PyTorch

交叉熵

主讲人：龙良曲

Why not MSE?

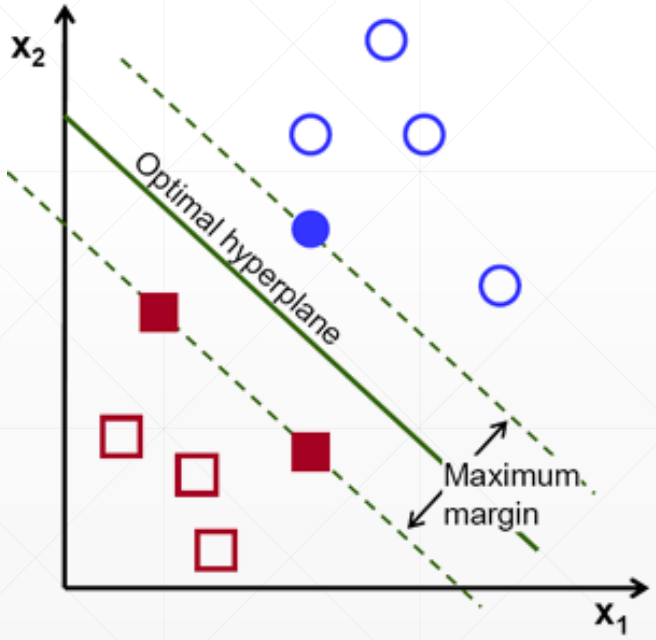
Label	predict	correct
3	[0.3, 0.3, 0.4]	yes
2	[0.3, 0.4, 0.3]	yes
1	[0.1, 0.2, 0.7]	no

Label	predict	correct
3	[0.1, 0.2, 0.7]	yes
2	[0.1, 0.7, 0.2]	yes
1	[0.3, 0.4, 0.3]	no

Loss for classification

- MSE
- Cross Entropy Loss
- Hinge Loss

$$\sum_i max(0, 1 - y_i * h_\theta(x_i))$$



Entropy

- Uncertainty
- measure of surprise

- higher entropy: higher uncertainty.

$$Entropy = -\sum_i P(i) \log P(i)$$

Claude Shannon

https://towardsdatascience.com/demystifying-cross-entropy-e80e3ad54a8

Lottery

```
In [2]: a=torch.full([4],1/4.)
tensor([0.2500, 0.2500, 0.2500, 0.2500])

In [4]: a*torch.log2(a)
Out[4]: tensor([-0.5000, -0.5000, -0.5000, -0.5000])

In [6]: -(a*torch.log2(a)).sum()
Out[6]: tensor(2.)

In [7]: a=torch.tensor([0.1,0.1,0.1,0.7])
In [8]: -(a*torch.log2(a)).sum()
Out[8]: tensor(1.3568)

In [15]: a=torch.tensor([0.001,0.001,0.001,0.999])
In [16]: -(a*torch.log2(a)).sum()
Out[16]: tensor(0.0313)
```

Cross Entropy

$$H(p,q) = -\sum p(x) \log q(x)$$

$$H(p,q) = H(p) + D_{\mathrm{KL}}(p|q).$$

- P=Q
  - cross Entropy = Entropy
- for one-hot encoding,
  - entropy = 1log1=0

Binary Classification

$$H(P,Q) = -P(cat)\log Q(cat) - (1 - P(cat))\log(1 - Q(cat))$$

$$P(dog) = (1 - P(cat))$$

$$H(P,Q) = -\sum_{i=(cat,dog)} P(i)\log Q(i)$$

$$= -P(cat)\log Q(cat) - P(dog)\log Q(dog)$$

$$-(y\log(p) + (1-y)\log(1-p))$$

for example

$$P_1 = [1 \quad 0 \quad 0 \quad 0 \quad 0]$$

$$Q_1 = [0.4 \quad 0.3 \quad 0.05 \quad 0.05 \quad 0.2]$$

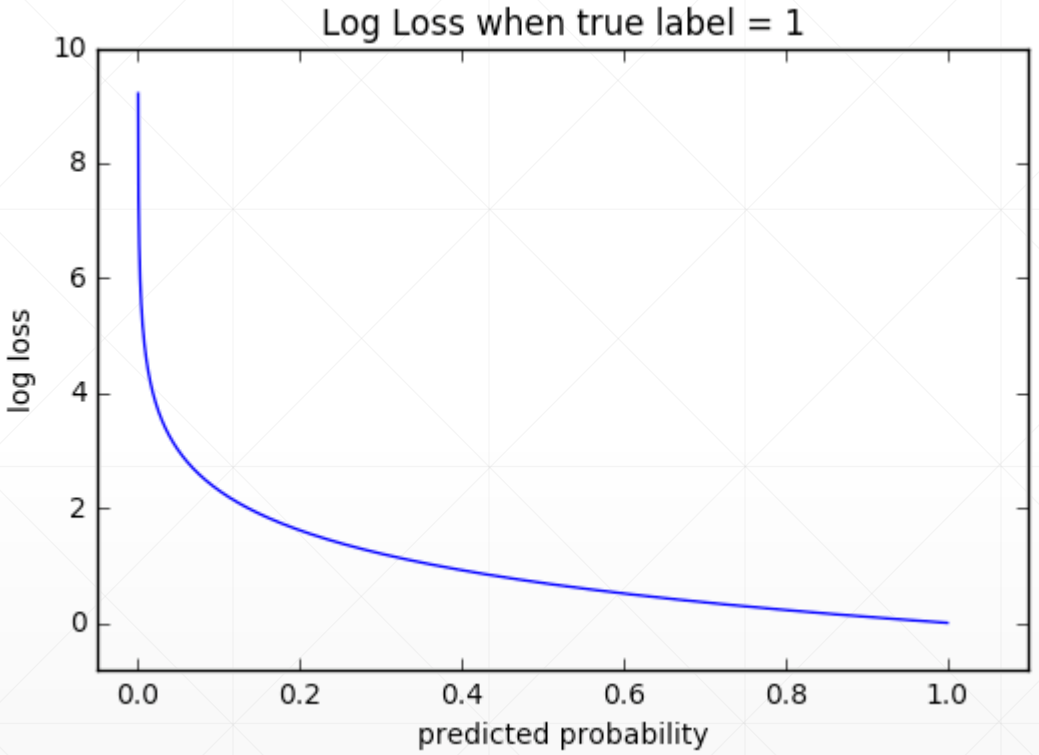
$$\begin{aligned} H(P_1, Q_1) &= -\sum_i P_1(i) \log Q_1(i) \\ &= -(1 \log 0.4 + 0 \log 0.3 + 0 \log 0.05 + 0 \log 0.05 + 0 \log 0.2) \\ &= -\log 0.4 \\ &\approx 0.916 \end{aligned}$$

$$Q_1 = [0.98 \quad 0.01 \quad 0 \quad 0 \quad 0.01]$$

$$\begin{aligned} H(P_1, Q_1) &= -\sum_i P_1(i) \log Q_1(i) \\ &= -(1 \log 0.98 + 0 \log 0.01 + 0 \log 0 + 0 \log 0 + 0 \log 0.01) \\ &= -\log 0.98 \\ &\approx 0.02 \end{aligned}$$

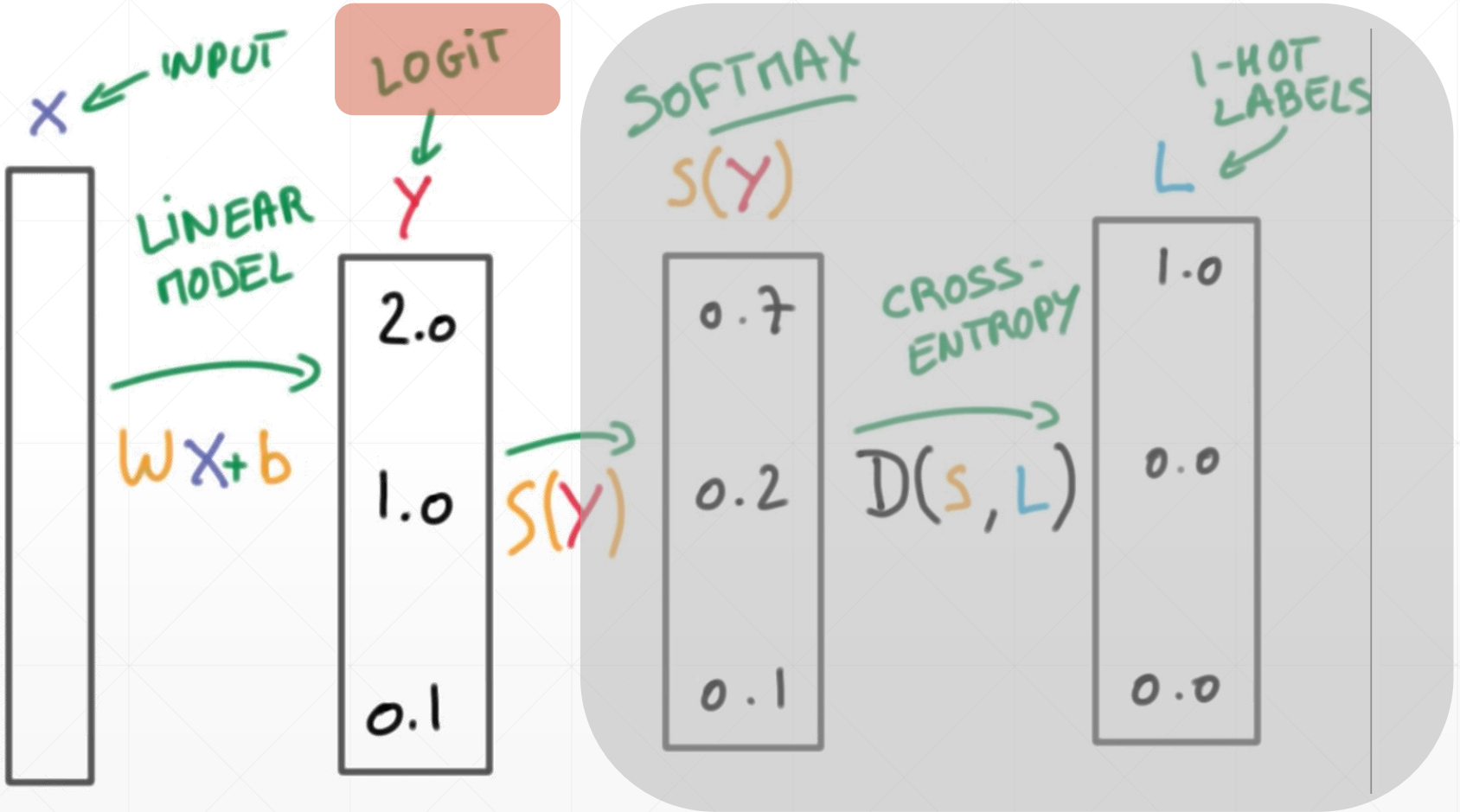
why not use MSE

- sigmoid + MSE
  - gradient vanish
- converge slower
- But, sometimes
  - e.g. meta-learning



https://ml-cheatsheet.readthedocs.io/en/latest/loss_functions.html

Therefore


INPUT
X
LOGIT
Y
LINEAR MODEL
WX+b
2.0
1.0
0.1
S(Y)
SOFTMAX
S(Y)
0.7
0.2
0.1
CROSS-ENTROPY
D(S, L)
1-HOT LABELS
L
1.0
0.0
0.0

Numerical Stability

```
In [17]: x=torch.randn(1,784)
In [18]: w=torch.randn(10,784)

In [19]: logits=x@w.t()
Out[20]: torch.Size([1, 10])

In [34]: pred=F.softmax(logits, dim=1)
Out[37]: torch.Size([1, 10])

In [40]: pred_log=torch.log(pred)

In [41]: F.cross_entropy(logits,torch.tensor([3]))
Out[41]: tensor(82.4905)

In [42]: F.nll_loss(pred_log,torch.tensor([3]))
Out[42]: tensor(82.4905)
```

下一课时

实战多分类问题

Thank You.