

Lecture 1: 机器学习的目标和灵感 | Goals and Inspirations of ML

拉普拉斯决定论 | Démon de Laplace

我堂妹在科大上大一，我帮她辅导力学时候看到郑永令《力学》2.8节里的一段话，大意是

[拉普拉斯在他的《概率论》中写](#)：知道了初始条件和所受的力，可以预测未来的一切。

然而，后来大家发现，有三种现象很难预测：第一种来自于量子力学（海森堡不确定性原理），第二种来自于复杂系统（[More Is Different](#)），第三种来自于混沌（微分方程的不稳定性、三体、蝴蝶效应）。

说得很对。这三种不可预测性中：

1. 第三种最好解决，每隔一段时间测一下就行了。天气预报就是这么做的。
2. 第一种，量子力学的几种诠释（[哥本哈根](#)、[德布罗意-波姆](#)、[多世界](#)）还没分出胜负，Einstein和Bohr关于上帝扔不扔骰子的辩论还在继续（德布罗意-波姆属于隐变量理论，相信量子现象是由隐变量决定的，因此并非随机；大家可以思考一下此事和教材上所谓的“真随机数和伪随机数”的关系；“隐变量”这个词后来被引进了统计、机器学习、神经科学，指那些无法直接观测到的变量）。但还不紧急，因为这几种诠释都承认薛定谔方程，只不过它们对于薛定谔方程的诠释不同。所以：“Shut Up and Calculate.”。
3. 第二种最紧急。统计物理、生物神经网络、深度学习、遗传学、气候、经济、蛋白质结构、意识都属于它。第一，ChatGPT已经让部分人（包括Bengio和Hinton）感到恐惧，人类第一次造出了一个自己无法理解的并且才华横溢的机器；第二，自动驾驶正逐渐走进生活，但很多人不信任一个黑盒子来开车；最后，经济危机是人类社会的最大也是最紧急难题之一，我猜每个想让自己和家人过上更好生活的人都会渴望预测并阻止它。

复杂系统 | More Is Different

包括狄拉克和费曼在内的众多物理学家都相信——万物的规律可以化简为基本单元的规律。这种思想被称之为还原论，或者叫bottom-up。凝聚态和化学只是量子力学的小小应用，生物学是化学的应用，社会科学则是生物学的应用。还原论的思想深入每个物理学生的脑海：力学和电磁学中的微元，光学中的独立传播和叠加，电动力学里计算场的能量密度，量子力学中尝试构建势能，统计物理中建立宏观量和微观量之间的联系。

1972年，Philip Anderson写了一篇短文——More Is Different，提出了构建论。此文可以用一句话总结：

The ability to reduce everything to simple fundamental laws does not imply the ability to start from those laws and reconstruct the universe.

所以凝聚态化学不是应用量子力学，生物学不是应用化学，社会科学不是应用生物学。所以只要基本单元的数量足够多，就可以“涌现”出一些不一样的性质。（这就是More Is Different）

还原论和构建论是哲学上的倾向。每个人都有自己的倾向，我还是偏向还原论。

More Is Different

Broken symmetry and the nature of
the hierarchical structure of science.

P. W. Anderson

X	Y
solid state or many-body physics	elementary particle physics
chemistry	many-body physics
molecular biology	chemistry
cell biology	molecular biology
•	•
•	•
•	•
psychology	physiology
social sciences	psychology

[Anderson, 1972](#)

PS:

我希望大家能尊重其他的学科。

就我个人感受而言，和数学、物理学一样，生物学、经济学同样需要高度的直觉与洞察。

物理学家喜欢把一切都简化成数学公式（特别是微分方程），但是自然界的一切规律都可以用数学公式描述吗？我认为未必——有的可能只能用自然语言表述，有的可能是自然语言和数学语言都表述不了的。

达尔文 (Charles Darwin)、孟德尔 (Gregor Mendel)、马克思 (Karl Marx)、凯恩斯 (John Keynes) 所发现的规律，大多用自然语言表述，附带小学数学，但也很重要。从求索的角度看，他们对人类认识世界做出了极其重要的贡献；从应用和功利的角度看，没有前两者，人均寿命不可能七十多岁，没有后两者，人类的经济会每几十年就崩一次。

他们都不是物理学家。

机器学习的目标 | Goals of ML

机器学习的目标是预测复杂系统。

我们来看几个非常有名的数据集：

表1.5.1 数据集vs计算机内存和计算能力

年代	数据规模	内存	每秒浮点运算
1970	100 (鸢尾花卉)	1 KB	100 KF (Intel 8080)
1980	1 K (波士顿房价)	100 KB	1 MF (Intel 80186)
1990	10 K (光学字符识别)	10 MB	10 MF (Intel 80486)
2000	10 M (网页)	100 MB	1 GF (Intel Core)
2010	10 G (广告)	1 GB	1 TF (Nvidia C2050)
2020	1 T (社交网络)	100 GB	1 PF (Nvidia DGX-2)

Dive Into Deep Learning

上图把各个年代最经典的数据集拿了出来，以显示数据对于ML/DL的推动作用。

除了上图，还有一些非常有趣的数据集：

1. [鸢尾数据集](#)：你需要通过4个特征（花萼长度、花瓣长度等）预测鸢尾的种类。
2. [波士顿房价数据集](#)：你需要通过13个特征（犯罪率、房间个数等）预测房子的价格。
3. [MNIST数据集](#)：你需要通过来784个特征（每个像素的亮度）预测图片中是哪个数字。（CIFAR, ImageNet无非是更大的MNIST）
4. [威斯康星乳腺癌数据集](#)：你需要通过来30个特征（肿瘤的大小、对称性等）预测它是良性还是恶性。
5. [泰坦尼克号数据集](#)：你需要根据9个特征（性别、年龄、船票等级等）来预测乘客是否生还。
6. [红酒数据集](#)：你需要根据11个特征（酒精含量、pH等）来预测红酒的品质（0-10分）
7. [皮马印第安人糖尿病数据集](#)：你需要根据8个特征（血糖含量、年龄、BMI等）来预测患者是否患有糖尿病。
8. [腰间盘突出数据集](#)：不要久坐，下课多动动腰，腰间盘突出是不可逆的。

仔细想一想，这些问题能否用微分方程来预测，就像牛顿、麦克斯韦、薛定谔方程那样？答案显然是否定的。一个肿瘤是良性还是恶性，影响它的因素太多太多；一个房子的价格是高是低，更不必说。

微分方程代表的是因果性，而在机器学习中，很少能找到因果性，通常只能找到相关性。[相关不一定蕴含因果](#)。

机器学习的动机 | Motivations of ML

机器学习的动机是分析、解释、预测复杂系统

机器学习的动机最像统计学。统计学家常说：[统计学是人类（面对复杂系统时）无能为力下的努力](#)。统计学从一开始就和遗传学、生物学、医学有千丝万缕的联系，很有名的辩论是吸烟和肺癌间的相关性到底是否蕴含因果，影响肺癌的因素同样太多太多。如果想说服大家吸烟确实能导致肺癌，就需要用实验证明烟草中的哪种物质引起了哪种小分子/蛋白质/细胞的变化，该分子的变化又导致了另一种小分子/DNA/蛋白质/细胞的变化，最后导致癌变。这条因果链非常难找，[这里](#)有一个尝试。

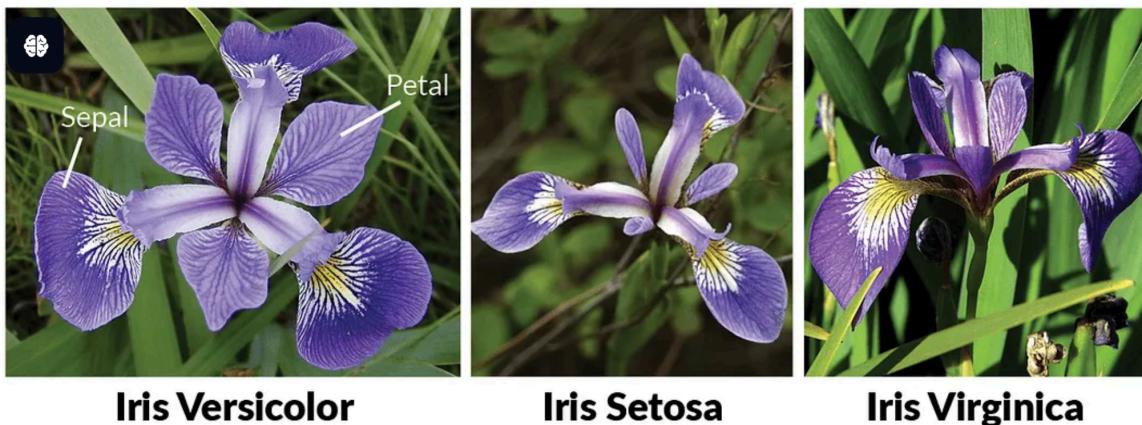
总而言之，机器学习和统计学都试图理解、分析、预测复杂系统。

机器学习的灵感 | Inspirations of ML

机器学习的灵感大多来自统计。学者们很喜欢给自己的算法找一个概率解释（逻辑回归、决策树、神经网络），也很喜欢从优化那里借用方法（梯度下降、惩罚项）。

一个例子 | An Example of ML

我们玩一下最简单的鸢尾数据集。鸢尾是一种美丽的植物。这里，我们要用Sepal Width, Sepal Length, Petal Width, Petal Length对于三种鸢尾进行分类。



[图片来自网络](#)

简单起见，我们只用Sepal Width和Sepal Length对前两种鸢尾进行分类。

以下是用Linear Regression (即最小二乘)去预测鸢尾数据集的代码：

```
# Load the Iris dataset
iris = datasets.load_iris()

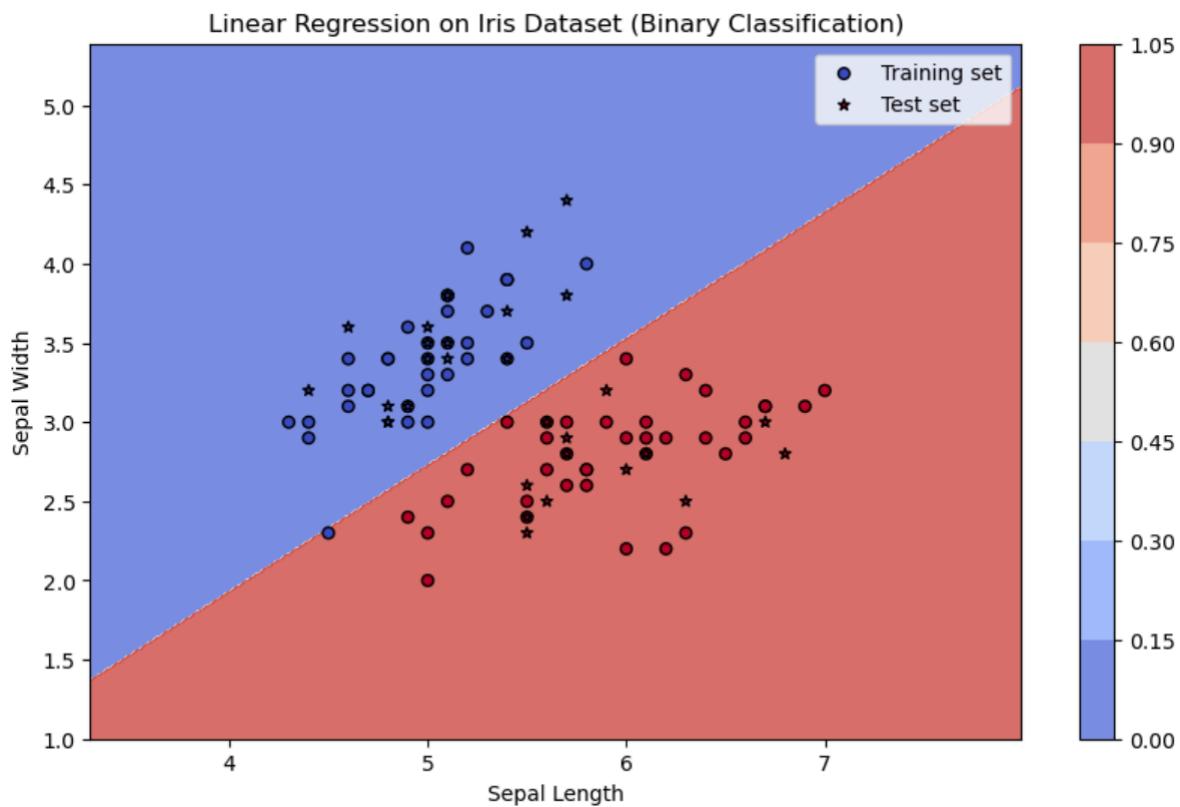
# Choose only two classes (e.g., Setosa and Versicolor)
X = iris.data[iris.target != 2, :2] # Use only the first two features (sepal
length, sepal width) for simplicity
y = iris.target[iris.target != 2] # Keep only Setosa (0) and versicolor (1)

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)

# Create and train the linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Predict on the train and test set
y_train_pred = model.predict(X_train)
y_test_pred = model.predict(X_test)
```

画个图



机器学习的流程就如上，其他的无非：

1. 换用其他的模型：逻辑回归、SVM、决策树、随机森林、贝叶斯。（换模型若用库函数只需要改一行代码）
2. 更好的划分训练集、验证集、测试集的方法：交叉验证、k折交叉验证。
3. 更好的寻找超参的方法：网格搜索。
4. 为了避免过拟合：惩罚项。
5. 为了利用不同模型之优点：集成学习。
6. 寻找更好的评价指标：Accuracy, Mean Square Error, Precision & Recall.
7. 当特征太多时（100、1000、10000个）：特征选择、降维。
8. 数据预处理：min-max归一化、中心化、z-score归一化。

另一个例子 | Another Example of ML

请参考 [Titanic.ipynb](#)。

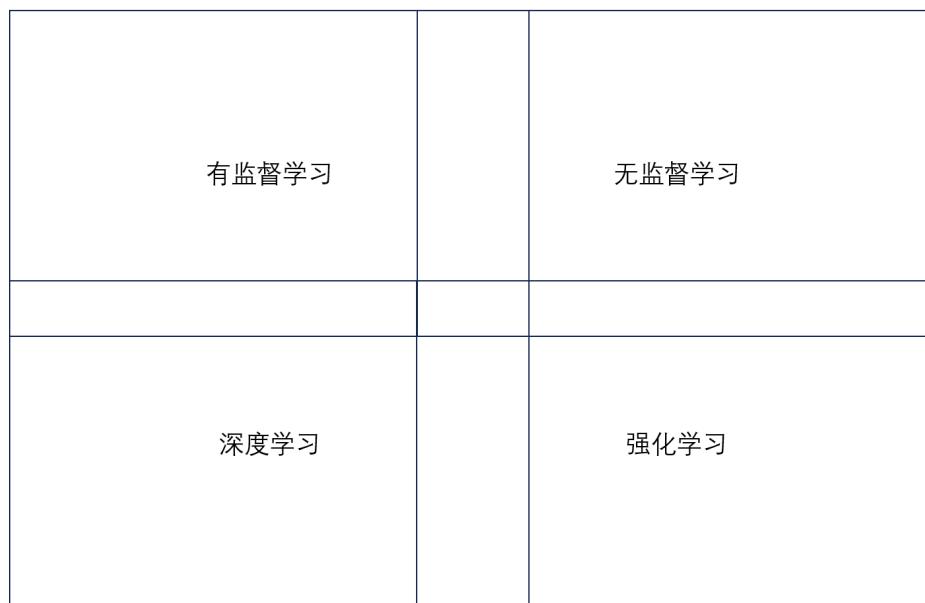


[图片来自网络](#)

机器学习的分类 | Classification of ML

1. 有监督学习：需要有标签数据的机器学习算法。比如线性回归、SVM、决策树用于鸢尾数据集。
2. 无监督学习：不需要有标签数据的机器学习算法。比如聚类和降维。
3. 深度学习：涉及人工神经网络的机器学习算法。Multi Layer Perceptron和CNN用于MNIST
4. 强化学习：涉及奖赏的机器学习算法。Markov Decision Process用于下跳棋。

这四种是相互之间有重叠的。若画成Venn图，如下：



交叉甚多，比如

1. 自回归LLM算是有监督学习和无监督学习的交集，它巧妙地用之前的词预测下一个词，所以把互联网上大量无标签的文本都可以利用。
2. AlphaGo是深度学习和强化学习的交集。

3. AlexNet是深度学习和有监督学习的交集。
4. ChatGPT是四者的交集。它用了自回归LLM和RLHF。

P.S.: 大家不要死扣字眼。这些词不重要，它们到底干了啥更重要。

问答 | QA

Q: 什么问题不适合机器学习？

A:

1. 预测木星的轨道：只要把当前时刻木星的 \vec{r} 和 \vec{v} 代入牛顿方程，我们可以预测几百年后某一时刻木星的 \vec{r} 和 \vec{v} 。这种“简单”的问题不需要用机器学习。
2. 预测三体问题：三体问题属于上文所提到的混沌，对初值有敏感性。用数学语言说，当初值改变 ϵ 时， t 时间后的解不能被 δ 控制住。[严格定义见此](#)。机器学习对长期预测混沌系统也无能为力，我们只能每隔一段时间测一下。（有些工作在用机器学习预测天气，这是因为天气同时也是复杂系统）

Q: 我是物理学家，我为什么要用机器学习？

A: 为了对付复杂系统。上述预测木星轨道的问题，物理学方法当然更优越。但是，预测蛋白质结构，用物理学方法和机器学习都可以，历史上两者也互有胜负，但现在后者超过了前者。

Q: 除了机器学习外，还有其他实现人工智能的方法吗？

A: 有。

人们很难给学习下一个定义。一种定义是：Learning is a process where a system improves performance from experience（学习是一个系统从经验中提高表现的过程）。如果这样定义，那么以下是和机器学习方法非常对立的工作，**因为它们不需要或者几乎不需要数据来训练**：

1. 1955, Allen Newell & Herbert Simon, [Logic Theorist](#)
 1. 目标：证明数学定理。
 2. 方法：Logic Theorist 使用了一套符号规则和搜索算法来探索证明路径。
 3. 成就：它证明了罗素[《数学原理》](#)中52条定理中的38个。
2. 1965, Edward Feigenbaum & Joshua Lederberg, [DENDRAL](#)
 1. 目标：预测有机物结构。
 2. 方法：内置化学知识、符号推理
 3. 成就：DENDRAL 是第一款专家系统
3. 1982, David Marr, [Vision](#)
 1. 目标：解释视觉。
 2. 方法：原始图片->初草图->2.5D草图->3D建模。
 3. 成就：计算机视觉的最高奖叫Marr奖
4. 1997, Deep Blue
 1. 目标：（在国际象棋上）超越人类。

2. 方法：内置知识为主、机器学习为辅

3. 成就：击败卡斯帕罗夫

现在回看，用数据训练的模式取得了远远超过前人的成功。但三十年河东，三十年河西，也许下一轮变革要从符号规则中借鉴。

Q: 那机器学习取得过哪些比肩或超越以上三者的成就？

A:

1. 1959, 西洋跳棋

1. 目标：（在西洋象棋上）超越人类

2. 方法：强化学习、Markov Decision Process

3. 成就：在整个CS的历史上都有震撼性——因为在那之前人们一直认为一个程序只能按照预设的指令形式，既完全否定了学习的可能。

2. 1957, 感知机

1. 目标：模拟人脑

2. 方法：把神经元建模成节点、把突出建模成节点之间的连线

3. 成就：让很多人认为我们马上就要得到通用人工智能，深度学习的第一次崛起。

3. 1982, Hopfield Network

1. 目标：解释记忆

2. 方法：类似Ising Model

3. 成就：在旅行商问题上得到了当时最好的结果，深度学习的第二次崛起。

4. 2012, AlexNet

1. 目标：图片语义识别

2. 方法：CNN

3. 成就：在ImageNet竞赛上远远甩开往年冠军和当年亚军，深度学习的第三次崛起。

5. 2022, ChatGPT

1. 目标：通用人工智能

2. 方法：Transformer + RLHF

3. 成就：自不必说。

后四者代表着DL的四次崛起。另外，SVM、决策树、K-Means、Boost、Hidden Markov Model、Bayes、MCMC也都取得过不错的成就。

Q: 有关理论机器学习，我需要知道哪些？

A: 以下概念至少要了解：

1. 没有免费的午餐: 若考虑所有潜在的问题，则所有学习算法都一样好。（因此你要根据你的问题选择合适的算法。）

2. 万能近似定理: 两层神经网络可以近似任何函数。

3. VC Dimension: 算法能“打散”的点集的势的最大值。可以用它描述模型能力。

Q: 为什么说DL像炼丹?

A: 俩原因

1. DL需要不断调参，有很大运气成分。
2. 炼丹需要炼丹炉，DL需要显卡，两者都是黑的，而且都是热的。

但是吧，人类发明造纸术、蒸汽机、电灯、飞机的时候，调参、试错也很多，也有很大运气成分，也很凭借经验。工程问题都是这样。

Q: 既然DL中每个参数都是知道的，那明明是白盒子，为什么说DL是黑盒子?

A: 这个问题很重要，自己思考下。

Lecture 2 & 3: 机器学习的常用概念 | Basic Concepts in ML

分类和回归 | Classification and Regression

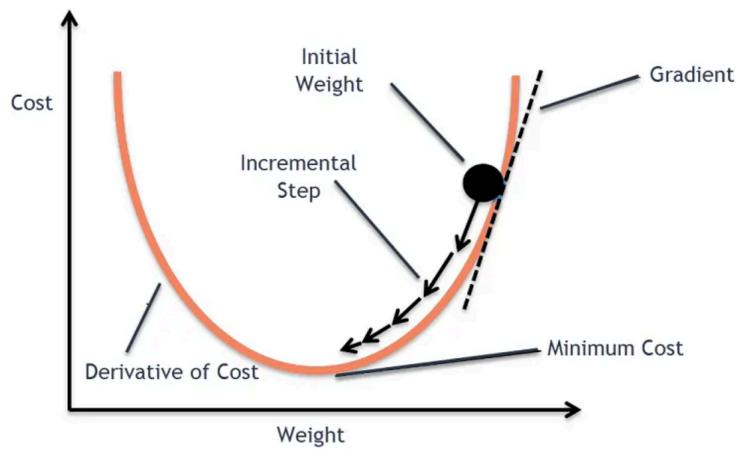
分类的标签是离散的；回归的标签是连续的。

梯度下降 | Gradient Descent

目标：最小化目标函数，并找到使其最小的参数。

灵感：一个球在山上滚来滚去。。。

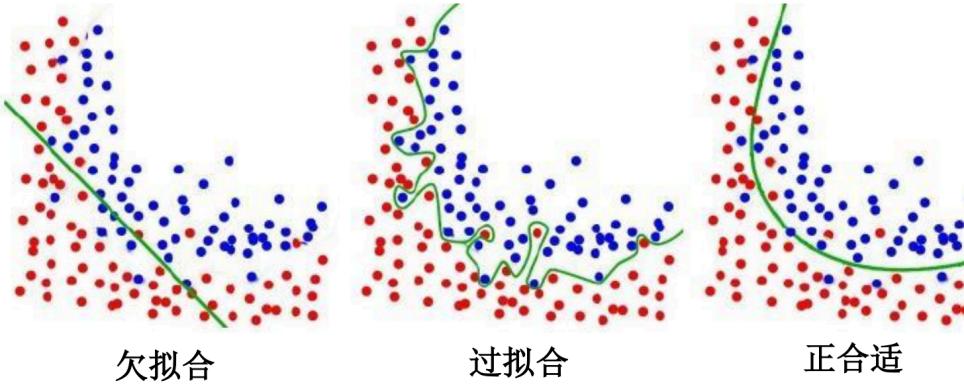
图像：能看懂下图就可以了。



[图片来自网络](#)

过拟合 | Overfit

图像：能看懂下图就可以了。



图片来自老师的幻灯片

防止过拟合的方法：各个算法均有，我只介绍一些最奇葩的

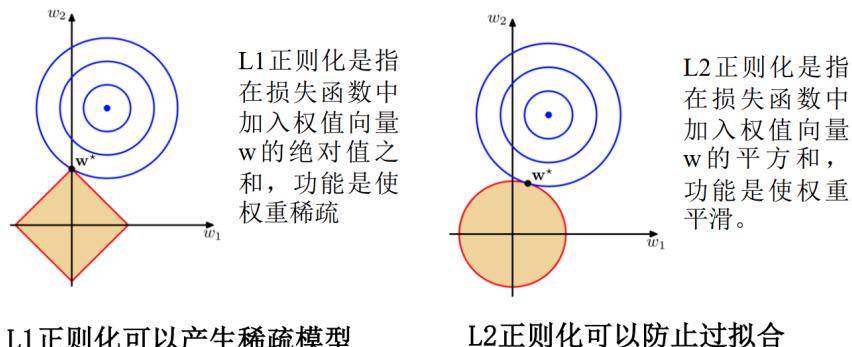
1. 后剪枝：决策树训好后，剪掉一部分枝叶。
2. Drop-out：神经网络训好后，随机扔掉一部分神经元。（我见过的最奇葩的算法之一）

正则化 | Regularization

目标：防止过拟合。

灵感：来自凸优化。统计学家经常玩不同的惩罚项。

图像：能看懂下图就可以了。（图片来自老师的幻灯片）



图片来自老师的幻灯片

评价指标 | Performance

目标：评价算法好坏

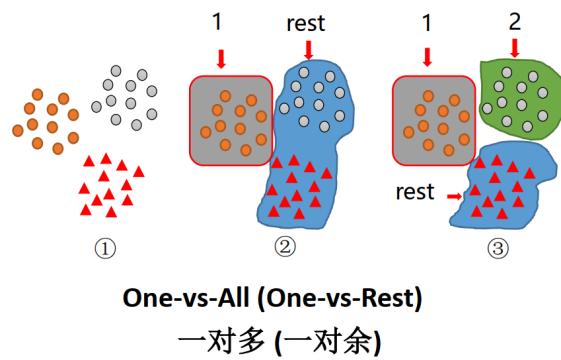
灵感：来自统计学。统计学家经常玩不同评价指标。

常用指标

1. 回归：Mean Square Error.
2. 分类：Accuracy; Precision, Recall.（后两个很重要，可以查查看）

多分类 | Multi-Class

图像：



图片来自老师的幻灯片

有三种方法：

1. One vs One
2. One vs Rest
3. Many vs Many

验证集 | Validation Set

目标：寻找超参。

灵感：以高考为例，训练集 (Training Set) 类似平时刷题，验证集类似模拟考试，训练集 (Test Set) 类似高考。

注：是否需要验证集，取决于你的算法是否需要调整超参数。

思考题：请大家思考一下——测试集是不是只能用一次？

泄露测试集 | Test Set Leakage

一种极其阴险的学术造假

简介：

机器学习里常见的造假方式是——故意把测试集泄漏到训练集和验证集里。这类似于，把高考题泄漏到平时刷题和模拟考试里。

学术造假图景 | The Big Picture of Scientific Misconduct:

我所知的，物理、生物、机器学习里的造假方式，按隐晦程度从低到高，有四种

1. 改数据 | Data Fabrication
2. 挑数据 | Cherry-Picking
3. p值操纵 | p-hacking
4. 泄露测试集 | Test Data Leakage

应对方法：

1. 对前三者的方法是——重复实验。
 2. 对第四者的方法是——构造新测试集。
-

来自ChatGPT-o1的详细介绍：

下面给你提供一段简要的说明，既包含了前面提到的四类常见造假方式及其隐蔽程度，也补充了为何“泄露测试集”不像前面三种那样能通过“重复实验”来识别，以及相对应的解决方案。你可以在此基础上自行调整措辞和篇幅。

常见的学术造假方式及其应对方法

在学术研究中，以下四类造假方式都可能给研究结果和学术共同体带来严重的危害。它们的隐蔽程度从低到高，大致可以分为：

1. 改数据 (Data Fabrication)

- **方式**：直接编造、篡改、伪造实验或观测数据。
- **危害**：最明显且严重，可完全扭曲结论。
- **应对**：通过重复实验、复核原始记录和数据来源等方式，可以相对容易地发现。

2. 挑数据 (Cherry-Picking)

- **方式**：只选择对自己有利的结果或符合预期的样本，忽略“不理想”或不支持假设的部分。
- **危害**：研究结论缺乏客观性，可能造成误导。
- **应对**：通过复核实验设计和完整的数据集，或由他人重复实验，可以查出被省略或忽视的数据点。

3. p值操纵 (p-hacking)

- **方式**：通过反复尝试不同分析方法、收集/剔除数据点、反复修饰统计模型，从而得到“显著性”结果。
- **危害**：结果看似“有意义”，但实际上缺乏真实统计依据。
- **应对**：重复实验、预先注册研究计划 (preregistration)，以及审阅详细的分析过程有助于避免或发现此类问题。

4. 泄露测试集 (Test Data Leakage)

- **方式**：在机器学习/深度学习中，将原本用于最终评估模型性能的测试数据 (test set) 故意或无意地泄露到训练和验证环节，从而使模型“见过”测试数据。
- **隐蔽性**：比 p-hacking 更难发现，因为只要还是使用同一批泄露的测试数据进行验证，就会得到“漂亮但虚假”的成绩。
- **危害**：在算法研究或比赛场景中，会极大地夸大模型的实际能力，误导后续应用和学术引用。
- **应对**：单纯“重复实验”并不能起到防范作用，因为如果使用同样的测试集，结果依然会被“复刻”。有效的做法是**构造全新的、没有被泄露的测试集**，并在真正的盲测环境下重新评估模型性能。

结论与忠告

- 前三者（改数据、挑数据、p值操纵）都可以通过重复实验来识别或验证其可靠性。

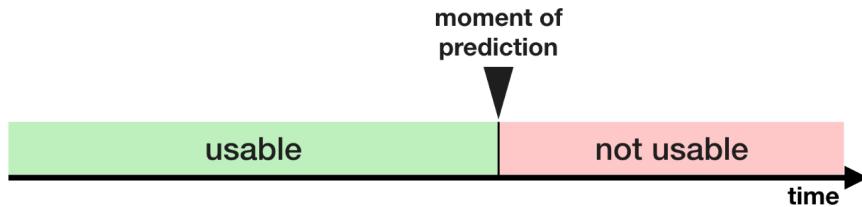
- 第四者（泄露测试集）则需要构造新测试集、采用更严格的评估流程才能发现和防范。

学术造假不仅影响个人学术发展，也会破坏学术共同体的信任与声誉。呼吁所有同学在科研和学习中秉持科学精神，以严谨诚实的态度对待每一次实验与数据分析。做学术要能经得起检验，让数据和事实“说话”，而不是想方设法去粉饰或制造“好看”的结论。

来自 Josh Starmer （统计出身，和 3Blue1Brown 并列是我最喜欢的科普视频制作者）的介绍：

[p-hacking: What it is and how to avoid it!](#)

来自 Kaggle 的介绍：[Data Leakage](#).



[图片来自Kaggle](#)

大物实验：我知道很多人在大物实验里干过改数据和挑数据的事（之所以没有p-hacking，是因为大物实验不要求做假设检验）。那也无所谓。**人类社会的法律，是要视危害程度给出惩罚**。在大物实验里造假，最大的危害是对其他不造假的学生不公平；在科研里造假，是霍霍老百姓纳税的钱让自己升官发财、是削弱公众对于科学家的信任、是浪费其他学者的时间和精力、是玩弄年轻人对于科学的热爱和感情。所以前者的惩罚是扣点分，后者的惩罚要视严重程度而定（[诈骗罪是刑事违法行为，诈骗五十万及以上对应十年以上有期徒刑](#)）。

匪夷所思：但让我很失望的是，目前世界各国对学术造假难以想象地宽容。实在是匪夷所思。

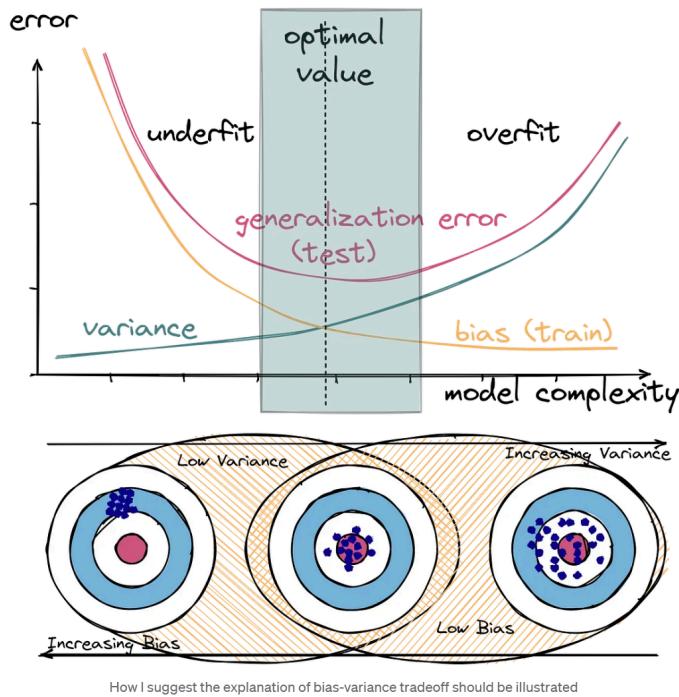
忠告：**如果我的朋友干这四件事情，我宣布他不再是我的朋友了。**

偏差-方差窘境 | Bias-Variance Tradeoff

目标：让模型达到最优

灵感：射箭

图像：



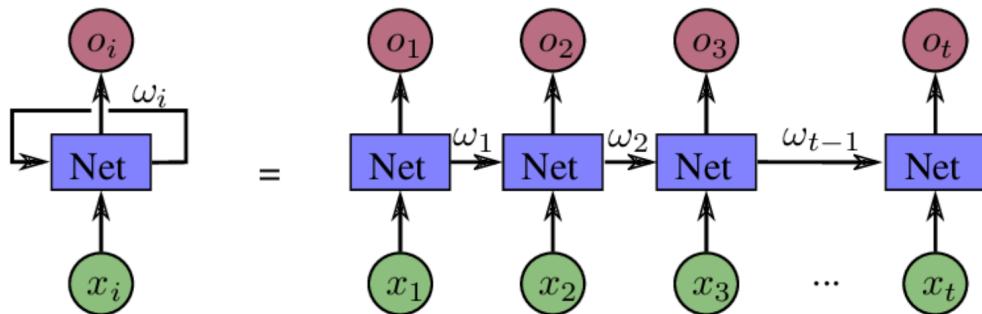
[图片来自Medium](#)

自回归 | Auto-Regressive

目标：自己预测自己。

灵感：统计中的Auto-Regressive-Moving-Average；控制论中的闭环控制。

图像：

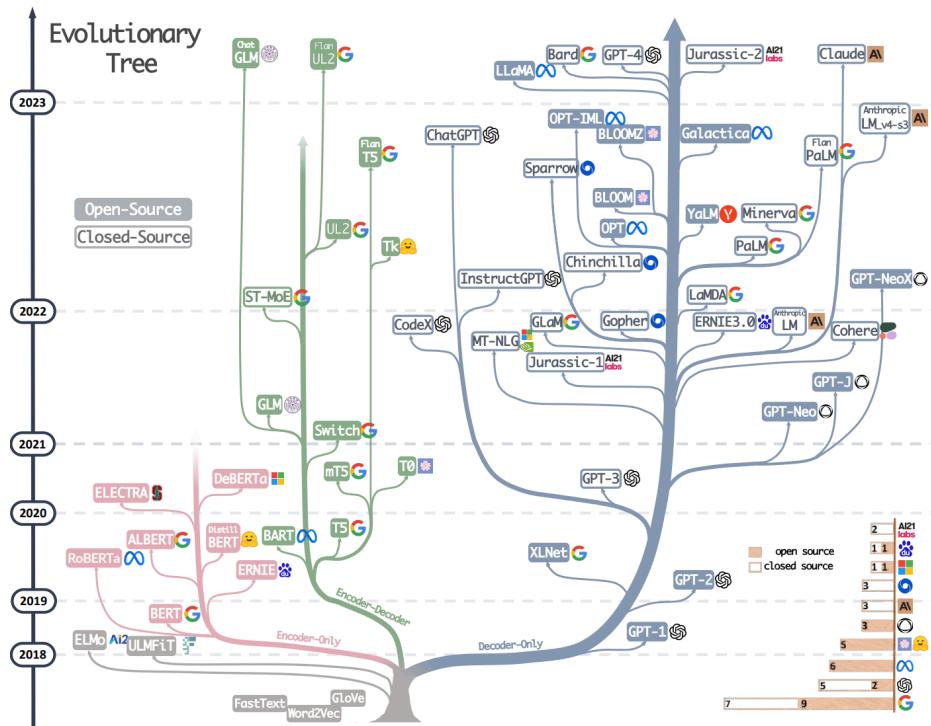


[图片来自Medium](#)

自然语言处理 | Nature Language Processing

目标：让AI学会自然语言

参考：[Chapter 14 of Dive Into Deep Learning](#)



LLM 进化树



Yann LeCun · Following
VP & Chief AI Scientist at Meta
31m ·

• • •

A survey of LLMs with a practical guide and evolutionary tree.

Number of LLMs from Meta = 7

Number of open source LLMs from Meta = 7

The architecture nomenclature for LLMs is somewhat confusing and unfortunate.

What's called "encoder only" actually has an encoder and a decoder (just not an auto-regressive decoder).

What's called "encoder-decoder" really means "encoder with auto-regressive decoder"

What's called "decoder only" really means "auto-regressive encoder-decoder"

Comment From LeCun

13年前的那些历史咱就不说了（和现在进展比起来，不值一提），以下是重点事件：

1. 2013年，Word2Vec，第一个将词嵌入 (Word Embedding) 发扬光大的工作。

1. 词嵌入，即，一个词对应一个向量

2. 它实现了：King - Man = Queen - Woman

3. 上图中的FastText, GloVe是实现词嵌入的不同方式。

4. GPT的词嵌入最像Word2Vec。

2. 2014年，Seq2Seq，可以将英文翻译成法文，这是LLM的雏形。

1. 在求 $P(y_1, y_2, \dots | x_1, x_2, \dots)$

3. 2018年，BERT

1. 在求 $P(x_i, \dots | \dots x_{i-2}, x_{i-1}, x_{i+1}, x_{i+2}, \dots)$

4. 2018年，GPT

1. 在求 $P(x_i | \dots x_{i-2}, x_{i-1})$

看起来是BERT更符合人的阅读习惯，GPT更符合人的写作习惯。

现在，GPT取得了最终的胜利，并且已经成为（或者即将成为）AGI。

除了少数文本理解任务，BERT不可能击败GPT。我知道BERT目前在科研里还有人用，有人用它处理EEG数据。但感觉都能被GPT取代。

注意力 | Attention

目标：让AI学会集中注意力。

灵感：生物的注意力。

参考：[Chapter 10 of Dive Into Deep Learning](#)

注：有多种实现注意力的方法。现在Transformer取得了最终的胜利。号称要挑战Transformer的Retention和Mamba，应该也以某种方式实现了注意力。

图形处理器 | GPU

3Blue1Brown做了一个直观的图：训练LLM，每秒能做1Billion计算，也需要 10^8 年。大约是 10^{24} 。

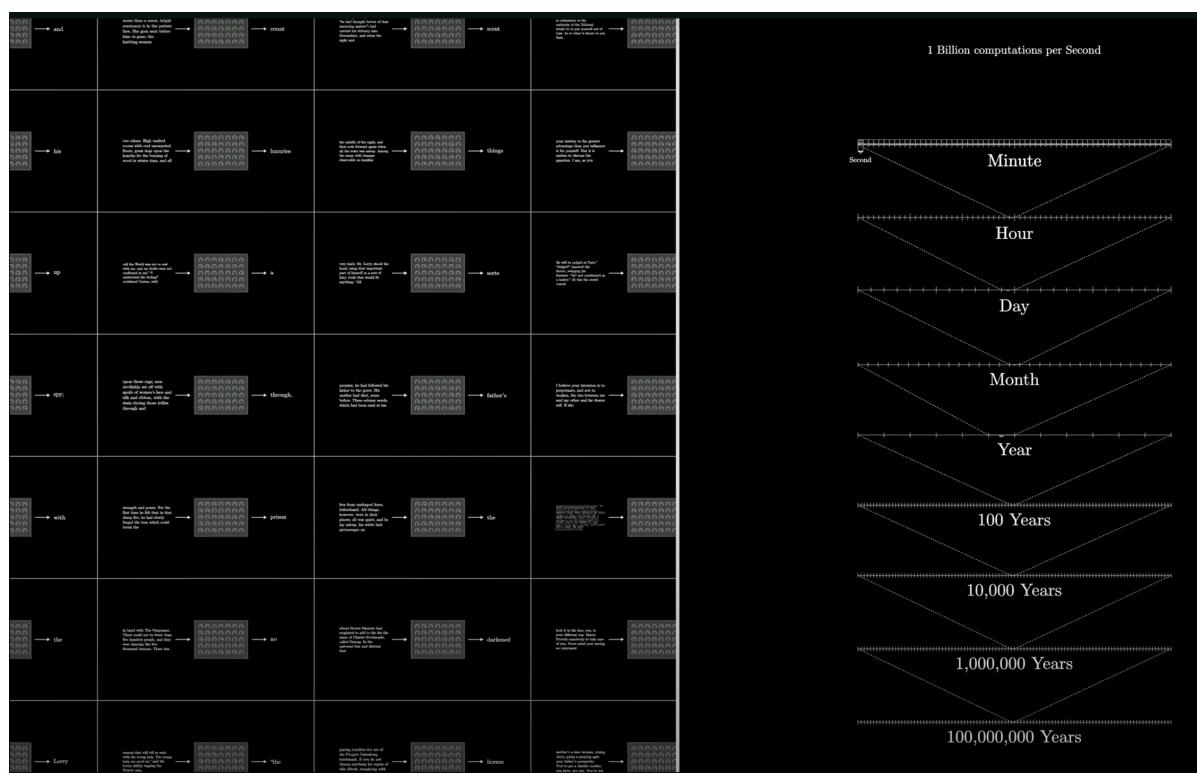


Figure from 3Blue1Brown

我让GPT-4o算了一下英特尔最新处理器的性能，不过 10^{12} 次/秒。

英特尔最新的酷睿i9-14900K处理器拥有24个核心和32个线程，最高睿频可达6.0GHz。虽然英特尔并未公开其每秒浮点运算次数（FLOPS）的具体数据，但我们可以通过理论计算进行估计。

假设每个核心每个时钟周期可以执行16个浮点运算指令（这一数值取决于处理器的架构和指令集，实际情况可能有所不同），则每个核心在6.0GHz下的浮点运算能力为：

$$16 \text{ FLOPs}/\text{周期} \times 6.0 \times 10^9 \text{ 周期}/\text{秒} = 96 \times 10^9 \text{ FLOPs} (\text{即} 96 \text{ GFLOPS})$$

对于24个核心的处理器，总的理论浮点运算能力为：

$$96 \times 10^9 \text{ FLOPs/核心} \times 24 \text{ 核心} = 2,304 \times 10^9 \text{ FLOPs (即2.304 TFLOPS)}$$

这就是为什么我们要用GPU：(PS: CPU串行，GPU并行)

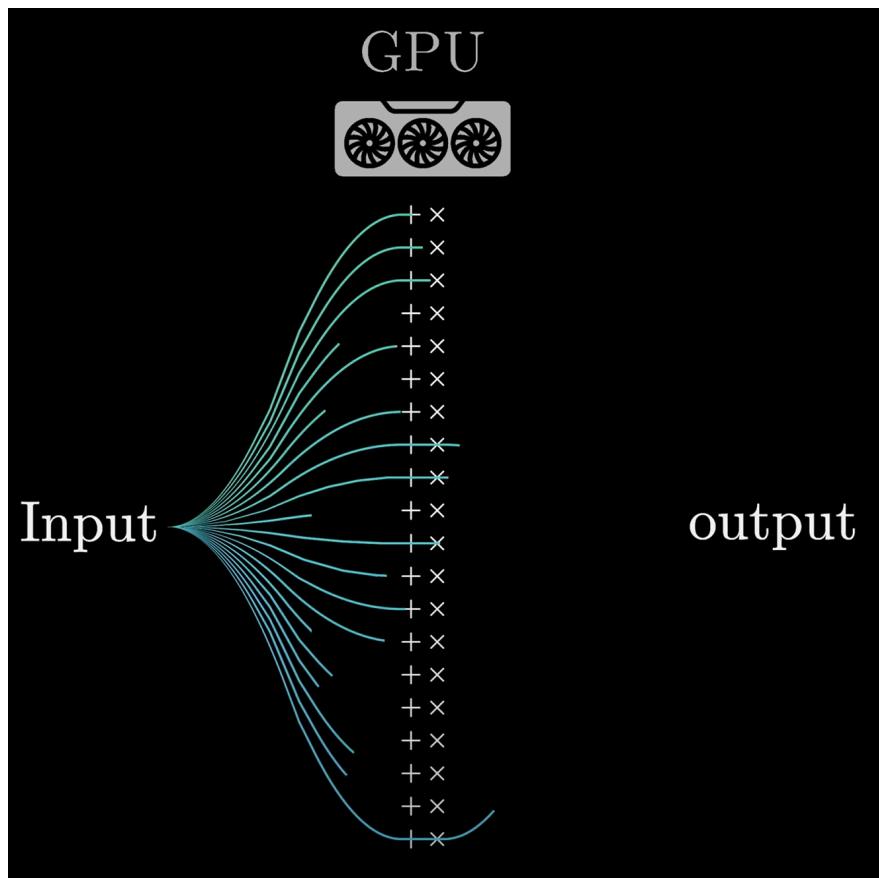


Figure from 3Blue1Brown

配环境 | Create Environments

很好，今天我来带大家配一下 [Dive Into Deep Learning](#) 的环境

安装

我们需要配置一个环境来运行 Python、Jupyter Notebook、相关库以及运行本书所需的代码，以快速入门并获得动手学习经验。

安装 Miniconda

最简单的方法就是安装依赖Python 3.x的[Miniconda](#)。如果已安装conda，则可以跳过以下步骤。访问Miniconda网站，根据Python3.x版本确定适合的版本。

如果我们使用macOS，假设Python版本是3.9（我们的测试版本），将下载名称包含字符串“MacOSX”的bash脚本，并执行以下操作：

```
# 以Intel处理器为例，文件名可能会更改  
sh Miniconda3-py39_4.12.0-MacOSX-x86_64.sh -b
```

如果我们使用Linux，假设Python版本是3.9（我们的测试版本），将下载名称包含字符串“Linux”的bash脚本，并执行以下操作：

```
# 文件名可能会更改  
sh Miniconda3-py39_4.12.0-Linux-x86_64.sh -b
```

接下来，初始化终端Shell，以便我们可以直接运行conda。

```
~/miniconda3/bin/conda init
```

现在关闭并重新打开当前的shell。并使用下面的命令创建一个新的环境：

```
conda create --name d2l python=3.9 -y
```

现在激活 d2l 环境：

```
conda activate d2l
```

流派之争 | Dispute of AI People

Q: 人工智能有哪些学派

A: 有名的有四个（它们当然不像整数分成正整数、0、负整数那样是不重不漏的，是有重叠、遗漏的）

1. 符号主义：即Minsky, Simon那派
2. 连接主义：即深度学习
3. 行为主义：即强化学习
4. 统计：即SVM



Figure from [Internet](#)

Q: 它们爆发过哪些激烈的斗争

A: 有两次最激烈

1. 符号主义 vs 连接主义：导致了深度学习第一次寒冬
2. 统计 vs 连接主义：导致了深度学习第二次寒冬

问答 | QA

Q: OpenAI等公司发布模型时候会有测试集泄露的问题？

A: 肯定有。比如说 AIME (American Invitational Mathematics Examination) 和 IMO (International Mathematical Olympiad) 的数学题，肯定泄露到互联网数据里了。虽然他们有些[方法](#)，但是我认为不能完全避免。

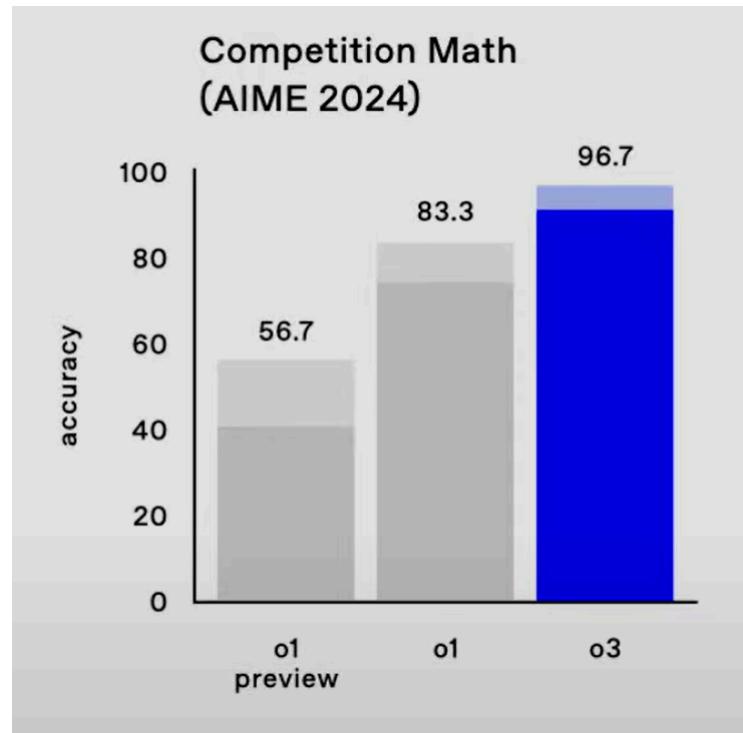
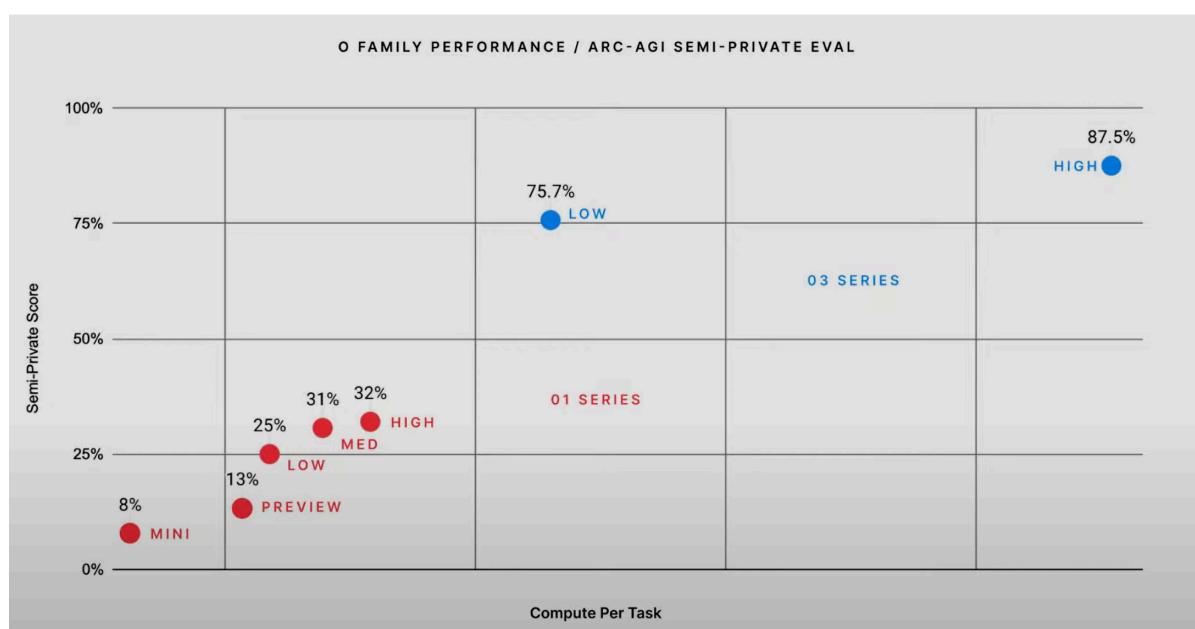
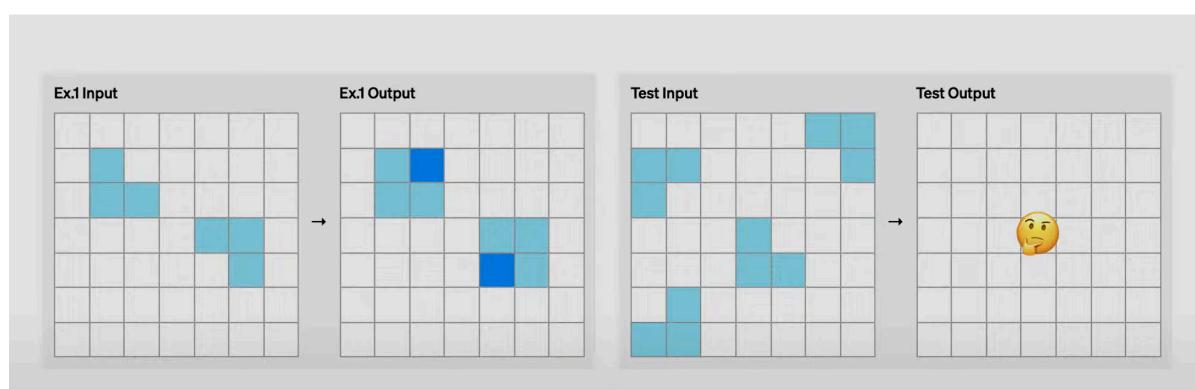


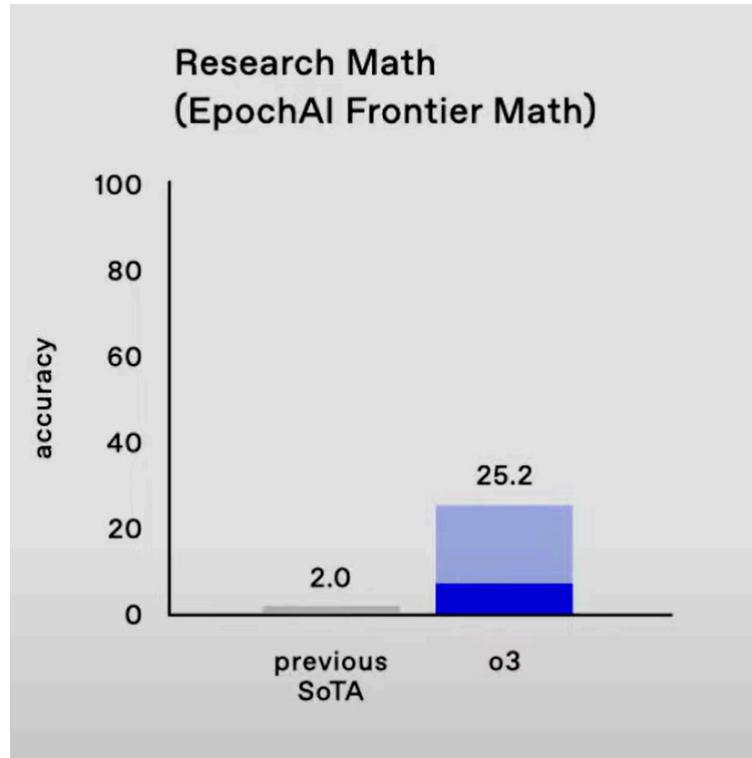
Figure from [OpenAI](#)

Q: 那咋办？

A: 构建新的数据集。比如，OpenAI 发布 o3 时，使用了ARC和FrontierMath，两个新构造的数据集。



ARC, Figure from [OpenAI](#)



Frontier Math, Figure from [OpenAI](#)

Lecture 2 & 3: 有监督学习 | Supervised Learning

目标：分析有人类标签的数据 (Data With Human Label)。

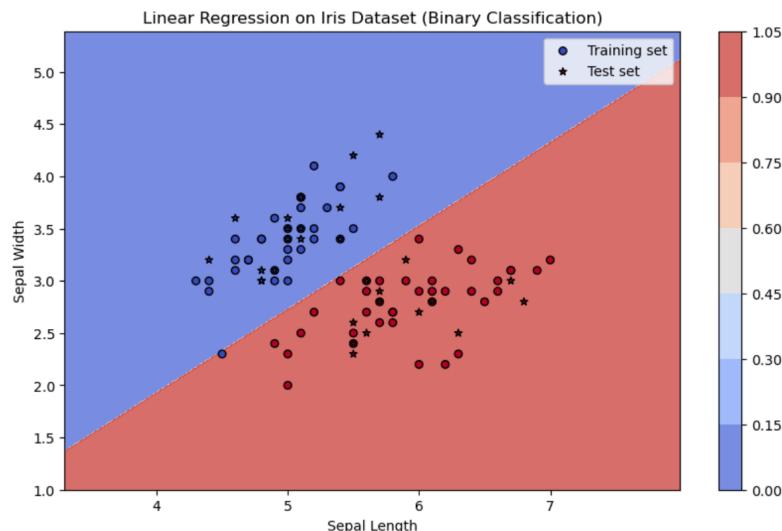
线性模型 | Linear Model

最小二乘 | Least Square

目标：回归（其实也可以分类，比如下图）。

灵感：勒让德和高斯观察天文学数据发明的，[俩人还吵起来了](#)。

图像：

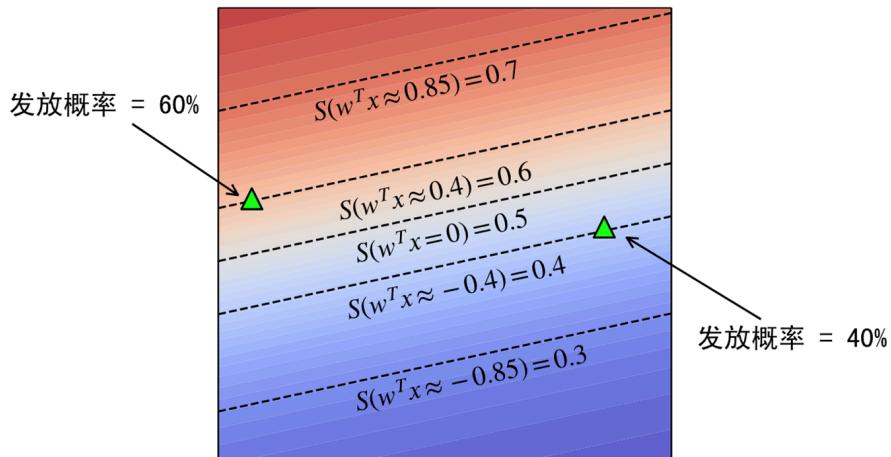


逻辑回归 | Logistic Regression

目标：分类。

灵感：当最小二乘用于分类时，找一个统计解释。

图像：



所以，在Sigmoid函数的帮助下，实现了逻辑回归的思路。

[图片来自马同学](#)

支持向量机 | Supporting Vector Machine

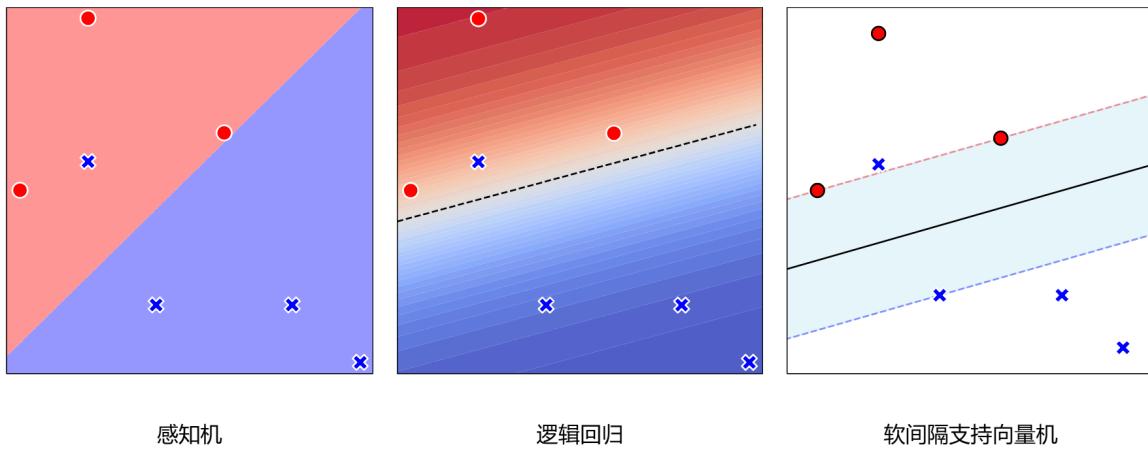
目标：分类。

灵感：凸优化、统计。

名称来源：下图最后侧，最靠近分类边界的那几个点叫支持向量(Supporting Vector)，因此这算法叫支持向量机器(Supporting Vector Machine)。

优点：可以用Kernel把低维线性不可分的样本转变为高维中线性可分的样本。这是SVM超越最小二乘和逻辑回归之处。

图像：



[图片来自马同学](#)

问答 | QA

Q: 正则到底是啥意思?

A: 正则一词来自古汉语

名余曰正则兮，字余曰灵均。战国·屈原《离骚》

辩方位而正则，五精帅而来摧。汉·张衡《东京赋》

它有美好、礼仪、法规的意思。

物理中的正则变换(Canonical Transformation)、机器学习中的正则化(Regularization)、CS中的正则表达式(Regular Expression)，都有“美好、法规、约束”的意思，因此都被翻译成了正则。

Q: LASSO (L1)和Ridge (L2)，这俩名字咋来的?

A:

1. LASSO: Least Absolute Shrinkage and Selection Operator.

2. Ridge: 几何上，L2正则化的约束集对应一个高维球体。当你将这些球体等式约束（如半径不同的球）在参数空间中与最小二乘误差等高线（一般为椭圆）叠加时，它们的交点像是在“山脊”上寻找解（即在约束下的优化点常位于“椭圆谷”边缘）。

Q: 这张图到底啥意思?

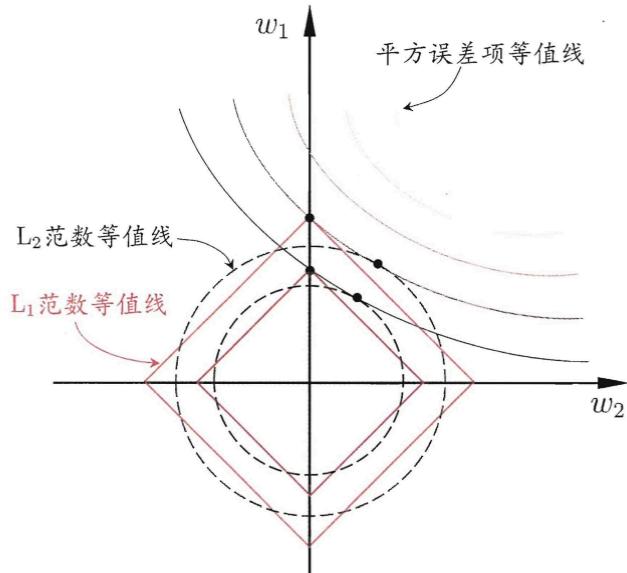


图 11.2 L₁ 正则化比 L₂ 正则化更易于得到稀疏解

图片来自西瓜书

A: 你要找两类等值线的交点。随后你就发现，L1的会在坐标轴上，L2的不在。

Q: 为什么不用L0, L3, L4,..., 作为惩罚项?

A:

它们分是：

1. L0: # non zero elements

2. L1: $\sum_i |w_i|$

3. L2: $\sum_i w_i^2$

4. L3: $\sum_i w_i^3$

5. L4: $\sum_i w_i^4$

6. ...

至于为什么主要用L1和L2

1. L0计算上是NP-Hard。

2. L3、L4、更高阶的效果和L2差不多，但是可解释性不如L2。肯定有人用，但是不是主流。

3. L1有稀疏选择性。

4. L2有可解释性，可以和物理中的能量搭上关系。

Q: “回归”这词是咋来的？

A: [Francis Galton](#), 统计学创始人之一，研究了很多对父子的身高的关系。他发现最高的那批人的子女身高会没那么高(rank会下降)，最低的那批人的子女身高会没那么矮(rank会上升)，因此他把这种现象称之为Regression。中文翻译为回归，挺传神的。后来，人们把所有预测一个连续变量的方法都叫 Regression，而Galton观察到的现象则被称之为[Regression Towards the Mean](#)。

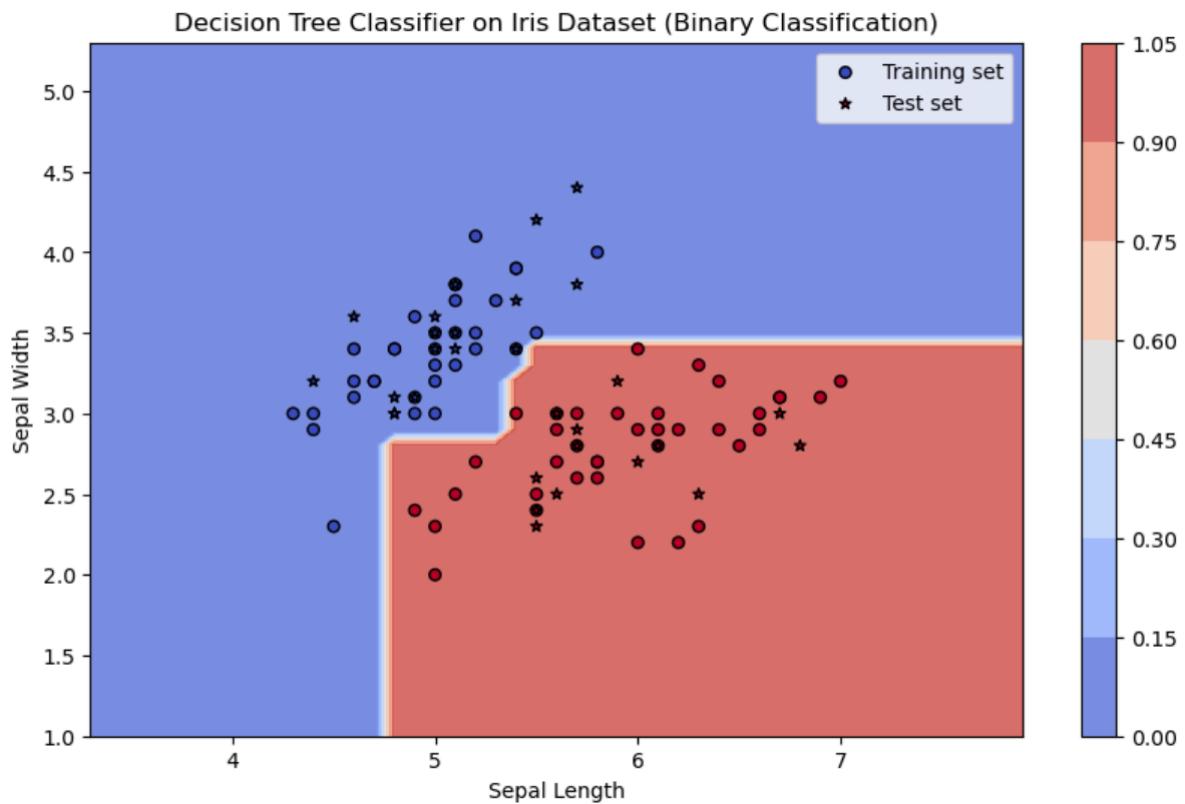
决策树 | Decision Tree

引入 | Intro

目标：分类

灵感：模仿人类做决定

图像：



分裂标准/特征选择：基尼指数；香农熵增益；香农熵增益率。（见附录）

剪枝 | Prune

目标：对付过拟合

预剪枝 | Pre-Pruning

图像：

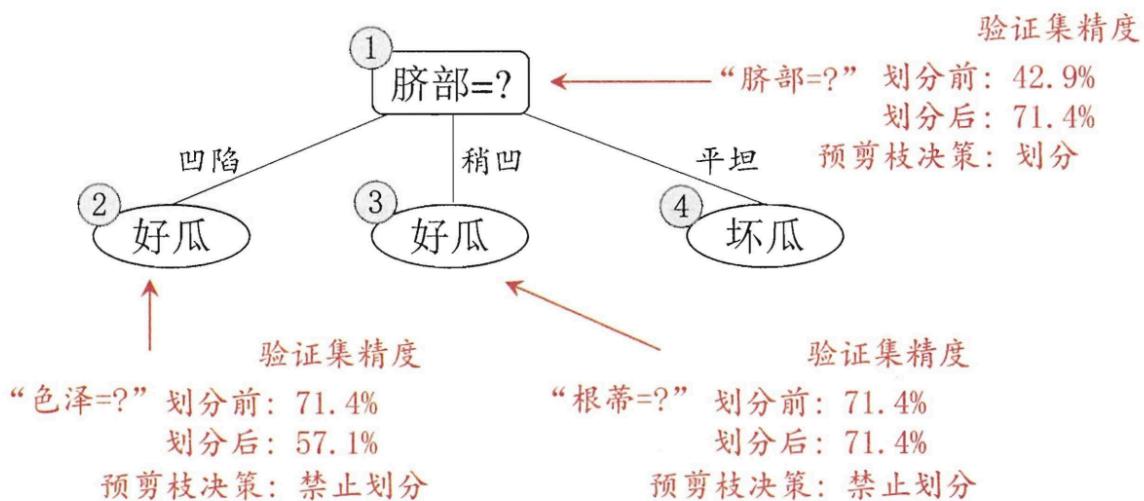


图 4.6 基于表 4.2 生成的预剪枝决策树

图片来自西瓜书

重点：得有验证集。

后剪枝 | Post-Pruning

图像：

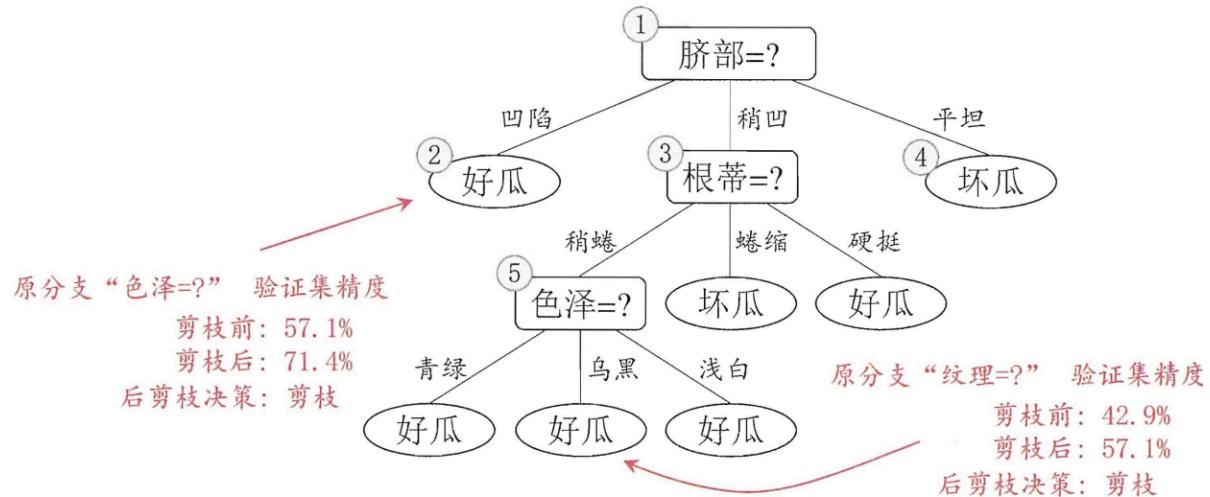


图 4.7 基于表 4.2 生成的后剪枝决策树

图片来自西瓜书

重点：同上，得有验证集。

问答 | QA

Q: 决策树用的是贪心算法，因此得到的树未必是最优的？

A: 是。

Q: 决策树相比最小二乘和逻辑回归的最大优点？

A: 非线性。

Q: 听说决策树有很多种，主要区别是什么？

决策树学习算法最著名的代表是 ID3 [Quinlan, 1979, 1986]、C4.5 [Quinlan, 1993] 和 CART [Breiman et al., 1984]. [Murthy, 1998] 提供了一个关于决策树文献的阅读指南。C4.5Rule 是一个将 C4.5 决策树转化为符号规则的算法 [Quinlan, 1993]，决策树的每个分支可以容易地重写为一条规则，但 C4.5Rule 算法在转化过程中会进行规则前件合并、删减等操作，因此最终规则集的泛化性能甚至可能优于原决策树。

图片来自西瓜书

算法	支持模型	树结构	特征选择	连续值处理	缺失值处理	剪枝	特征属性多次使用
ID3	分类	多叉树	信息增益	不支持	不支持	不支持	不支持
C4.5	分类	多叉树	信息增益率	支持	支持	支持	不支持
CART	分类回归	二叉树	基尼指数 均方差	支持	支持	支持	支持

图片来自老师的幻灯片

A: 主要区别是分裂标准/特征选择。

Q: CART, ID3, C4.5的名字咋来的?

A:

1. CART: Classification And Regression Tree
2. ID3: Iterative Dichotomiser 3
3. C4.5: Classifier 4.5

Q: 没有其他特征选择方法吗?

A:

本质上，各种特征选择方法均可用于决策树的划分属性选择。特征选择参见第 11 章。

在信息增益、增益率、基尼指数之外，人们还设计了许多其他的准则用于决策树划分选择，然而有实验研究表明 [Mingers, 1989b]，这些准则虽然对决策树的尺寸有较大影响，但对泛化性能的影响很有限。[Raileanu and Stoffel, 2004] 对信息增益和基尼指数进行的理论分析也显示出，它们仅在 2% 的情况下会有所不同。4.3 节介绍了决策树剪枝的基本策略；剪枝方法和程度对决策树泛化性能的影响相当显著，有实验研究表明 [Mingers, 1989a]，在数据带有噪声时通过剪枝甚至可将决策树的泛化性能提高 25%。

图片来自西瓜书

Q: 决策树做好之后，如果收了新的数据，只能从头训练吗？

A:

有一些决策树学习算法可进行“增量学习”(incremental learning)，即在接收到新样本后可对已学得的模型进行调整，而不用完全重新学习。主要机制是通过调整分支路径上的划分属性次序来对树进行部分重构，代表性算法有 ID4 [Schlimmer and Fisher, 1986]、ID5R [Utgoff, 1989a]、ITI [Utgoff et al., 1997] 等。增量学习可有效地降低每次接收到新样本后的训练时间开销，但多步增量学习后的模型会与基于全部数据训练而得的模型有较大差别。

图片来自西瓜书

Q: 上述决策树都只能画横平竖直的线，不能画斜线吗？

A: 能。称之为多变量决策树。见西瓜书4.5。

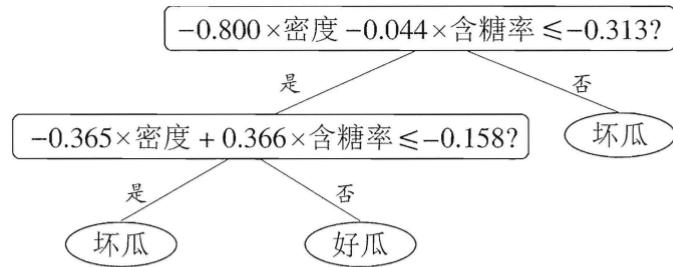


图 4.13 在西瓜数据集 3.0 α 上生成的多变量决策树

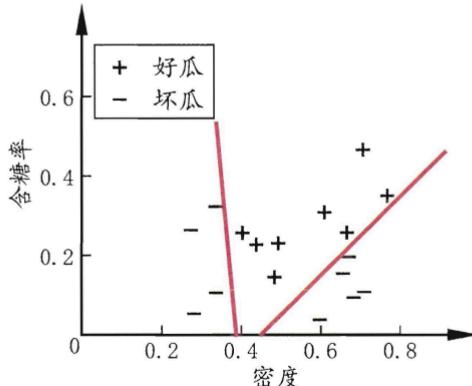


图 4.14 图 4.13 多变量决策树对应的分类边界

图片来自西瓜书

多变量决策树算法主要有 OC1 [Murthy et al., 1994] 和 [Brodley and Utgoff, 1995] 提出的一系列算法。OC1 先贪心地寻找每个属性的最优权值，在局部优化的基础上再对分类边界进行随机扰动以试图找到更好的边界；[Brodley and Utgoff, 1995] 则直接引入了线性分类器学习的最小二乘法。还有一些算法试图在决策树的叶结点上嵌入神经网络，以结合这两种学习机制的优势，例如“感知机树”(Perceptron tree) [Utgoff, 1989b] 在决策树的每个叶结点上训练一个感知机，而 [Guo and Gelfand, 1992] 则直接在叶结点上嵌入多层神经网络。

图片来自西瓜书

集成学习 | Ensemble Learning

引入 | Intro

目标：结合不同学习器的优点

核心：好而不同

分类：

根据个体学习器的生成方式，目前的集成学习方法大致可分为两大类，即个体学习器间存在强依赖关系、必须串行生成的序列化方法，以及个体学习器间不存在强依赖关系、可同时生成的并行化方法；前者的代表是 Boosting，后者的代表是 Bagging 和“随机森林”(Random Forest)。

图片来自西瓜书

增强 | Boosting

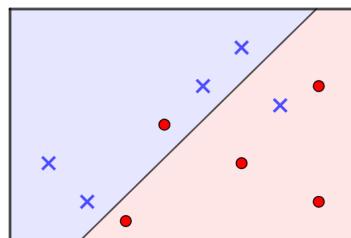
目标：结合不同学习器的优点

核心：串行

图像：

1 第一个模型

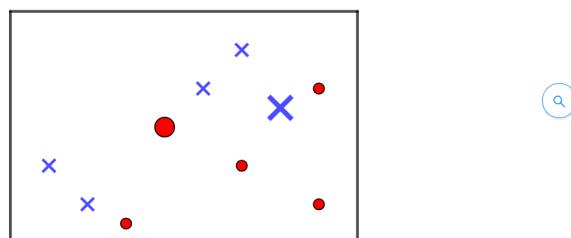
首先找一个简单的模型，比如感知机的口袋算法，然后求出第一个模型：



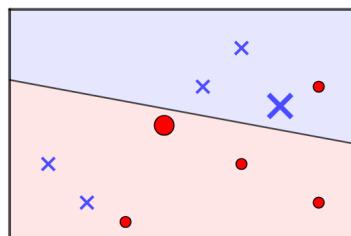
容易看到在该模型中有两个点分错了，所以我们决定对它们进行重点研究，以此生成第二个模型。

2 第二个模型

为了做到这一点，可以新生成的一个数据集，加大刚才分错的两个点在数据集中的权重（最简单的作法就是将分错的点在数据集中多复制几个，这样在计算误差时，该点的比重就会增大），下图中将它们画得很大表示其权重的增加：



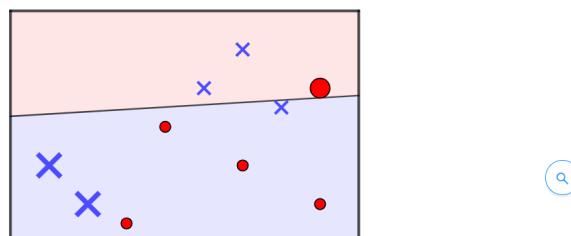
根据新的数据集，可以用感知机的口袋算法得到第二个模型：



根据第二个模型的决策边界，可以看到有三个点分错了，它们就是下一个模型的研究重点。

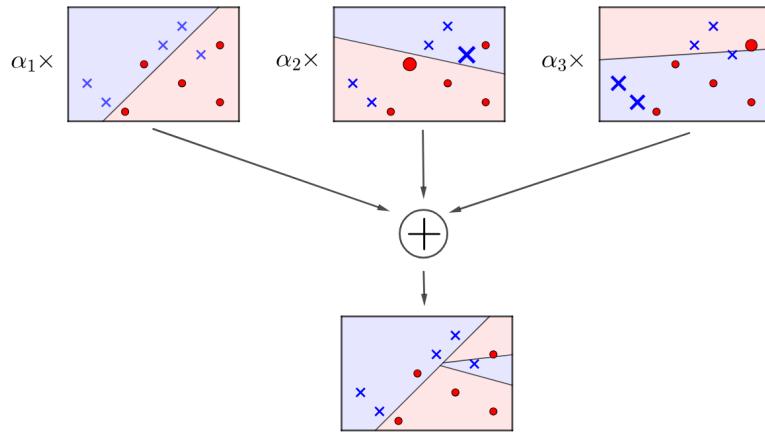
3 第三个模型

根据分错的三个点可以又生成一个新的数据集，从而得到第三个模型：



4 集成

将上述三个模型 $h_m(x)$ 按照一定的权重 α_m 组合在一起就得到了最终的集成模型（其中还是有分错的点，可以继续增加模型来集成）：

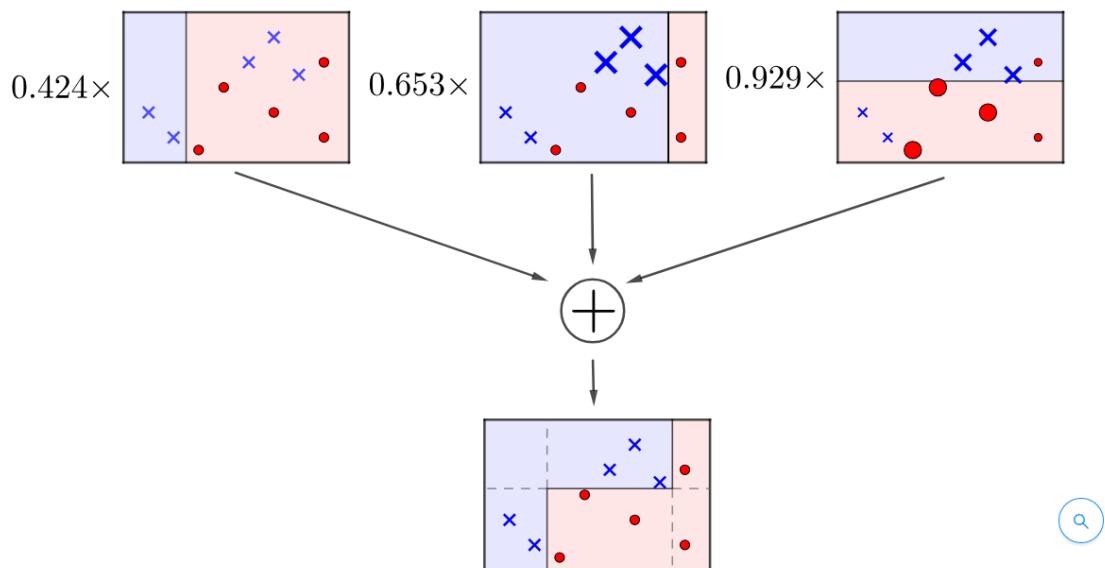


[图片来自马同学](#)

自适应增强 | AdaBoost

核心：在Boost基础上加了权重

图像：



[图片来自马同学](#)

梯度增强树 | Gradient Boosted Decision Trees

核心：[对梯度进行一些操作](#)

变种

1. [XGBoost, 2014](#)
2. [LightGBM, 2016](#)
3. [CatBoost, 2017](#)

(这三个我也不懂，但应该不难，像工程。SVM的那些算法推导是真难，像数学/统计。)

袋子 | Bagging

目标：结合不同学习器的优点

核心：并行

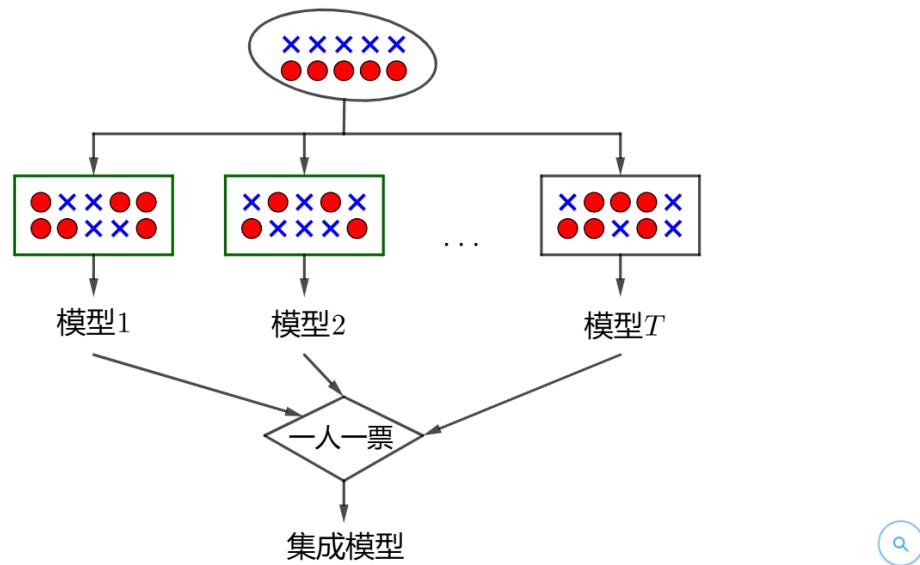
名称来源：Bootstrap Aggregating

图像：

将 Bootstrap 用于集成学习，就是 Bagging 算法：

$$\text{Bagging} = \underbrace{\text{Bootstrap}}_{\text{自助法}} + \underbrace{\text{Aggregating}}_{\text{聚合}}$$

具体的做法就是在现有数据集的基础上，通过 Bootstrap 生成 T 个数据集，然后分别训练出 T 个模型，最后用一人一票的方式投出集成模型：



每个模型都是分开训练的，所以这是并行的集成学习算法。

[图片来自马同学](#)

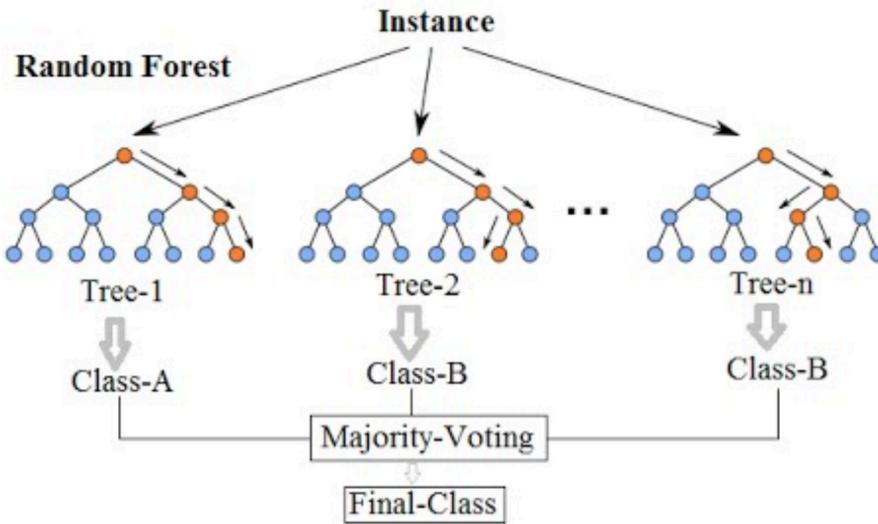
随机森林 | Random Forest

核心：Bagging 只靠数据集不同来打造不同的模型，随机森林还可以加入决策树划分时随机性。

名称来源：如果 Bagging 算法都采用决策树的话，最终得到的集成模型就称为随机森林。“树”多了自然就组成了“森林”。

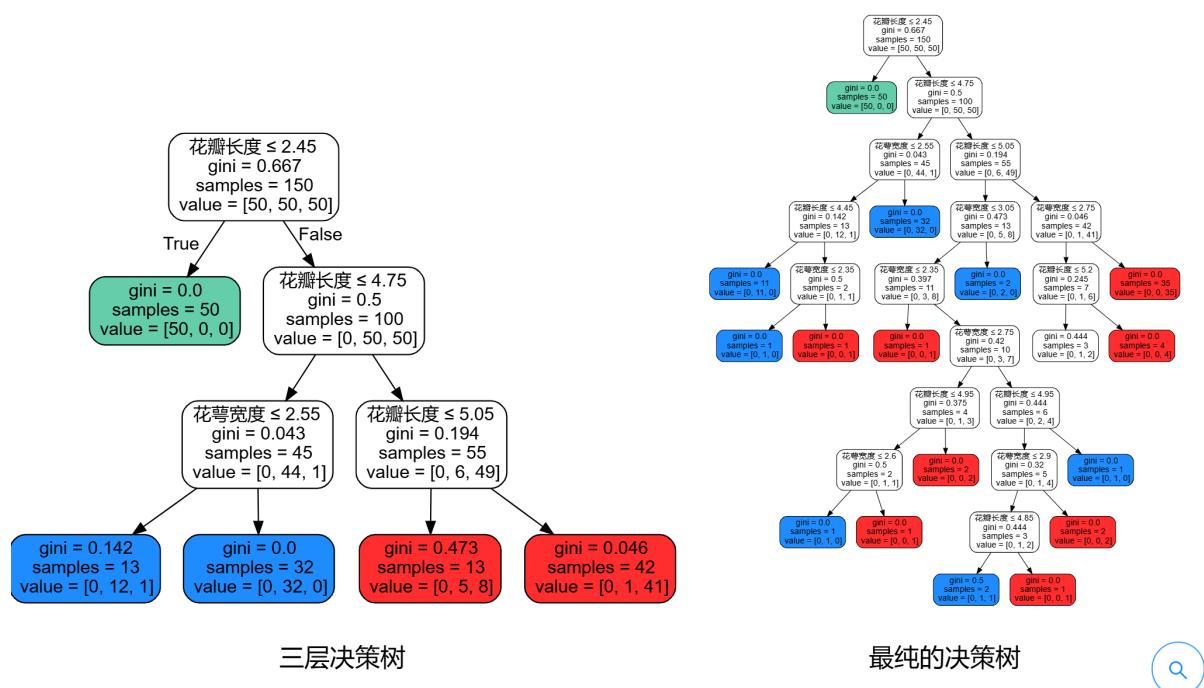
图像：

Random Forest Simplified



图片来自NVIDIA

注：据说，在随机森林中，使用最纯的决策树更好。下图是对鸢尾数据集的决策树分类，左侧为三层，右侧为最纯。



注：选择属性时候没有随机性的随机森林可以叫决策森林。

8.3.2 随机森林

随机森林(Random Forest, 简称 RF) [Breiman, 2001a] 是 Bagging 的一个扩展变体. RF 在以决策树为基学习器构建 Bagging 集成的基础上, 进一步在决策树的训练过程中引入了随机属性选择. 具体来说, 传统决策树在选择划分属性时是在当前结点的属性集合(假定有 d 个属性)中选择一个最优属性; 而在 RF 中, 对基决策树的每个结点, 先从该结点的属性集合中随机选择一个包含 k

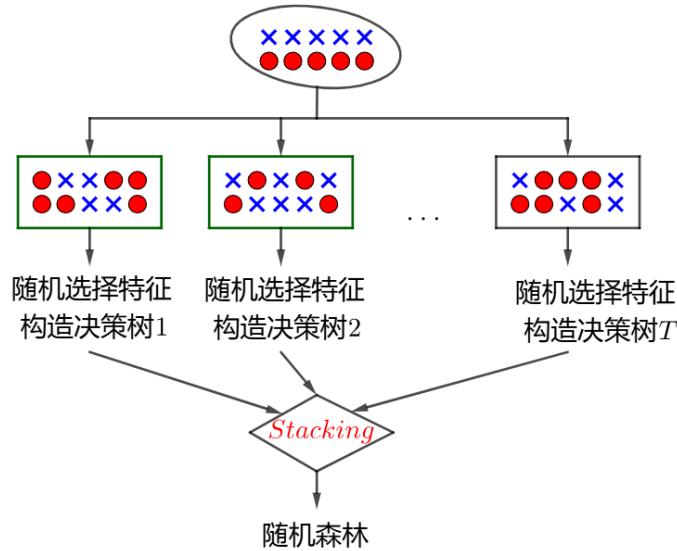
个属性的子集，然后再从这个子集中选择一个最优属性用于划分。这里的参数 k 控制了随机性的引入程度：若令 $k = d$ ，则基决策树的构建与传统决策树相同；若令 $k = 1$ ，则是随机选择一个属性用于划分；一般情况下，推荐值 $k = \lceil \frac{d}{11097} \rceil$ [Breiman, 2001a]。

图片来自西瓜书

改变投票方式 | Stacking

图像：

一定是“一人一票”吗？其实不一定，可以对输出的决策树再进行学习，从而决定每个模型的投票权重。因为学习分了两层，第一层学习获得决策树，第二层学习获得投票权重，所以该算法称为 **Stacking**：



Stacking 肯定不光可以和随机森林配合，实际上可以和很多算法进行配合，这里不再赘述。

[图片来自马同学](#)

问答 | QA

Q: 集成学习在现实中用得多吗？

A: 据我耳闻，是传统机器学习里用得最多的。最火的是XGBoost和随机森林。

Q: 决策树和集成学习比起深度学习如何？

A: 课上所讲的决策树和集成学习算法大多是深度学习第二次寒冬时（1990-2006）提出的。那时候它们还完全压制了深度学习。

Q: XGBoost (2014) 和 LightGLM (2016) 是深度学习第三次崛起后提出的，它们比之后者如何？

A:

优势

1. 前两者在小规模数据集上有可能超过深度学习。
2. 前两者不需要众多GPU。

3. 前两者适合Kaggle竞赛和天池竞赛。

劣势

1. 前两者在大规模数据集上逊于深度学习。
2. 前两者需要人类专家设计特征。 (因此可解释性也比深度学习好)。

Q: 什么叫Boosting?

A: Boosting, 是不断增强的意思。

Q: 什么叫Bagging?

A: Bootstrap + Aggregating

Q: 集成学习可以用GPU加速吗?

A: 可以。只要是能并行的算法都可以 (回忆一下高性能计算)。

Lecture 4: 深度学习 | Deep Learning

目标：模仿人脑，用一大堆人工神经元实现通用人工智能。

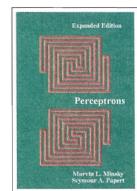
数学保证：两层神经网络可以近似任何函数。

一些历史：

休息一会儿

小故事：神经网络的几起几落

二十世纪四十年代 M-P 神经元模型、Hebb 学习律出现后，五十年代出现了以感知机、Adaline 为代表的一系列成果，这是神经网络发展的第一个高潮期。不幸的是，MIT 计算机科学研究的奠基人马文·闵斯基 (Marvin Minsky, 1927—) 与 Seymour Papert 在 1969 年出版了《感知机》一书，书中指出，单层神经网络无法解决非线性问题，而多层网络的训练算法尚看不到希望。这个论断直接使神经网络研究进入了“冰河期”，美国和苏联均停止了对神经网络研究的资助，全球该领域研究人员纷纷转行，仅剩极少数人坚持下来。哈佛大学的 Paul Werbos 在 1974 年发明 BP 算法时，正值神经网络冰河期，因此未受到应有的重视。



闵斯基于 1969 年获图灵奖。

此书中有不少关于神经网络的真知灼见，但其重要论断所导致的后果，对神经网络乃至人工智能整体的研究产生了极为残酷的影响，因此在神经网络重又兴起后，该书受到很多批判。1986 年再版时，闵斯基专门增加了一章以作辩护。

1983 年，加州理工学院的物理学家 John Hopfield 利用神经网络，在旅行商问题这个 NP 完全问题的求解上获得当时最好结果，引起了轰动。稍后，UCSD 的 David Rumelhart 与 James McClelland 领导的 PDP 小组出版了《并行分布处理：认知微结构的探索》一书，Rumelhart 等人重新发明了 BP 算法，由于当时正处于 Hopfield 带来的兴奋之中，BP 算法迅速走红。这掀起了神经网络的第二次高潮。二十世纪九十年代中期，随着统计学习理论和支持向量机的兴起，神经网络学习的理论性质不够清楚、试错性强、在使用中充斥大量“窍门”(trick)的弱点更为明显，于是神经网络研究又进入低谷，NIPS 会议甚至多年不接受以神经网络为主题的论文。

2010 年前后，随着计算能力的迅猛提升和大数据的涌现，神经网络研究在“深度学习”的名义下又重新崛起，先是在 ImageNet 等若干竞赛上以大优势夺冠，此后谷歌、百度、脸书等公司纷纷投入巨资进行研发，神经网络迎来了第三次高潮。

图片来自西瓜书

MP 模型 | MP Model

目标：模仿人类神经元

时间：1943年

数学： $y = \sum_i w_i x_i$

单层感知机 | Perceptron

目标：通用人工智能（从 Rosenblatt 提出感知机的那天起，这就一直是连接学派的目标和信仰）

时间：1957年

本质：线性模型，和逻辑回归非常类似。

数学： $y = \text{Sign}(\sum_i w_i x_i + b)$

图像：

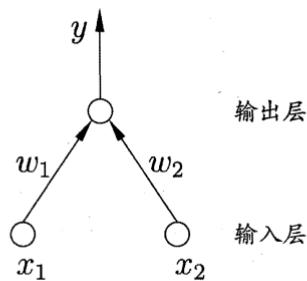


图 5.3 两个输入神经元的感知机网络结构示意图

图片来自西瓜书

能力：我估计它的能力和逻辑回归差不多。之所以当年掀起深度学习第一次高潮，是因为它是生物学可行的，让大家看到了实现通用人工智能的希望。

一些历史：Frank Rosenblatt, Perceptron发明者，吹了一些不该吹的牛。他对媒体和公众表示，感知器未来可能能够“自动学习、自动解决问题，甚至可能会达到与人类大脑竞争的水平”（这说的不就是ChatGPT）。1969年，符号学派的Marvin Minsky和Seymour Papert合著的《感知器》一书中揭示单层感知机连异或都解决不了，而多层感知机的训练还束手无策。因此美国和苏联都停止了资助。这就是所谓的“第一次深度学习寒冬”。

（但是吧，如果不吹牛，深度学习怎么可能有后两次崛起？！）

（现在他们做成了，ChatGPT是由一层Tranformer、一层MLP组成的，Frank Rosenblatt也可以瞑目了。）

（所以吹牛无所谓，最后把吹的牛都实现了就可以了。）

Hopfield 网络 | Hopfiled Network

目标：通用人工智能

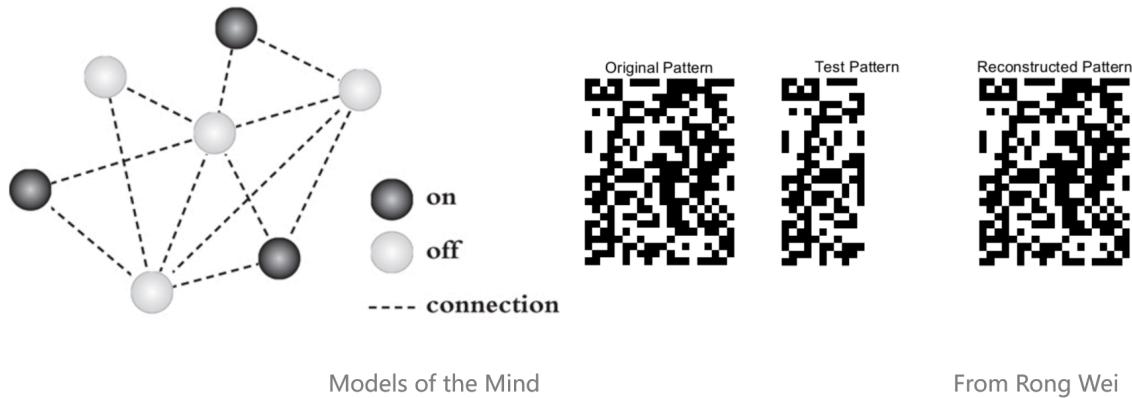
时间：1982年

数学：感知机里， $x \in [0, +\infty]$ ；此处， $x \in \{-1, +1\}$

灵感：Ising Model

能力：在旅行商问题上取得了当时最好的结果。

图像：



全连接神经网络/多层感知机 | DNN/MLP

目标：通用人工智能

时间：1986年

成功原因：1986年提出的反向传播。

反向传播的本质：微积分中的莱布尼茨法则。推断 (Infer) 时是把输入一层一层前向传播，学习 (Learning) 时是把梯度下降的结果一层一层反向传播。

图像：

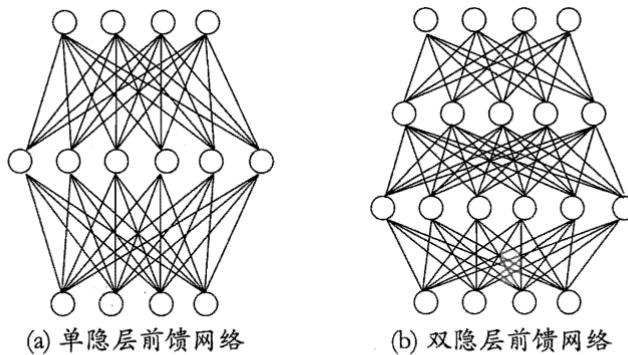


图 5.6 多层前馈神经网络结构示意图

图片来自西瓜书

能力：1988年，Terry Sejnowski用它做出来人类历史上第一个能朗读自己从未见过的文本的机器(txt to speech)——[NETtalk](#)。

能力瓶颈：[Terry Sejnowski 1988年就想过用 MLP 来预测蛋白质结构](#)，但是效果不佳，应该是数据、算力不足，所以模型没法做大。

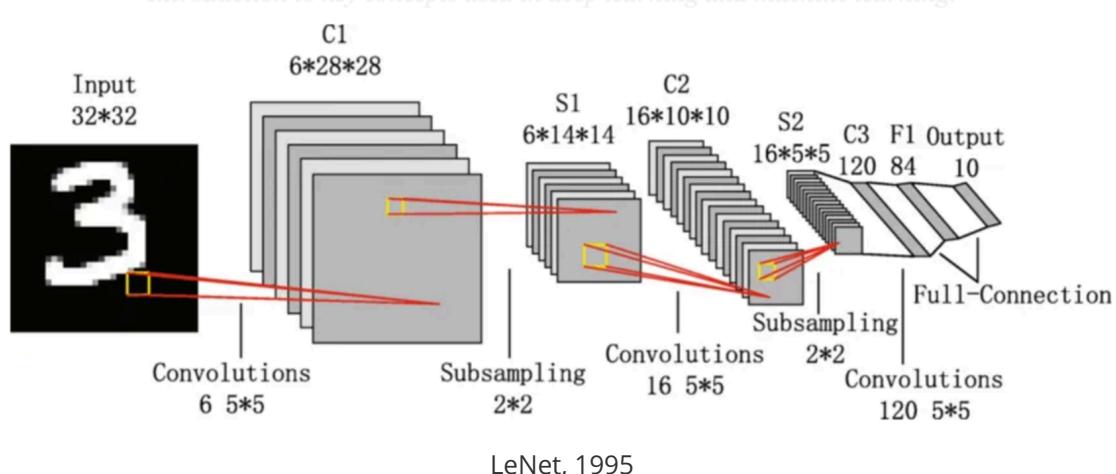
参考：3Blue1Brown；西瓜书第五章。

卷积神经网络 | CNN

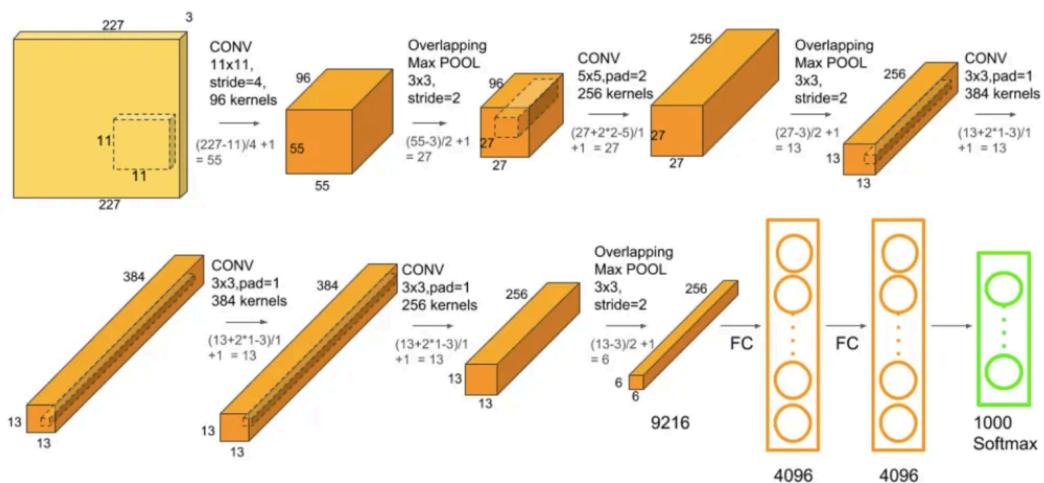
目标：处理图片。

时间：1995年由 Yann LeCun 发明，2012年由 Alex 发扬光大。

图像：（每个块块里是一大堆人工神经元）



LeNet, 1995



AlexNet, 2012

注：每层由好多复制。LeNet第一层有6个，AlexNet第一层有96个。他们的目标是提取不同特征。他们的卷积核不同是因为初始化的 \vec{w}_0 不同。

数学：

1. 卷积：和微积分、概统里的卷积是一个意思。
2. 池化：就是取mean或者取max。

为什么适合处理图片？：因为图片中的特征是有平移不变性的，所以CNN中权重可以共享。（即，同一特征（如边缘、角点、纹理等）可能出现在不同的位置。为了有效地检测这些特征，使用相同的卷积核在整个图像上滑动检测是合理且高效的。）

能力：AlexNet之后没几年，CNN就在图片识别上超过了人类。

循环神经网络 | RNN

目标：处理时间序列。

时间：2000年左右。（其实Hopfield也是RNN，但是现代的RNN主要是2000年左右提出来的，特别是1997年的LSTM）

图像：

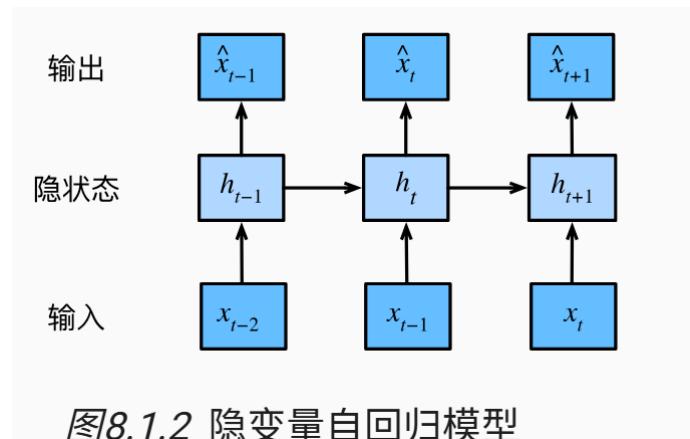


图8.1.2 隐变量自回归模型

图片来自Dive Into Deep Learning

能力：2006-2012，深度学习已有崛起前兆——当时RNN处理音频效果非常不错，为DL保留了一些火种和希望。

变形金刚 | Transformer

目标：处理自然语言。（2020年被拓展到了视觉上，所谓的ViT）

时间：2017年。

图像：

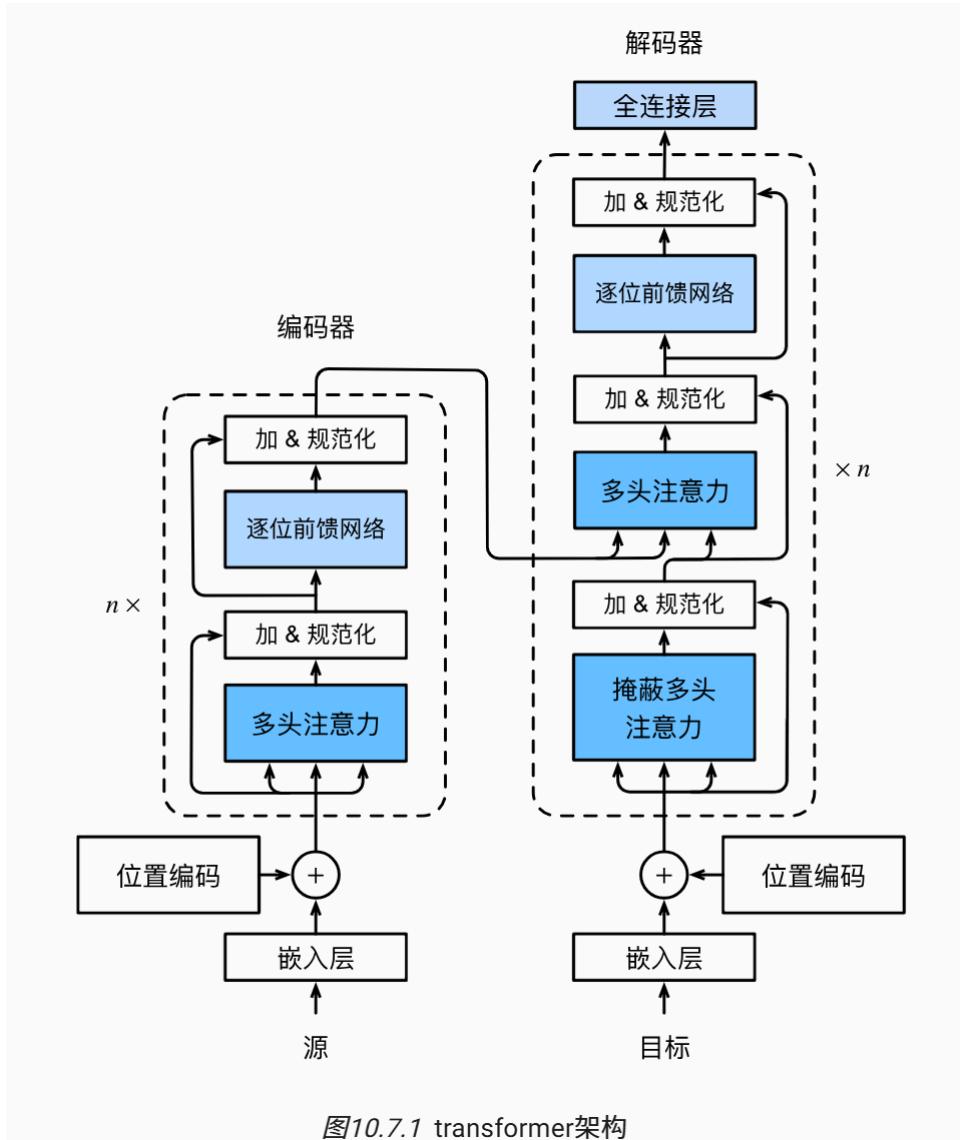


图10.7.1 transformer架构

图片来自Dive Into Deep Learning

能力：只要数据和算力足够，目前啥问题都用Transformer。

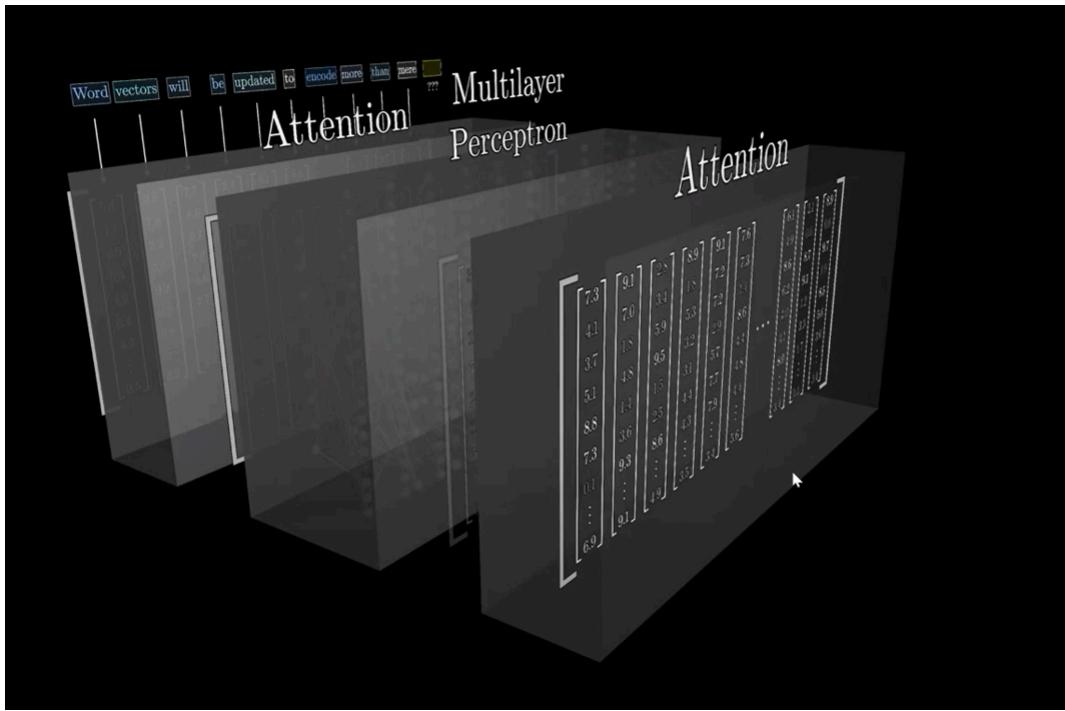
能力：取代了LSTM。

生成式预训练变形金刚 | GPT

目标：通用人工智能。

时间：2018年。

图像：变形金刚、多层感知机、变形金刚、多层感知机、…



图片来自3Blue1Brown

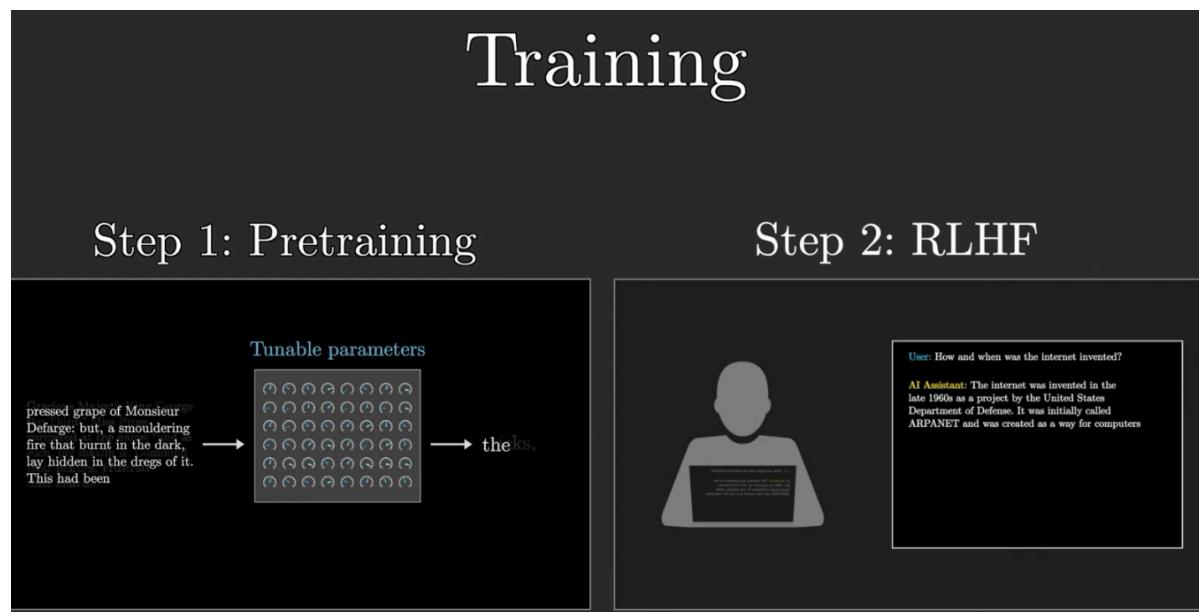
为什么叫预训练？：此处的预训练指的是在通用数据集训练，之后再在特定任务上微调。与之相对的概念是从头训练，即直接在特定任务上训练。

能聊天的生成式预训练变形金刚 | ChatGPT

目标：通用人工智能。

时间：2022年。

图像：GPT + RLHF。



图片来自3Blue1Brown

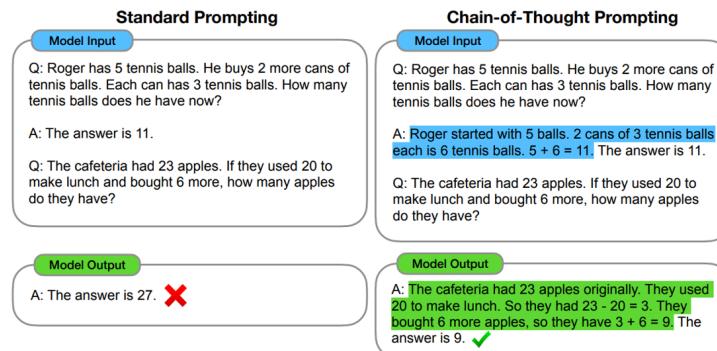
能力：在ChatGPT之前，没多少人相信深度学习能实现通用人工智能；在ChatGPT之后，没多少人不相信深度学习能实现通用人工智能。

加入了思维链的能聊天的生成式预训练变形金刚 | ChatGPT-O

目标：通用人工智能。

时间：2024年。

图像：（思维链这想法最迟22年1月就有了，但发扬光大的是OpenAI）



Jason Wei, 2022-01

ChatGPT-o1

能力：在AIME (American Invitational Mathematics Examination) 上拿到了80+的分数。

ChatGPT-o3

能力：在AIME (American Invitational Mathematics Examination) 上拿到了90+的分数。

方法：可能只是把o1做大了，也可能有其它重要方法

滞留 | RetNet

目标：取代 Transformer

时间：2023年。

图像：

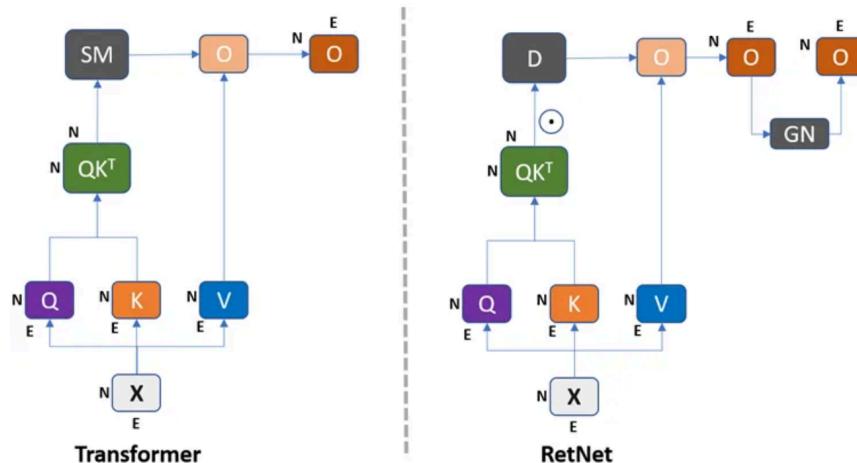


Fig 4: Transformer computation on the left and RetNet on the right. Can you spot the difference?

图片来自Medium

曼巴 | Mamba

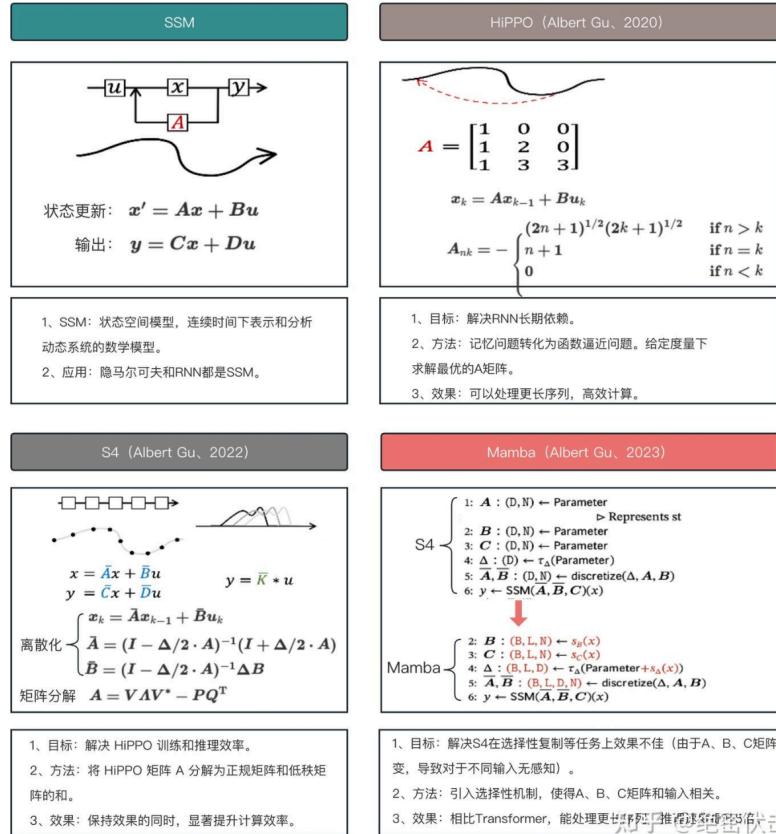
目标：取代 Transformer

时间：2023年。

灵感：控制论

参考：见数学附录

图像：



图片来自网络

问答 | QA

Q: 逻辑回归和单层感知机的相同点和不同点？

A: 两者都有 $w^T x + b$ ，但是激活函数、损失函数不一样。

Q: 为什么深度学习先驱们都在模仿生物神经网络？

A: 现在假设你就是 Rosenblatt 或者 Hinton，你想要造一个很厉害的机器，比如说，原子弹。而你的邻居刚好有一个原子弹，那模仿它显然比从头造要快。

Q: 为什么从八十年代开始又逐渐不模仿了？

A: 后来你发现你邻居的原子弹是用自然选择制造的、费时费力（每次把进化好的留下来，差的丢弃），并且他总是藏着掖着不把核心技术告诉你（神经科学中的学习和记忆很难探究）。那你与其猜他的，还不如从头造。

Q: 为什么有人说BP是生物学不可行的(Biological Implausible)?

A: 因为，在生物神经网络中，误差不太可能一层一层准确地传回来。反着传一层有可能，一层一层传回来也太离谱了。

Q: 那梯度下降是生物学可行(Biological Plausible)的吗？

A: 争议很大。以前有人认为，凡是用全局信息的都不行，而梯度下降用了全局的能量函数；但现在有人提出，神经递质浓度也许可以代表全局信息。人类需要更多、更先进的实验手段才能回答这个问题

Q: 我听说深度学习成功有三个原因——数据、算力、算法，缺一不可，是吗？

A: 前两个肯定不能缺，第三个嘛，有的时候可以缺，有的时候不能

1. AlexNet ($6 \cdot 10^7$ 个参数) 和 LeNet ($6 \cdot 10^4$ 个参数) 相比，仅仅是做大了一千倍。这就是深度学习的苦涩教训——Bigger Is Better。所以不要像 LSTM 那样瞎想了，直接往大了做。
2. Transformer, RLHF, Chain of Thought, 都是人类智慧的结晶，也是 ChatGPT o1 成功的关键。

Lecture 4 & 5: 无监督学习 | Unsupervised Learning

目标：分析无人类标签的数据 (Data Without Human Label)。

聚类 | Clustering

目标：揭示无标签的数据背后的规律。

重要性：你别看他简单，其实非常实用！

方法：把无标签的数据划分为不同的集合。

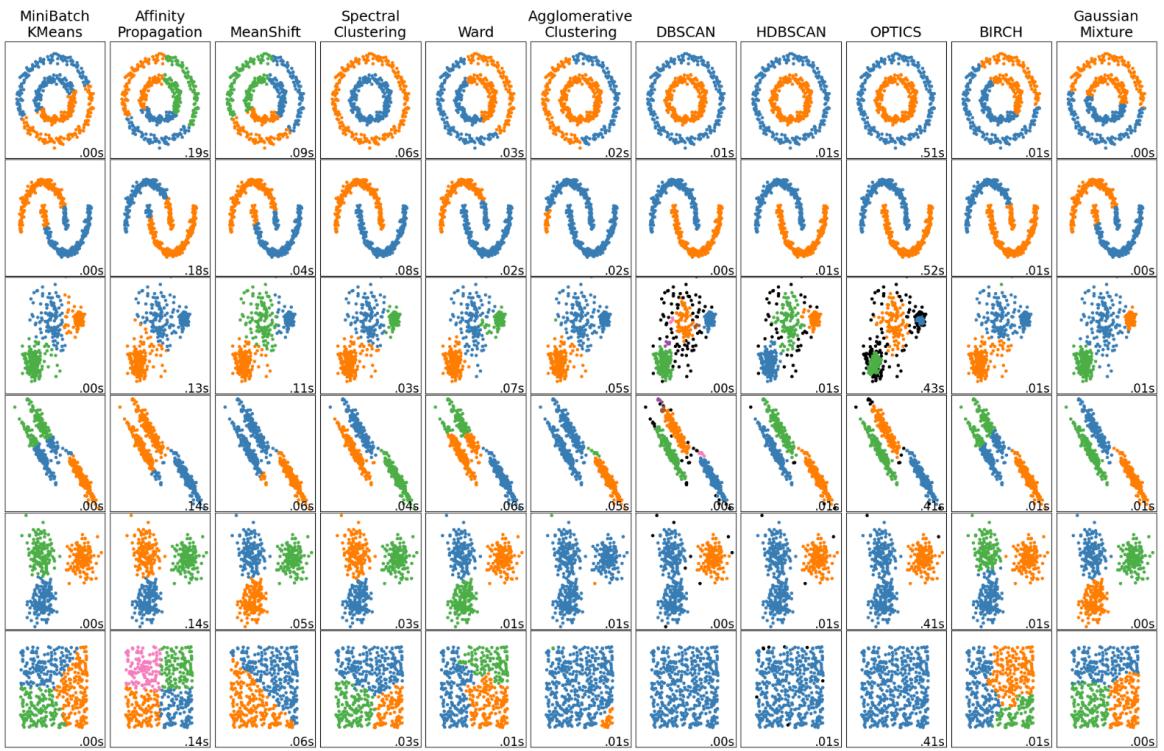
参考：西瓜书第九章；[Sklearn](#)。

性能：见西瓜书9.2（要看一下，可能会考）

距离：见西瓜书9.3（此事较难，因为高维欧几里得空间有一些诡异的性质。所以要仔细思考怎么定义距离。）

图像：

1. 在前两行，DBSCAN（和它右侧的两个小弟）超过了 KMeans。
2. 在第五行，KMeans超过了DBSCAN（和它右侧的两个小弟）。
3. 第六行我也不知道是啥意思。
4. 前二行和第五行，ClusterDP可以收拾。这就是为什么我认为它强过KMeans和DBSCAN。



[图片来自sklearn](#)

基于原型 | Prototype Based

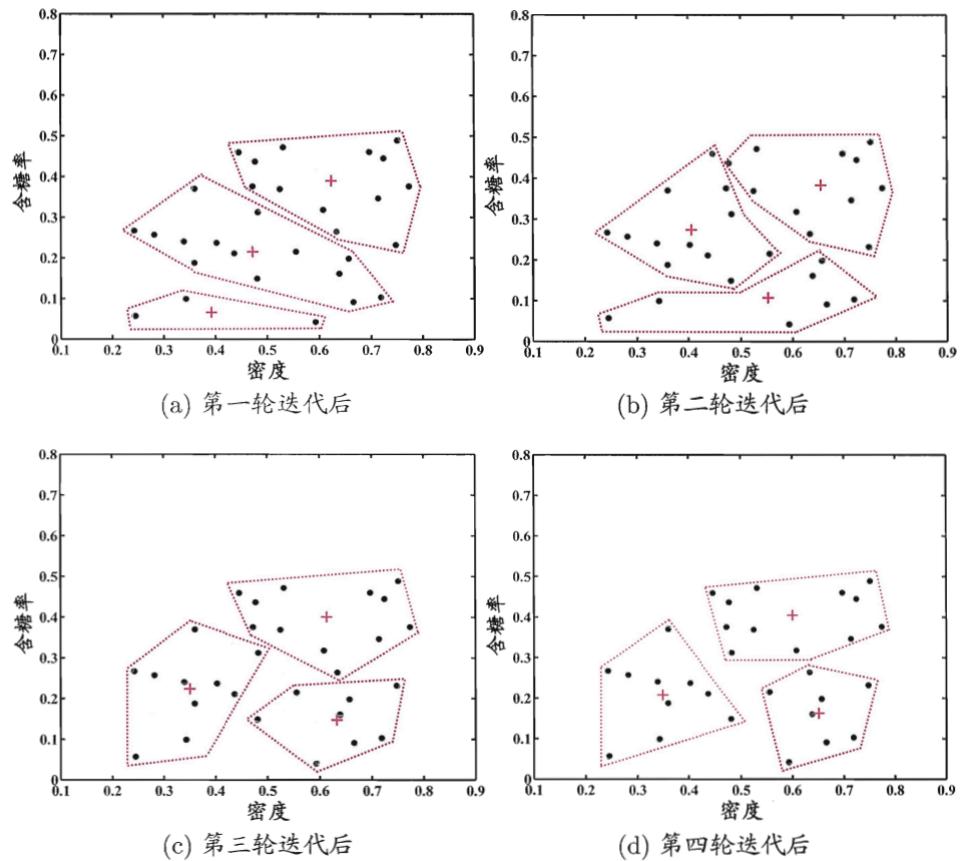
目标：聚类。

灵感：人们聚成小团体时候，每个团体总有一个领导者。我们把它称之为原型 (Prototype)。

PS: Prototype有领导者的意思，也有同一批中第一个被造出的机器的意思。有一个很有名的游戏，叫虐杀原型 (Prototype)，即为后者之意。

K 均值 | K-Means

图像：（看好，每个类都是球状的，即，[凸集合](#)）



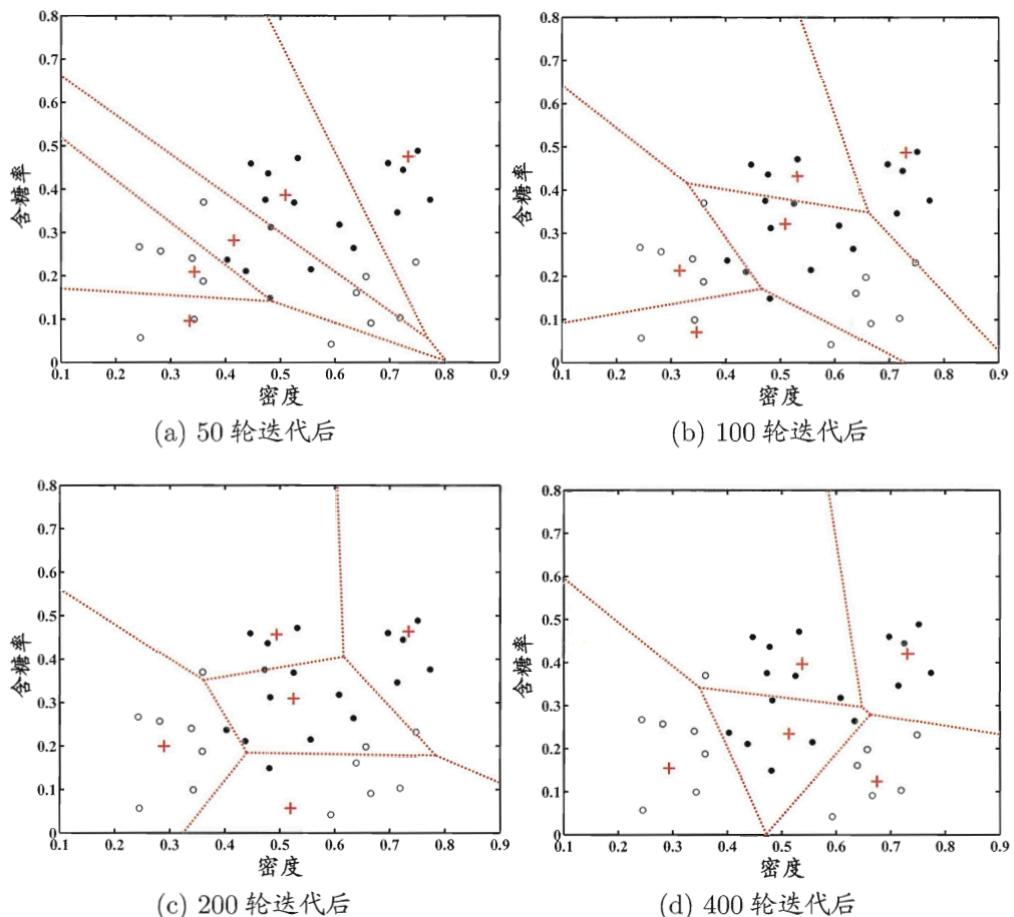
图片来自西瓜书

缺点：

1. 需要预先指定聚类数量 K
2. 只适合球状数据（即，凸的）

学习向量量化 | Learning Vector Quantization

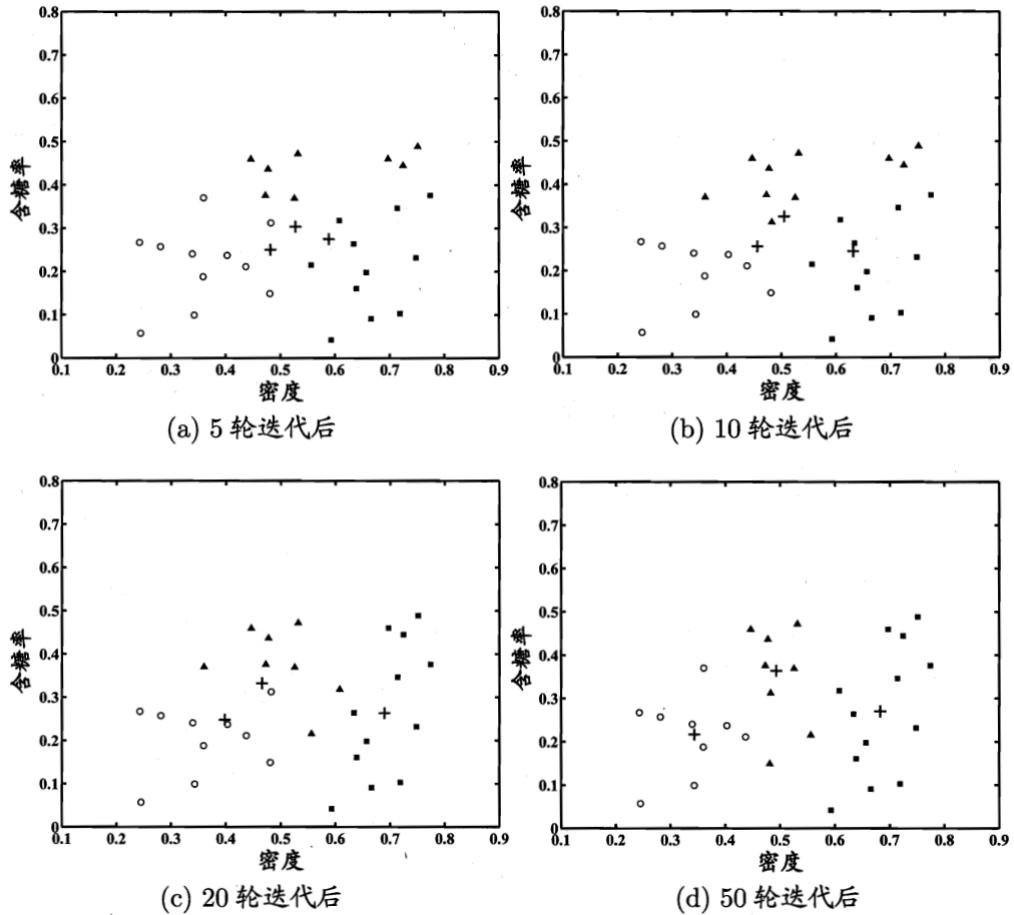
图像：（每个类不一定是球状的）



图片来自西瓜书

高斯混合 | Gauss Mixture

图像：（三个十字是三个高斯分布的中心）



图片来自西瓜书

基于密度 | Density Based

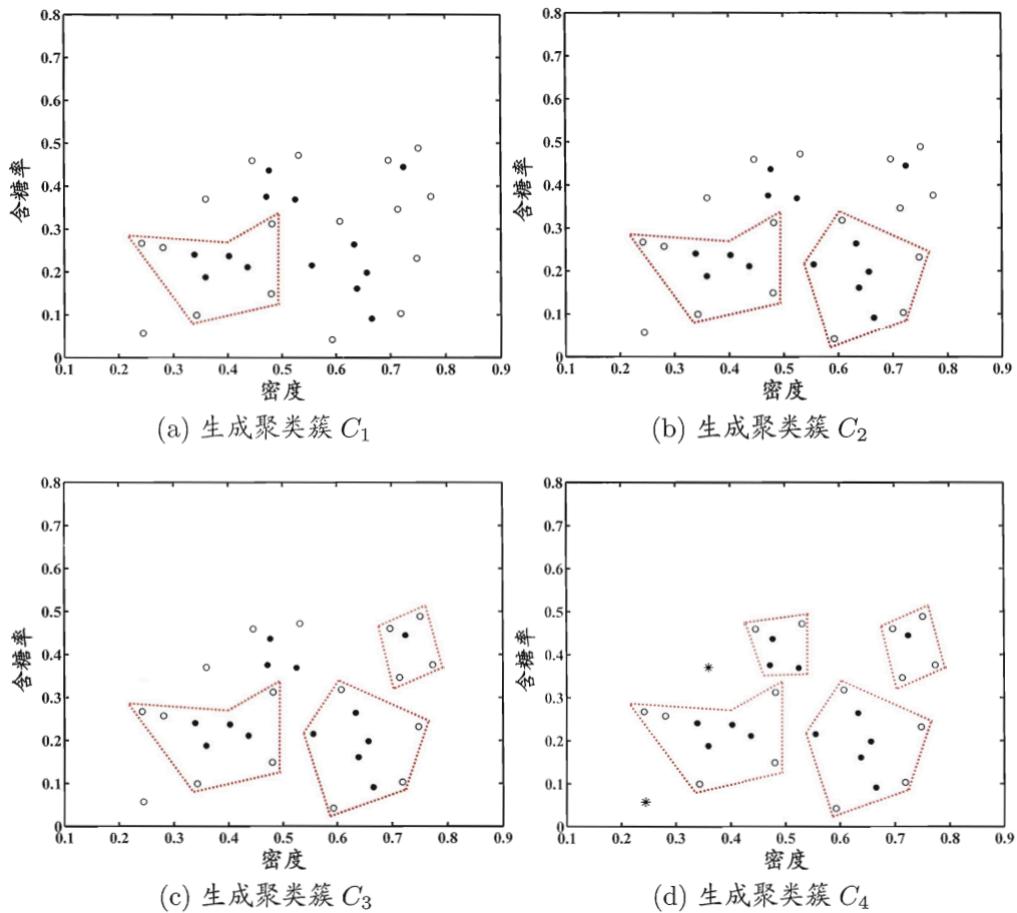
目标：聚类。

灵感：每个集合内部应该靠得比较近，并且距离其他集合都应该有一定距离。

例子

DBSCAN

图像：（每一步生成一个新类）



图片来自西瓜书

缺点：有两个超参数要调。 (K-Means, ClusterDP只有一个)

OPTICS

Search the Internet.

基于层次 | Hierarchy Based

目标：聚类。

灵感：集合之间可能是树状图的关系。

AGNES

图像： (哈哈，很简单吧)

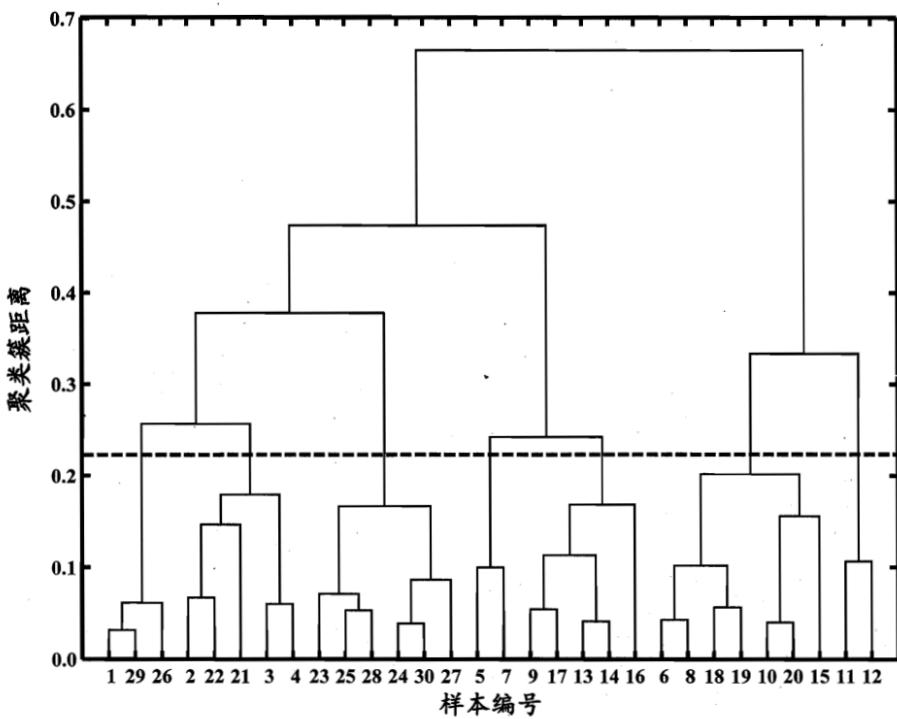


图 9.12 西瓜数据集 4.0 上 AGNES 算法生成的树状图(采用 d_{\max}). 横轴对应于样本编号, 纵轴对应于聚类簇距离.

图片来自西瓜书

DIANA

Search the Internet.

基于原型和密度 | Prototype and Density Based

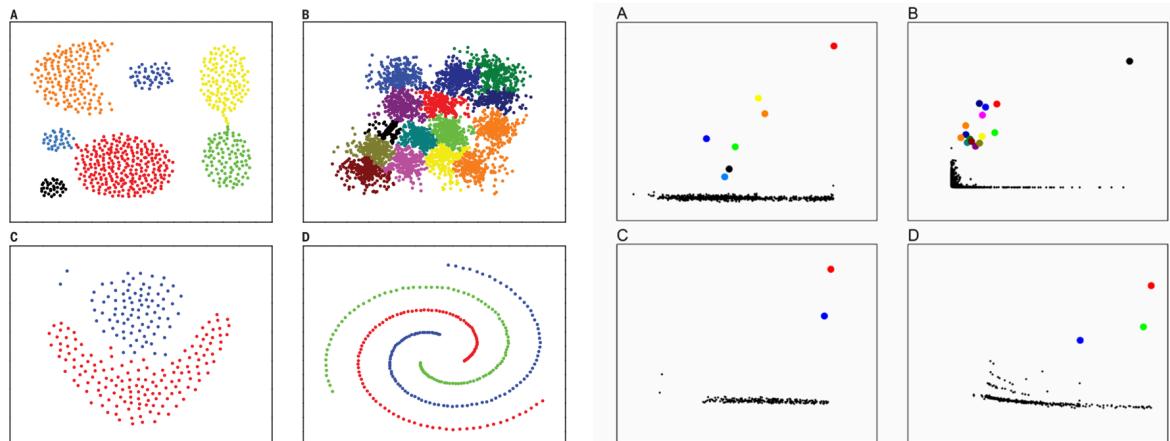
目标: 聚类。

灵感: 既考虑原型, 又考虑密度

寻峰聚类 | ClusterDP

[原论文。](#)

图像: (A图和C图, K-Means已经寄了; D图, 上述的那几个除了估计都做不出来, DBSCAN精确调参有可能)



Rodriguez et al., 2014

重要性：这个算法非常有名、有用、美观，要不然我也不会介绍给你们。

永远不要限制自己：这算法不是机器学习专家提出的，是两个已经转做计算物理/计算生物的人不务正业搞出来的。

后续：他俩2014年发明这算法之后，就继续搞自己的事去了。后来很多学者做了更进一步的工作，有的有用，有的只是水论文。他俩做了从0到1，从1到100交给别人，很酷。

降维 | Dimension Reduction

目标：避免维度诅咒。

重要性：我已说过，机器学习的目标是收拾复杂系统。既然是复杂系统，那么通常一个样本可能有成千上万个特征。这就是有名的“维度诅咒”。这会导致运算时间很长，也会导致模型可解释性变差。

方法：把无标签的数据从高维映射到低维。

参考：西瓜书第十章

线性 | Linear

主成分分析 | Principle Component Analysis

方法：做一个线性变换罢了。

新坐标怎么找？：对 $X^T X$ 做一个特征值分解（Eigen Value Decomposition）罢了。或者说，对 X 做一个奇异值分解，Singular Value Decomposition。 X 是数据集，一个方向是特征1、特征2、特征3.....，另一个方向是样本1、样本2、样本3.....

参考：[看看Josh Starmer的吧，这是他的成名之作](#)。（PS：别人做过并且做得很好的东西，我就不想写了，我只想写别人没写过的东西）

可解释性：比[tSNE](#), [UMAP](#)好，也比用[VAE](#)（一种深度学习方法）做降维好（当然！），但不如原始特征（当然！）。

缺点：线性的，无法捕捉非线性关系。（神经科学里的论文经常上来就做PCA，这是有问题的）

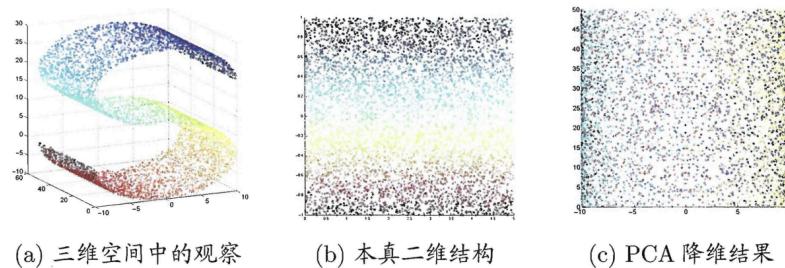


图 10.6 三维空间中观察到的 3000 个样本点，是从本真二维空间中矩形区域采样后以 S 形曲面嵌入，此情形下线性降维会丢失低维结构。图中数据点的染色显示出低维空间的结构。

图片来自西瓜书，一个S就让PCA瞬间破防

线性判别分析 | Linear Discrimination Analysis

参考：西瓜书 3.4。

非线性 | non-linear

例子

1. Kernelized PCA
2. tSNE
3. UMAP

参考：西瓜书 10.4、维基百科。

聚类和降维联用 | Joint Use of Clustering and Dim Reduction

方法：先降维，再聚类

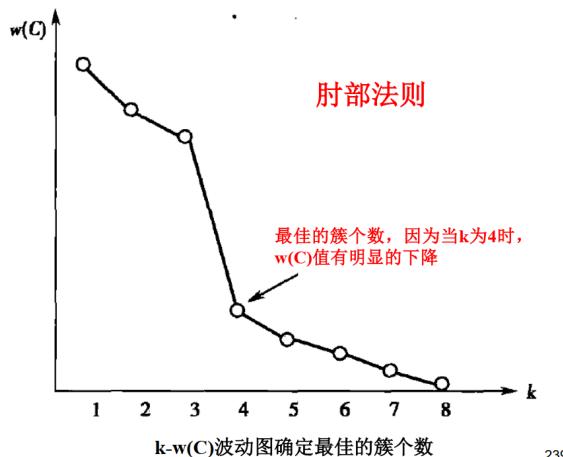
注：通常嘛，拿到一大堆高维数据之后，首先要数据清洗，其次就是降维和特征选择，之后再搞别的事。别的事包括有监督学习、深度学习、聚类。（不过，如果是OpenAI和Google做大语言模型，他们应该不会搞降维和特征选择，但数据清洗依然很重要，[Garbage In, Garbage Out](#)）

问答 | QA

Q: K-Means要求制定集合的数目，那这不就是说它在实际中几乎没用吗？你在实际中咋可能提前知道有多少集合？

A: 可以用肘部法（Elbow Method）。除此之外，还有一些炫酷方法（我没用过）——Silhouette Score, Akaike 信息准则和贝叶斯信息准则。除了它们之外，还可以根据先验知识——至少知道簇的个数的大致范围。比如说，你可以根据电脑使用时间、一教出没时间、环资楼出没时间等对科大本科生聚类。你预先知道科大有10-20个院。

如何确定簇个数？



239

图片来自老师的幻灯片

Q: K-Means要求数据为球状（凸集）的，那这不也是在说它在实际中几乎没用吗？

A: 很多情况下的真实数据确实是球状的。

Q: 高斯混合模型中的概率是频率学派的还是贝叶斯学派的？

A: 贝叶斯学派的。和逻辑回归一样，如果能在某一个点继续收集数据，再用频率估计概率，得到的结果不一定是贝叶斯学派给的结果。至于为什么高斯混合模型能得到 $T = 2.26$ ，我也感到非常迷惑。

强化学习 | Reinforcement Learning

目标：模仿生物，用试错、奖赏、交互实现通用人工智能。

参考：

1. *Reinforcement Learning, An Introduction*。(其实我不喜欢这书，勉强及格罢了——很多地方废话很多，很多地方故意把简单的东西搞复杂——比西瓜书、李航书、*Dive Into Deep Learning*、3Blue1Brown、Josh Starmer、马同学差远了，但我也找不到更好的RL书。**朋友们，知道更好的可以告诉我。**) 如果咱们看不懂一本书或者一节网课，通常是书和课的问题，而不是咱们的问题。
2. [*Reinforcement Learning | Mutual Information*](#): 比上面那本书直观，但总感觉还差点意思。

非参数方法 | Nonparametric

目标：用非参数方法进行强化学习。

参考: *Reinforcement Learning, An Introduction, CH2-CH8*。

图像：无非就是下图的(2.4)罢了。

As you might suspect, this is not really necessary. It is easy to devise incremental formulas for updating averages with small, constant computation required to process each new reward. Given Q_n and the n th reward, R_n , the new average of all n rewards can be computed by

$$\begin{aligned} Q_{n+1} &= \frac{1}{n} \sum_{i=1}^n R_i \\ &= \frac{1}{n} \left(R_n + \sum_{i=1}^{n-1} R_i \right) \\ &= \frac{1}{n} \left(R_n + (n-1) \frac{1}{n-1} \sum_{i=1}^{n-1} R_i \right) \\ &= \frac{1}{n} \left(R_n + (n-1)Q_n \right) \\ &= \frac{1}{n} \left(R_n + nQ_n - Q_n \right) \\ &= Q_n + \frac{1}{n} [R_n - Q_n], \end{aligned} \tag{2.3}$$

which holds even for $n = 1$, obtaining $Q_2 = R_1$ for arbitrary Q_1 . This implementation requires memory only for Q_n and n , and only the small computation (2.3) for each new reward.

This update rule (2.3) is of a form that occurs frequently throughout this book. The general form is

$$\text{NewEstimate} \leftarrow \text{OldEstimate} + \text{StepSize} [\text{Target} - \text{OldEstimate}]. \tag{2.4}$$

Chapter 2 of *Reinforcement Learning, An Introduction*

参数方法 | Parametric

目标：用参数方法进行强化学习，从而把深度学习加进来。

参考: *Reinforcement Learning, An Introduction, CH9-CH13*。

图像：引入一个参数 \vec{w} 。当然啦，通常它就是深度学习里的那个 \vec{w} 。

In this chapter, we begin our study of function approximation in reinforcement learning by considering its use in estimating the state-value function from on-policy data, that is, in approximating v_π from experience generated using a known policy π . **The novelty in this chapter is that the approximate value function is represented not as a table but as a parameterized functional form with weight vector $\mathbf{w} \in \mathbb{R}^d$.** We will write $\hat{v}(s, \mathbf{w}) \approx v_\pi(s)$ for the approximate value of state s given weight vector \mathbf{w} . For example, \hat{v} might be

Chapter 9 of *Reinforcement Learning, An Introduction*

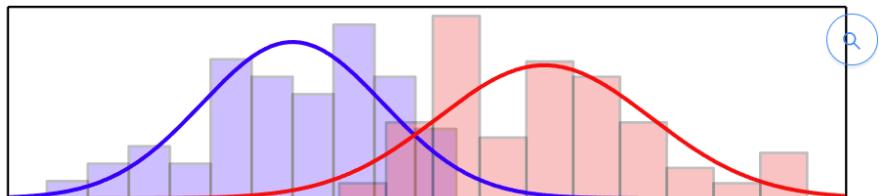
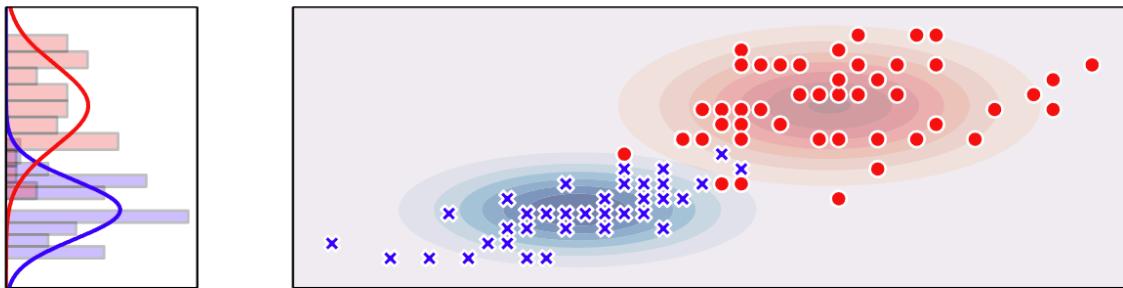
其他 | Others

贝叶斯 | Bayes

目标：用贝叶斯学派的方式思考机器学习

方法：最简单的是朴素贝叶斯，即假设所有东西都独立

图像：



[图片来自马同学](#)

趣事：曾经有人（休谟）问拉普拉斯——**如果不知道任何和太阳有关的物理规律，怎么计算太阳明天会升起的概率？** 拉普拉斯采用贝叶斯的方式思考，并运用拉普拉斯平滑，给出了一个有趣的答案。你们可以猜一猜。

概率图 | Probability Graphical Model

目标：因果推断

图像：

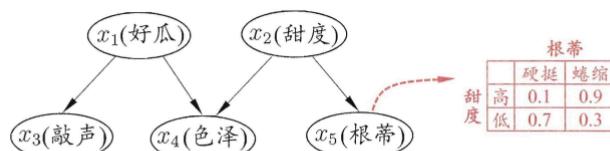


图 7.2 西瓜问题的一种贝叶斯网结构以及属性“根蒂”的条件概率表

图片来自西瓜书

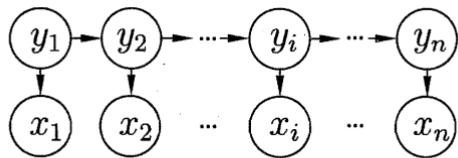


图 14.1 隐马尔可夫模型的图结构

图片来自西瓜书

懒惰学习 | Lazy Learning

目标：节省训练时间。

方法：懒惰地学习。等收到测试集再开始学习。

图像：以 K 近邻 (K Nearest Neighbor) 为例

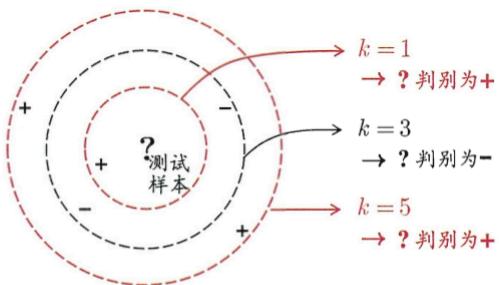


图 10.1 k 近邻分类器示意图. 虚线显示出等距线; 测试样本在 $k = 1$ 或 $k = 5$ 时被判别为正例, $k = 3$ 时被判别为反例.

图片来自西瓜书

特征选择 | Feature Selection

目标：对付唯独诅咒

方法：选出有用的特征，丢弃无用的特征

例子：请了解下RELIEF，是非常漂亮的算法，见西瓜书CH11；我们以前学的正则化也可以视为特征选择。

Pankaj Mehta 老兄的其他代码 | Other Codes of Pankaj Mehta

我都概括一下好了：

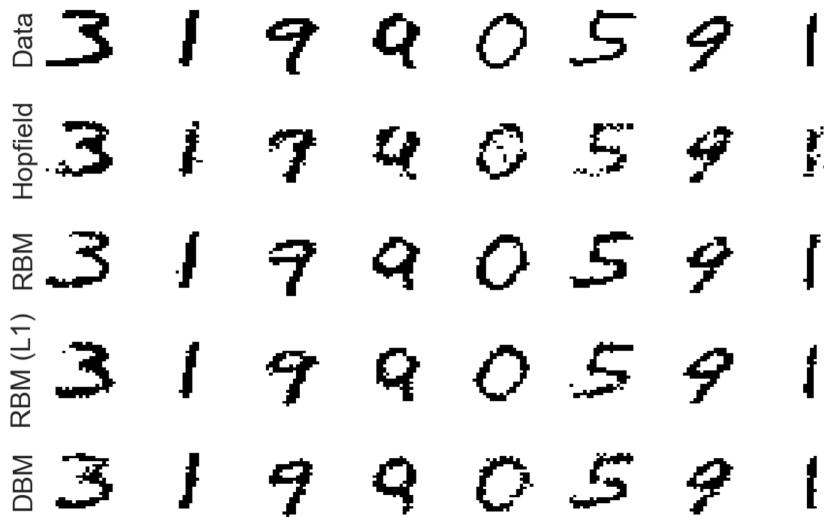
8：用Bagging集合多个感知机玩人造数据集。

10：用XGBoost玩超对称数据集（我也不知道超对称是什么意思）

14：用Pytorch造CNN玩Ising

16：用EM算法（贝叶斯算法之一）玩硬币。

17：用 Hopfield 和 Boltzmann Machine 重构、生成新的手写数字。



图片来自Pankaj的[NB17](#)

18: 用 Hopfield 和 Boltzmann Machine 重构、生成新的Ising。

19: 用VAE生成新的手写数字。

20: 用VAE生成新的Ising。

库 | Libraries

1. [sklearn](#): 开源组织负责，始于2007年。
2. [tensorflow](#): Google开发，始于2015年。
3. [pytorch](#): Meta开发，始于2016年。

问答 | QA

Q: 我写的算法怎么才能被收录进sklearn? (来自某同学)

A: 只要你写得够好，就可以。贡献方法见[此处](#)。

大语言模型 | Large Language Models

竞技场 | Arena

<https://lmarena.ai/>

目标：让用户投票决定两个模型谁更厉害。

可以用什么：GPT, Gemini, Grok, LLAMA, Claude, qwen, Yi, GLM

我能做什么：多用。多传播（你们可以把它传播到班群或者院群）。我感觉这是世界上为数不多的又对自己有利又对全人类有利的事情。

ChatGPT-3.5/4 水平 | ChatGPT-3.5/4 Level

美国

1. [ChatGPT | OpenAI](#) —— 最近和微软闹掰了，投奔苹果的怀抱。
2. [Gemini | Google](#) —— 最有钱。一心取代OpenAI。
3. [Claude | Anthropic](#) —— OpenAI 前员工，一对兄妹，创办的公司。一心取代OpenAI。
4. [LLAMA | Meta](#) —— 和谷歌一样有钱。一心取代OpenAI。
5. [Grok | xAI](#) —— 和谷歌一样有钱。一心取代OpenAI。
6. [Nova | Amazon](#) —— 和谷歌一样有钱，但应该更侧重于和别人合作。
7. [Nemotron | NVIDIA](#) —— 和谷歌一样有钱，但应该更侧重于和别人合作。
8. [Command R | Cohere](#) —— 初创公司。Transformer最年轻作者成立的。
9. [Perplexity | Perplexity](#) —— 初创公司。以搜索见长。

中国

1. [豆包 | 字节跳动](#)
2. [Kimi | 月之暗面](#)
3. [GLM | 智谱](#) —— 和清华合作。应该是此节唯一一个走Encoder-Decoder路线的。
4. [Qwen | 阿里巴巴](#)
5. [星火 | 科大讯飞](#)
6. [文心一言 | 百度](#)
7. [Yi | 零一万物](#)
8. [Deep Seek | 深度求索](#)

法国

1. [Mistral | Mistral](#)

我的评价：以上，使用体验都差不多。

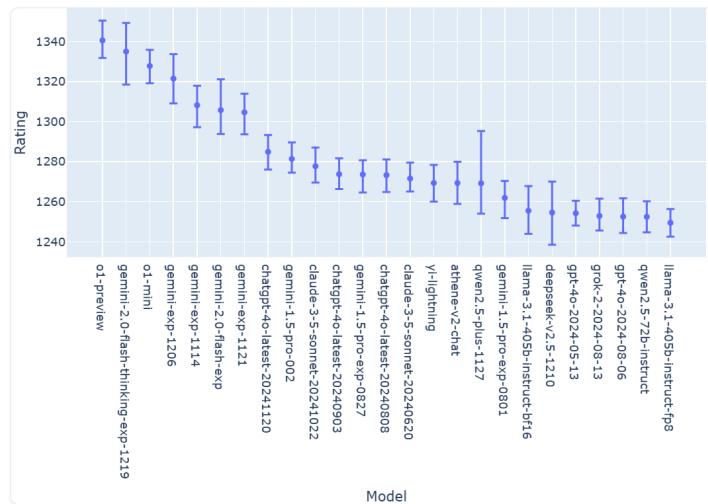
开源和闭源：以上要么是开源的，要么是免费使用的。

ChatGPT-o1 水平 | ChatGPT-o1 Level

1. ChatGPT-o1 | OpenAI
2. Gemini-Thinking | Google

More Statistics for Chatbot Arena - Math

Figure 1: Confidence Intervals on Model Strength (via Bootstrapping)



图片来自竞技场

3. Deep Seek V3-Thinking | Deep Seek (我问了一个问题感觉还不错。再过一周，竞技场排名就会出来，到时看看！)

广告 | ADS

深度学习的三起三落 | Ups and Downs of DL

1. 第一次兴起: 1950, Perceptron
 2. 第一次衰落: 1969-1982
 3. 第二次兴起: 1982, Hopfield
 4. 第二次衰落: 1990-2006
 5. 第三次兴起: 2012, AlexNet
 6. 第三次衰落: 2020-2022
 7. 第四次兴起: 2022, ChatGPT

前两次衰落时，深度学习很少能申请到funding——于是，大多数人都离开了。

只有极少数人(Geff Hinton, Yann LeCun, Jungen)坚持了下来，坚信这种方法一定能实现AGI。

欲知更多，请来第三次习题课。

深度学习 vs 神经科学 | DL vs Neuro

1. Perceptron的灵感来自于[Louis Lapique](#), [Hodgkin & Huxley](#)对于单神经元的研究, 以及[Hebb](#)对于突触强度变化的猜测。
 2. LeNet (第一个CNN) 的灵感来自于[Hubel & Wiesel](#)对于哺乳动物视觉的研究。AlexNet仅仅是把LeNet做大了。

欲知更多，请来第三次习题课。

强化学习 vs 神经科学 | RL vs Neuro

1. 2000年左右，[神经科学家发现多巴胺浓度和强化学习中的Reward Prediction Error非常一致](#)。这就是你小时候常常听说的——吃糖可以产生多巴胺。

欲知更多，请来第三次习题课。

数学附录 | Math Appendix

最小二乘的解析解 | Analytic Solution of Least Square

重复：线性回归和最小二乘是一个意思。

在最小二乘中，E的定义如下

$$E(\vec{w}) := \sum_{i=1}^n (y_i - \vec{w} \cdot \vec{x}_i)^2 = Y^T Y + w^T X^T X w - 2w^T X^T Y$$

求一阶导，我们得到

$$\frac{\partial E}{\partial \vec{w}} = 2(X^T X w - X^T Y)$$

于是

$$\frac{\partial E}{\partial \vec{w}} = 0 \Rightarrow X^T X \hat{w} = X^T Y$$

那么有两种情况：

1. 当 $X^T X$ 可逆时

$$\hat{w} = (X^T X)^{-1} X^T Y = X^\dagger Y$$

where $X^\dagger := (X^T X)^{-1} X^T$ and X^\dagger is Moore-Penrose inverse.

2. 当 $X^T X$ 不可逆时， $X^T X \hat{w} = X^T Y$ 这个线性方程组有多个解，选择哪一个作为结果由算法的偏好决定。

什么时候 $X^T X$ 可逆呢？

线性代数中有一个很有名的定理： X 列满秩 $\Leftrightarrow X^T X$ 可逆。而 X 的行数越多，线性无关的行向量很可能越多。线性无关的行向量越多，行秩、秩、列秩越大， X 越容易列满秩。以下是几个例子：

- 在波士顿房价数据集中， X 有506行和14列， X 大概率列满秩。
- 在生物信息学中，通常feature数目比sample多， X 大概率列不满秩。

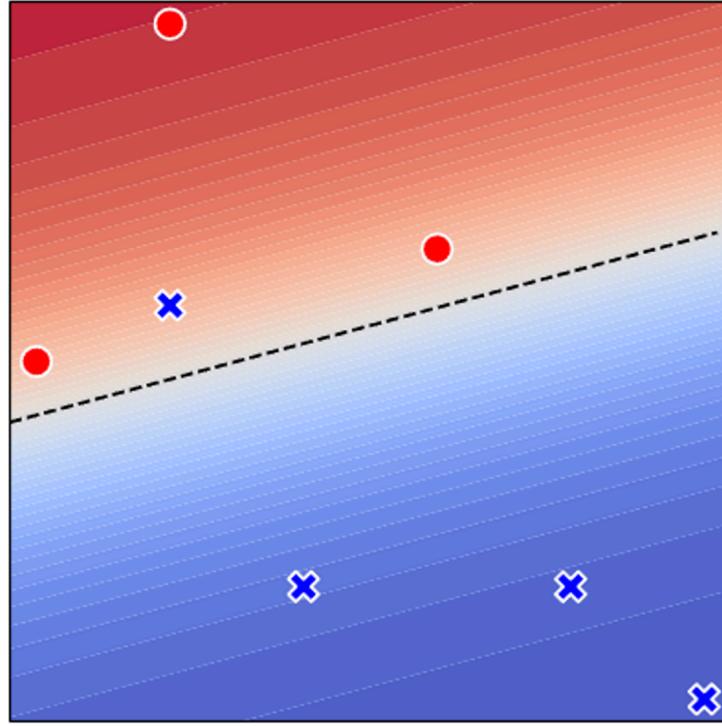
逻辑回归的数值解 | Numerical Solution of Logistics Regression

逻辑回归（周志华书中称之为对数几率回归）中，我们假设

$$\begin{cases} P(y = +1 | \vec{x}, \vec{w}, b) = \frac{1}{1+e^{-\vec{w} \cdot \vec{x}}} \\ P(y = 0 | \vec{x}, \vec{w}, b) = \frac{1}{1+e^{+\vec{w} \cdot \vec{x}}} \end{cases}$$

注意，这是一个假设。

正是这个假设赋予了LR概率含义。 $P(y = +1|\vec{x}, \vec{w}, b)$ 即样本 \vec{x} 是正例的概率， $P(y = 0|\vec{x}, \vec{w}, b)$ 即样本 \vec{x} 是反例的概率。如下图所示，离分类直线越远，属于对应类的概率越大，在直线上的点属于任何一点的概率都为0.5。



[图片来自马同学](#)

接下来推导E：

把伯努利分布的两个P写到一起

$$P(y_i|\vec{x}_i, \vec{w}, b) = (P(y_i = +1|\vec{x}_i, \vec{w}, b))^{y_i} (P(y_i = 0|\vec{x}_i, \vec{w}, b))^{1-y_i}$$

似然函数为

$$\text{likelihood}(Y|X, \vec{w}, b) = \prod_{i=1}^n P(y_i|\vec{x}_i, \vec{w}, b)$$

对数似然函数为

$$\begin{aligned} \ln(\text{likelihood}(Y|X, \vec{w}, b)) &= \sum_{i=1}^n \ln(P(y_i|\vec{x}_i, \vec{w}, b)) \\ &= \sum_{i=1}^n y_i \vec{w} \cdot \vec{x}_i - \ln(1 + e^{\vec{w} \cdot \vec{x}_i}) \end{aligned}$$

最大化对数似然函数等价于最小化负对数似然函数，因此我们可以把E定义为

$$E := \sum_{i=1}^n -y_i \vec{w} \cdot \vec{x}_i + \ln(1 + e^{\vec{w} \cdot \vec{x}_i})$$

It's hard to calculate $\frac{\partial E}{\partial \vec{w}}$ (According to Wikipedia, it doesn't have a close-form solution), so we use numerical method like gradient descent and Newton method to get the optimal solution $\hat{\vec{w}}$.

P.S.: 此处的概率不是频率学派的概率，是贝叶斯学派的概率。逻辑回归的目的是给线性模型找一个概率解释，恰好 $\frac{1}{1+e^{-x}}$ 可以输出一个0到1之间的值，而且直觉上也符合（ x 越大越接近于1，越小越接近于0），就吧它的输出当成概率好了。如果能在某一个点继续收集数据，再用频率估计概率，得到的结果不一定是逻辑回归给的结果。

香农熵 | Shannon Information

数学公式：

$$Shannon\ Info := - \sum_i P_i \log_2 P_i$$

目标：定量地定义“信息”。

灵感： $S = k \ln W$; 猜数游戏。

注：我录制过一个视频——[如何从玻尔兹曼公式推导出香农对于信息的定义？](#)。看了你就懂了。

基尼指数 | Gini-Simpson index

数学公式：

$$Gini := \sum_i \sum_{j \neq i} P_i P_j$$

目标：定量地衡量一个集合的多样性。

灵感：它等于随机抽两个样本，不属于同一类的概率。

注：

1. 它仅仅是[多样性指标](#)之一。
2. 它和经济学中用来衡量贫富差距的[Gini Coefficient](#)不是一个意思。

自助法 | Boot-Strap

数学公式：

“自助法”(bootstrapping)是一个比较好的解决方案，它直接以自助采样法(bootstrap sampling)为基础 [Efron and Tibshirani, 1993]. 给定包含 m 个样本的数据集 D ，我们对它进行采样产生数据集 D' : 每次随机从 D 中挑选一个样本，将其拷贝放入 D' ，然后再将该样本放回初始数据集 D 中，使得该样本在下次采样时仍有可能被采到；这个过程重复执行 m 次后，我们就得到了包含 m 个样本的数据集 D' ，这就是自助采样的结果。显然， D 中有一部分样本会在 D' 中多次出现，而另一部分样本不出现。可以做一个简单的估计，样本在 m 次采样中始终不被采到的概率是 $(1 - \frac{1}{m})^m$ ，取极限得到

$$\lim_{m \rightarrow \infty} \left(1 - \frac{1}{m}\right)^m \rightarrow \frac{1}{e} \approx 0.368, \quad (2.1)$$

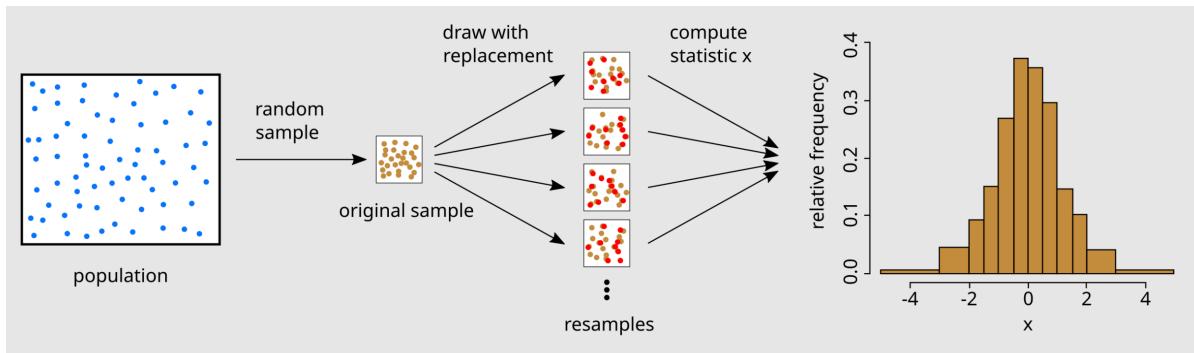
即通过自助采样，初始数据集 D 中约有 36.8% 的样本未出现在采样数据集 D' 中。于是我们可将 D' 用作训练集， $D \setminus D'$ 用作测试集；这样，实际评估的模型与期望评估的模型都使用 m 个训练样本，而我们仍有数据总量约 $1/3$ 的、没在训练集中出现的样本用于测试。这样的测试结果，亦称“包外估计”(out-of-bag estimate)。

图片来自西瓜书

目标：数据增强（即，当你数据不足时，如何凭空造出更多数据？）。

灵感：来自统计学，她们估算统计量和做假设检验时经常用。

图像：



[图片来自维基百科](#)

注：

1. 其名字来自于《吹牛大王历险记》。
2. 作为物理学家，你可以认为——此法可以在不能再做实验时，凭空造出无数组实验。

控制论 | Control Theory

起源：控制论最早起源于麦克斯韦对于回旋加速器的研究。

目标：施加外力，让机器保持在你想要的范围内。

例子：我只举一个例子

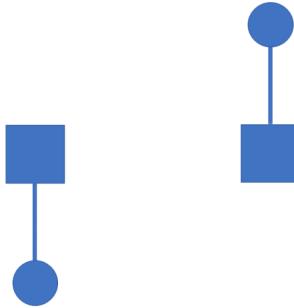


Figure 8: Two Fixed Points

[图片来自我的控制论笔记](#)

通过合适的控制，我们可以单摆稳定在它的不稳定平衡点

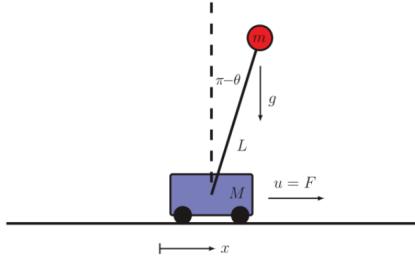


Figure 13: Inverted Pendulum on a Cart[2]

[图片来自我的控制论笔记](#)

我们可以证明，只要你知道系统的方程，你可以把它控制在任何地方。

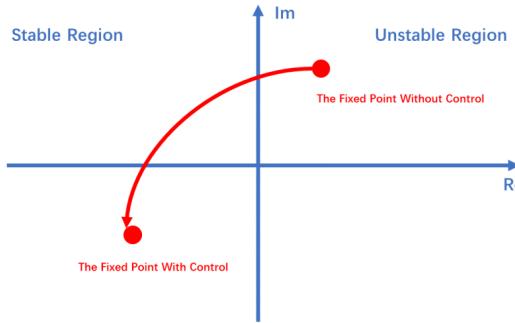


Figure 6: Drive the Fixed Point

其实在MATLAB里用一行代码就能实现——`K = place(A, B, eigs_wanted)`。

To add control, we simply add Bu to the right hand side of [Equation 6](#):

$$\dot{X} = AX + Bu \quad (9)$$

where $X \in \mathbb{R}^n$, $u \in \mathbb{R}^q$, $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times q}$.

To make things simpler, we shall not consider the estimator at the first place and we assume linear relation of u and X : $u = -KX$.

Put them together:

$$\begin{cases} \dot{X} &= AX + Bu \\ u &= -KX \end{cases} \quad (10)$$

where $K \in \mathbb{R}^{q \times n}$.

[Equation 10](#) is the math form of [Figure 4](#).

So

$$\dot{X} = (A - BK)X \quad (11)$$

Mathematicians have proved that, if we define $C := [B, AB, A^2B, \dots, A^{n-1}B]$

$$\text{rank}(C) = n \Leftrightarrow (A - BK) \text{ has any eigenvalues} \Leftrightarrow X \text{ can reach any point in } \mathbb{R}^n \quad (12)$$

And we call the system *controllable* when $\text{rank}(C) = n$.

There is an one-line-code in MATLAB to calculate C .

```
1 C = ctrb(A, B)
```

Attention. In real world, A, B are usually fixed for a given system while K is not. In other worlds, you can only change K , rather than A, B .

There is an one-line-code in MATLAB to calculate K , based on eigenvalues you want and known A, B .

```
1 K = place(A, B, eigs_wanted)
```

There is a similar concept to controllability, observability.

Consider

$$\begin{cases} \dot{x} = Ax + Bu \\ y = CX \\ u = -K\hat{x} \end{cases} \quad (13)$$

where $y \in \mathbb{R}^m, C \in \mathbb{R}^{m \times n}$.

Equation 13 is the math form of [Figure 5](#).

We call a system *observable* if:

$$\text{rank}(O) = n \quad (14)$$

where $O := [C, CA, CA^2, \dots, CA^{n-1}]^T$.

Roughly speaking, a system is observable means that y conveys all the information of x and u . To be more precise, a system is said to be observable if and only if, for every possible trajectory of state and control vectors (x and u), the current state vector x can be estimated using only the information from outputs y . On the other hand, if the system is not observable, there are state-control trajectories that are not distinguishable by only measuring the outputs.

There is also an one-line-code in MATLAB to calculate O.

```
O = obsv(A, C)
```

[图片来自我的控制论笔记](#)

作为一些特例，单摆能控制，双摆、三摆自然也能控制。

YouTube上的实验视频（强烈建议画几秒看看）：[Inverted Single Pendulum](#), [Inverted Triple Pendulum](#).

为什么要在这里讲控制论？因为 Mamba 用的就是控制论方程！（我笔记中的式子13）（[式子13的出处](#)）

深度学习掀起的科学革命 | Science Revolution Caused by DL

如果终将失事，你是否还会扬帆远航？

有一位科学家叫 Jim Gray (1944-2007)。1998年因“对数据库和事务处理研究的开创性贡献以及系统实现中的技术领导地位”获得图灵奖（我至今也不理解他的工作，毕竟我非CS出身）。

他很喜欢一个人出去航海。2007年，他打算前往一个小岛，把母亲的骨灰撒在岛上。

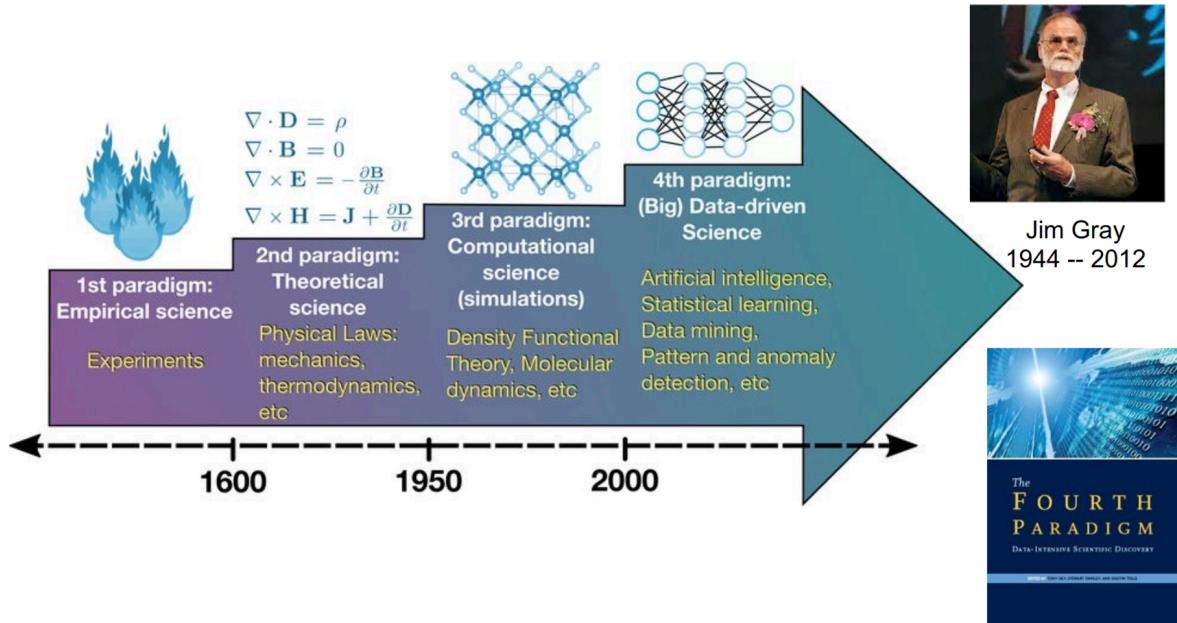
但是，他再也没有回来。

他的朋友、学生中有不少都是计算机视觉的高手。他们在几千张影像中希望找到Gray的身影，但是一无所获。

Jim Gray 极富洞察。2006年左右（注意这个年份），他提出，在大数据时代，我们正在经历第四次科学革命。（那时候是深度学习第二次寒冬，他指的不是深度学习，仅仅是大数据。）他认为：

1. 第一范式：人类无脑地做实验，比如学会用火，比如建造埃及金字塔，比如发明造纸术。
2. 第二范式：始于伽利略。先提出一个理论，再做实验看实验和理论是否符合。如果不符合，修改理论。
3. 第三范式：始于电子计算机。计算机模拟现实中难做的实验。
4. 第四范式：始于大数据。Jim 认为，辅以大数据，计算机能理解复杂系统。

他去世后，他的思想、演讲、草稿被几位朋友整理成一本书，[the Fourth Paradigm](#)，并出版。他生前效力于微软公司，想必后者在书籍的整理和出版中也帮了不少忙。



Slides from [Pengcheng Zhou](#)

Science Paradigms

- Thousand years ago:
science was **empirical**
describing natural phenomena
- Last few hundred years:
theoretical branch
using models, generalizations
- Last few decades:
a computational branch
simulating complex phenomena
- Today: **data exploration** (eScience)
unify theory, experiment, and simulation
 - Data captured by instruments or generated by simulator
 - Processed by software
 - Information/knowledge stored in computer
 - Scientist analyzes database/files using data management and statistics

Figure 1 of [the Fourth Paradigm](#)

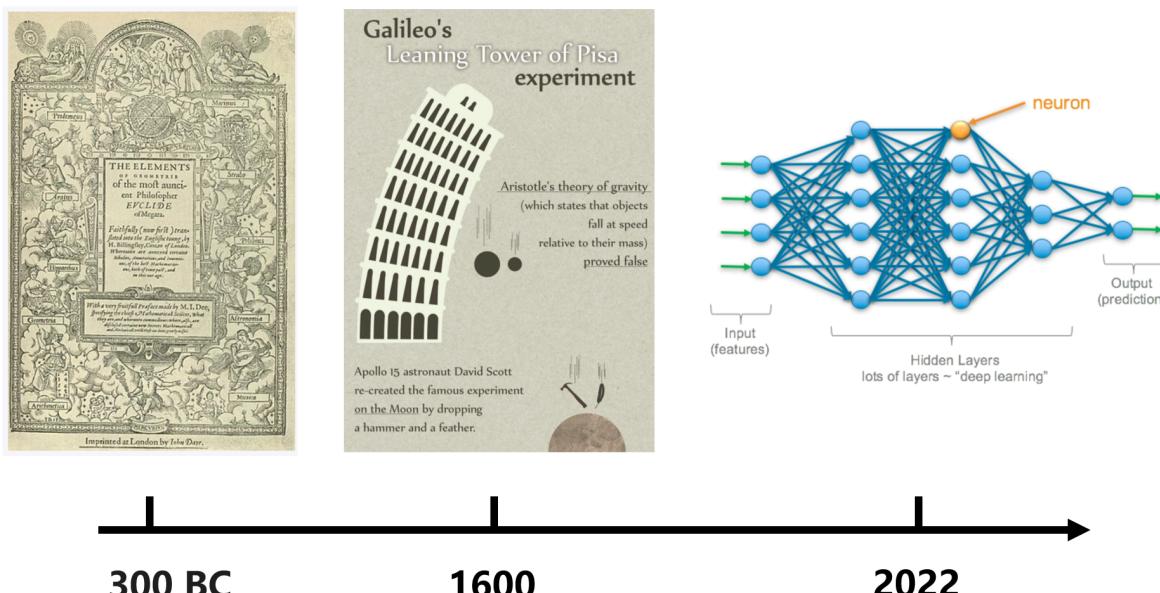
2014年，一家AI公司，[第四范式](#)，在中国成立。和大家猜的一样，[创始人说，名字是为了致敬Jim Gray。](#)

我的想法和 Jim Gray 不太一样：

1. 伽利略之前的无脑实验：当然对人类进步很重要，当然是工程，但是似乎不能称之为自然科学。
2. 计算机模拟（这正是我们这门课，计算物理）：我认为和实验没有本质区别，只是把现实中难做的实验拿到计算机上了。依旧是“先提出一个理论，再做实验（或模拟）看实验（或模拟）和理论是否符合”。至于用计算指导实验，我认为，类似于预实验。

我觉得，人类历史上有三次科学革命：

1. 欧几里得等人撰写《几何原本》
2. 伽利略撰写《两种新科学》
3. Deep Learning (AlphaGo, AlphaFold, ChatGPT, AlphaGeometry)



第一点标志着数学的真正开端。我们把一些规则当成公理，再用它们构建我们的体系，而不是只是算术和画图。这就是数学。后来，统计的皮尔逊 (Karl Pearson)、费雪 (Ronald Fisher)，计算机的布尔 (Bull)、香农 (Claude Shannon)、司马贺 (Herbier Simon)，也都采用类似的思考方式。

第二点标志着自然科学的真正开端。先提出一个理论，再做实验，看实验和理论是否符合，而不是单纯思辨或者无脑实验。这就是自然科学。后来，化学的拉瓦锡 (Antoine Lavoisier)，生物的孟德尔 (Gregor Mendel)、巴斯德 (Louis Pasteur)、巴甫洛夫 (Ivan Pavlov)，医学的弗莱明 (Alexander Fleming)，心理学的冯特 (Wilhelm Wundt)，社会科学的众多政治家（各个经济政策都会先在几个城市搞试点），也用类似的思考方式。（天文和地质因实验较难，所以观察、模拟仍占主流。这只是无奈之举。如果能做实验他们肯定做实验。）

第三点，来自于很多人心底的疑惑——上帝的语言真的是数学吗？自然界的规律真的都可以用简单的数学语言或自然语言描述吗？也许答案是否定的。到目前为止，人类能理解的规律要么能用几句数学语言描述、要么能用几张图描述、要么能用几句自然语言描述。但复杂系统也许不能。而未来，深度学习可以充当人类和自然界（特别是复杂系统）的“翻译软件”。

在新的范式下，收拾复杂系统只需要三步：

1. 找到输入
2. 找到输出
3. 把输入和输出扔给人工神经网络

More Is Different，刚刚提出时是凝聚态物理的“独立宣言”，现在是笼罩整个人类社会的诅咒。

1. 人类能理解DNA -> RNA -> 蛋白质 (Crick Dogma)；但一个蛋白质发生改变后，会对个体的性状造成什么改变？
2. 人类能理解原子分子之间的相互作用 (Newton or Schrodinger)；但基于物理的方法在预测蛋白质结构上如此彻底地败给深度学习。

3. 人类能理解单神经元并准确地给出动作电位 (Hodgkin & Huxley); 但神经元一多，人类就束手无策。没有任何一个人知道生物神经网络如何实现学习和记忆，也没有任何一个人知道它为什么需要睡眠。
 4. 人类发明了CNN, RNN, Transformer，知道其中的每一个单元、每一个相互作用、全部的学习方法；但人类无法理解它们为什么工作得如此之好，也无法预测它们对新事物的响应。
-

如果终将失事，我仍会扬帆远航。

因为航行本身就是一种追寻，是对未知的探索，是对梦想的执着。

因为扬帆远航的瞬间，生命才真正充盈。

因为那是一种无惧无悔的勇气。

因为那是在过程中寻找意义的慨然。

与其在岸边踟蹰不前，不如以热忱拥抱风雨，哪怕前路注定风高浪急，甚至必将失败，过程却充满了属于自己的光辉。

你呢，会选择出发吗？

深度学习掀起的技术革命 | Technology Revolution Caused by DL

和上一小节极富争议不同，深度学习将掀起新一轮技术革命、解放生产力，是不争的事实。我想它至少可以和以下几次技术革命并列：

1. 火
2. 第一次工业革命
3. 第二次工业革命
4. 电子计算机

在十年的时间尺度上，深度学习可能导致失业率飙升，也可能导致经济危机。但从百年的时间尺度上，这当然是好事。

这是最好的时代，这是最坏的时代。