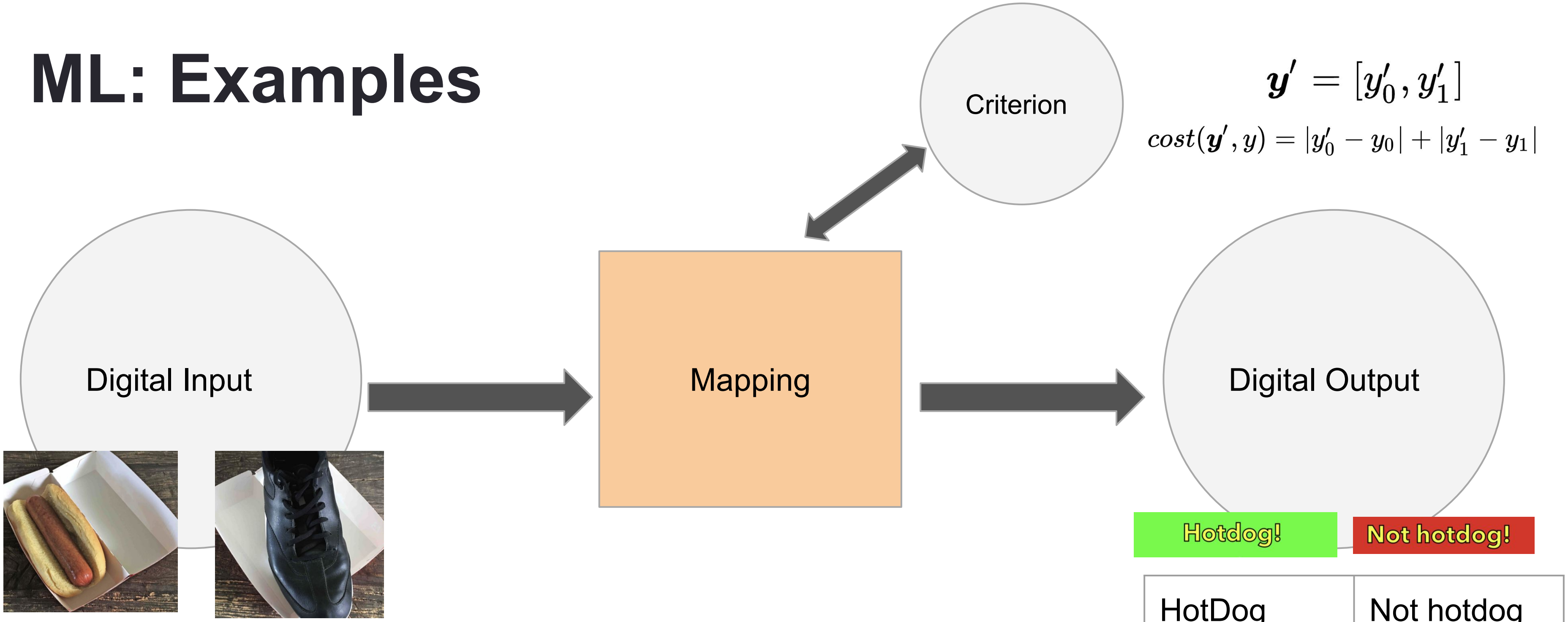
Classification · Clustering · Regression · Anomaly detection · AutoML · Association rules · Reinforcement learning · Structured prediction · Feature engineering · Feature learning · Online learning · Semi-supervised learning · Unsupervised learning · Learning to rank · Grammar induction



ML: Four key parts



ML: Examples

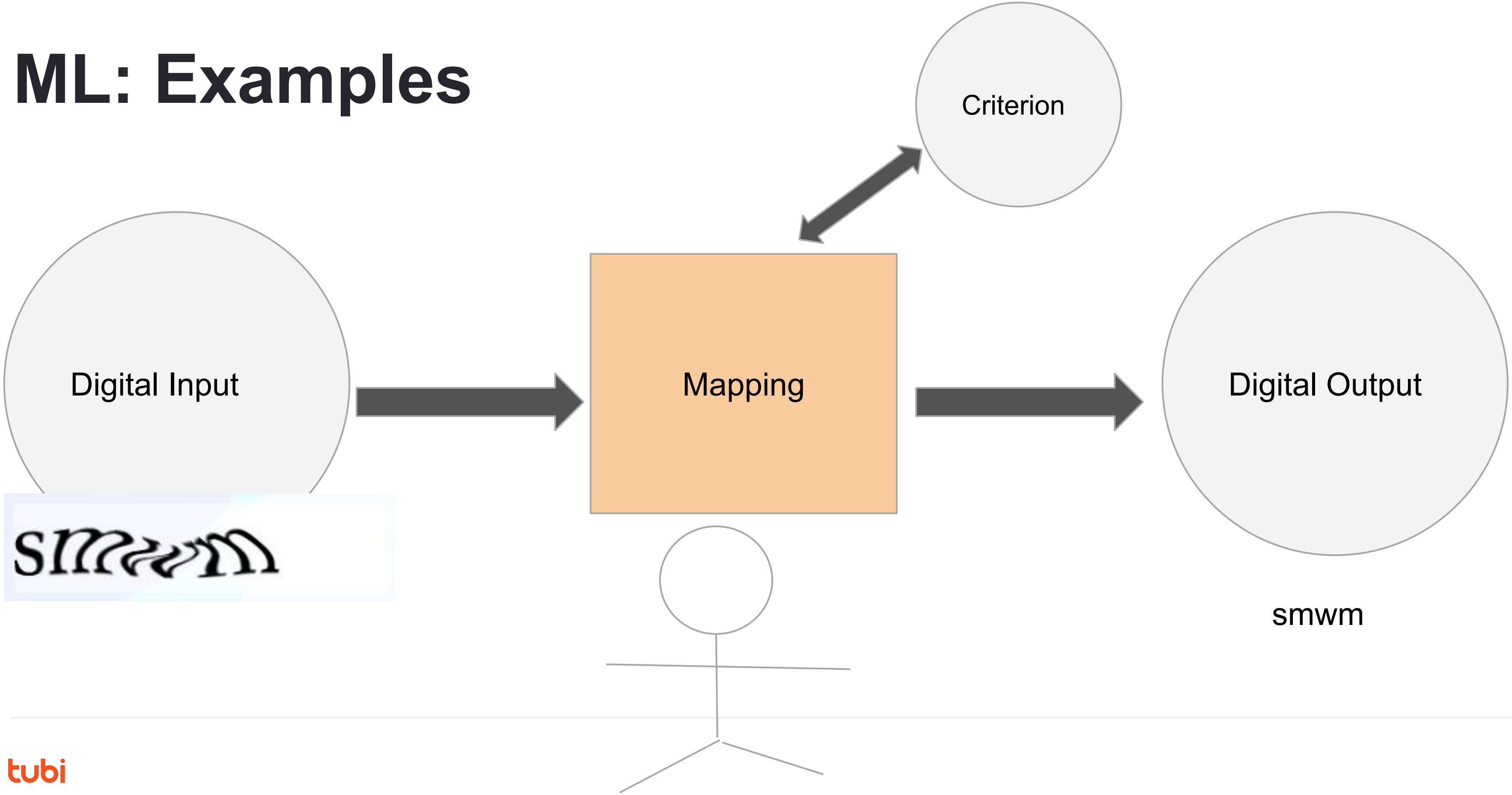


$$\boldsymbol{y}' = [y'_0, y'_1]$$

$$cost(\boldsymbol{y}', y) = |y'_0 - y_0| + |y'_1 - y_1|$$

HotDog	Not hotdog
[0, 1]	[1, 0]

ML: Examples



Criterion

Digital Input

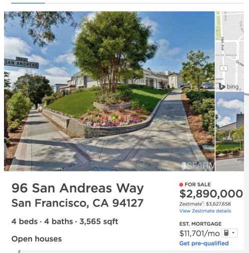
Mapping

Digital Output

smwm

$$cost(\boldsymbol{y}', y) = |y' - y|$$

ML: Examples



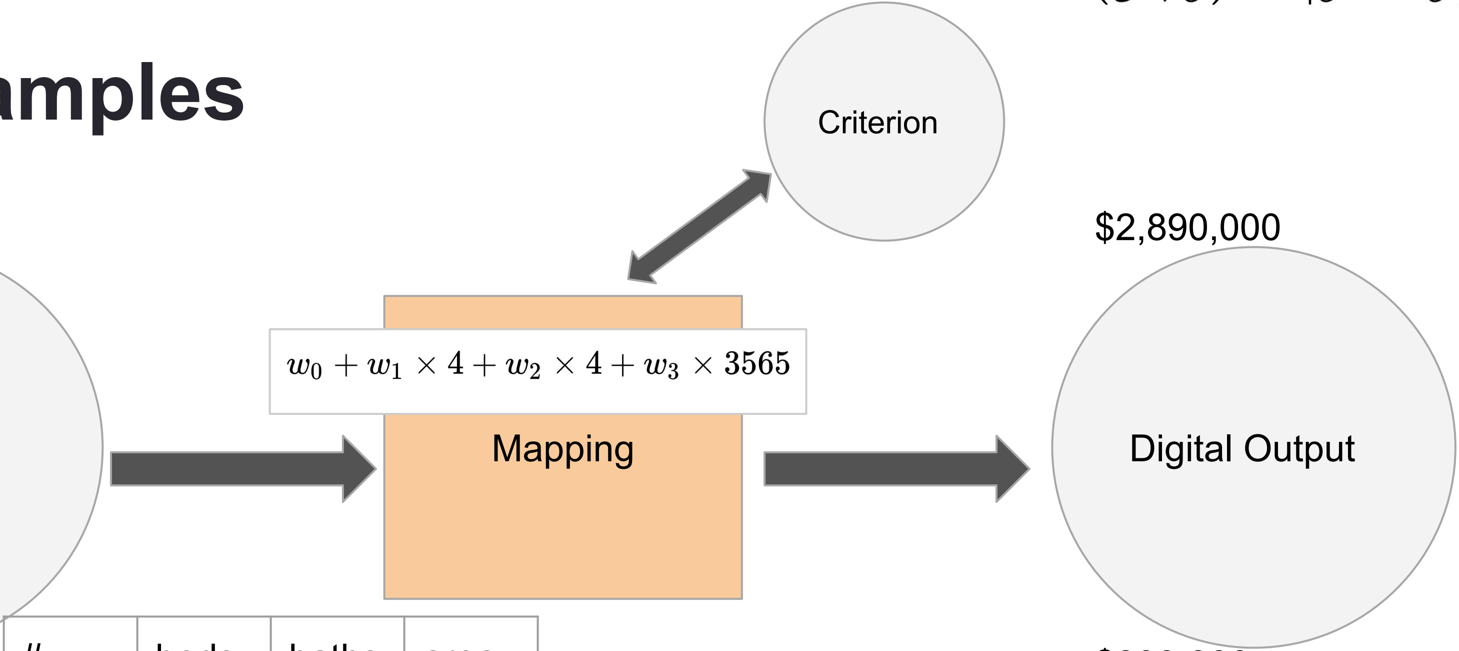
96 San Andreas Way
San Francisco, CA 94127

4 beds · 4 baths · 3,565 sqft

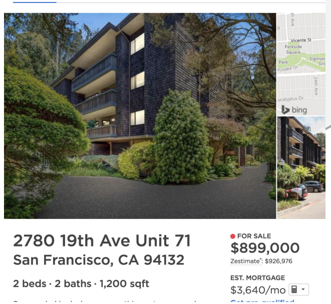
Open houses

FOR SALE
$2,890,000
Zestimate®: $3,627,658
View Zestimate details

EST. MORTGAGE
$11,701/mo
Get pre-qualified



$$w_0 + w_1 \times 4 + w_2 \times 4 + w_3 \times 3565$$



2780 19th Ave Unit 71
San Francisco, CA 94132

2 beds · 2 baths · 1,200 sqft

FOR SALE
$899,000
Zestimate®: $926,976

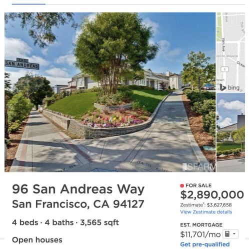
EST. MORTGAGE
$3,640/mo

#	beds	baths	area
#1	4	4	3565
#2	2	2	1200

ML: Optimization, Derivative and gradient descent

$$y' = w_0 + w_1 \times 4 + w_2 \times 4 + w_3 \times 3565$$

$$cost(y', y) = |y' - y| = |y' - 2890| = |w_0 + w_1 \times 4 + w_2 \times 4 + w_3 \times 3565 - 2890|$$



96 San Andreas Way
San Francisco, CA 94127

4 beds · 4 baths · 3,565 sqft

Open houses

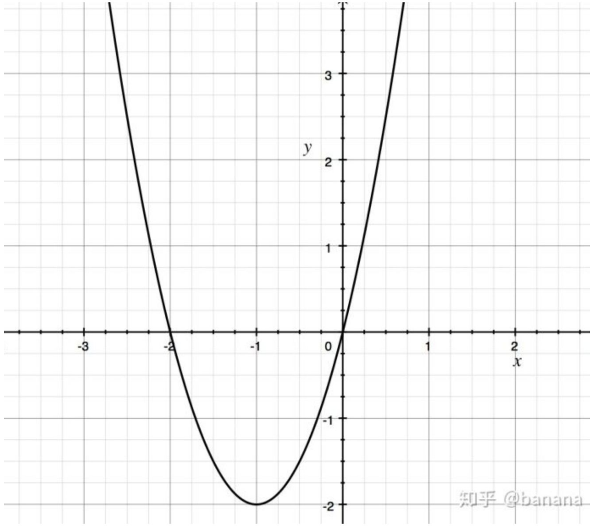
● FOR SALE
$2,890,000
Zestimate®: $3,627,658
View Zestimate details

EST. MORTGAGE
$11,701/mo

Get pre-qualified

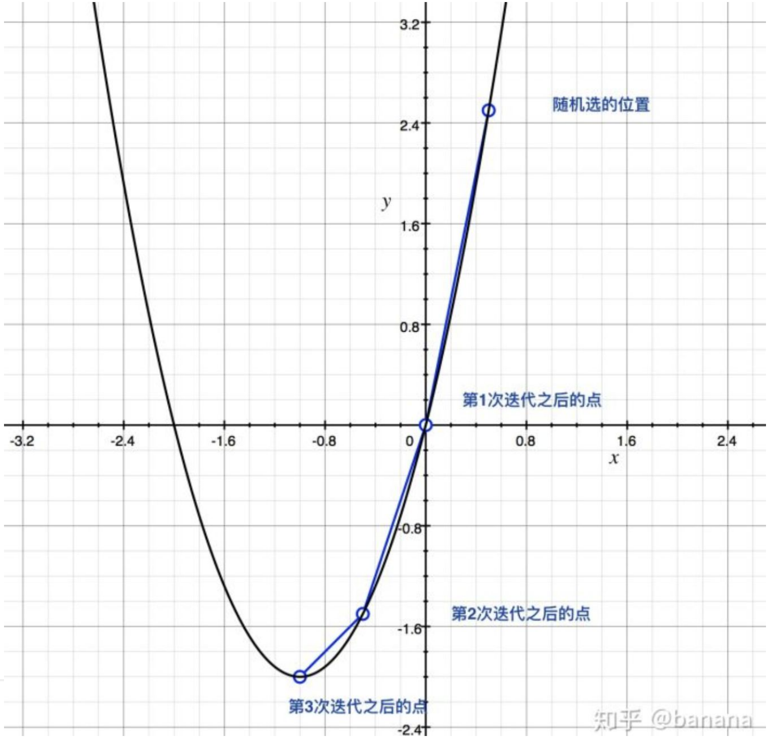
ML: Optimization, Derivative and gradient descent

$$cost = 2 \times w_1^2 + 4 \times w_1$$



ML: Optimization, Derivative and gradient descent

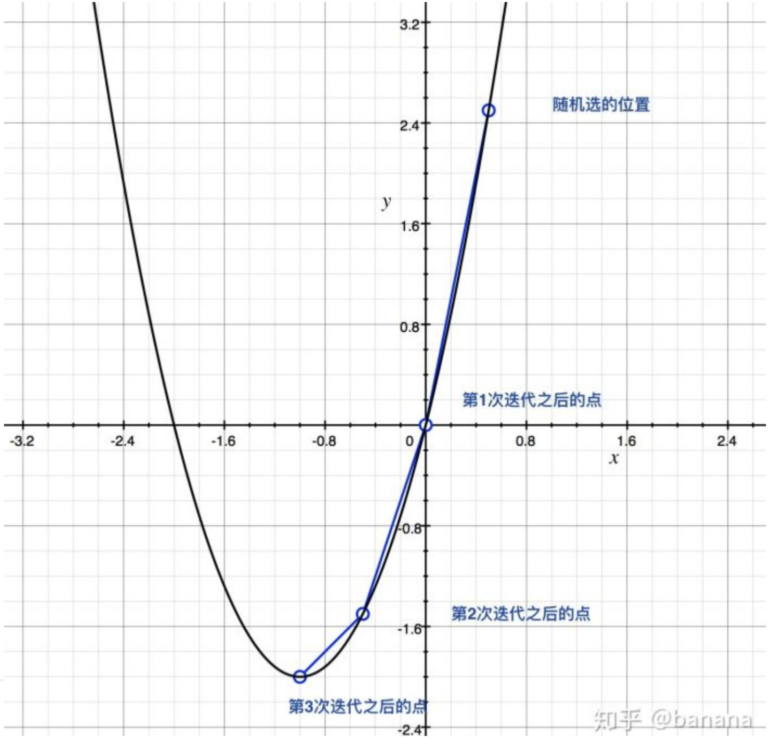
#	w_1	$cost = 2 \times w_1^2 + 4 \times w_1$	$\frac{d(cost)}{d(w_1)} = 4 \times w_1 + 4$
0	0.5	2.5	6
1	0	0	4
2	-0.5	-1.5	2
3	-1	-2	0



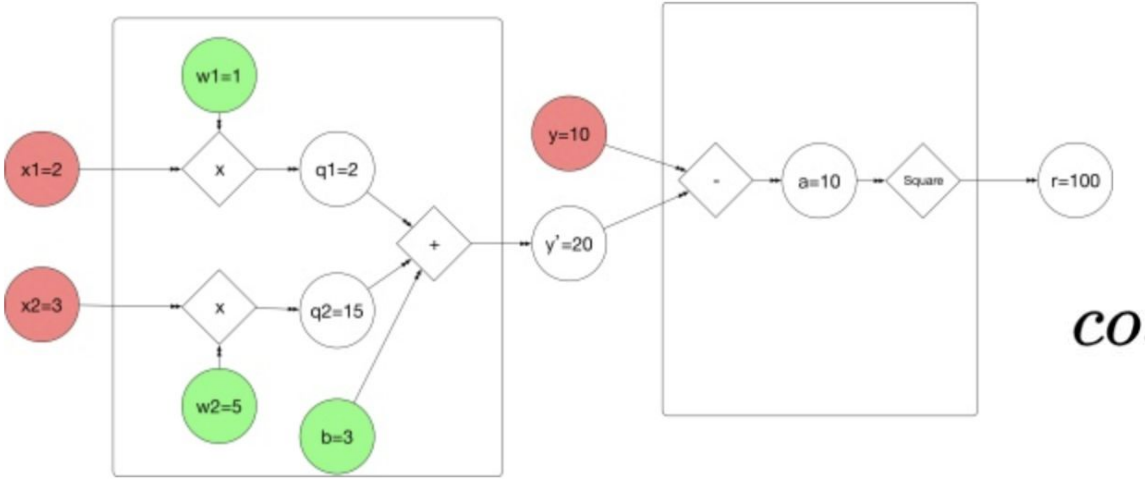
ML: Optimization, Derivative and gradient descent

$$cost = 2 \times w_1^2 + 4 \times w_1$$

$$\frac{d(cost)}{d(w_1)} = 4 \times w_1 + 4$$



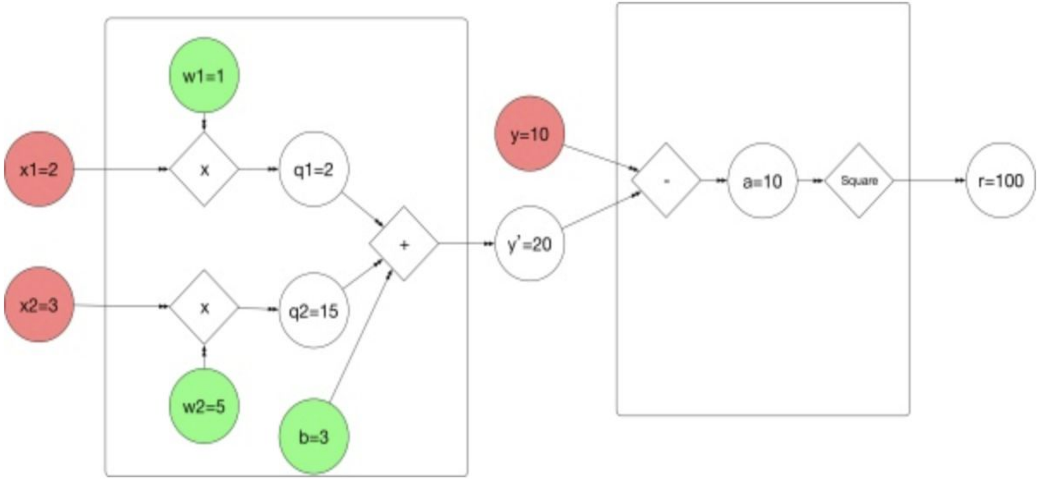
ML: Optimization, implementation

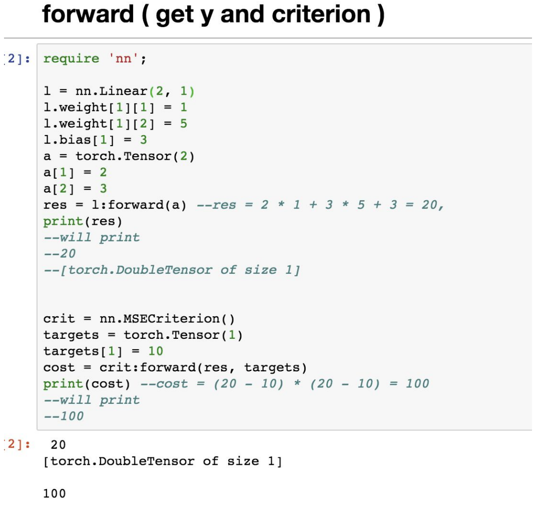


$$cost = (y - y')^2$$

$$y' = w1 \times x1 + w2 \times x2 + b$$

ML: Optimization, implementation





forward (get y and criterion)

```
2]: require 'nn';

    l = nn.Linear(2, 1)
    l.weight[1][1] = 1
    l.weight[1][2] = 5
    l.bias[1] = 3
    a = torch.Tensor(2)
    a[1] = 2
    a[2] = 3
    res = l:forward(a) --res = 2 * 1 + 3 * 5 + 3 = 20,
    print(res)
    --will print
    --20
    --[torch.DoubleTensor of size 1]


    crit = nn.MSECriterion()
    targets = torch.Tensor(1)
    targets[1] = 10
    cost = crit:forward(res, targets)
    print(cost) --cost = (20 - 10) * (20 - 10) = 100
    --will print
    --100
```

```
2]:  20
    [torch.DoubleTensor of size 1]

    100
```

Code: optimization_implementation

ML: Optimization, implementation

$$cost = (y - y')^2$$

$$\frac{d(cost)}{d(y')} = 2(y - y')\frac{d(y - y')}{d(y')} = 2(y - y') \times -1 = 2(y' - y)$$

$$y' = w1 \times x1 + w2 \times x2 + b$$

$$\frac{d(y')}{d(w1)} = x1$$

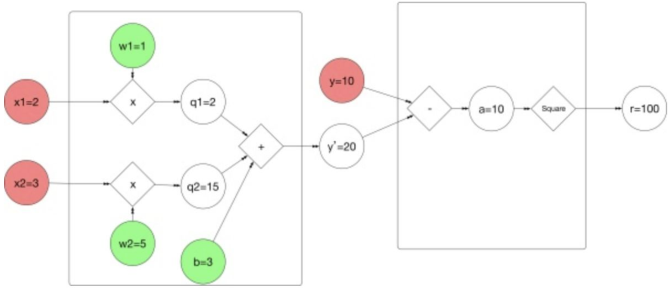
$$\frac{d(y')}{d(w2)} = x2$$

$$\frac{d(y')}{d(b)} = 1$$

$$\frac{d(cost)}{d(w1)} = \frac{d(cost)}{d(y')}\frac{d(y')}{d(w1)} = 2(y' - y) \times x1$$

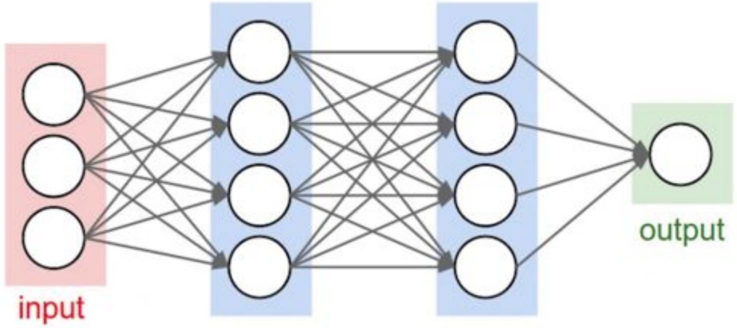
$$\frac{d(cost)}{d(w2)} = \frac{d(cost)}{d(y')}\frac{d(y')}{d(w2)} = 2(y' - y) \times x2$$

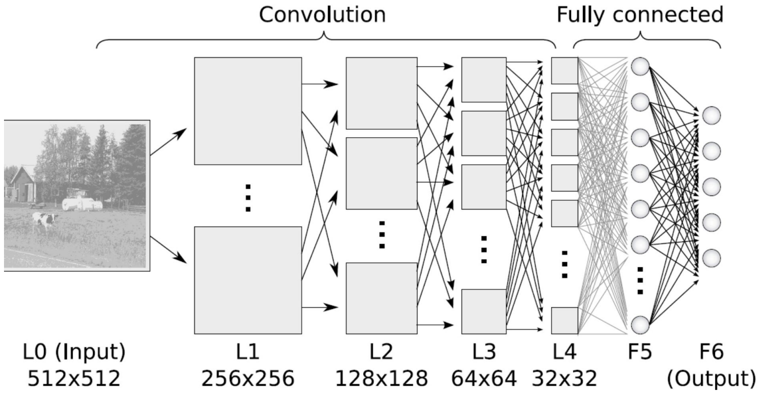
$$\frac{d(cost)}{d(b)} = \frac{d(cost)}{d(y')}\frac{d(y')}{d(b)} = 2(y' - y)$$

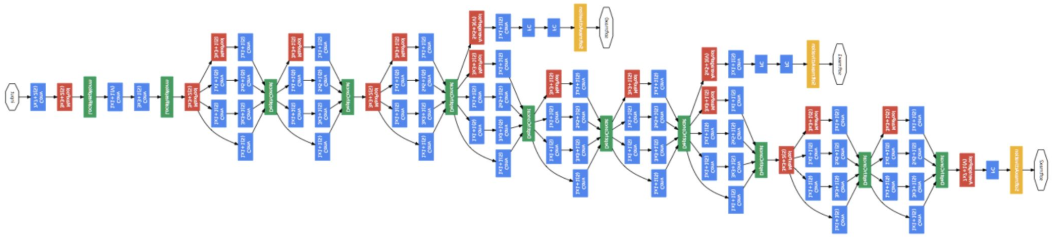


Code: optimization_implementation

DL: Chain Rule of Calculus and Back-Propagation

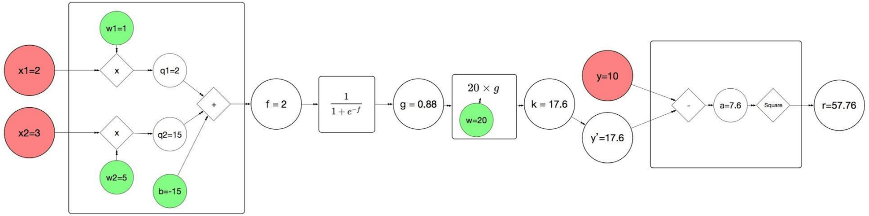






DL: Chain Rule of Calculus and Back-Propagation

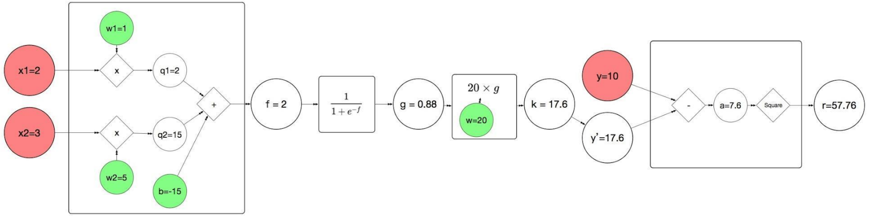
$$\begin{aligned} f' &= f(\boldsymbol{x}, \boldsymbol{w_f}) \\ g' &= g(f') \\ y' &= k(g') \\ cost &= criterion(y, y') \end{aligned}$$



$$\frac{d(cost)}{d(\boldsymbol{w_f})} = \frac{d(f')}{d(\boldsymbol{w_f})} \times \frac{d(g')}{d(f')} \times \frac{d(y')}{d(g')} \times \frac{d(cost)}{y'}$$

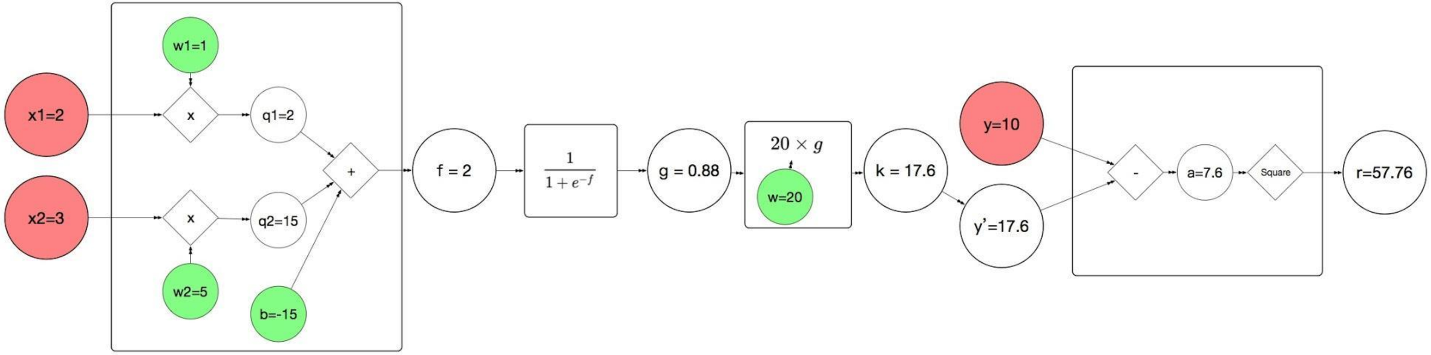
DL: Chain Rule of Calculus and Back-Propagation

$$
\begin{aligned}
f' &= f(\boldsymbol{x}, \boldsymbol{w_f}) \\
g' &= g(f', \boldsymbol{w_g}) \\
y' &= k(g') \\
cost &= criterion(y, y')
\end{aligned}
$$



$$
\frac{d(cost)}{d(\boldsymbol{w_g})} = \frac{d(g')}{d(\boldsymbol{w_g})} \times \frac{d(y')}{d(g')} \times \frac{d(cost)}{y'} 。
$$

DL: Implementation



$$f' = f(\boldsymbol{x}, \boldsymbol{w_f})$$
$$g' = g(f', \boldsymbol{w_g})$$
$$y' = k(g')$$
$$cost = criterion(y, y')$$

$$\boldsymbol{x} = [x1, x2] = [2, 3]$$

$$f' = f(\boldsymbol{x}) = w1 \times x1 + w2 \times x2 + b = 1 \times 2 + 5 \times 3 + -15 = 2$$

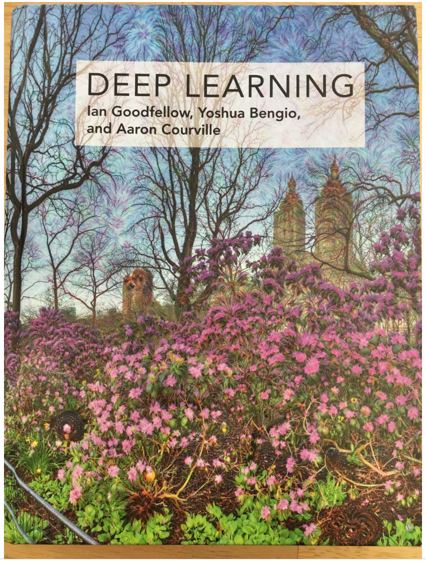
$$g' = g(f') = \frac{1}{1 + e^{-f'}} = \frac{1}{1 + e^{-2}} = 0.8808$$

$$k' = k(g') = w \times g' = 20 \times 0.8808 = 17.6160$$

Code: DL_implementation

Learn more about deep learning

Book



DEEP LEARNING
Ian Goodfellow, Yoshua Bengio,
and Aaron Courville

Online Course

1. CS231n
2. ML taught by Andrew Ng

Implementation

torch

A SCIENTIFIC COMPUTING FRAMEWORK FOR LUAJIT

PyTorch

Supported by facebook

References

- [Video]Lecture 4 | Introduction to Neural Networks, Backpropagation and Neural Networks
- [Slides]Lecture 4: Backpropagation and Neural Networks
- Torch | Developer Documentation, Define your own layer
- 知乎专栏:机器学习与数学

A tiny project: digit recognizer