EDU

CSDN学院　IT实战派

Python 深度学习

深度学习入门篇：如何做出好模型

讲师： 彭靖田

本章目录

- 机器学习的三大分支
- 模型训练：如何评估一个模型的好与坏
- 模型训练：常用数据预处理方法
- 模型训练：如何解决过拟合问题

EDU

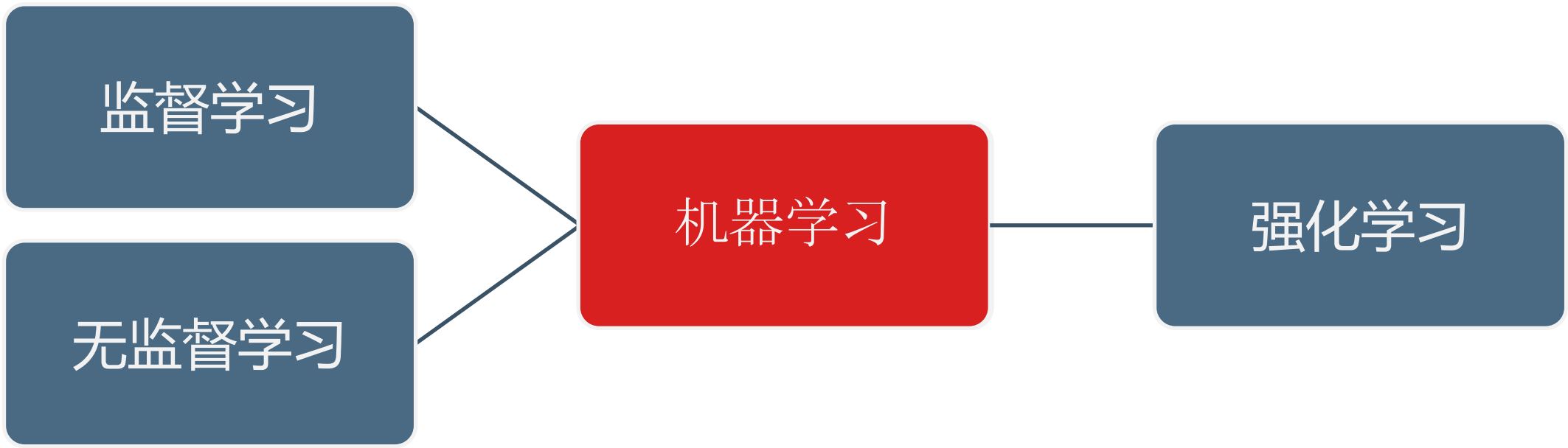
CSDN学院　IT实战派

Python 深度学习

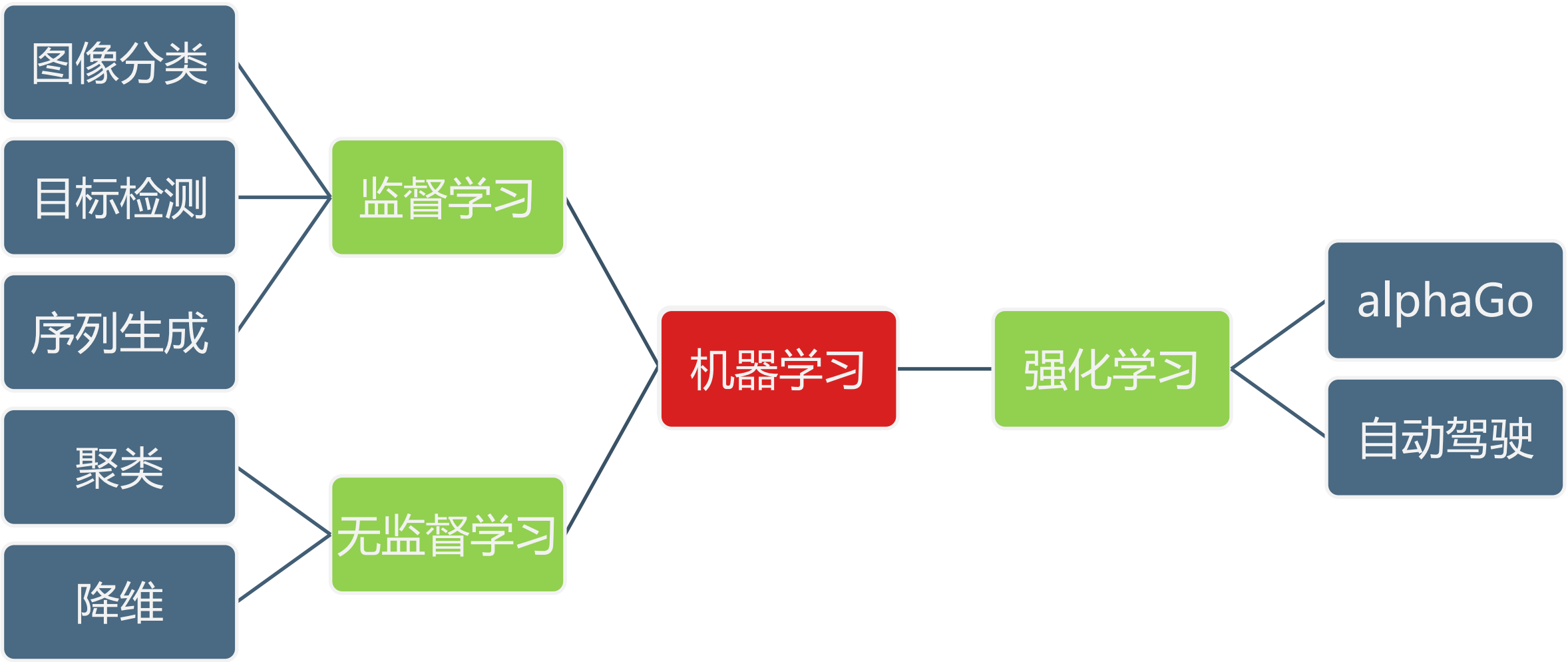
机器学习的三大分支

讲师： 彭靖田

机器学习的三大分支



机器学习的三大分支



监督学习

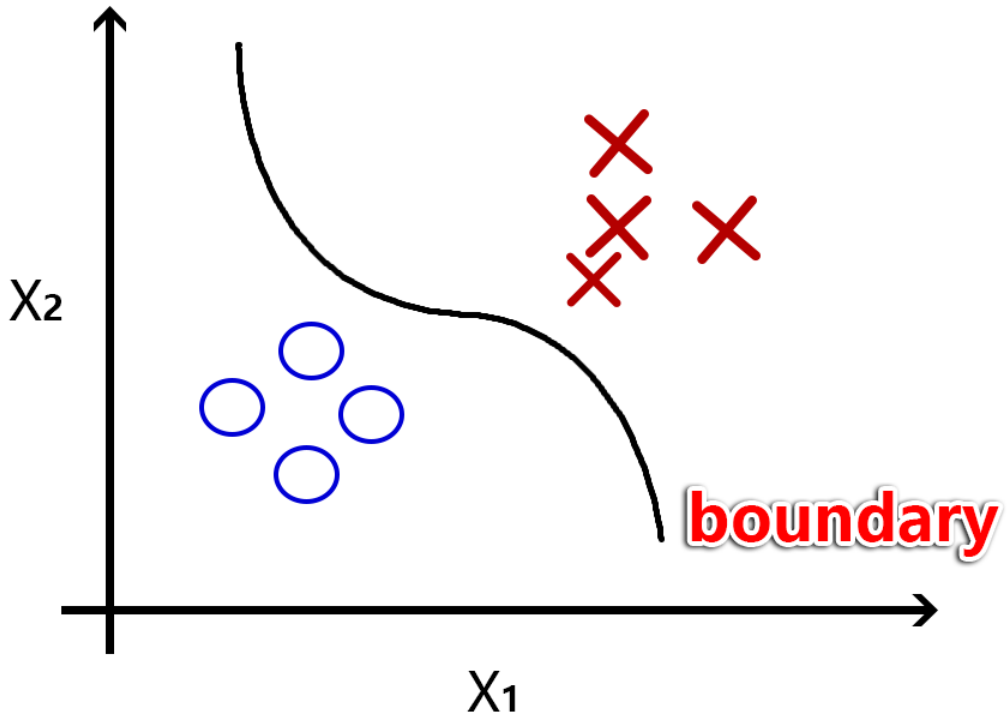
监督学习

监督学习

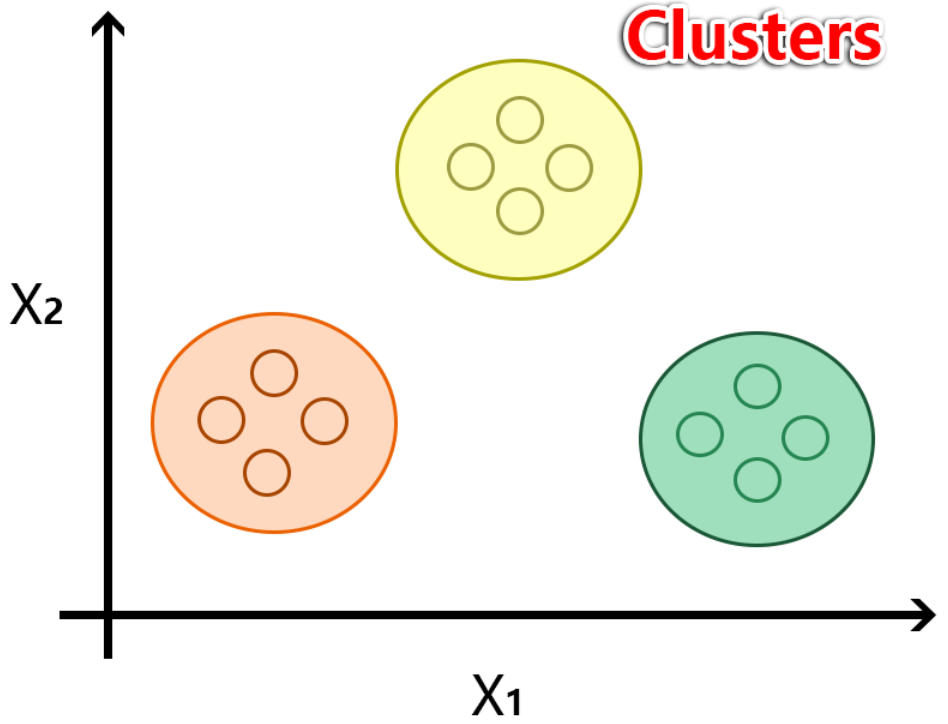
图像分类	支持向量机	K-近邻算法	决策树	朴素贝叶斯

无监督学习

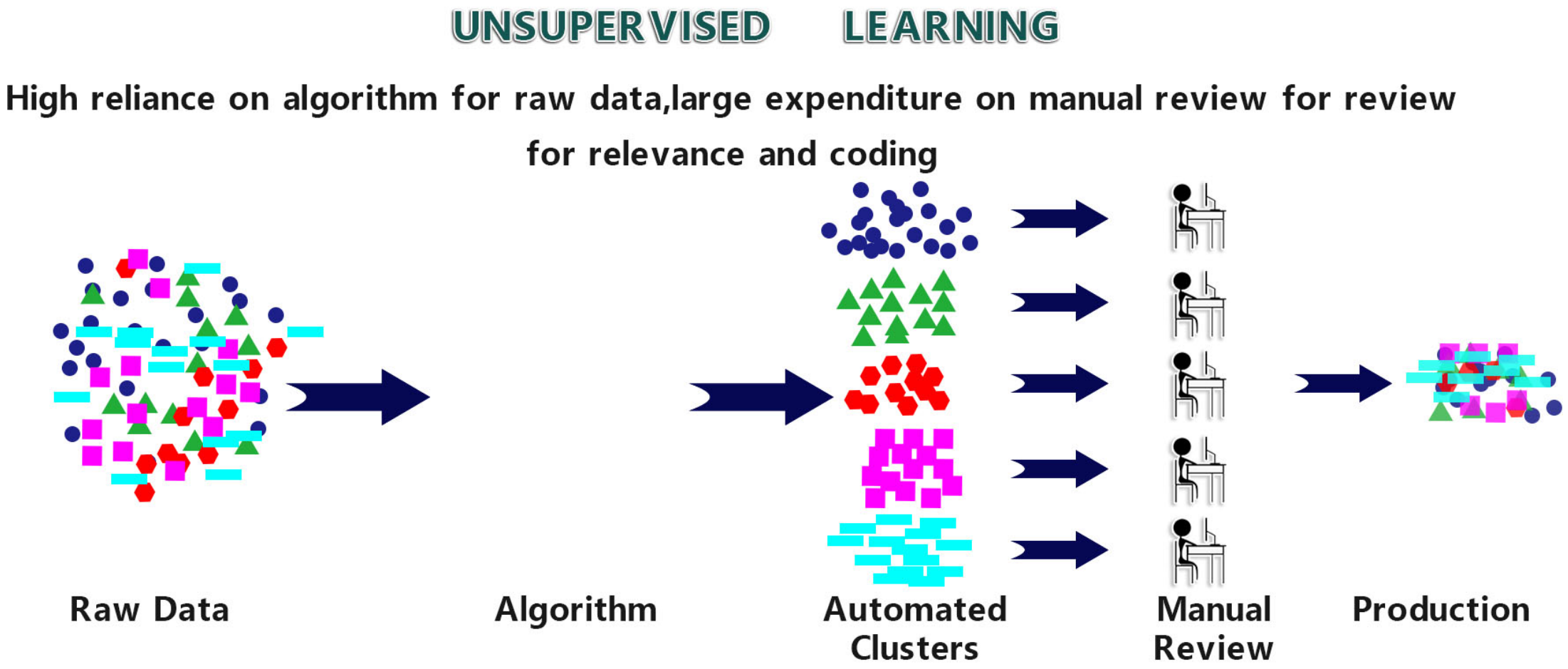
Supervised learning



Unsupervised learning



无监督学习


UNSUPERVISED LEARNING
High reliance on algorithm for raw data,large expenditure on manual review for review for relevance and coding
Raw Data
Algorithm
Automated Clusters
Manual Review
Production

无监督学习

自编码器

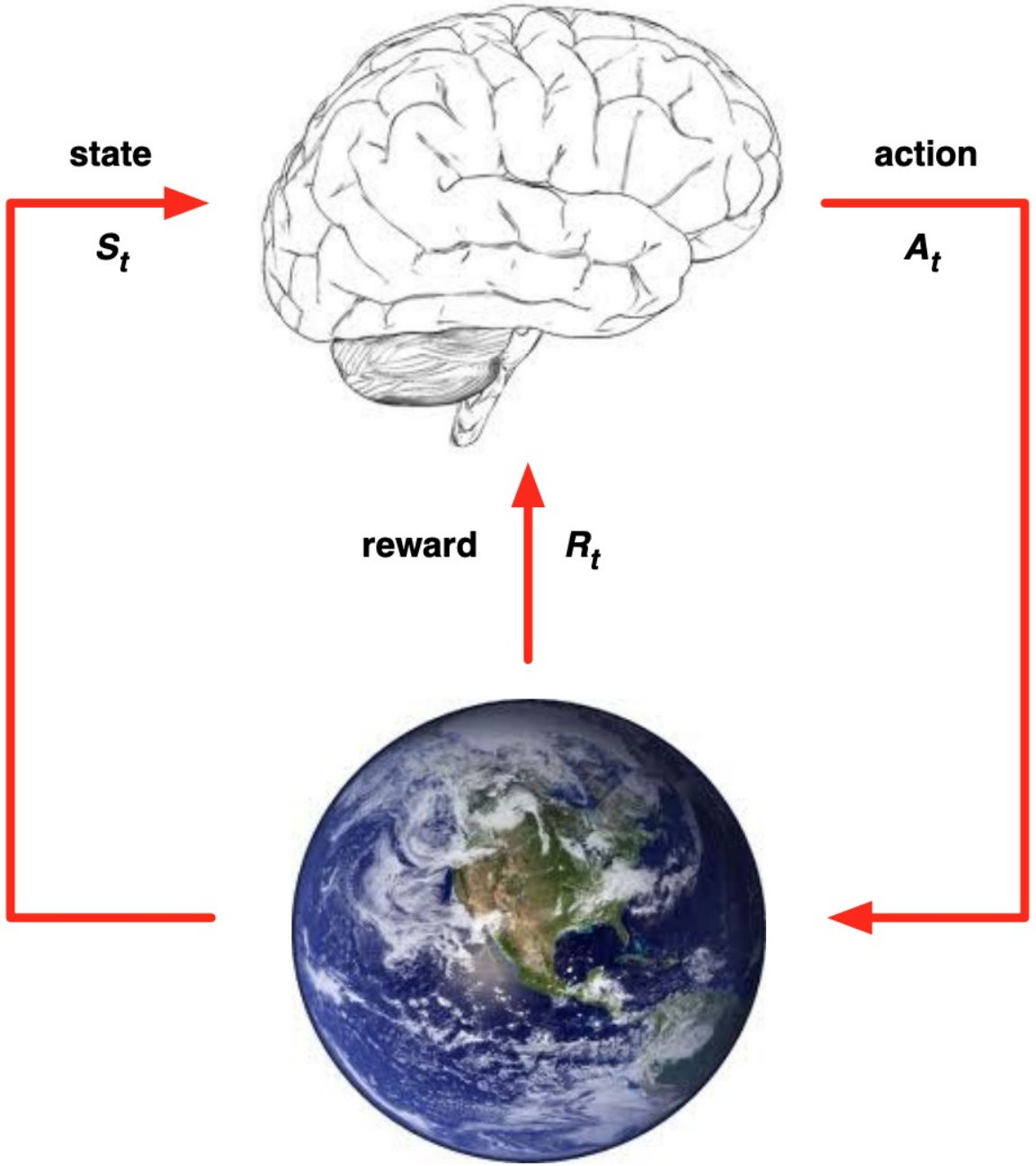
生成对抗网络

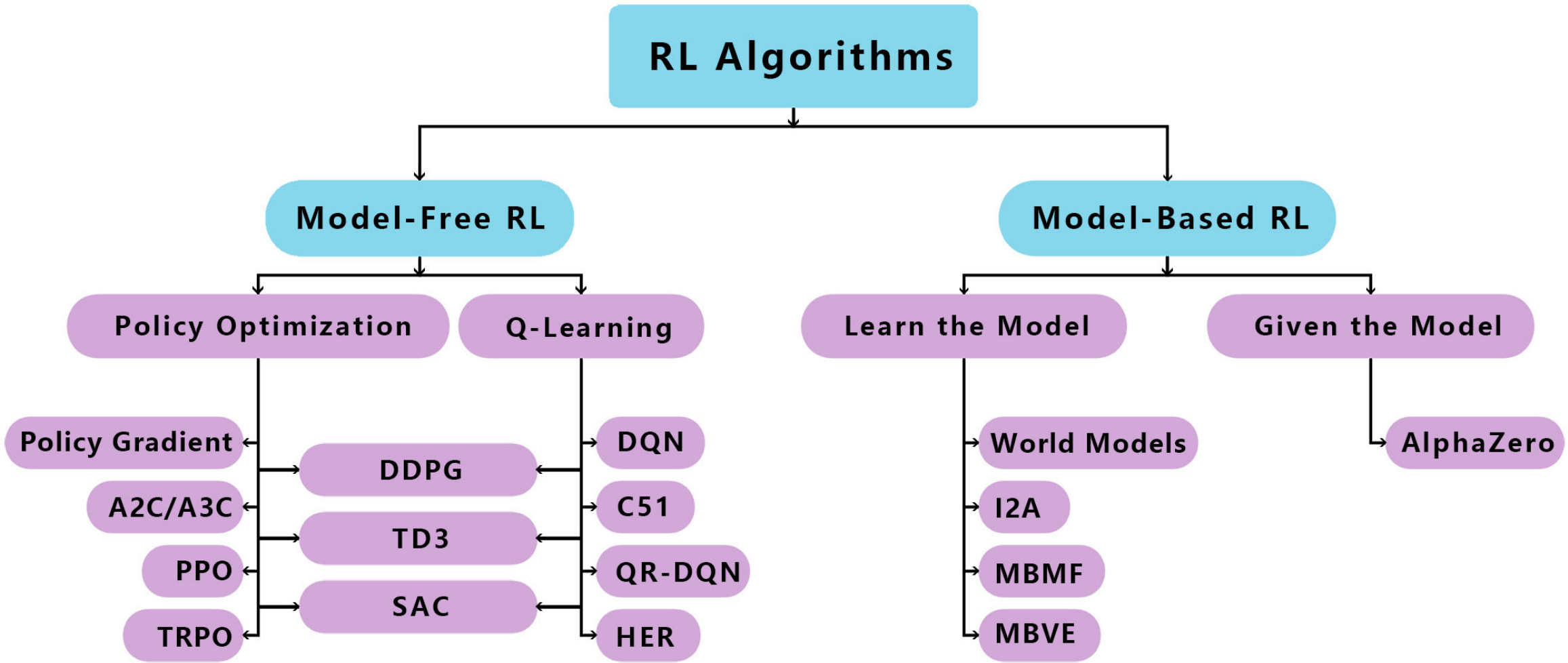
k-means

算法

主成分分析

强化学习




RL Algorithms
Model-Free RL
Model-Based RL
Policy Optimization
Q-Learning
Learn the Model
Given the Model
Policy Gradient
A2C/A3C
PPO
TRPO
DDPG
TD3
SAC
DQN
C51
QR-DQN
HER
World Models
I2A
MBMF
MBVE
AlphaZero


EDU
CSDN学院 IT实战派

EDU

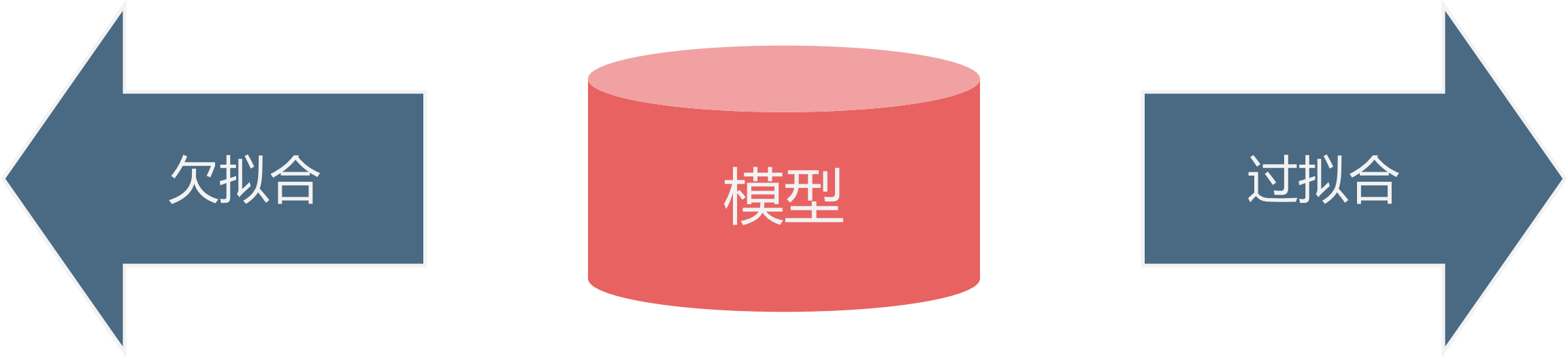
CSDN学院　IT实战派

Python 深度学习

模型训练：如何评估一个模型的好与坏

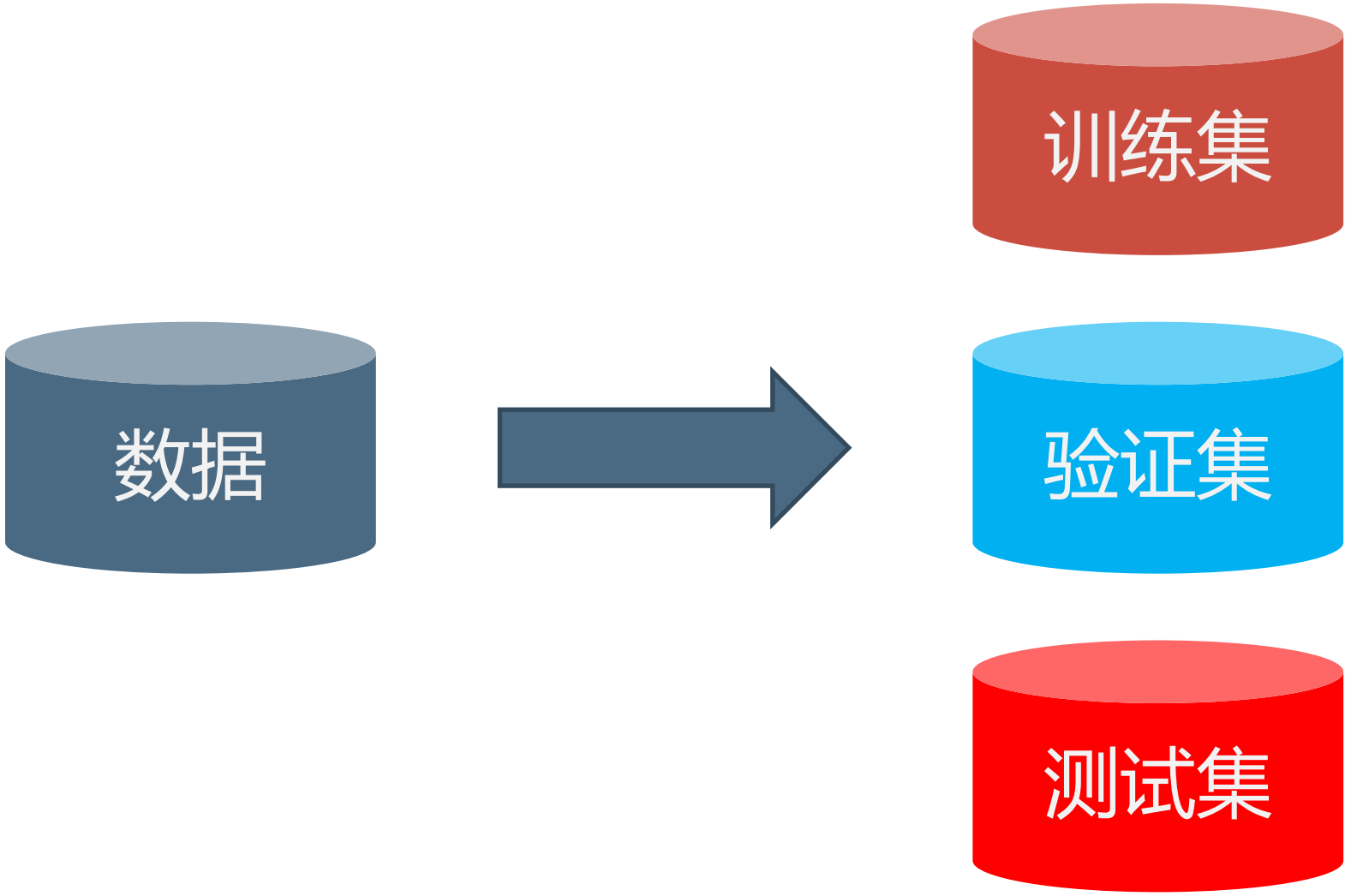
讲师： 彭靖田

模型评估



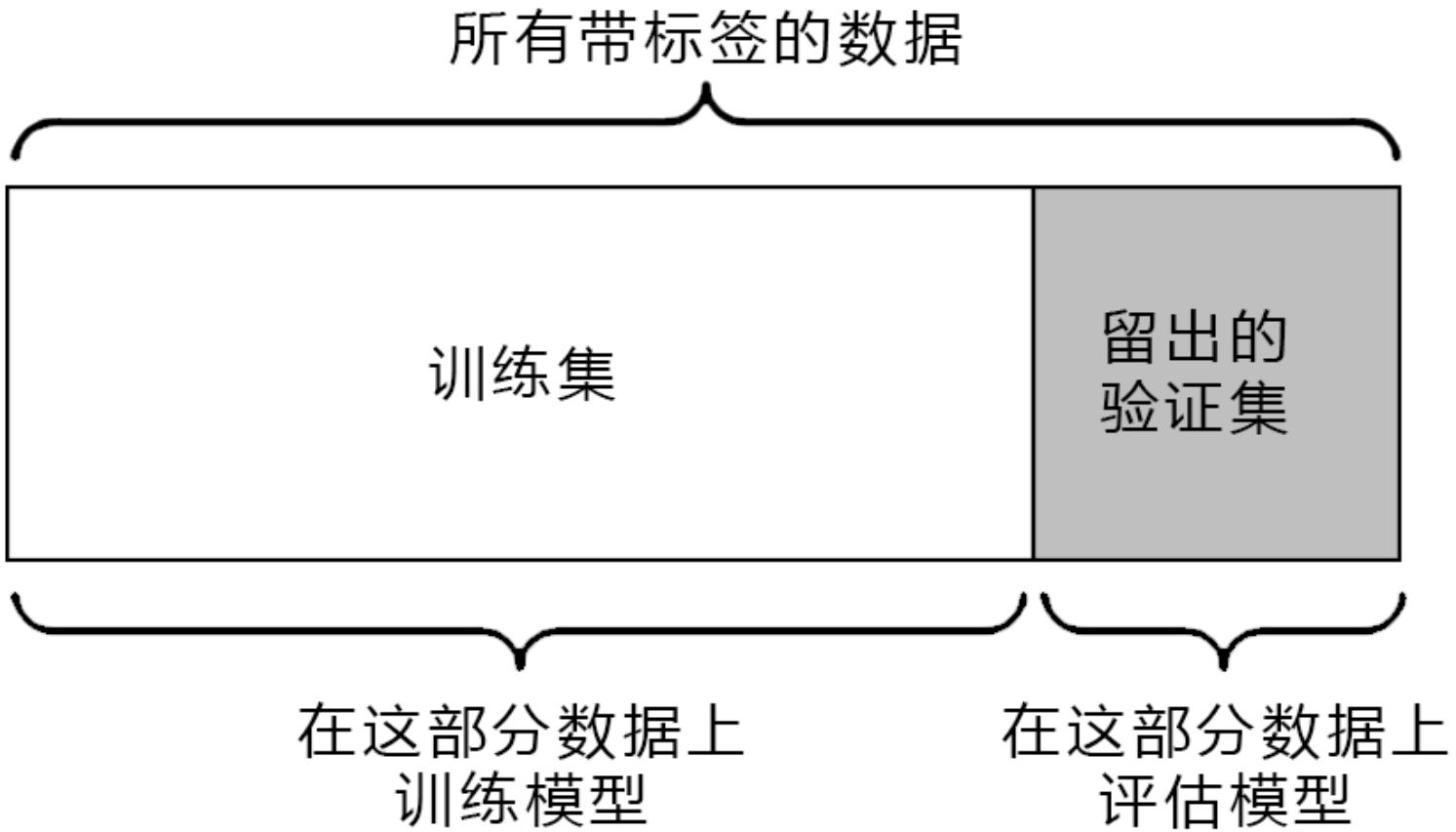
模型评估

数据划分

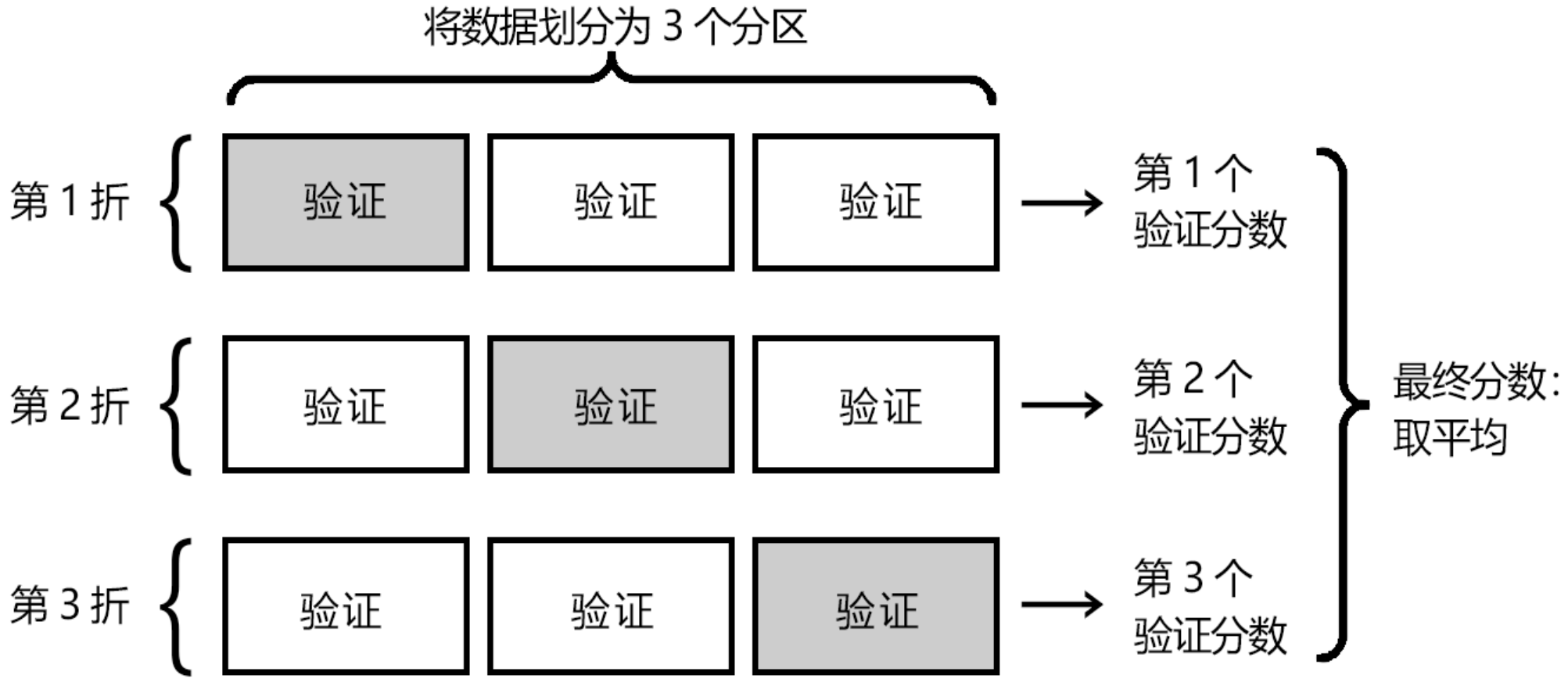


数据很少怎么办？？？

简单划分



K折验证


将数据划分为 3 个分区
第 1 折
第 2 折
第 3 折
验证
验证
验证
验证
验证
验证
验证
验证
验证
第 1 个
验证分数
第 2 个
验证分数
第 3 个
验证分数
最终分数:
取平均


EDU
CSDN学院 IT实战派

EDU

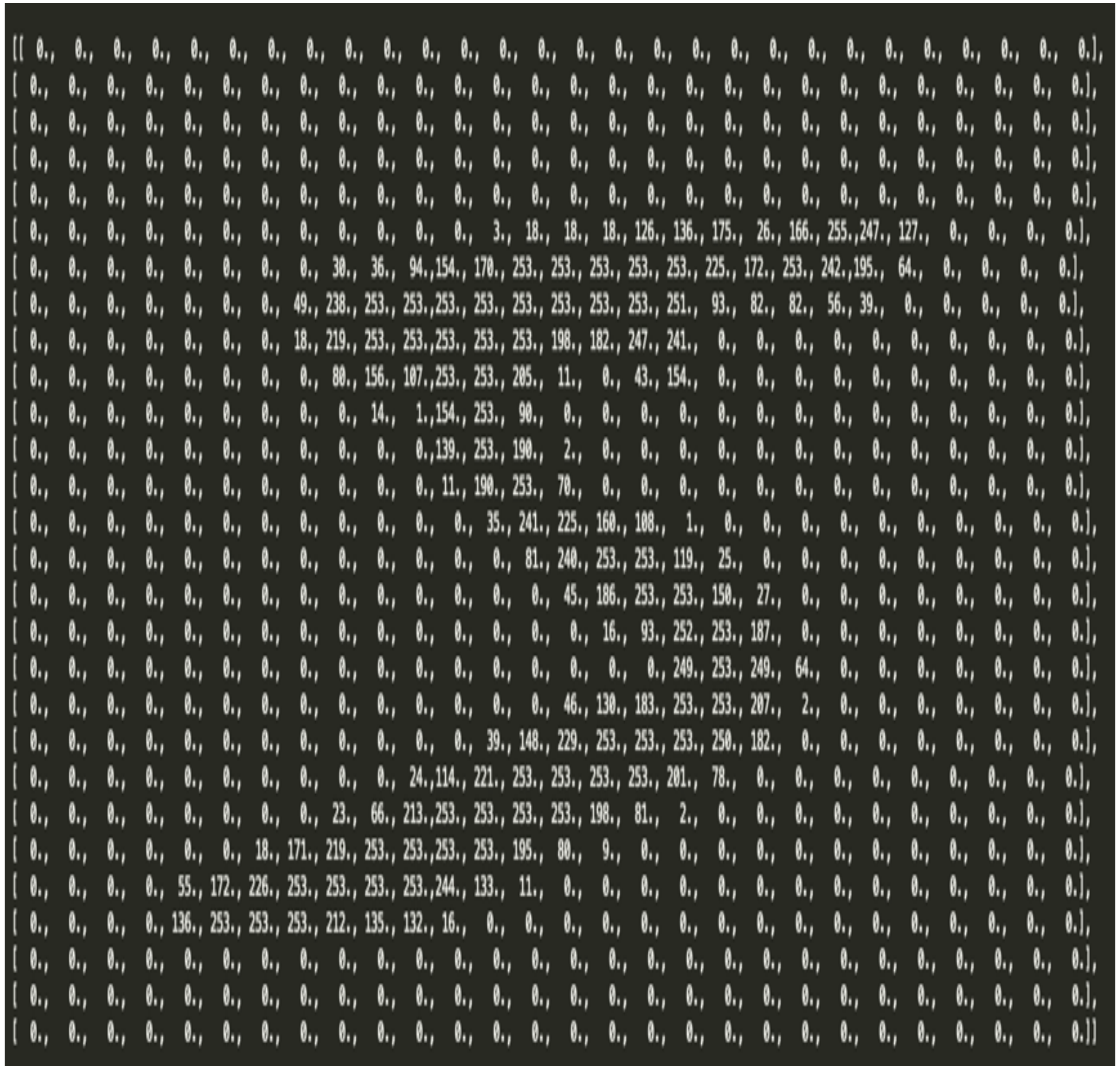
CSDN学院　IT实战派

Python 深度学习

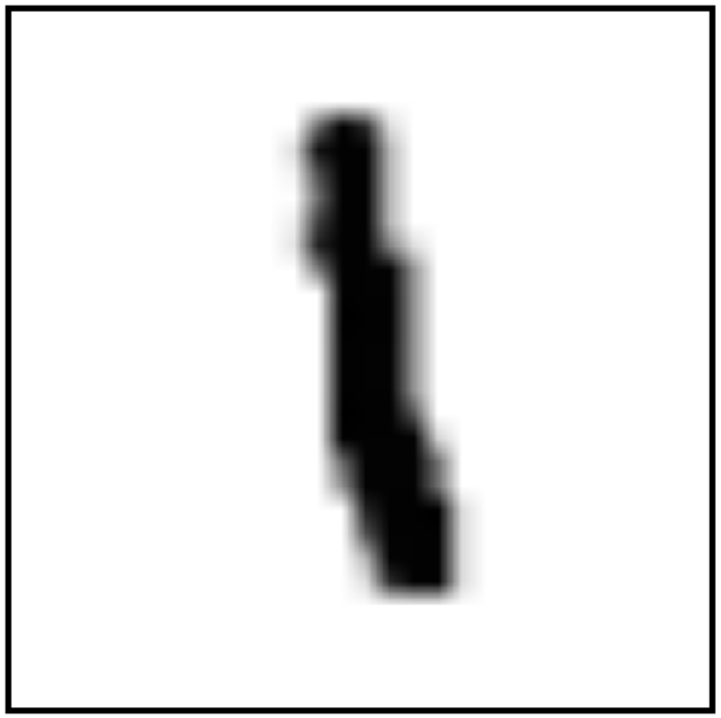
模型训练：常用数据预处理方法

讲师： 彭靖田

数据预处理-向量化

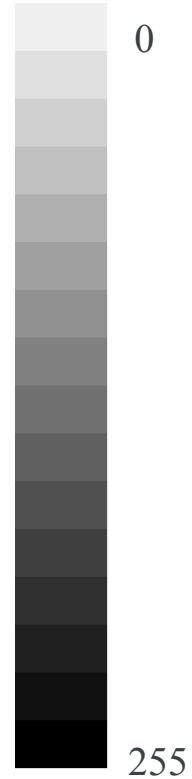

[[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 3., 18., 18., 18., 126., 136., 175., 26., 166., 255., 247., 127., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 0., 30., 36., 94., 154., 170., 253., 253., 253., 253., 253., 225., 172., 253., 242., 195., 64., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 49., 238., 253., 253., 253., 253., 253., 253., 253., 253., 251., 93., 82., 82., 56., 39., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 18., 219., 253., 253., 253., 253., 253., 198., 182., 247., 241., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 0., 80., 156., 107., 253., 253., 205., 11., 0., 43., 154., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 0., 0., 14., 1., 154., 253., 90., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 139., 253., 190., 2., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 11., 190., 253., 70., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 35., 241., 225., 160., 108., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 81., 240., 253., 253., 119., 25., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 45., 186., 253., 253., 150., 27., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 16., 93., 252., 253., 187., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 249., 253., 249., 64., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 46., 130., 183., 253., 253., 207., 2., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 39., 148., 229., 253., 253., 253., 250., 182., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 24., 114., 221., 253., 253., 253., 253., 201., 78., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 0., 23., 66., 213., 253., 253., 253., 253., 198., 81., 2., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 18., 171., 219., 253., 253., 253., 253., 195., 80., 9., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 55., 172., 226., 253., 253., 253., 253., 244., 133., 11., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 136., 253., 253., 253., 212., 135., 132., 16., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.]]

数据预处理-值标准化

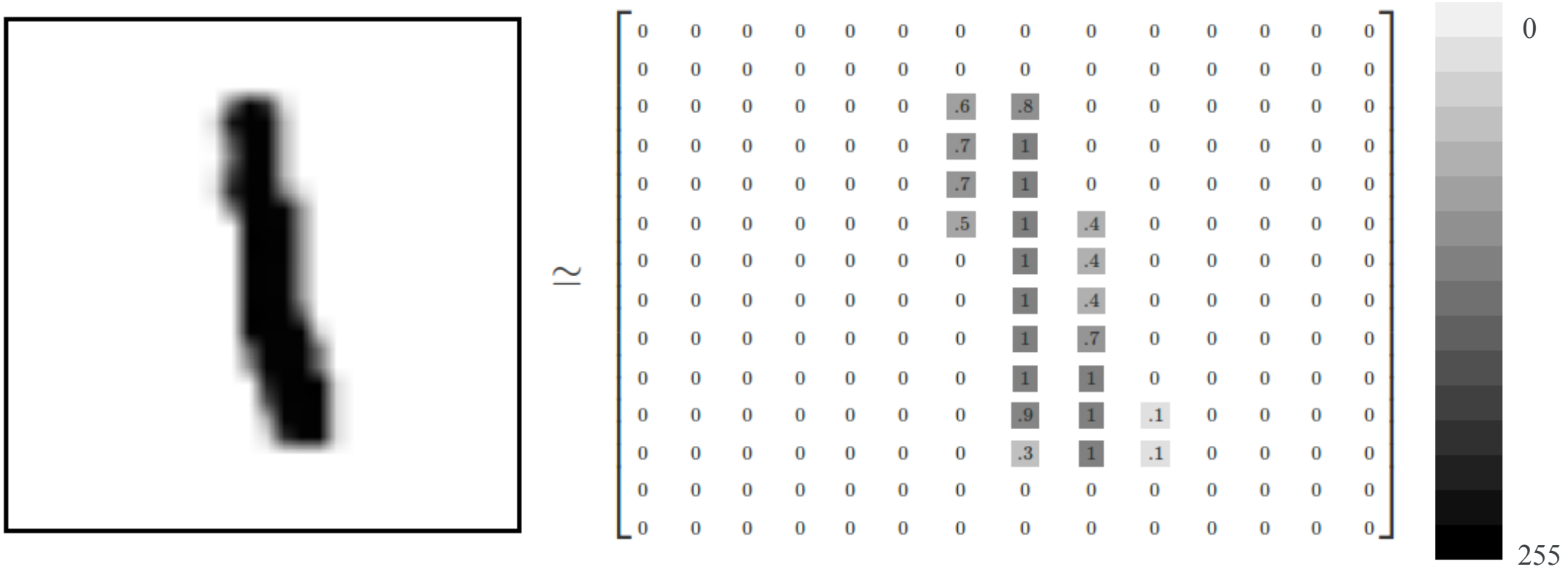


≈

0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	.6	.8	0	0	0	0	0	0
0	0	0	0	0	0	.7	1	0	0	0	0	0	0
0	0	0	0	0	0	.7	1	0	0	0	0	0	0
0	0	0	0	0	0	.5	1	.4	0	0	0	0	0
0	0	0	0	0	0	0	1	.4	0	0	0	0	0
0	0	0	0	0	0	0	1	.4	0	0	0	0	0
0	0	0	0	0	0	0	1	.7	0	0	0	0	0
0	0	0	0	0	0	0	1	1	0	0	0	0	0
0	0	0	0	0	0	0	.9	1	.1	0	0	0	0
0	0	0	0	0	0	0	.3	1	.1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

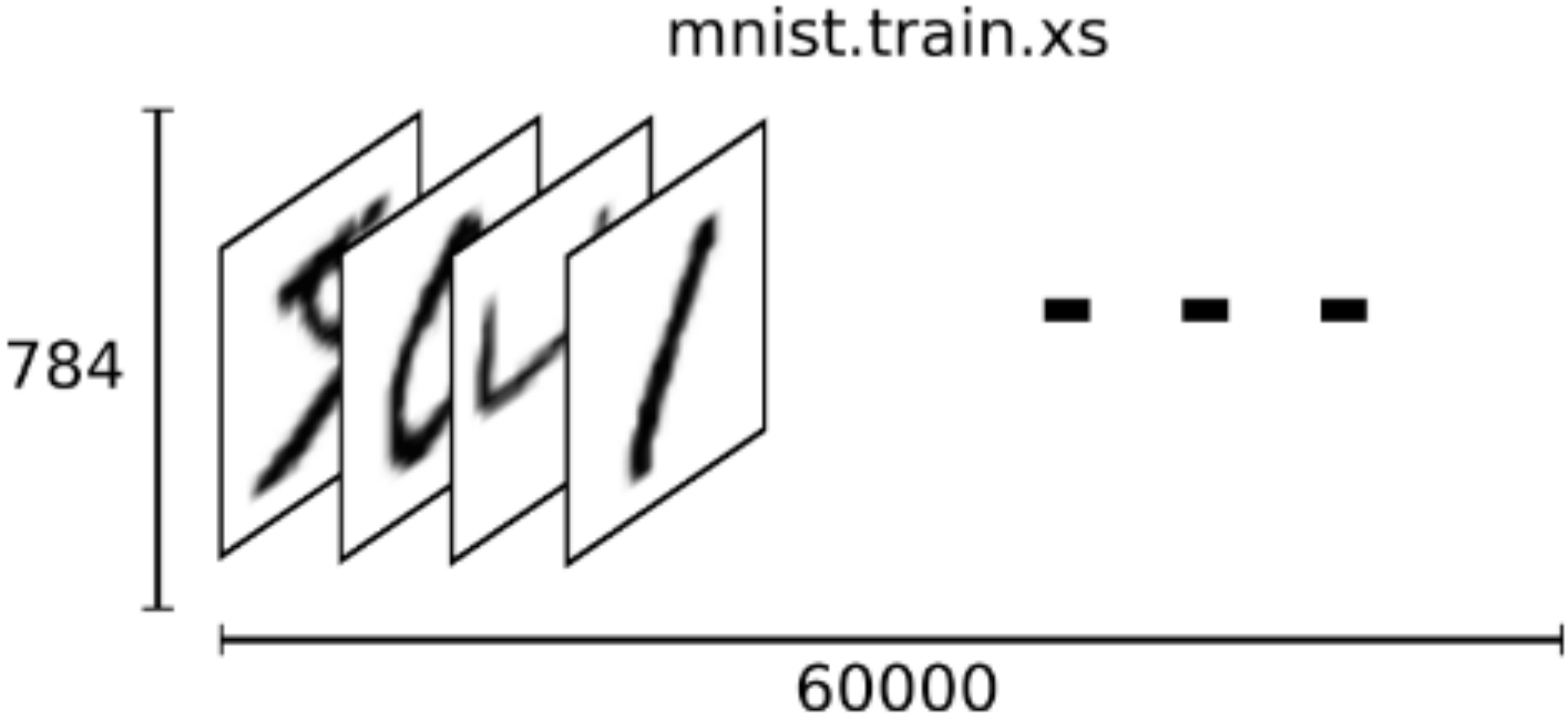


数据预处理-值标准化

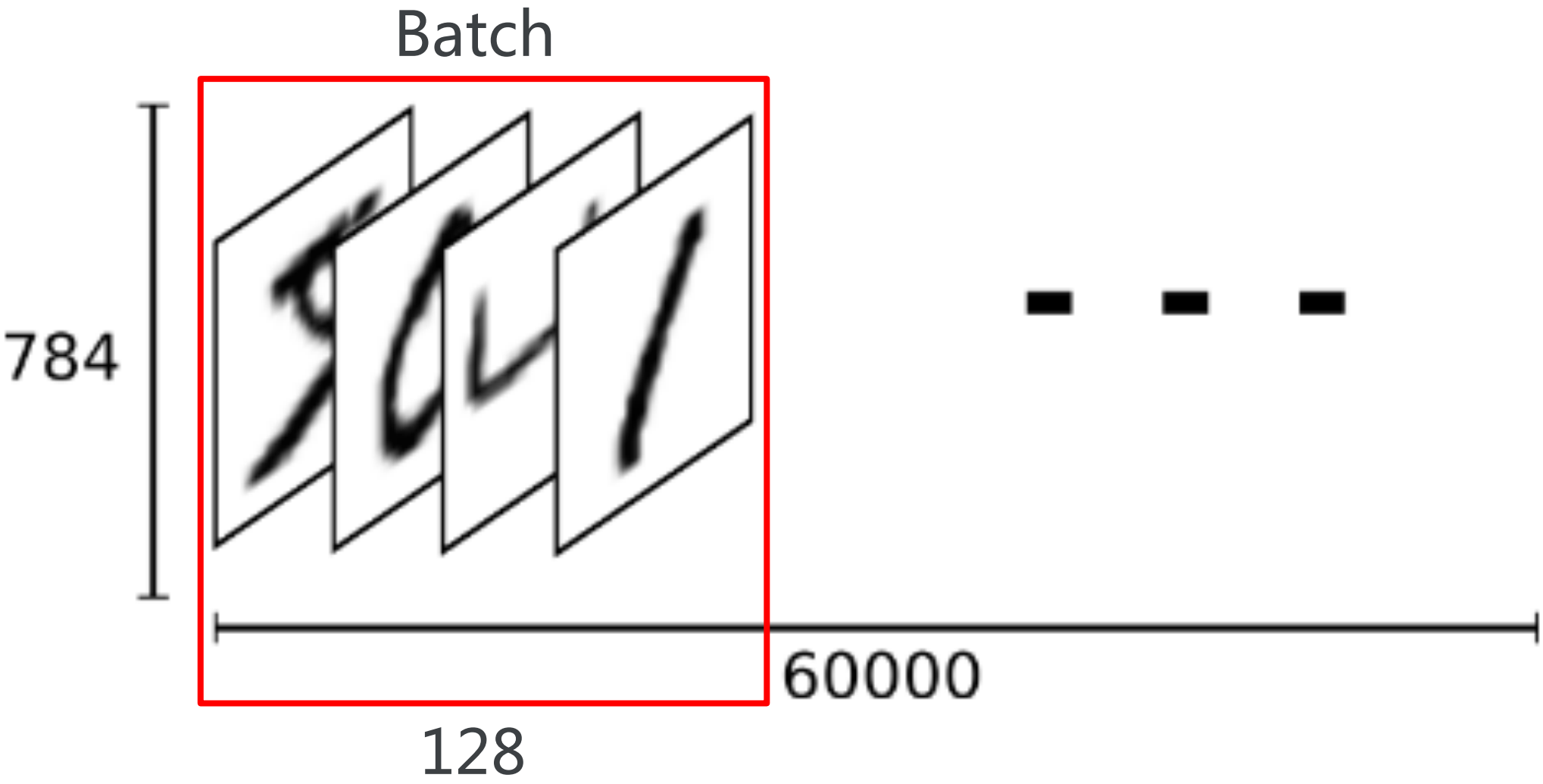


Reshape: (28,28) -> (784, 1)

数据预处理-数据批量化



数据预处理-数据批量化




EDU
CSDN学院 IT实战派

EDU

CSDN学院　IT实战派

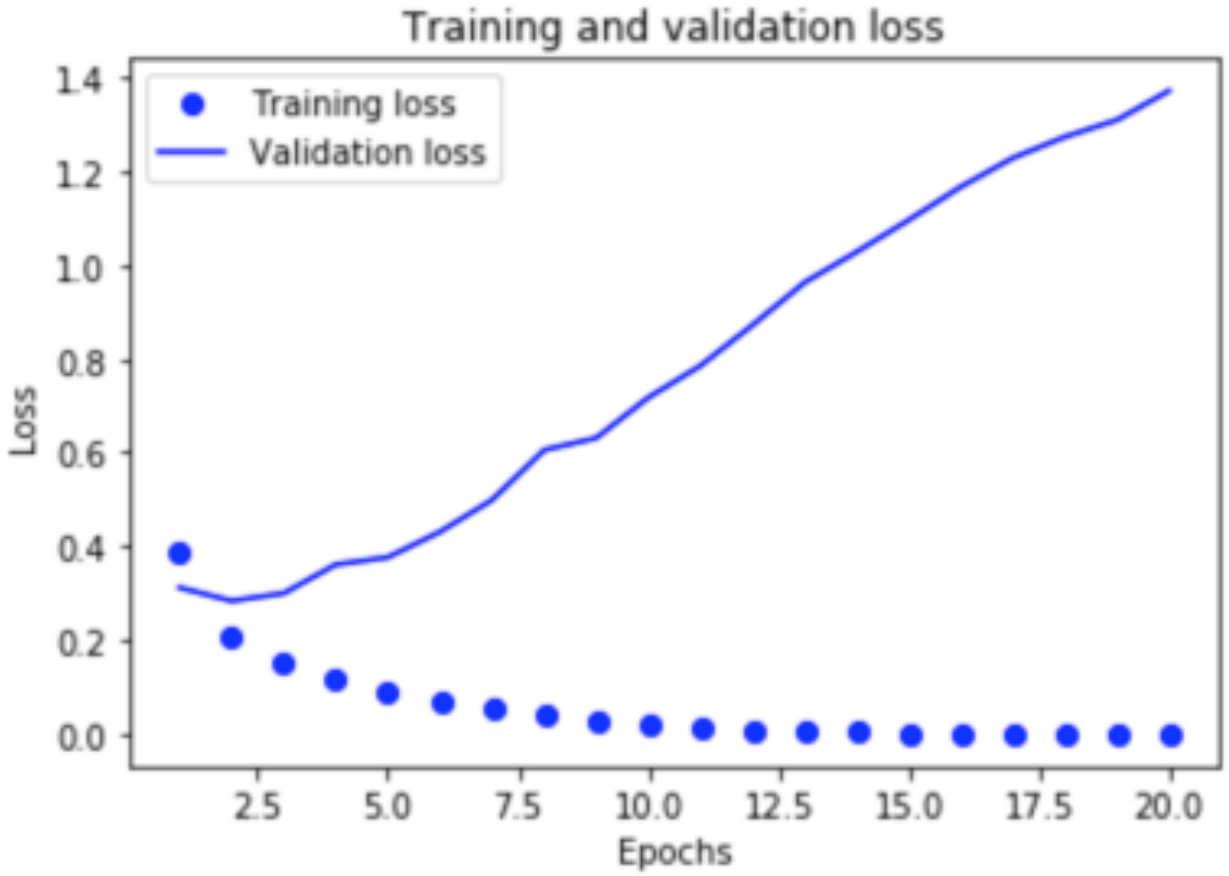
Python 深度学习

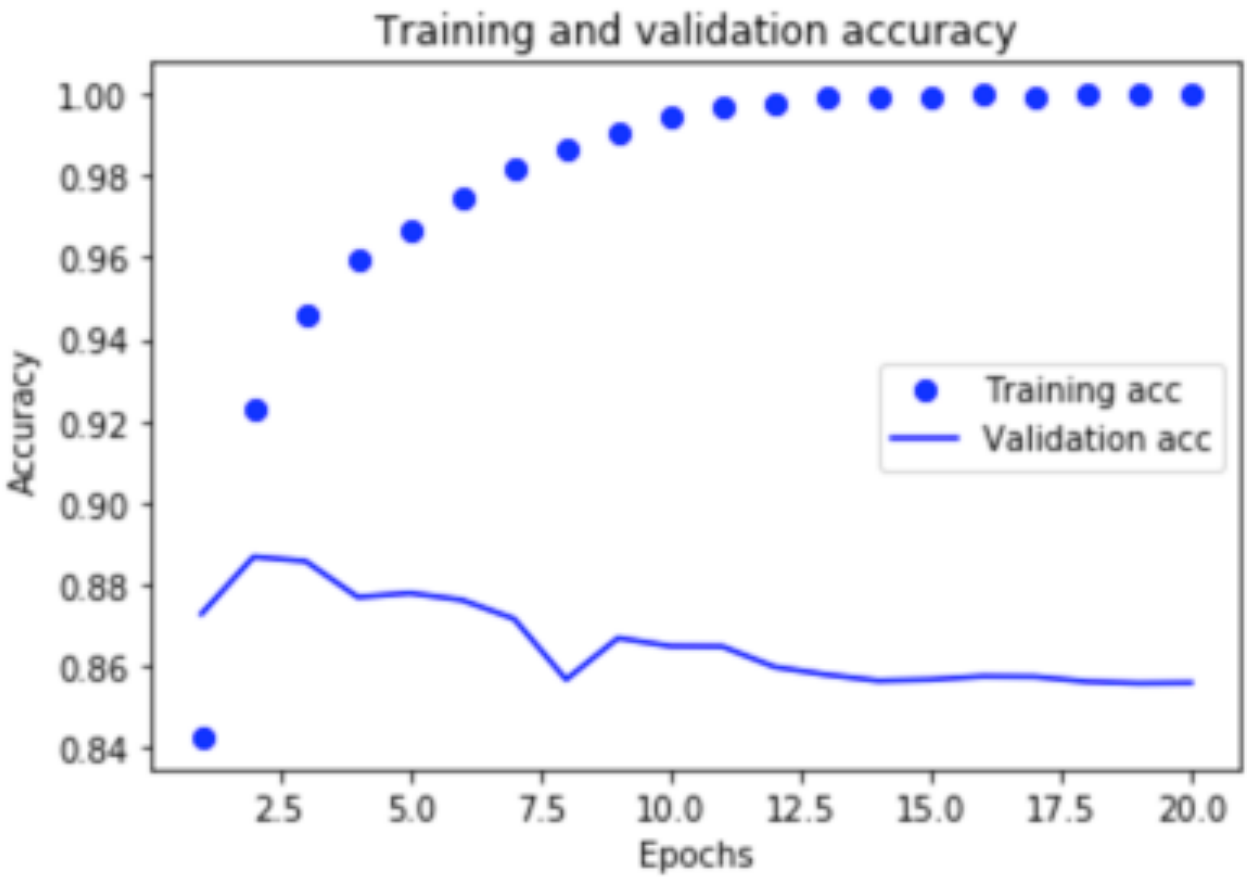
模型训练：如何解决过拟合问题

讲师： 彭靖田

模型过拟合

模型过拟合


Training and validation loss
Training loss
Validation loss
Loss
Epochs
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0


Training and validation accuracy
Training acc
Validation acc
Accuracy
Epochs
0.84
0.86
0.88
0.90
0.92
0.94
0.96
0.98
1.00
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

降低模型复杂度

Large Model

```
model = models.Sequential()
model.add(layers.Dense(512, activation='relu', input_shape=(10000,)))
model.add(layers.Dense(512, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))
```

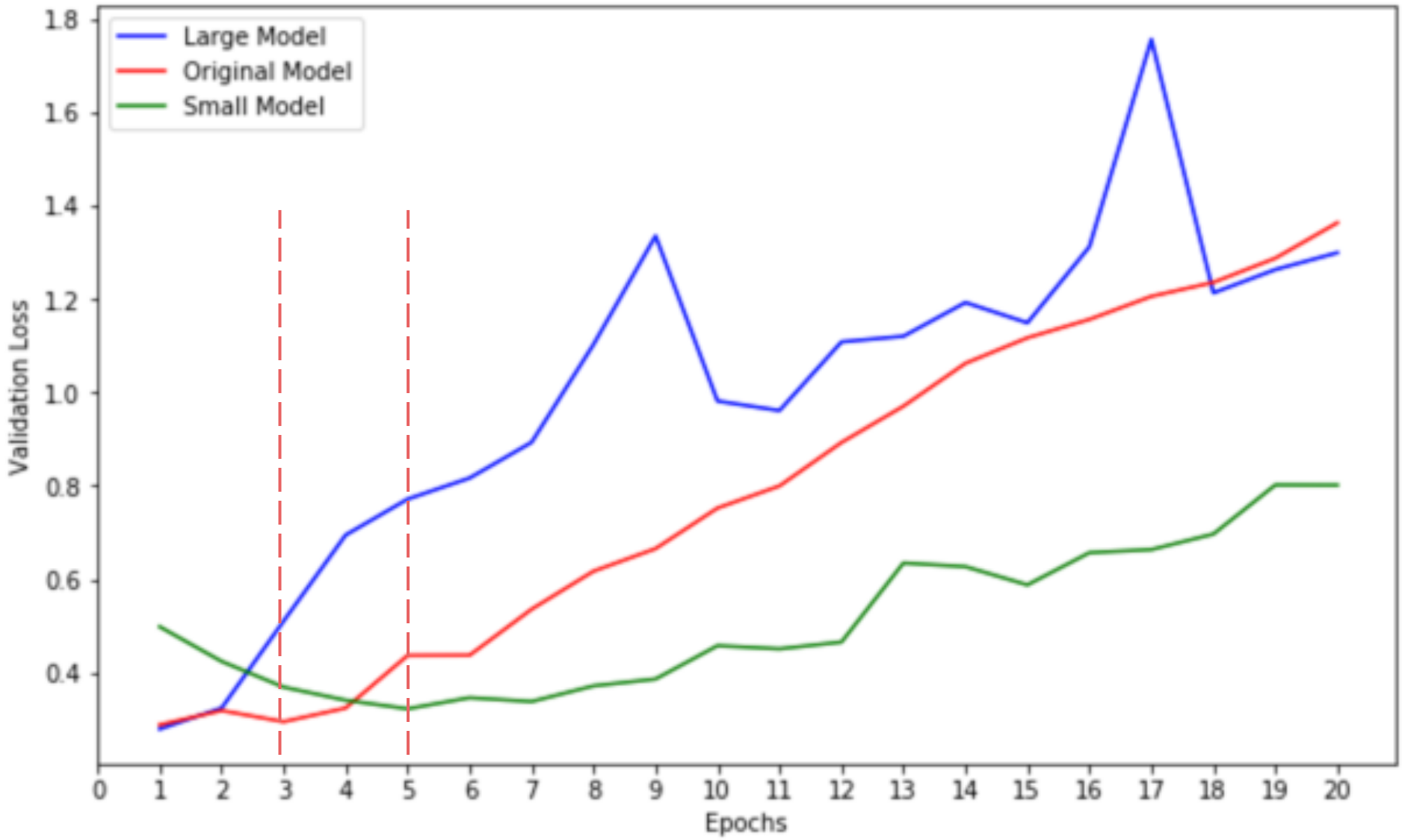
Original Model

```
model = models.Sequential()
model.add(layers.Dense(16, activation='relu', input_shape=(10000,)))
model.add(layers.Dense(16, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))
```

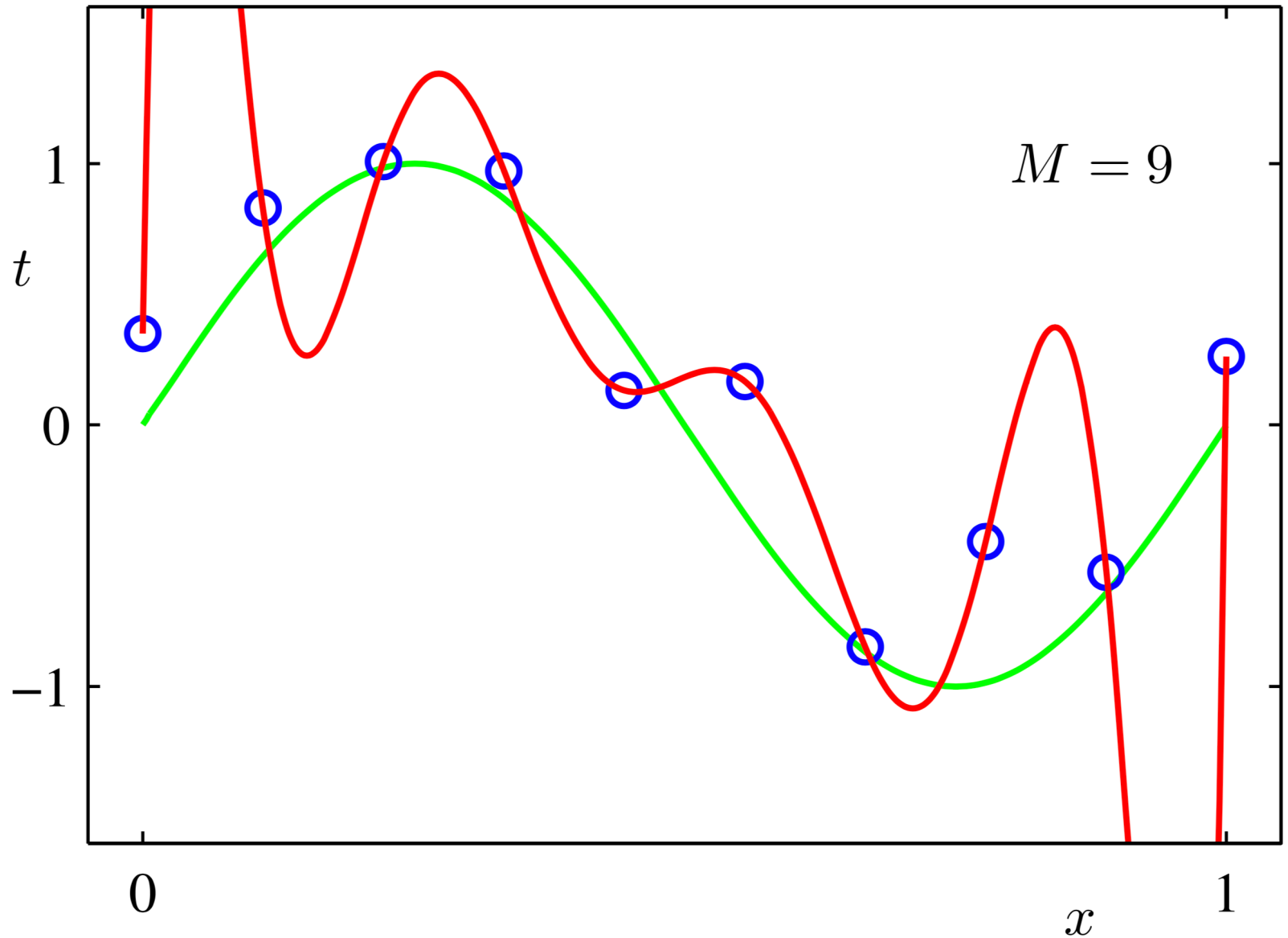
Small Model

```
model = models.Sequential()
model.add(layers.Dense(4, activation='relu', input_shape=(10000,)))
model.add(layers.Dense(4, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))
```

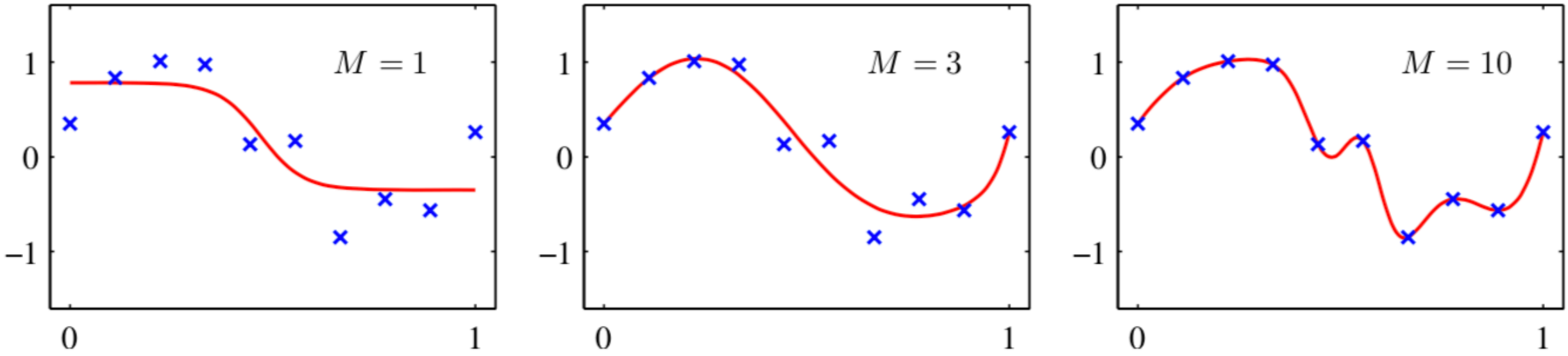
降低模型复杂度


Large Model
Original Model
Small Model
Validation Loss
Epochs
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

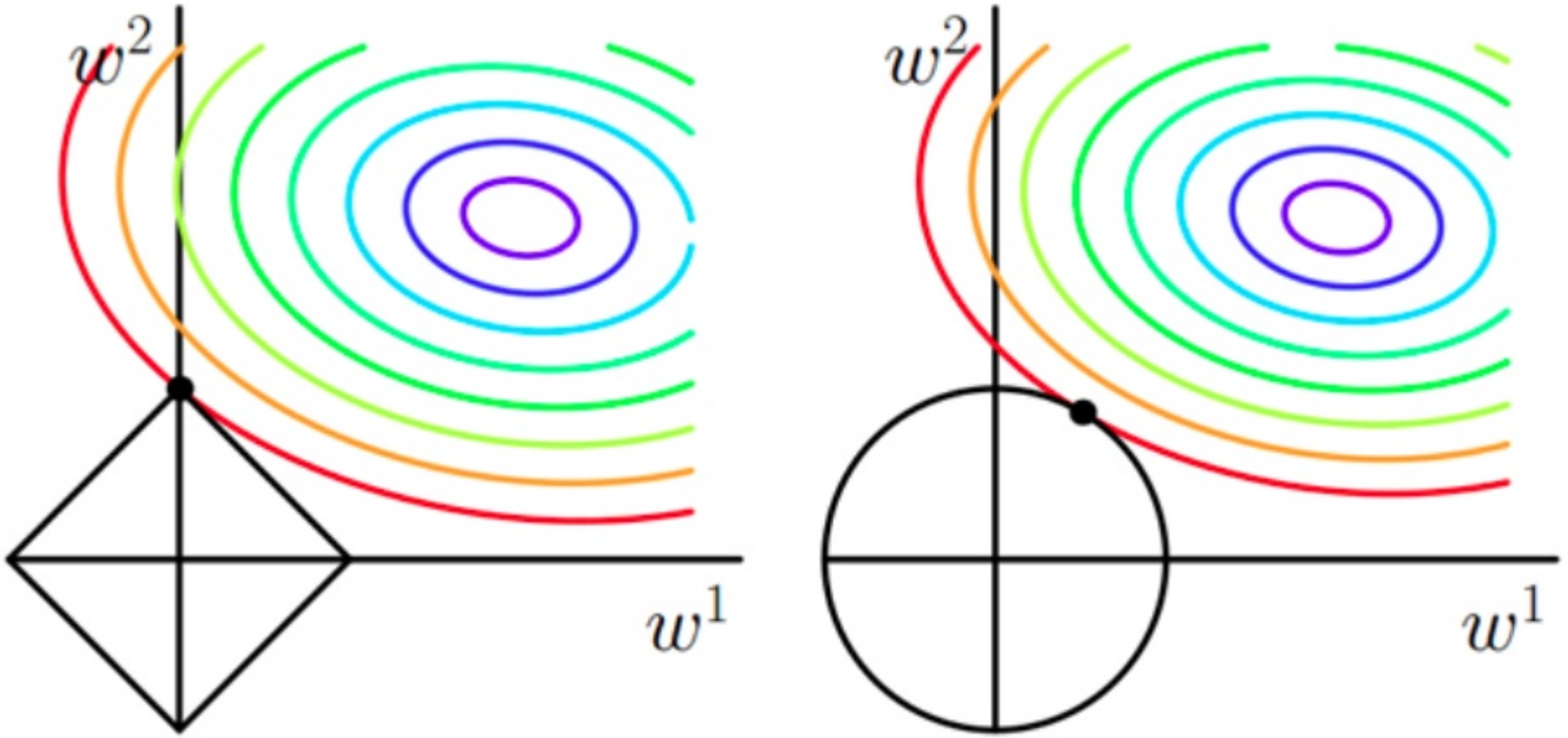
权重正则化



权重正则化



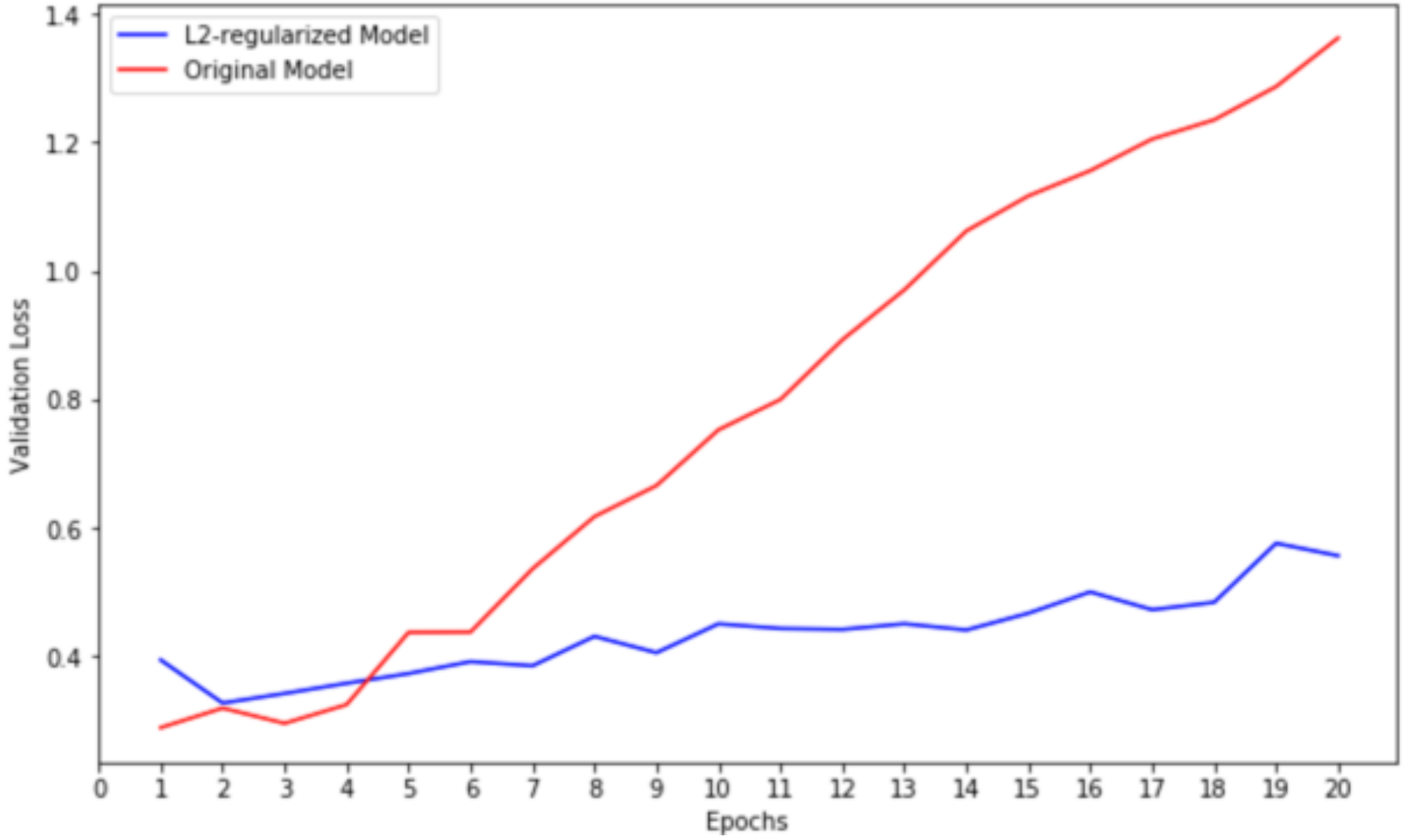
L1 正则 vs L2 正则



L2 正则化

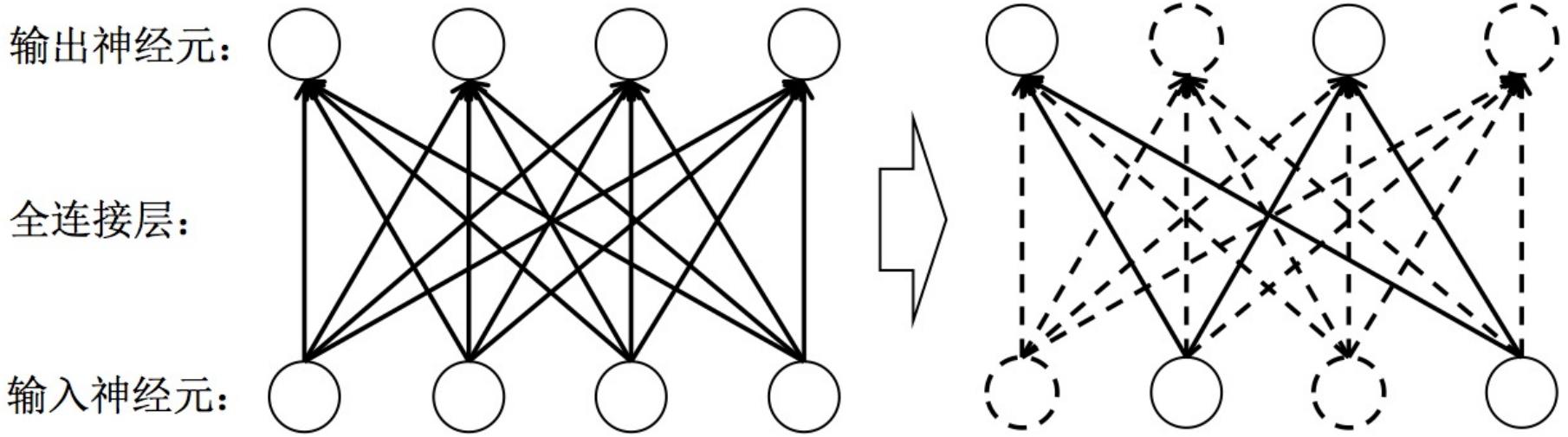
```
from keras import regularizers

model = models.Sequential()
model.add(layers.Dense(16,
                       kernel_regularizer=regularizers.l2(0.001),
                       activation='relu',
                       input_shape=(10000,)))
model.add(layers.Dense(16,
                       kernel_regularizer=regularizers.l2(0.001),
                       activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(loss='binary_crossentropy',
              optimizer=optimizers.RMSprop(lr=0.001),
              metrics=['accuracy'])

history = model.fit(partial_x_train,
                    partial_y_train,
                    batch_size=batch_size,
                    epochs=epochs,
                    verbose=1,
                    validation_data=(x_val, y_val))
```

L2 正则化


L2-regularized Model
Original Model
Validation Loss
Epochs
1.4
1.2
1.0
0.8
0.6
0.4
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20

Dropout

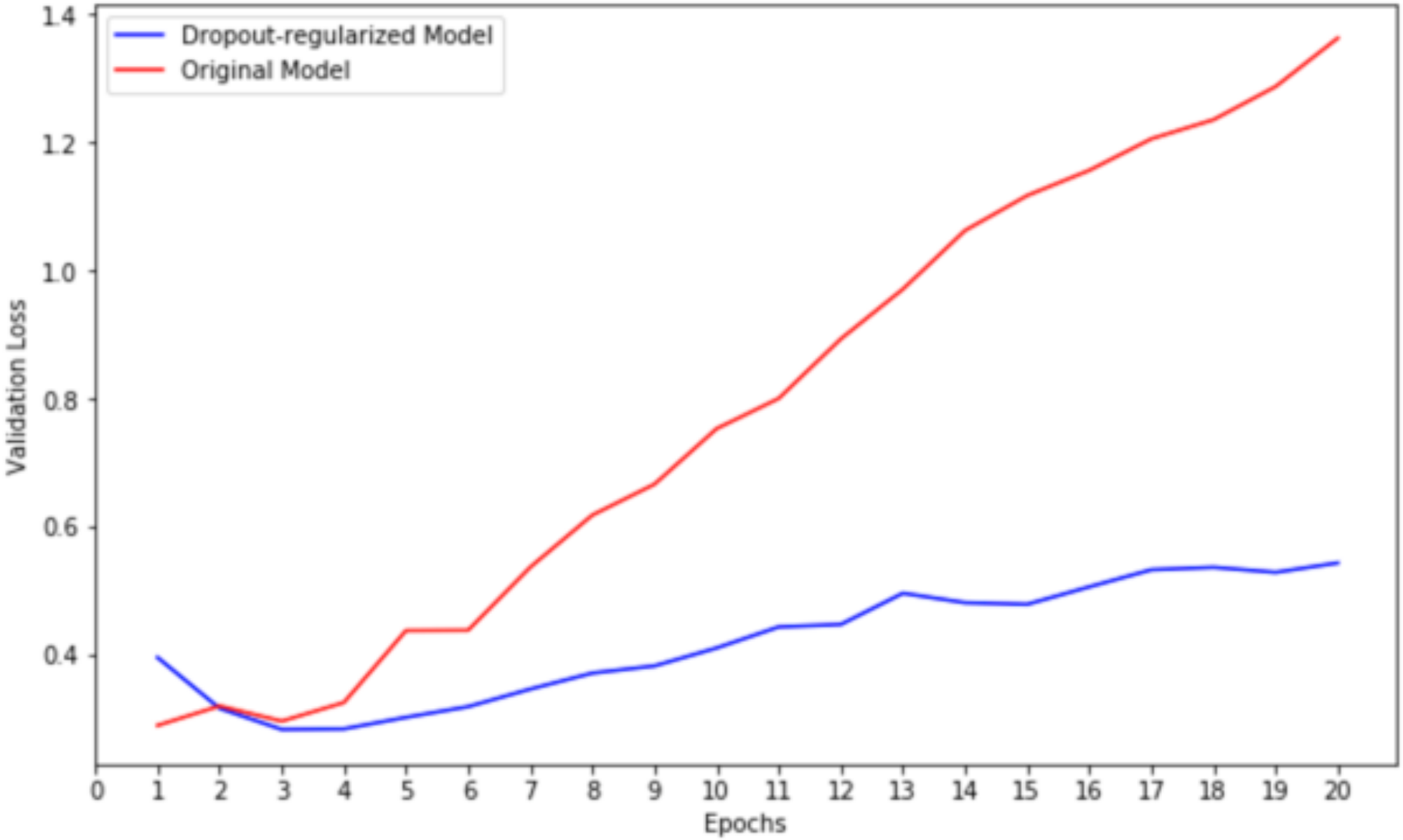
Dropout 是常用的一种正则化方法，Dropout层是一种正则化层。全连接层参数量非常庞大（占据了CNN模型参数量的80%～90%左右），发生过拟合问题的风险比较高，所以我们通常需要一些正则化方法训练带有全连接层的CNN模型。在每次迭代训练时，将神经元以一定的概率值□暂时随机丢弃，即在当前迭代中不参与训练。



Dropout

```
model = models.Sequential()
model.add(layers.Dense(16, activation='relu', input_shape=(10000,)))
model.add(layers.Dropout(0.5))
model.add(layers.Dense(16, activation='relu'))
model.add(layers.Dropout(0.5))
model.add(layers.Dense(1, activation='sigmoid'))


model.compile(loss='binary_crossentropy',
              optimizer=optimizers.RMSprop(lr=0.001),
              metrics=['accuracy'])

history = model.fit(partial_x_train,
                    partial_y_train,
                    batch_size=batch_size,
                    epochs=epochs,
                    verbose=1,
                    validation_data=(x_val, y_val))
```

Dropout


Dropout-regularized Model
Original Model
Validation Loss
1.4
1.2
1.0
0.8
0.6
0.4
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
Epochs


EDU
CSDN学院 IT实战派