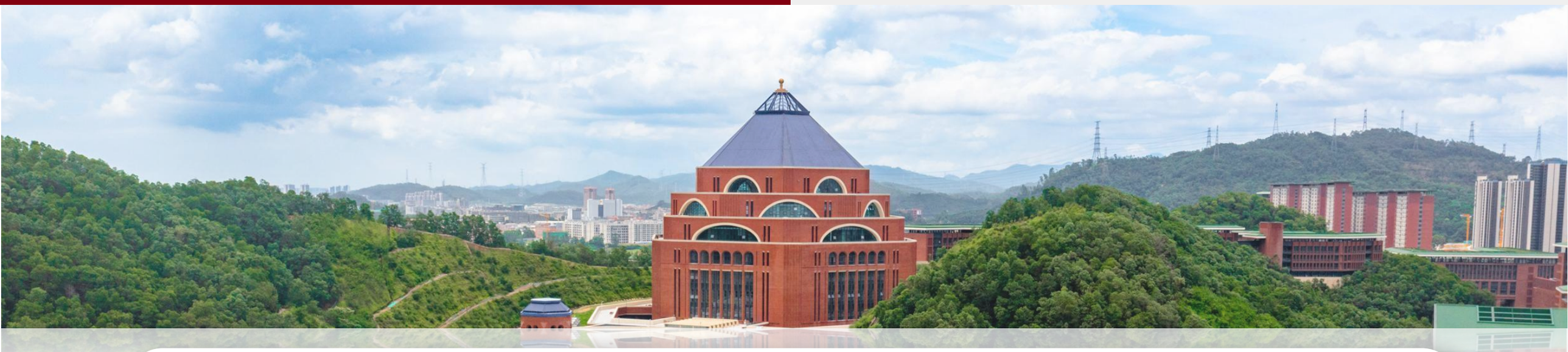


数据分析练习: 粒子物理中的机器学习方法实践（有监督学习）

Practice on data analysis: Hands-on session of machine learning methods in particle physics (*supervised learning*)

TeV 高能实验物理暑期学校 · iSTEP 2025

中山大学深圳校区

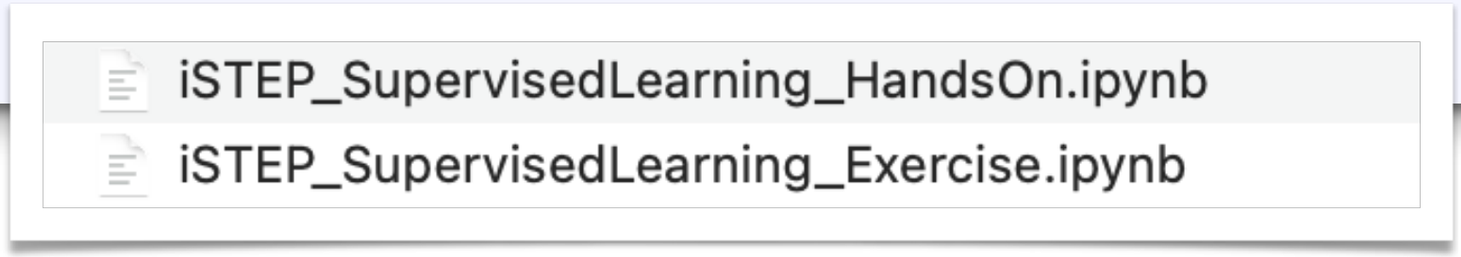
主讲人　李聪乔（北京大学）

Outline

This tutorial will cover

- ➔ Very brief introduction of supervised learning, gradient boosted decision trees (BDT), and neural networks
- ➔ Practical aspects in BDT/NN training
- ➔ **Hands-on session (our major focus):**

Materials shared in the group chat; also available on GitHub



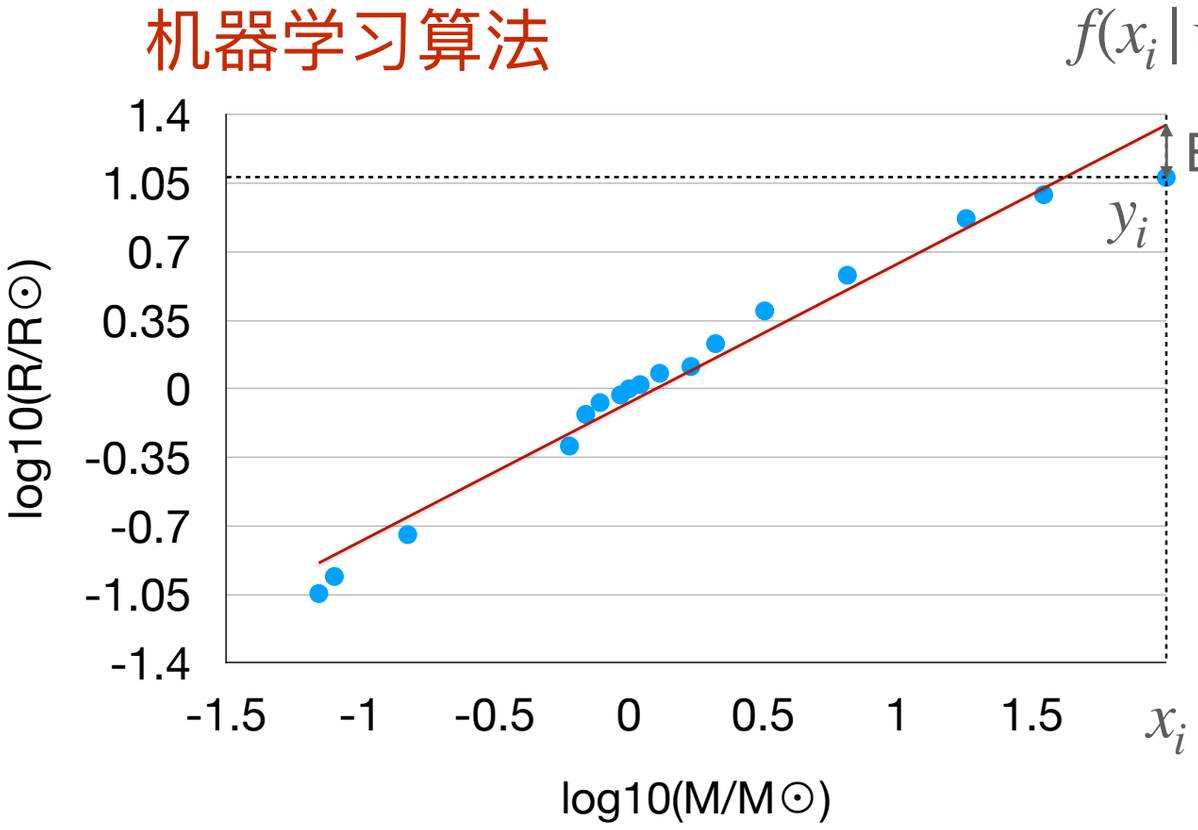
References:

- CMS data analysis school materials
- BDT/NN/transformers tutorials

Machine Learning?

机器学习：“数据驱动”的算法，基于样本数据来进行预测或者做出判断，其规则并不显式写入程序之内。

例如：**线性回归** 就是一种机器学习算法



- Linear model:

$$f(x \mid w) = w^\intercal x \quad (w \in \mathbb{R}^{D+1})$$

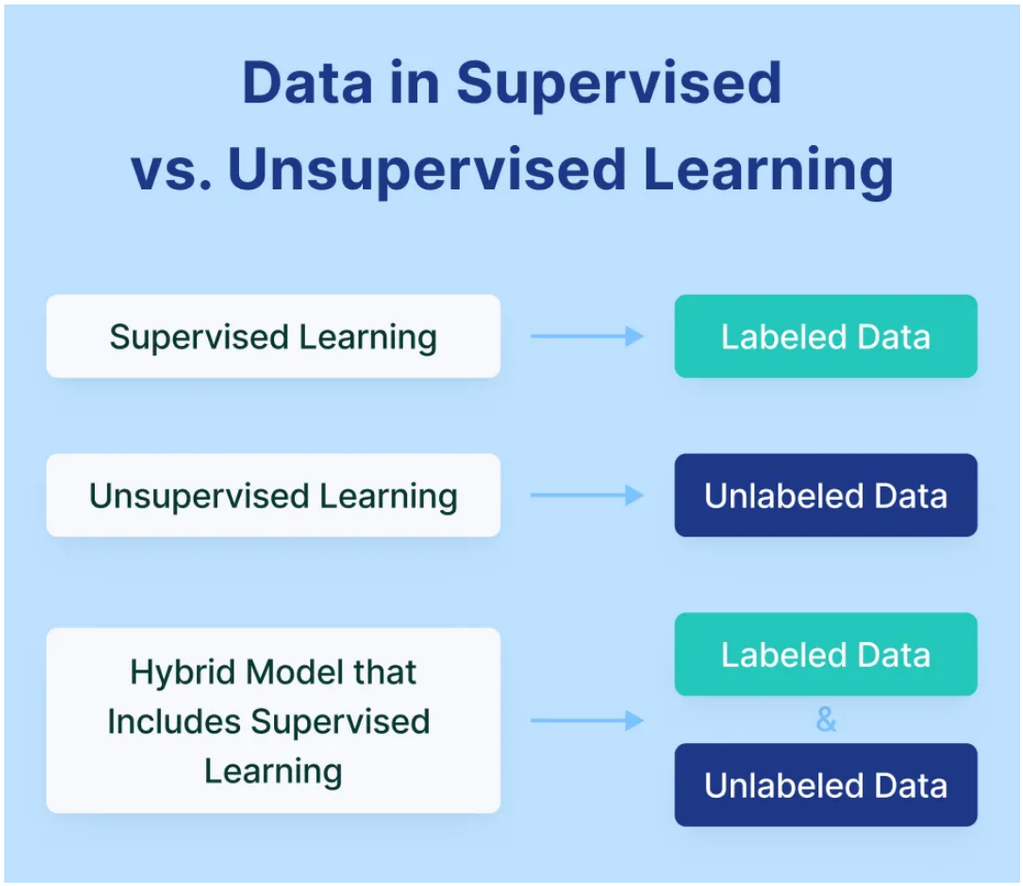
- How do we select the parameters w?
  - We want $y_i \approx f(x_i \mid w)$
- Squared loss: $L(y, y') = (y - y')^2$

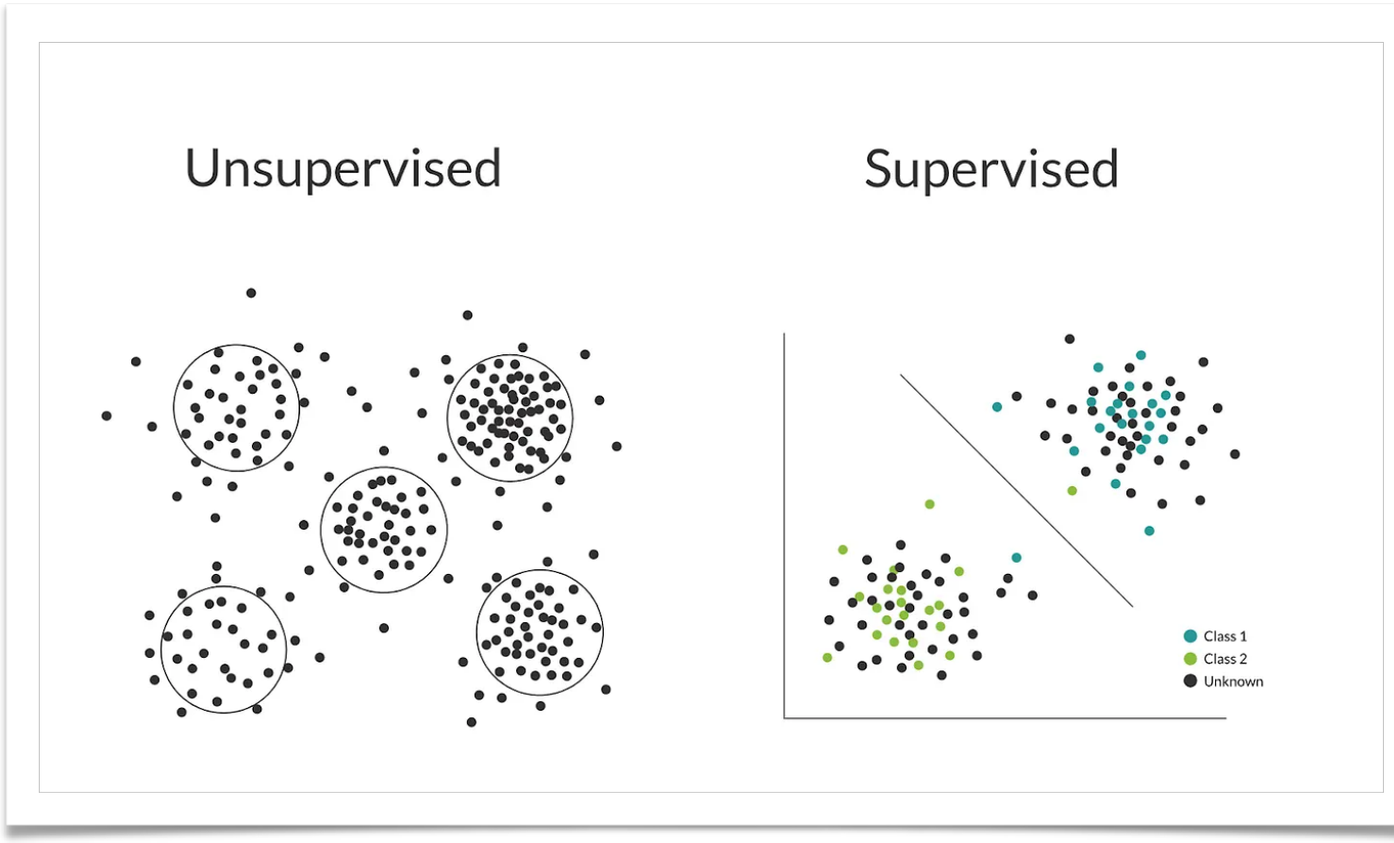
(Least squares)

Learning objective: $\arg\min_w \sum_{i=1}^{N} L(y_i, f(x_i \mid w)) = \arg\min_w \sum_{i=1}^{N} (y_i - w^\intercal x_i)^2$

Supervised & unsupervised learning

监督 & 无监督学习区别:
是否使用带标注的样本集?





直观理解

Supervised & unsupervised learning

粒子物理中的应用场景

Classification (分类问题):
信号-本底分类,
喷注识别
异常检测中的弱监督分类

Regression (回归问题):
喷注质量、动量回归,
中微子重建问题, …

Supervised

Multi-task learning (多任务学习)
e.g. 喷注基础模型 (foundation model for jets)

主要技术手段：
Decision trees (决策树)
Neural networks (神经网络)

Today’s focus!

Unsupervised

Generative tasks (生成式任务):
探测器响应的快速重建,
粒子重建算法,
…

Anomaly detection (异常检测):
Autoencoder用于探测反常事例,
数据质量实时监控,
…

Typical tasks of supervised machine learning

X

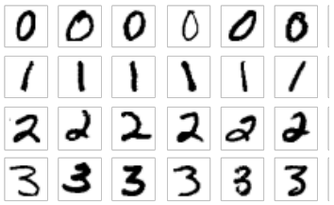
Model with learnable parameters

$$Y = f(X)$$

Y

分类

输入手写数字图像



0 1 ··· 0 0

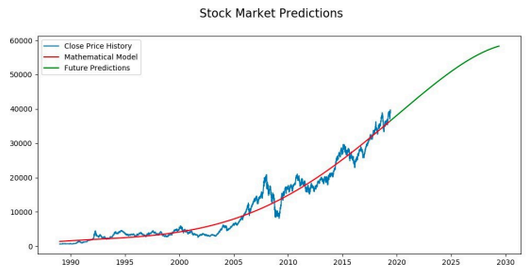
手写数字分类
10 种类别

输入图像

0 1 ··· 0 0

ImageNet 分类：
1000 种类别

回归



历史股价

嵌入到输入矢量

?

下一时刻股价预测

Machine learning: general pipelines

- Training dataset: $S = \{(x_1, y_1), \ldots, (x_N, y_N)\}$ where $x \in \mathbb{R}^D$ and $y \in \mathbb{R}$
- Model / hypothesis class: $f(x \mid w) = w^\intercal x$ (linear models)
- Loss function: $L(y, y') = (y - y')^2$ (squared loss)
- Optimization algorithm to minimize the learning objective:

$$\arg\min_w \sum_{i=1}^{N} L(y_i, f(x_i \mid w))$$

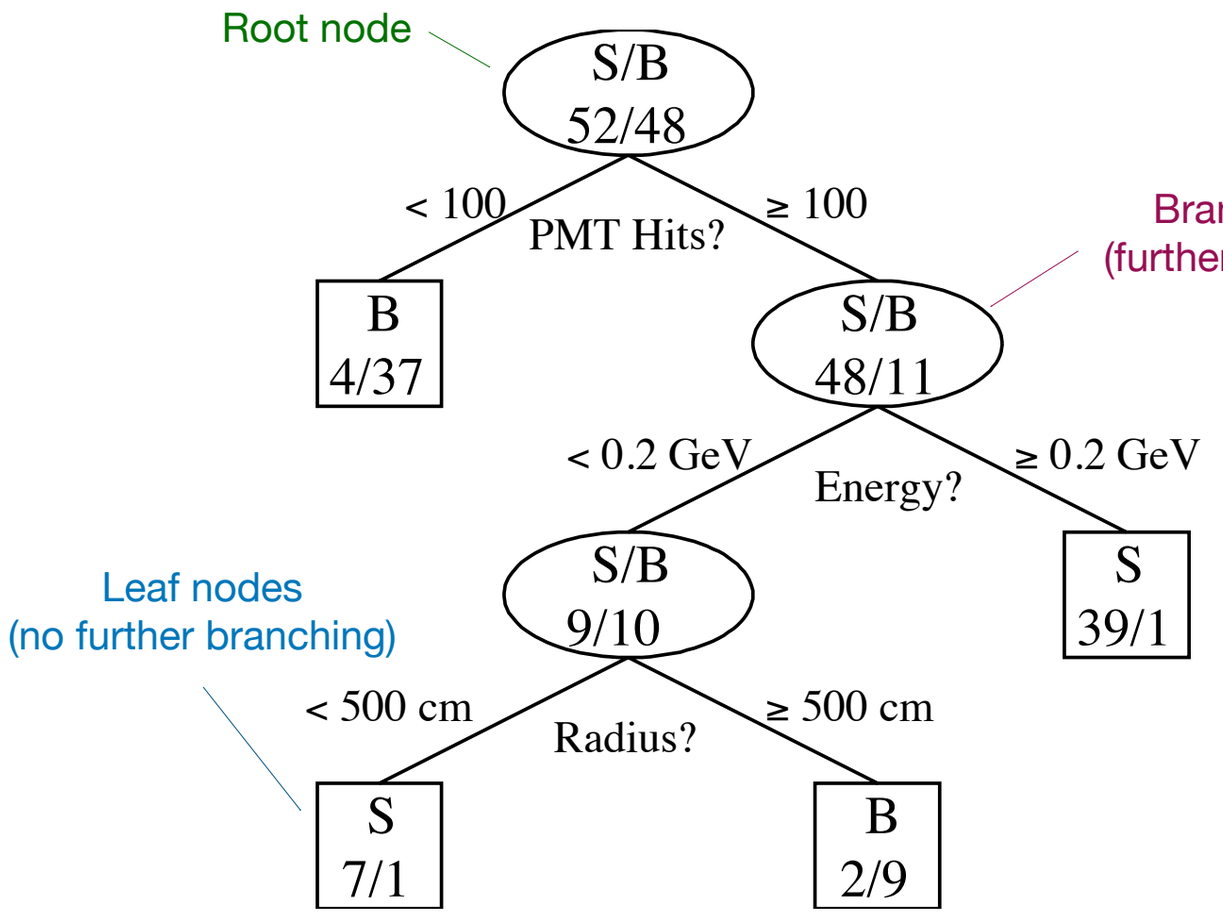
- Cross validation and model selection:

- Testing and deployment

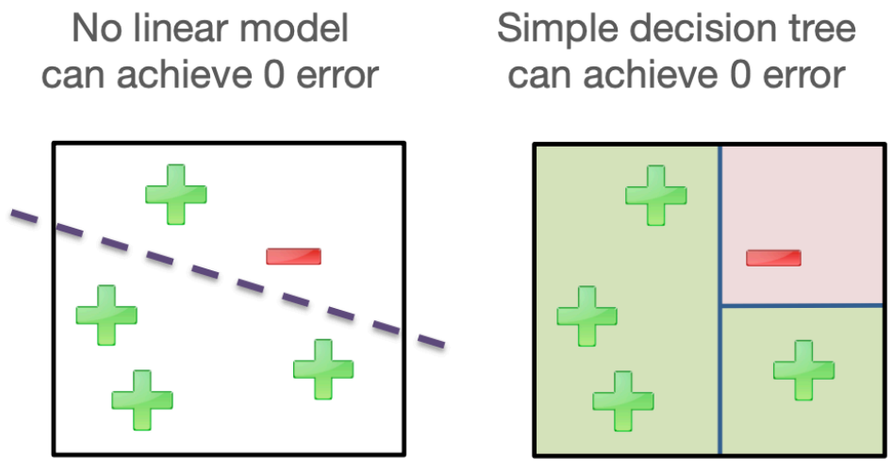
  Important: if a testing set is available, never use it to make decisions on the model!

验证集 (evaluation set) 决定选哪个模型，
测试集 (test set) 是最终部署用的数据集

Decision trees



用多个变量进行决策



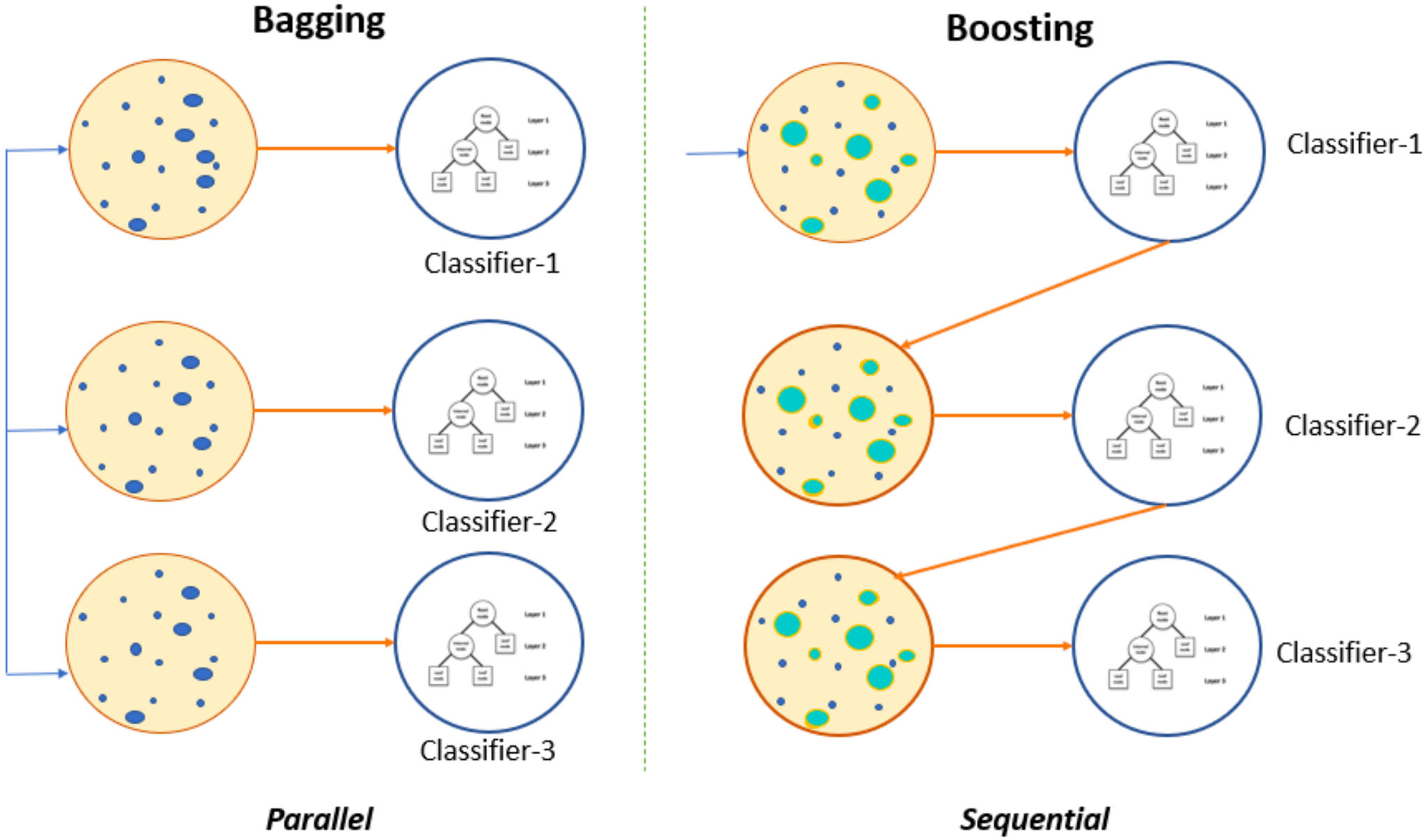
决策树：1) 非线性模型；2) 单个决策只用单一变量

Decision trees

用多棵“决策树”提升性能, 历史上的两种方法

Bagging:
选不同数据子集，训练多个模型

Boosting: (高能物理常用)
- 不断拟合新的决策树，作为之前决策残差
- 决策树的weight依赖于判断正确率



Boosted decision tree (BDT)

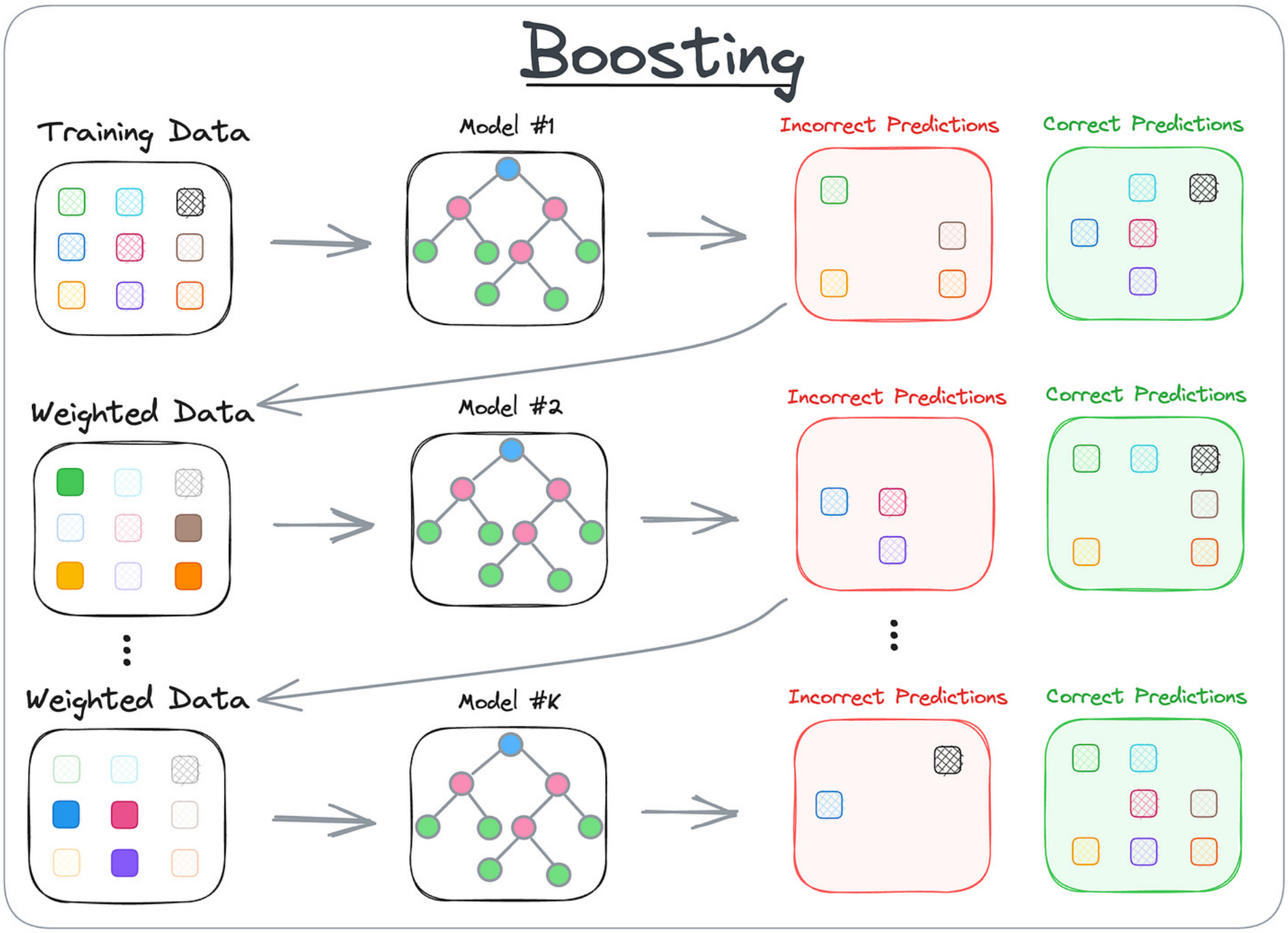
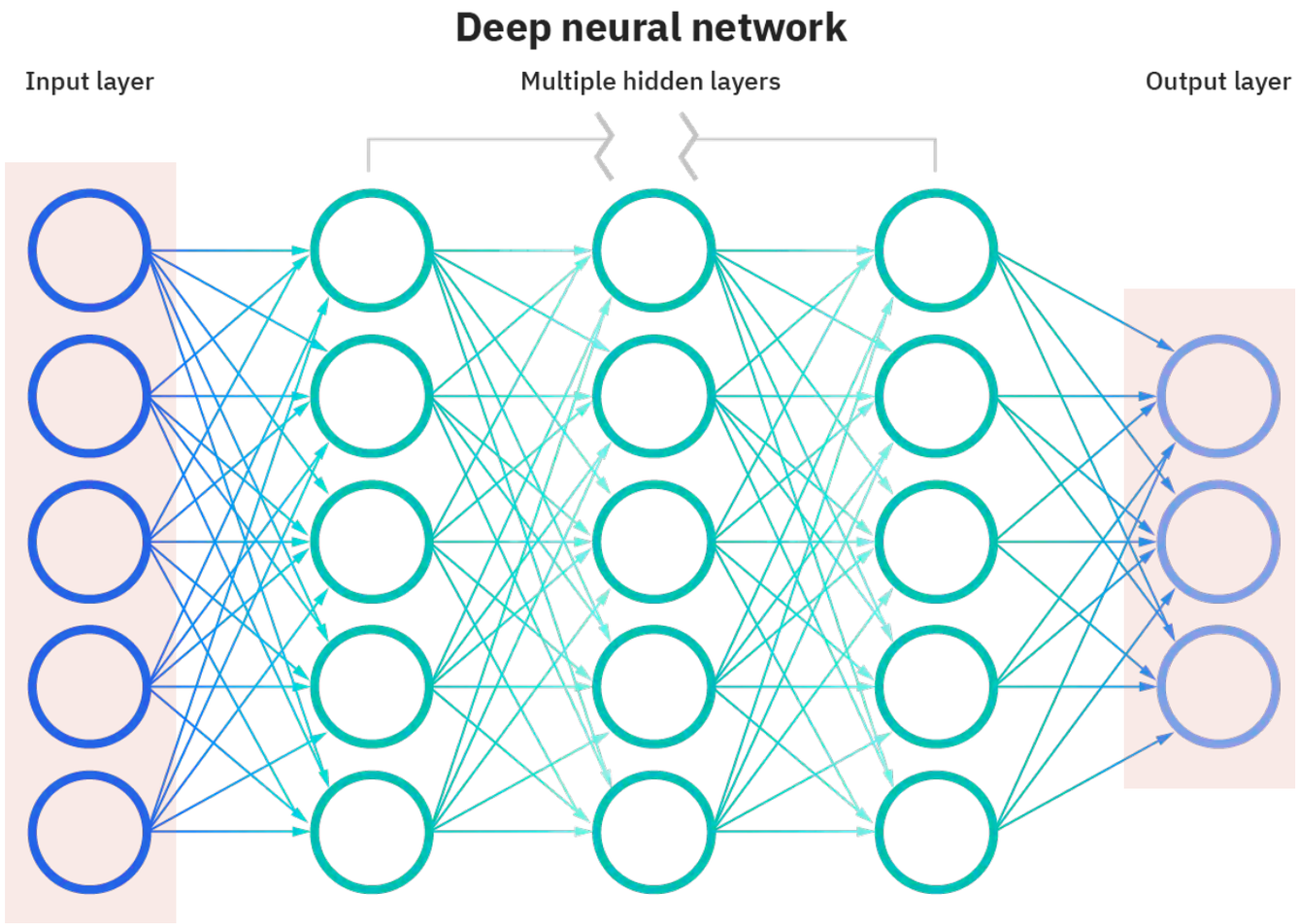


Image credit: link

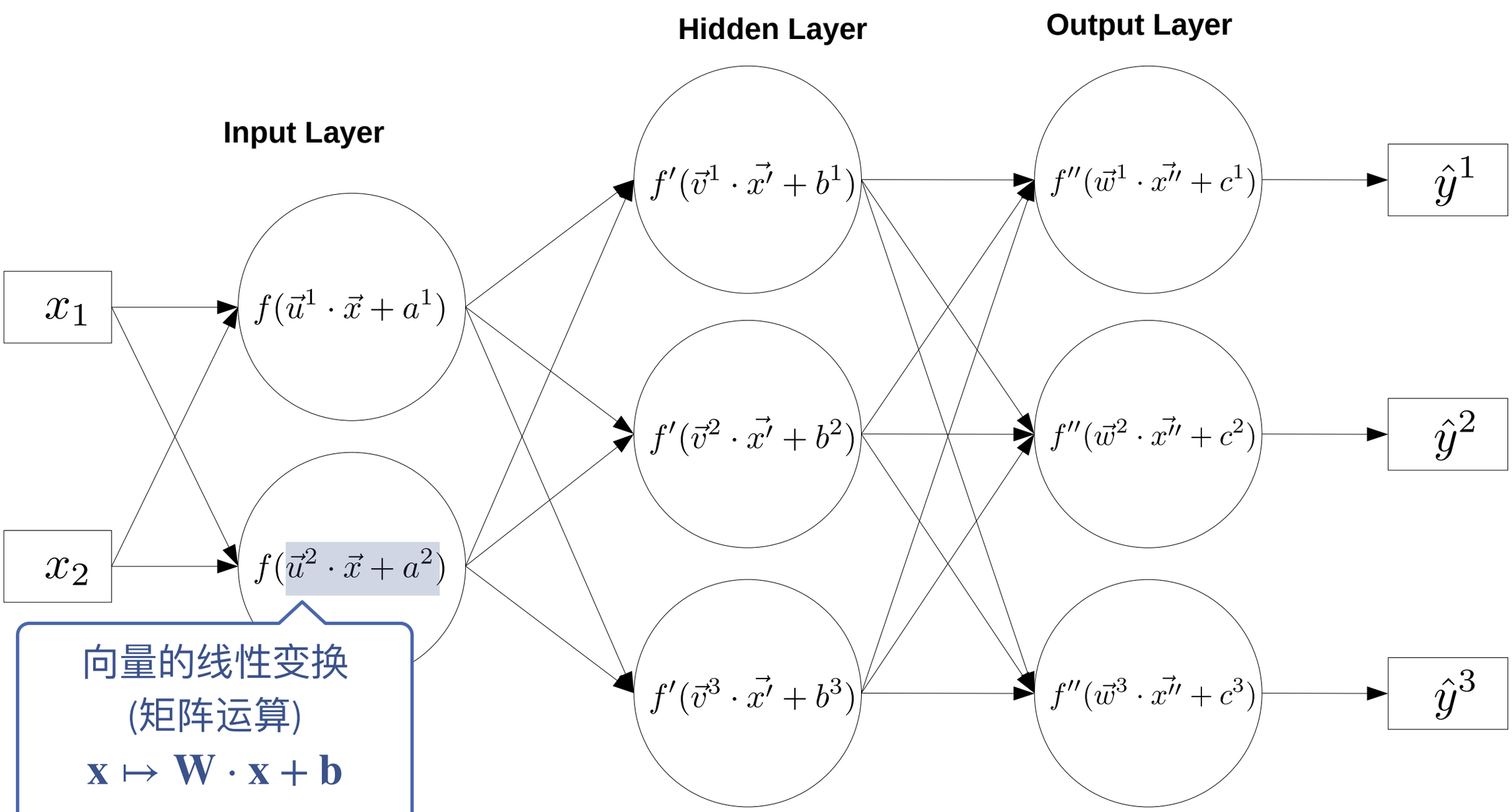
Deep neural networks



X $\quad Y = f(X) \quad$ Y

构成DNN函数 $(f(X))$的基础组件：
线性变换+非线性激活

Deep neural networks

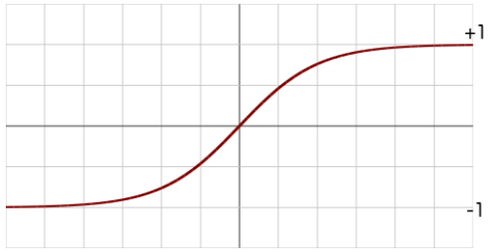


$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

Deep neural networks

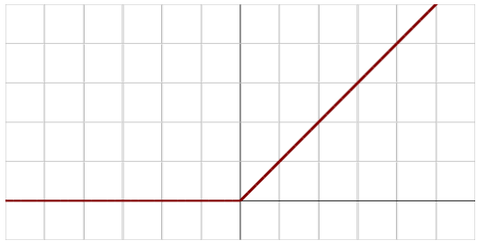
常用的几类非线性函数

Hyperbolic tangent



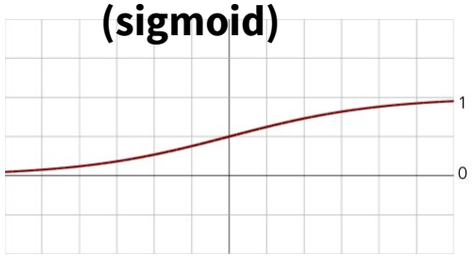
$$\hat{y} = \tanh x$$

Rectified **L**inear **U**nit (ReLU)

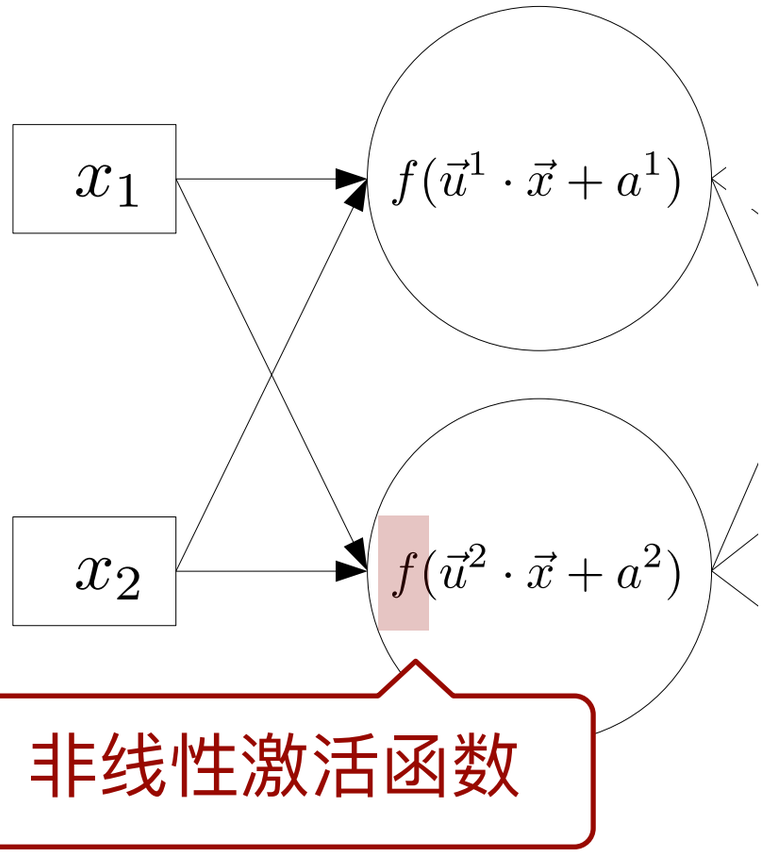


$$\hat{y} = \begin{cases} 0 & \text{if } x < 0 \\ x & \text{otherwise} \end{cases}$$

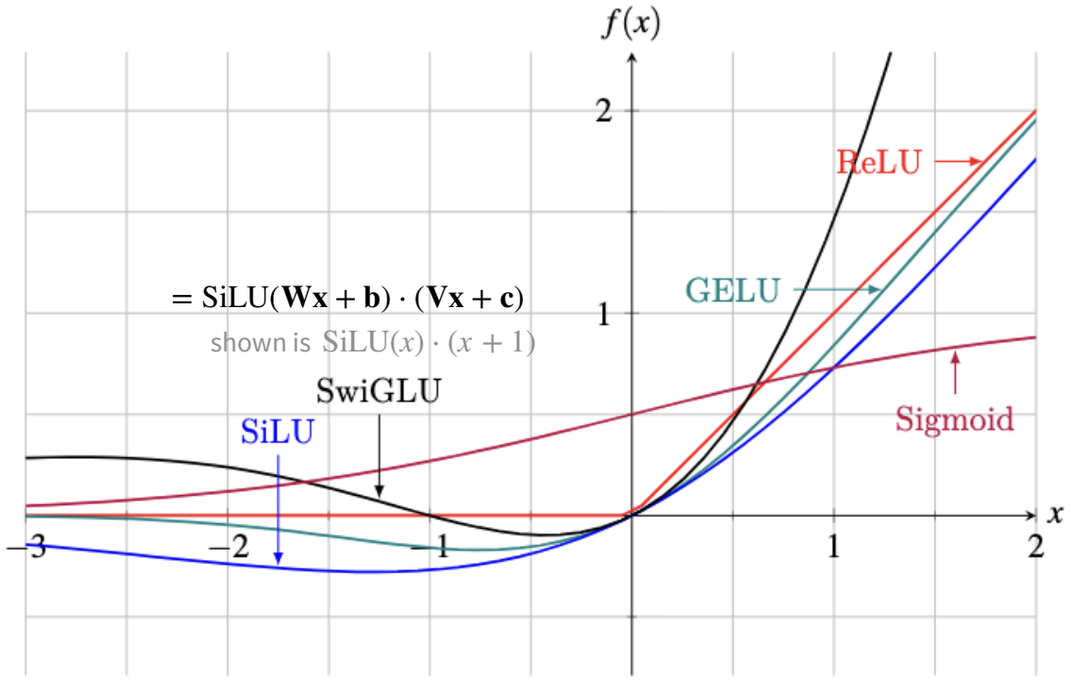
Logistic function **(sigmoid)**



$$\hat{y} = \frac{1}{1 + \exp(-x)}$$



现代架构常用的 ReLU 变体

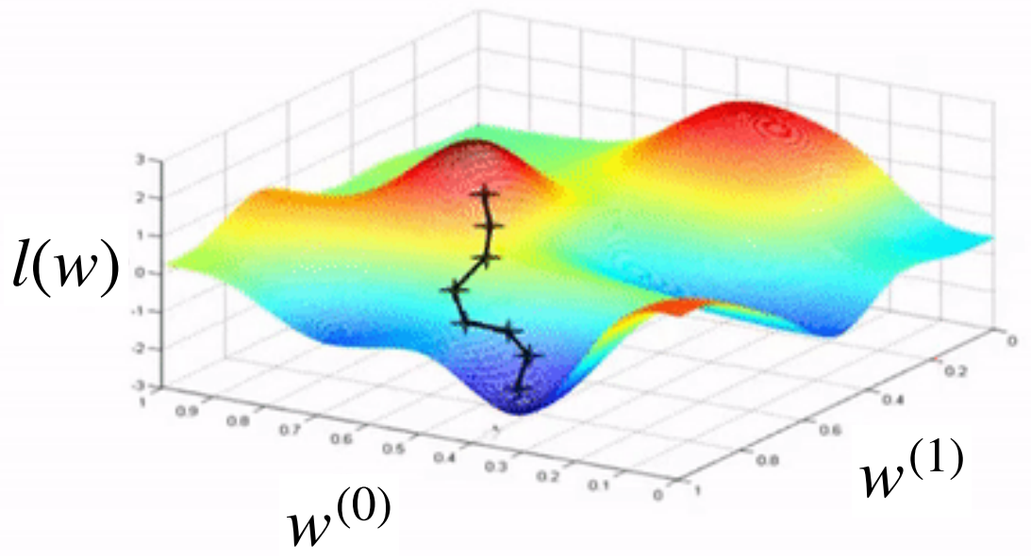


DNN optimization: gradient descent

- Set $w(t=0)$ to some values (e.g., $w(0)=0$ or some random value)
- At iteration t,
  - Compute the gradient $\nabla_w l(w(t))$: direction of steepest increase of $l(w)$ at $w(t)$
  - Take a small step in the opposite direction:

$$w(t+1) = w(t) - \eta \nabla_w l(w(t))$$

Step size / learning rate



Loss functions

- For classification: Use cross-entropy

$$p_i = p(y_i = 1|\mathbf{x}_i) = \sigma(h(\mathbf{x}_i))$$

$$L(\mathbf{w}, \mathbf{U}) = -\sum_i y_i \ln(p_i) + (1 - y_i) \ln(1 - p_i)$$

- For regression: Use squared error or something similar

$$L(\mathbf{w}, \mathbf{U}) = \frac{1}{2} \sum_i (y_i - h(\mathbf{x}_i))^2$$

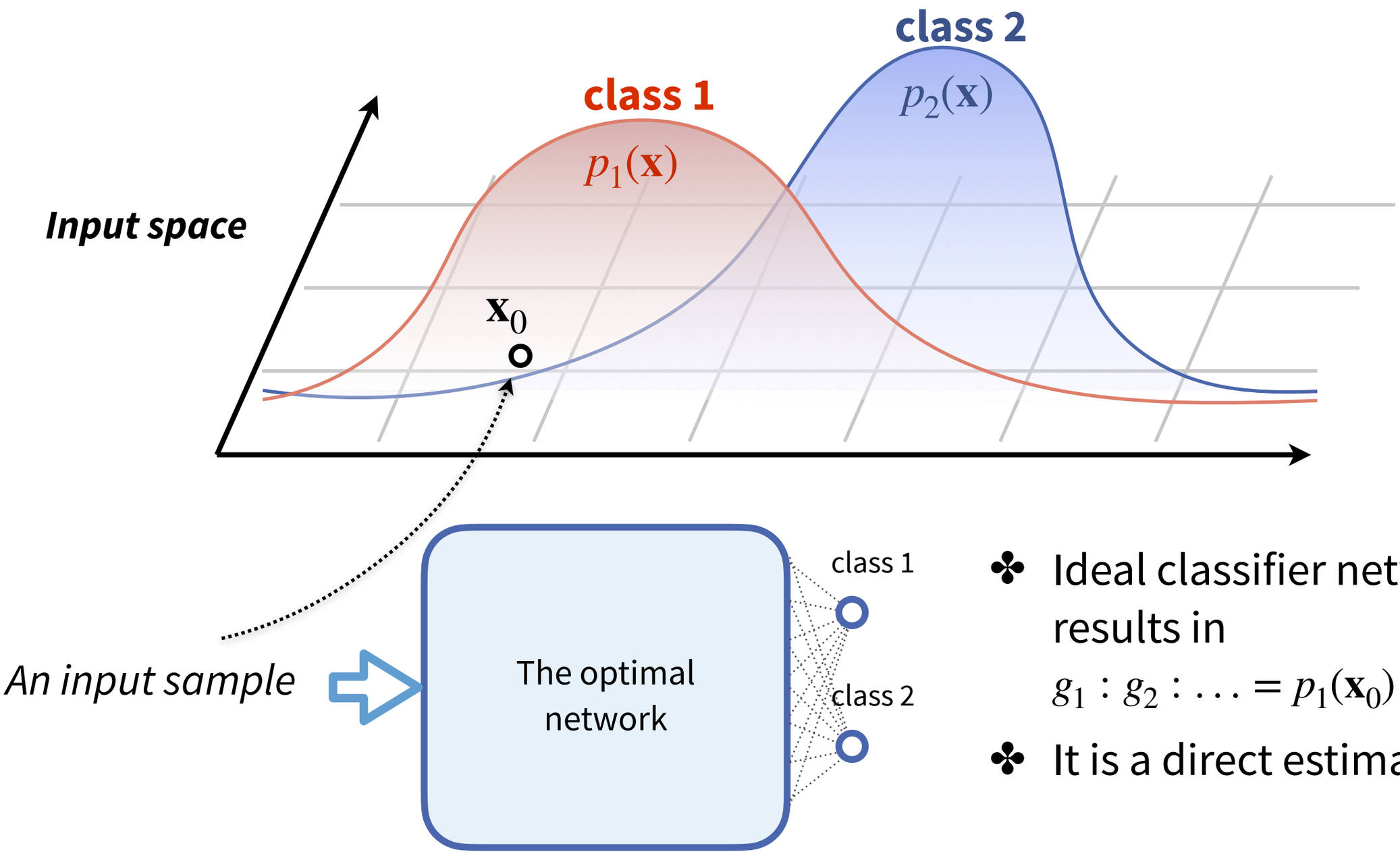
举例：

- 真实标签 $y = 0$ (本底)
  网络输出是信号的得分 0.2
  $L = -\ln(0.8)$
- 真是标签 $y = 1$ (信号)
  网络输出是信号的得分 0.9
  $L = -\ln(0.9)$

- 但为什么分类问题不用这种更简单的均方差 loss?

Classification with cross-entropy loss

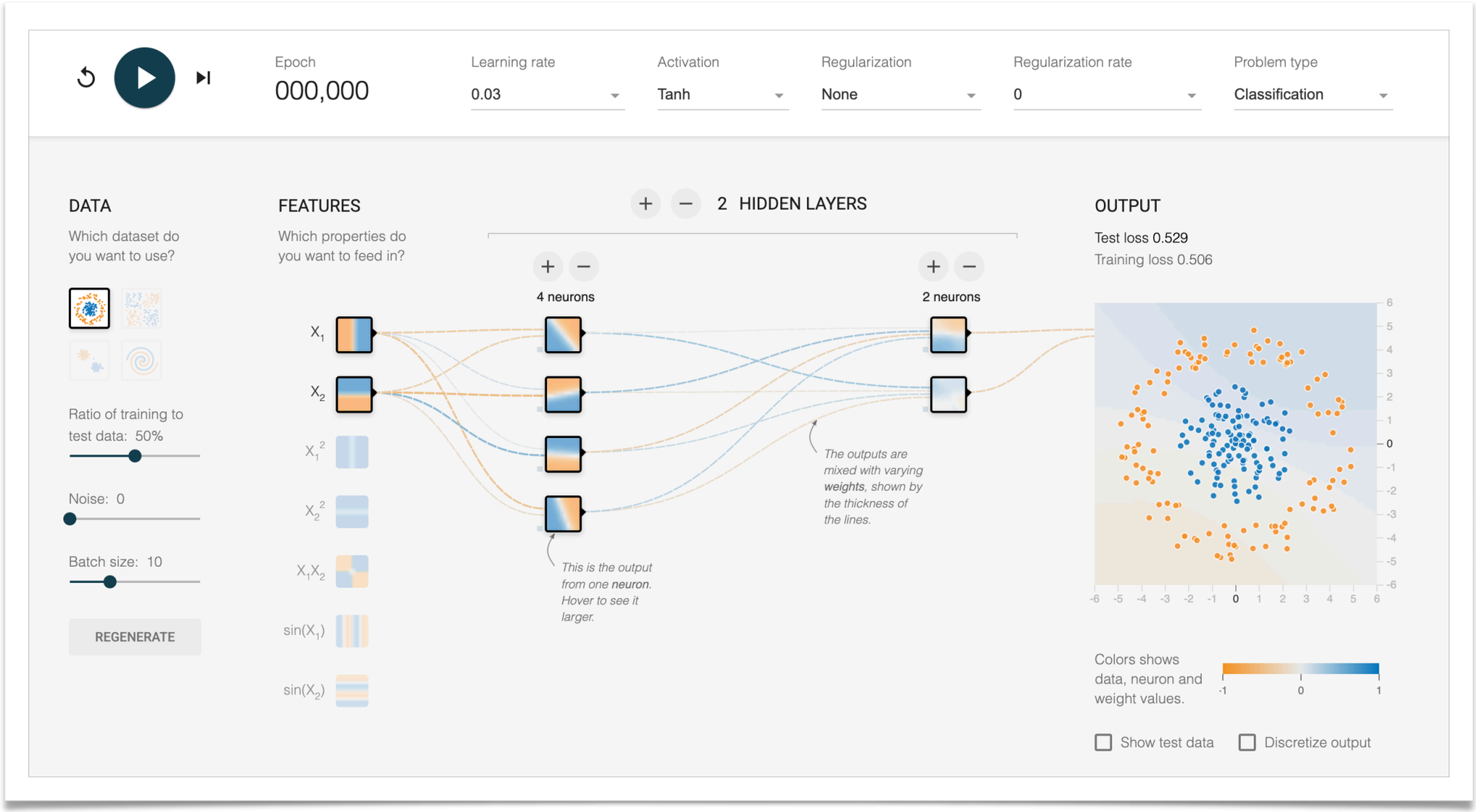
➔ Optimize **cross-entropy (交叉熵)** loss for classification tasks



- Ideal classifier network results in
  $g_1 : g_2 : \ldots = p_1(\mathbf{x}_0) : p_2(\mathbf{x}_0) : \ldots$
- It is a direct estimation of p

DNN optimization

➔ An intuitive example!



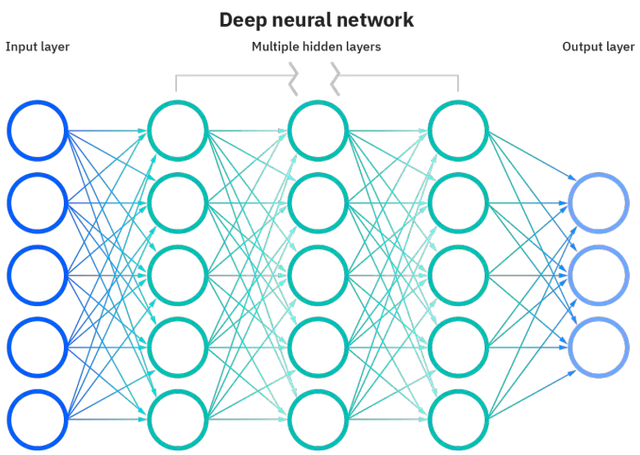
https://playground.tensorflow.org/

DNN optimization: input engineering

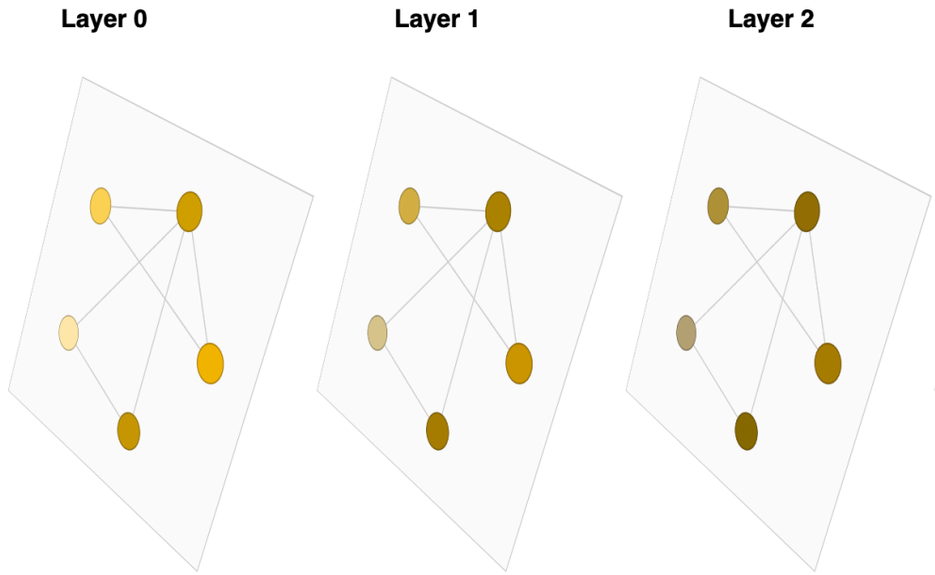
➔ 为了便于DNN优化，需要进行数据预处理：

- ❖ shift + scaling → 例如，使输入变量分布满足均值为0，方差为1
- ❖ 长尾分布 (e.g. jet p_T)，通过 log(|x|) 、log(|x|+1) 变换获得接近高斯的分布

➔ flattened 1D vector & tokenized input?



- 展开为1D vector 输入 DNN



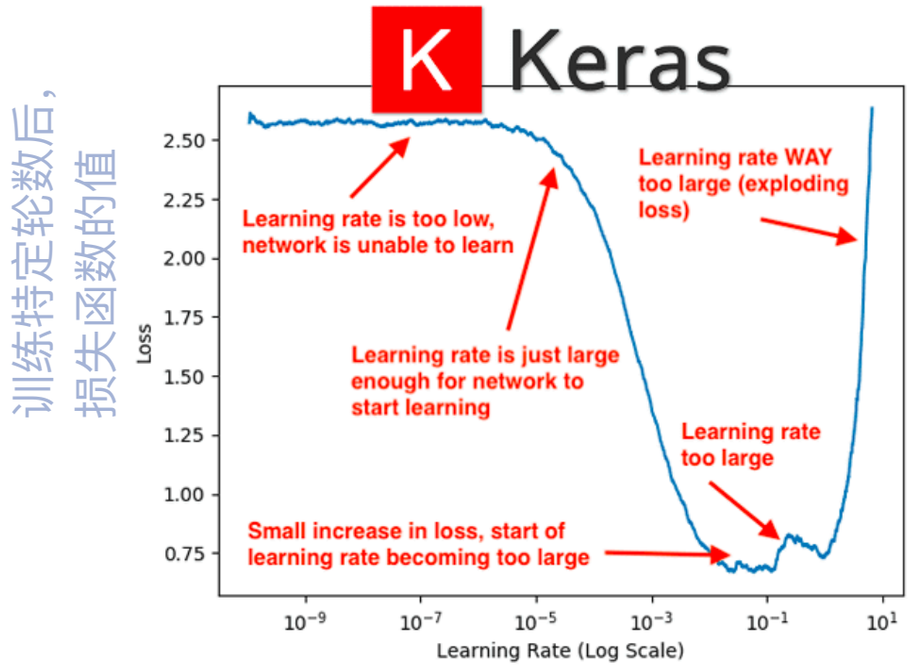
- 构造有层级的输入 (每个lepton / jet 等object当做一个 input node / token) → 尊重数据原本的属性
- 用更具表达性的 GNN / Transformer 来分析

见 hands-on 实例！

DNN optimization: hyperparameter tuning

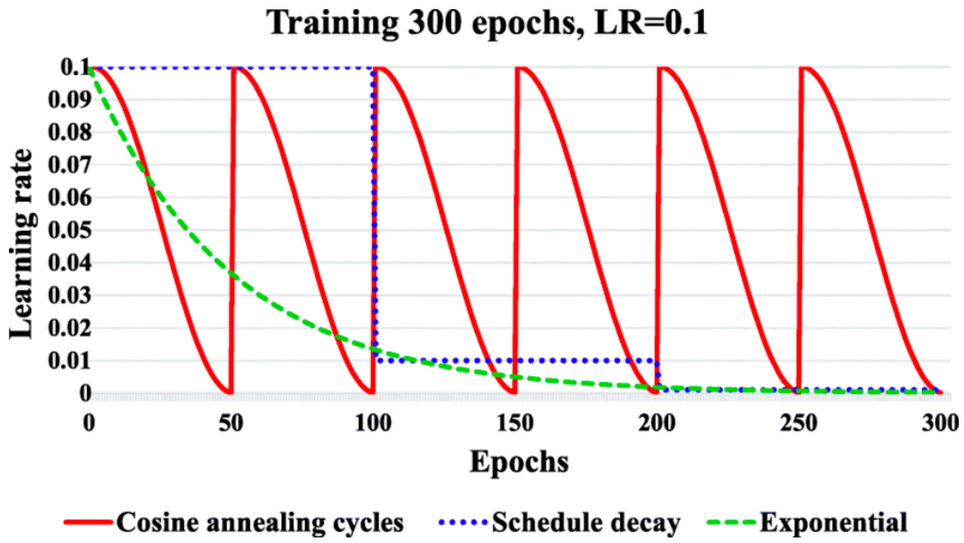
➔ 学习率 (learning rate)

- ❖ learning rate 是最重要的超参量
- ❖ 反映训练的每一步按照梯度更新参数的步长，可以以指数为跨度进行调节，最后挑选始终的值

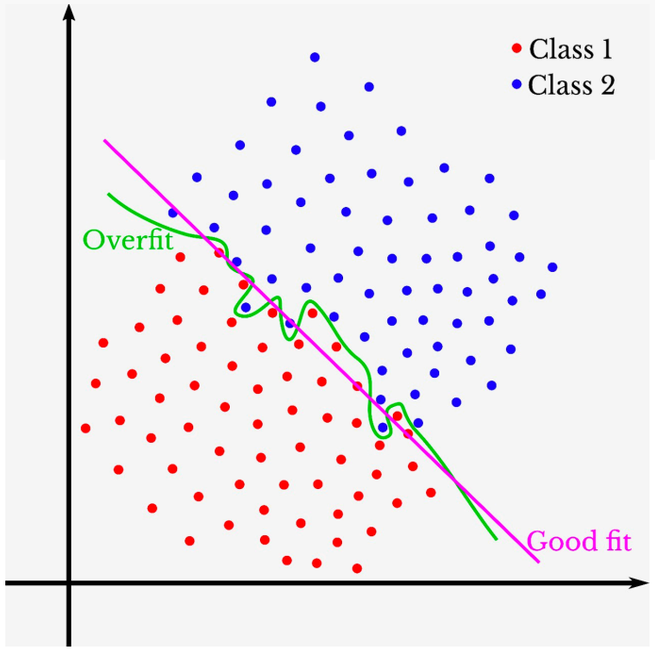


➔ 学习率的衰减 (decay rate)

- ❖ 学习率选用何种模式衰减？阶梯式衰减、指数衰减、余弦退火衰减，在不同的场景下各有应用
- ❖ 对于基础的神经网络，选择简单的阶梯式或指数衰减即可

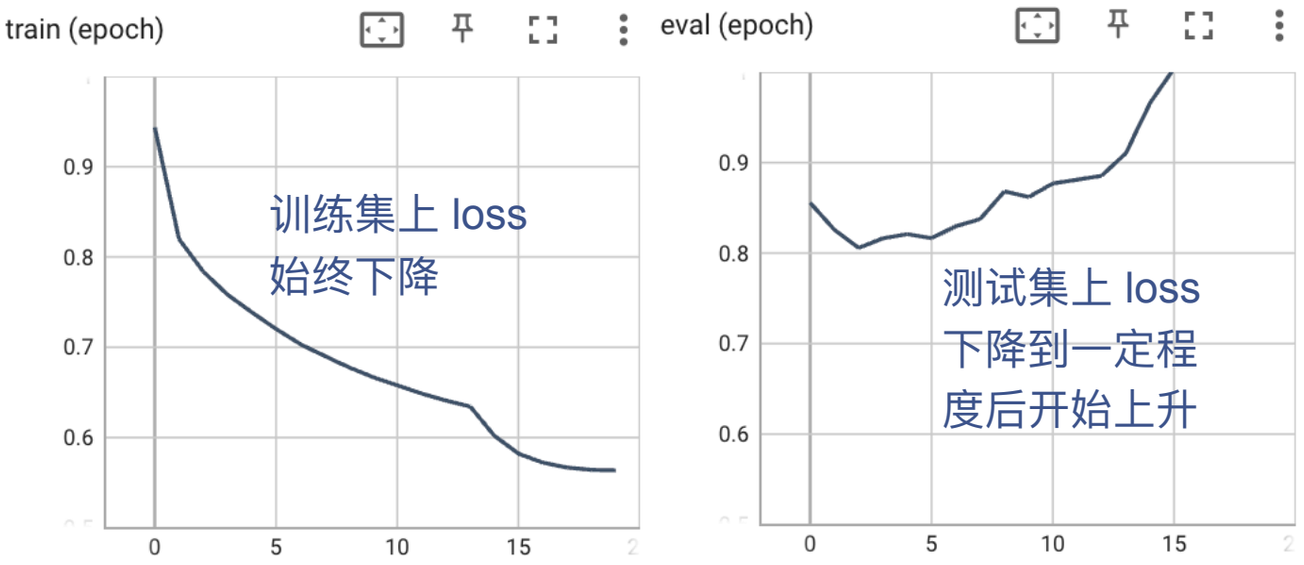


DNN optimization: overfitting



➔ 过拟合

- ❖ 表现：loss 在训练样本上始终下降，但每轮 epoch 结束后在测试样本上运行发现不再下降，甚至有所上升



- ❖ 原因：
  - 模型设计过大，有很多冗余参数
  - 训练的样本量太小，无法支持深层神经网络进行训练（经验上，NN 比 BDT 先天需更多样本）
- ❖ 改进方案：
  - 减小模型尺寸（如减少层数、减小每层的大小）
  - 加 dropout：一种训练技巧，每批训练时随机删除一些神经元
  - …

Let’s begin our hands-on session