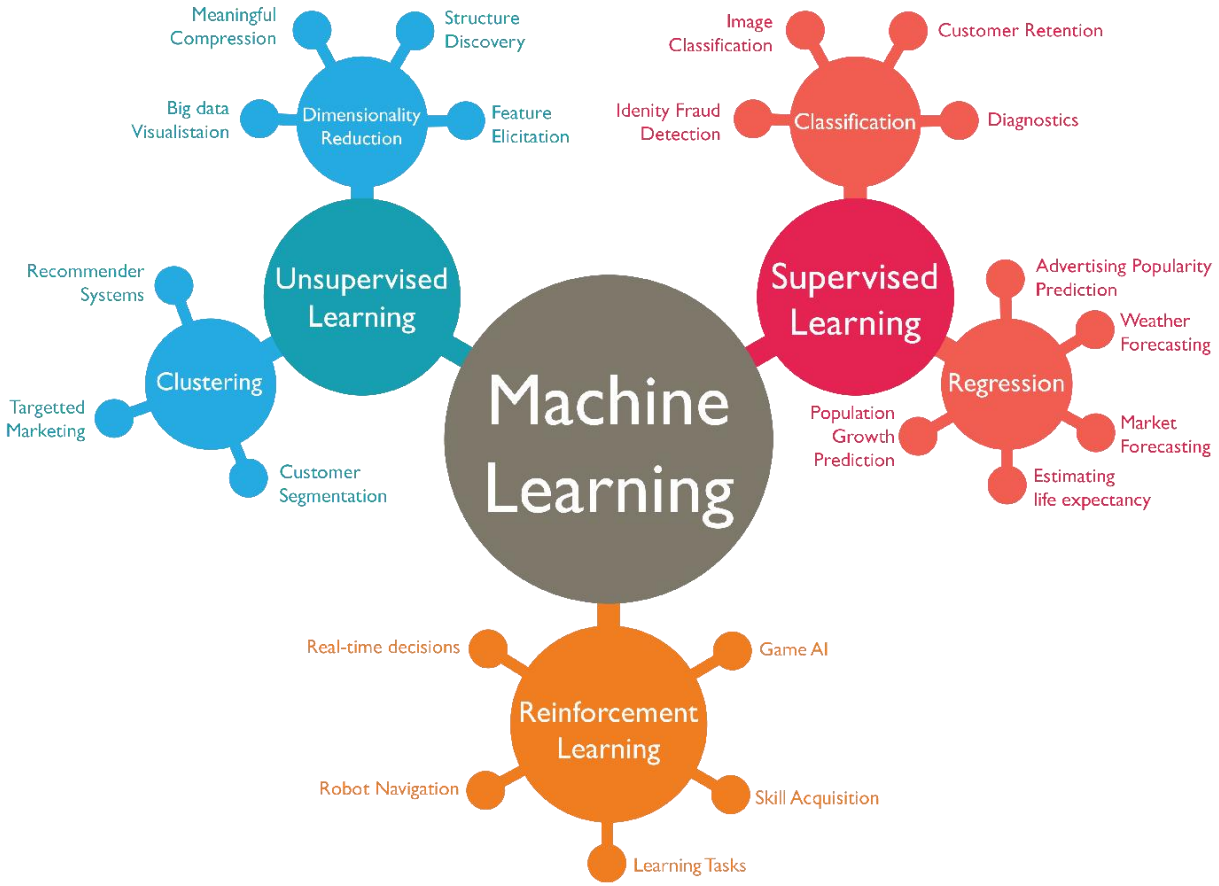
智能问答系统实践

第四课扩展：机器学习

姜文斌
北京师范大学人工智能学院
2025.03.20

什么是机器学习

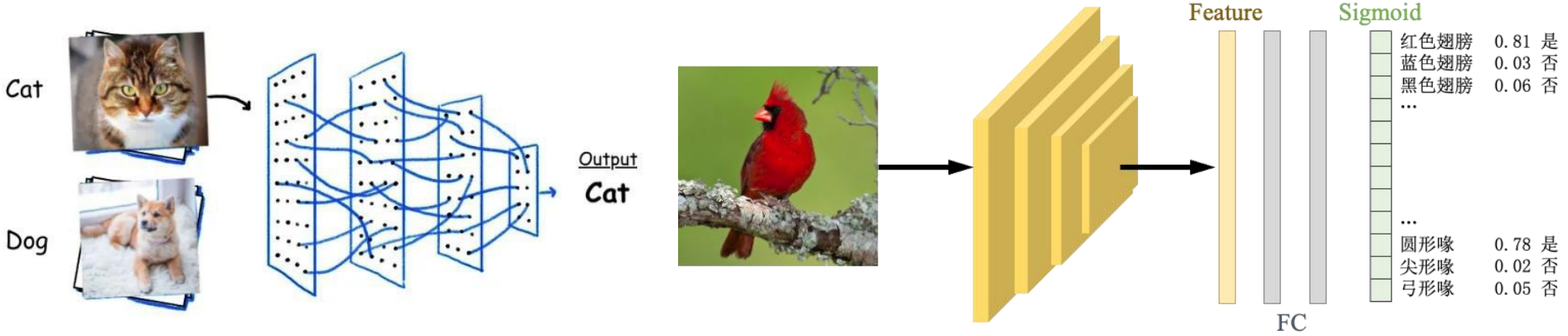
- 机器学习（Machine Learning）是一种人工智能技术，是实现智能化的关键
  - 机器学习使计算机能够自动从数据中学习规律和模式，并基于这些学习成果进行预测或决策，而无需明确的规则编程
  - 机器学习本质上通过训练算法来分析数据并学习如何完成特定任务



目录

分类问题

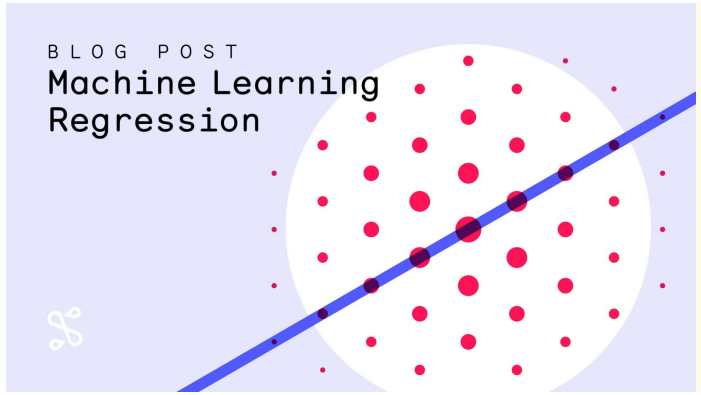
- 分类问题是把数据划分到不同类别中的过程
  - 模型的任务是学习如何根据输入信息来判断数据属于哪个类别
  - 适用于结果是离散类别的情况，比如垃圾分类、邮件分类（垃圾邮件或正常邮件）、肿瘤检测（良性或恶性）等，都是分类问题

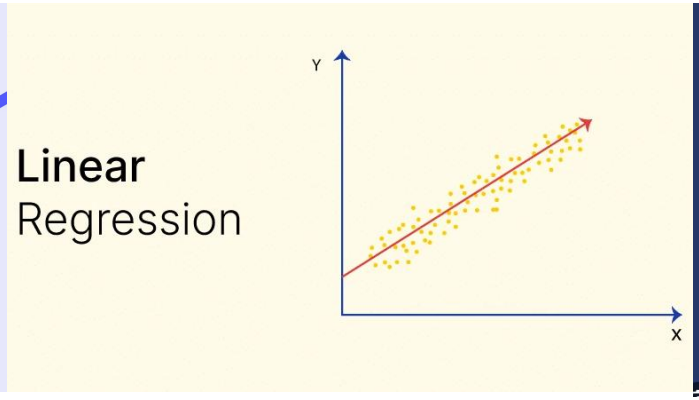


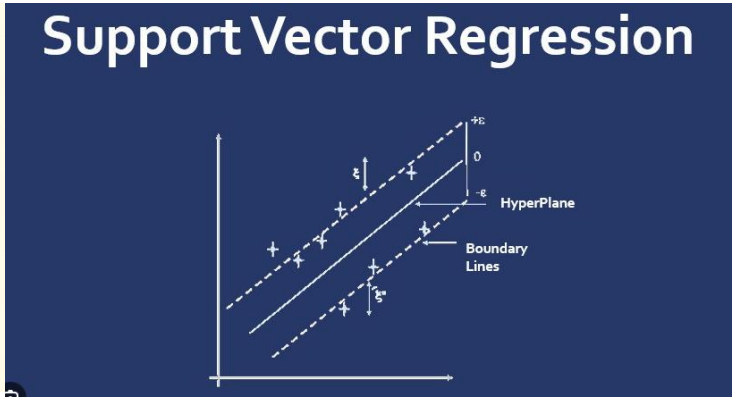
  - 输入是模型用来进行分类的已知信息或特征，输出是模型预测的类别标签，每个输入数据都会被分到一个具体的类别中

回归问题

- 回归问题是在已知数据的基础上，预测一个连续数值结果的过程
  - 在回归问题中，模型的任务是找到输入数据和输出之间的关系，然后根据这种关系，预测一个具体的数值
  - 回归问题适用于结果是连续数值的情况，特别是需要对结果进行精确预测的任务



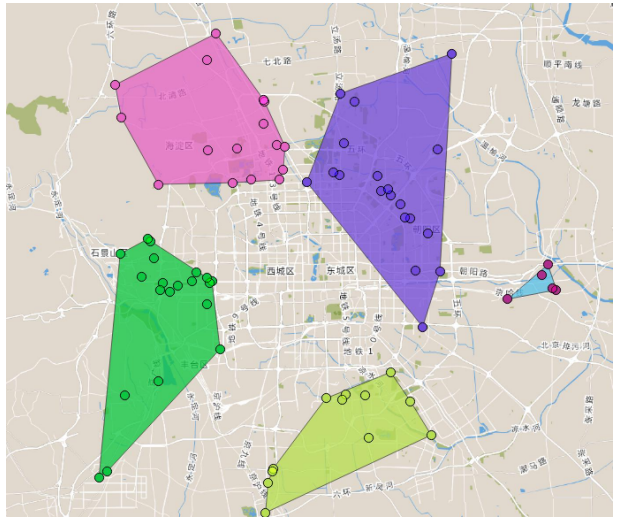




  - 输入是模型用来做预测的已知数据，也叫特征，输出是模型根据输入数据预测的数值结果

聚类问题

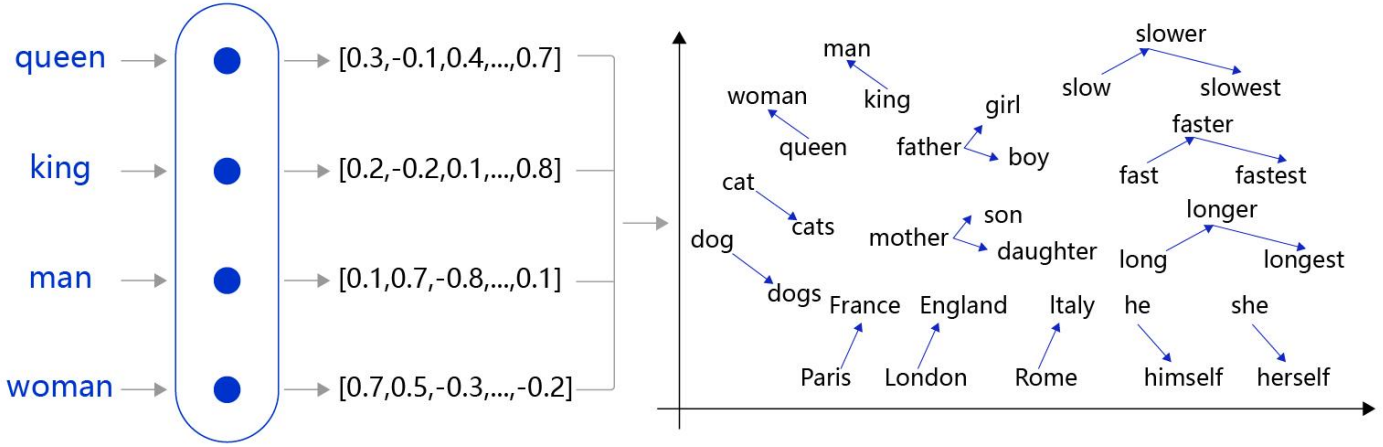
- 聚类问题是把一堆数据按相似性分成不同组的过程
  - 模型的任务是把没有标签的原始数据按相似性自动分组
  - 与分类问题不同，聚类不需要事先知道有哪几类，也没有“正确答案”。模型会分析数据中的模式，把相似的数据归为一组，不同的数据归为不同的组

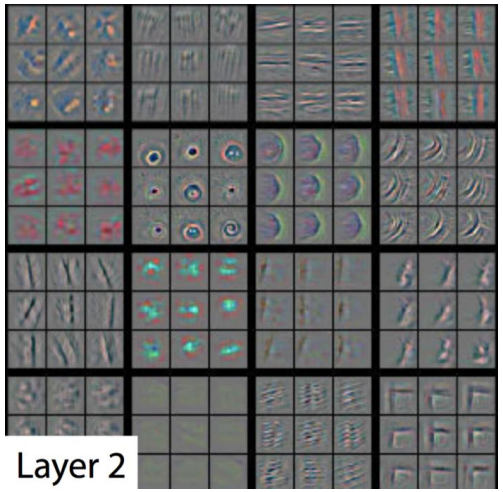


  - 输入是一组没有标签的数据，数据可以是文本、图片等，输出是若干个数据的分组结果，每组数据构成一个“簇”

表示问题

- 表示问题就是如何把复杂的信息转化为机器能够理解和处理的形式
  - 在机器学习中，模型不能直接理解像图片、文字或声音这样的原始信息，需要把这些复杂的信息转化成模型可以理解的数值形式或特征
  - 得到的“表示”既要包含足够的有用信息，又要便于模型处理





- 输入是原始的数据或信息，输出是把原始信息转化后的特征表示，是模型可以理解和使用的数值形式

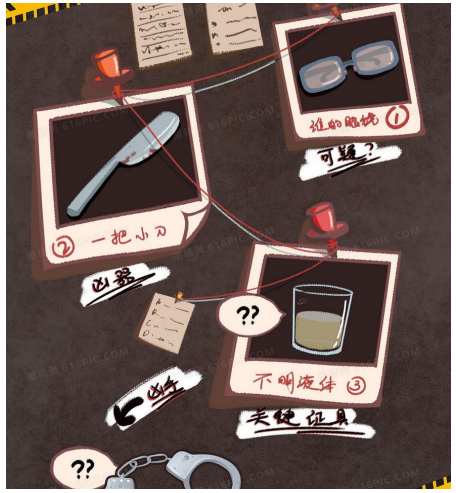
理解问题

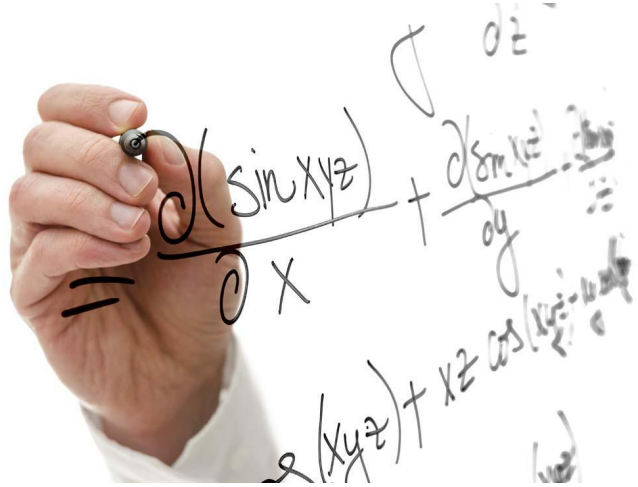
- 理解问题是让模型理解输入内容的含义，基于这个理解给出合理的答案或反应
  - 理解问题主要考察模型的“理解能力”，即能否正确领会输入内容的意思、意图或情感等，与单纯的字面匹配不同，模型需要对输入内容有更深层的理解
  - 通常出现在语言、视觉等领域，涉及到信息的分析和解释

  - 输入是需要理解的信息或内容，输出是模型基于理解给出的分析结果或结构化描述

推理问题

- 推理问题是在已知信息的基础上，推导出新的结论或答案的过程
  - 模型并不创造新的数据，而是根据已有的数据或条件，找出隐藏的关系、模式，或者作出合理的判断和预测，这种结论通常并不显而易见，需要通过分析和逻辑推导来获得
  - 推理问题类似于“根据线索找到答案”的过程

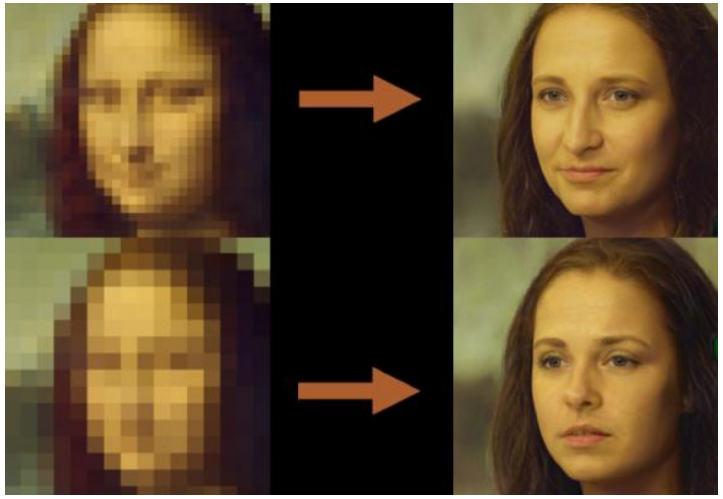




  - 输入是已知的信息、数据或条件，输出是基于输入信息得出的结论或答案

生成问题

- **生成问题就是根据输入的信息，生成新的数据或内容的过程**
  - 生成问题的目标是生成符合逻辑、连贯或符合某种特定风格的全新内容，通常存在于图像、文本、音乐等领域
  - 模型不仅是要“复制”已有数据，还要在此基础上“创造”出符合输入条件的新数据



  - 输入是提供给模型的起始信息或条件；输出是模型生成的新数据或内容

决策问题

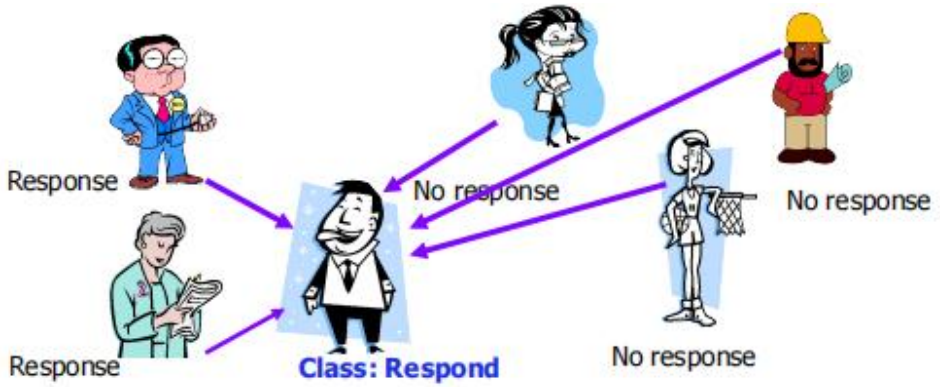
- 决策问题是在多种选择中找到最优方案或做出某种决策的过程
  - 在决策问题中，目标是找到一种能最好地达成目标或满足需求的选择。这类问题通常有明确的目标和一套可能的选择，需要根据给定的信息（输入）来决定哪个选择最优（输出）
  - 决策问题广泛存在于生活中，比如选择最佳路线、确定广告投放等

- 输入是有待决策的问题以及帮助做决策的信息，输出是最终做出的决策或选择

目录

基于实例的方法

- **基于实例的方法：无需显式模型的预测方法**
  - 基于实例的方法（Instance-Based Methods）是一类机器学习方法，也称为记忆式学习，通过存储和利用样本直接进行预测，而不是构建显式的模型



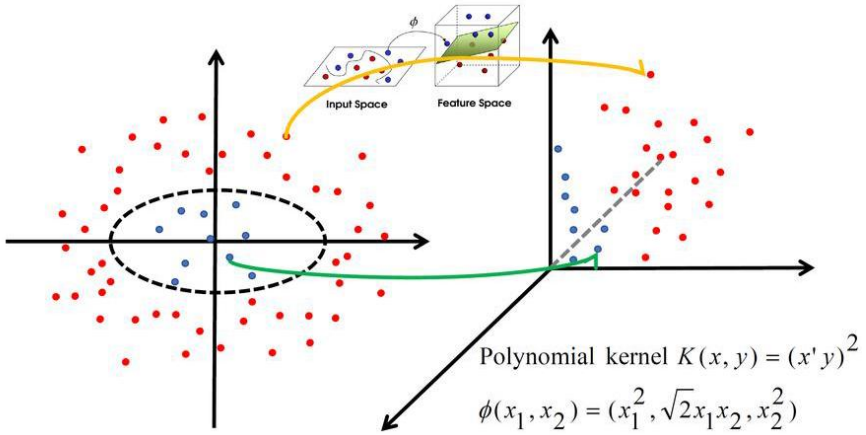
- **基本原理**
  - 在预测时，将新的样本与已知样本进行相似性比较，根据最相似的样本来确定结果
  - 基于实例的方法的关键是如何度量样本之间的相似性
    - 欧氏距离：用于度量数值型数据的相似性
    - 曼哈顿距离：适合处理稀疏数据的相似性
    - 余弦相似性：适合处理向量型数据的相似性

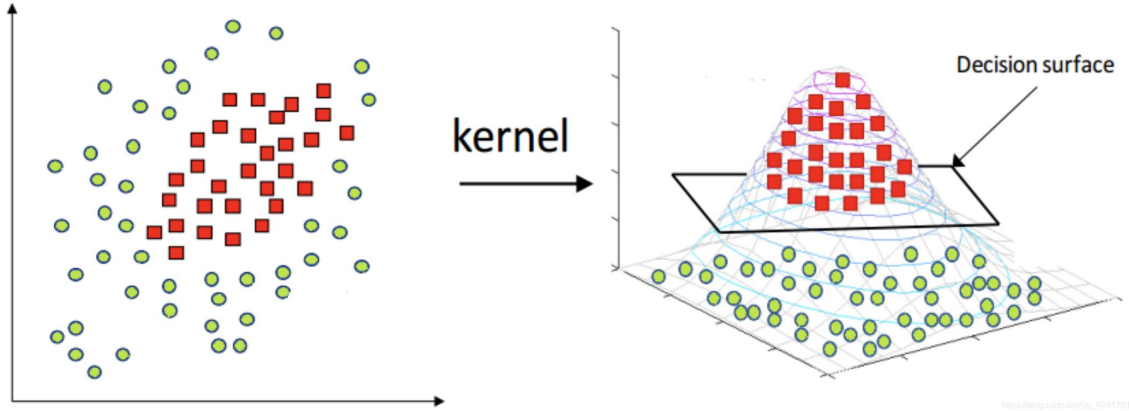
经典算法

- K-最近邻（K-Nearest Neighbors，KNN）
  - KNN是最经典的基于实例的方法之一。它通过计算样本间的距离，将每个新样本分配给距离最近的K个样本所在的类别（分类）或取K个邻居的平均值（回归）
  - **步骤**
    - 计算相似性：根据选定的相似性度量计算新样本与训练样本之间的距离
    - 选取K个最近邻：按照距离从小到大排序，选取最接近的K个样本
    - 多数投票或加权平均：对于分类任务，K个邻居中出现次数最多的类别作为新样本的类别；对于回归任务，K个邻居的加权平均作为新样本的预测值
  - **超参数选择**
    - K值的选择：K值是KNN算法的重要超参数。较小的K值会导致模型对噪声敏感，而较大的K值会平滑结果，可能会忽略局部特征
    - 权重选择：可以对距离较近的邻居赋予更大的权重，从而增强局部模式的影响

基于核的方法

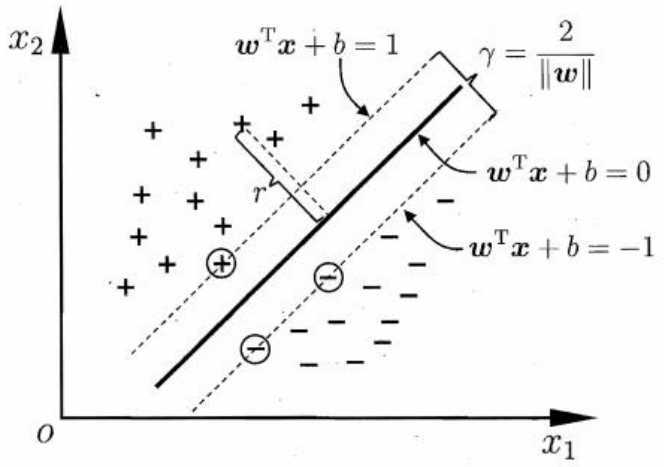
- **核方法：利用高维映射解决非线性问题的机器学习策略**
  - 基于核的方法（Kernel-Based Methods）是一类通过核函数将样本映射到高维特征空间，以处理非线性问题的机器学习方法。核函数在不显式计算高维映射的情况下，直接在原始输入空间中计算样本间的相似性，从而解决线性模型在低维空间中的局限性
- **基本原理**
  - 核方法的核心思想是：将原始空间中的非线性问题转化为高维特征空间中的线性问题。通过核函数，样本被隐式地映射到高维空间，使得在该空间中线性可分或可拟合



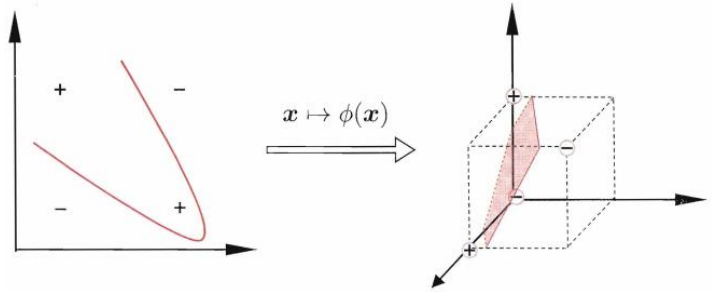


经典算法

- 支持向量机（Support Vector Machine，SVM）
  - 基本原理
    - 在样本特征空间中找到一个超平面，使得不同类别的样本在该超平面两侧尽可能分离开来，并最大化两类样本之间的间隔（Margin）



  - 线性SVM
    - 线性可分：如果数据在原始特征空间中线性可分，则SVM可以找到一个线性超平面将样本划分为两类
    - 最大化间隔：SVM通过找到使间隔最大化的超平面，提高分类的鲁棒性和泛化能力
  - 非线性SVM
    - 当数据在原始空间中不可线性分离时，SVM利用核函数将数据映射到高维空间，使得在高维空间中线性可分



贝叶斯方法

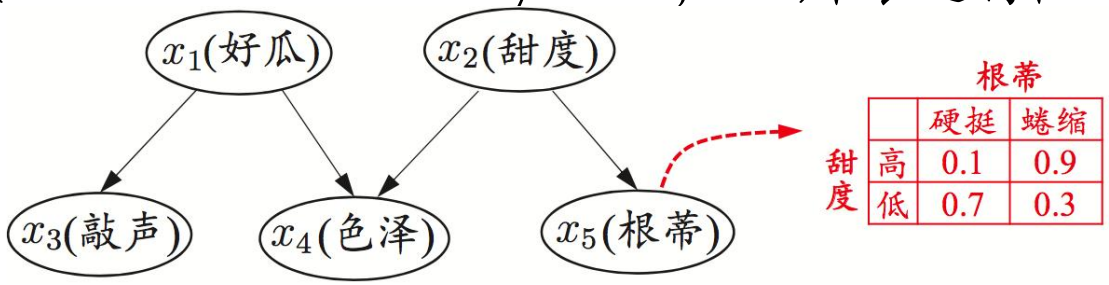
- 贝叶斯方法（Bayesian Methods）是一类基于贝叶斯定理的统计推断方法
  - 贝叶斯方法利用先验知识，结合当前观测到的数据，来更新关于未知参数的信息，以此获得修订后的参数的可能性，也就是后验概率，是一种灵活且解释性强的推断框架
- 基本原理
  - 贝叶斯定理描述了如何根据新的数据更新先验概率，从而得到后验概率，公式为：

$$P(\theta|\text{data}) = \frac{P(\text{data}|\theta) \cdot P(\theta)}{P(\text{data})}$$

    - $P(\theta)$:先验概率，表示在观察数据之前对参数θ的信念
    - $P(data|\theta)$:似然，表示在给定参数θ的情况下，数据出现的概率
    - $P(data)$:边际似然，表示数据出现的总概率，是归一化常数
    - $P(\theta|\text{data})$:后验概率，表示在观察数据之后更新的对参数θ的信念
  - 核心思想：通过结合先验知识和新获得的数据，对不确定性进行更新，从而得出更精确的估计或预测

经典算法

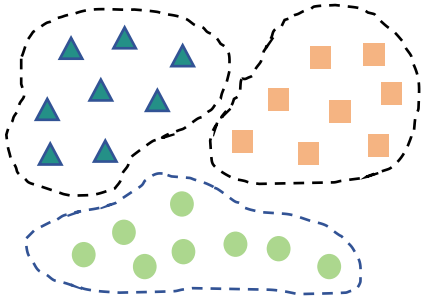
- 贝叶斯网络（Bayesian Network）
  - 贝叶斯网络是一种基于有向无环图（Directed Acyclic Graph, DAG）的概率图模型，用于表示变量之间的条件独立性和依赖关系
  - 使用条件概率表(Conditional Probability Table, CPT)来表述属性的联合概率分布



- 基本构成
  - **节点**（Nodes）：图中的每个节点代表一个随机变量，可以是离散或连续的
  - **有向边**（Directed Edges）：节点之间的有向边表示变量之间的条件依赖关系，箭头方向表明“因果”关系
  - **条件概率表**（CPT, Conditional Probability Table）：每个节点都会与一个条件概率表相关联，描述该节点在其父节点给定的条件下取不同值的概率

聚类方法

- 聚类算法（Clustering Algorithms）是一类无监督学习方法
  - 用于将样本划分为多个组或簇，使得同一个簇内的样本尽可能相似，而不同簇间的样本尽可能不同
  - 聚类广泛应用于数据分析、模式识别和图像处理等领域
- 基本原理
  - 聚类算法的核心是基于样本间的相似性或距离进行分组。目标是找到数据中的自然模式或分布，使得每个簇代表一个独立的类或类别



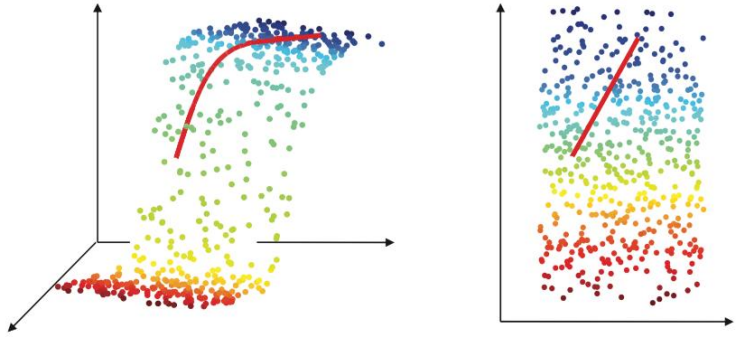
- 簇的定义
  - 不同聚类算法对“簇”的定义不同，有些算法定义簇为高密度区域，有些算法则定义为与中心距离较近的样本集合

经典算法

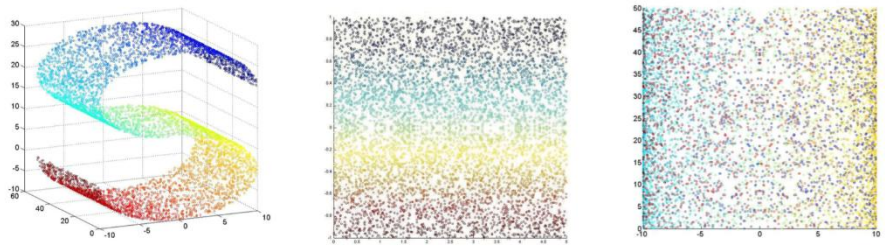
- K-Means聚类
  - **基本思路**
    - K均值将数据划分为K个簇，每个簇由一个质心（Centroid）代表
    - 通过迭代优化簇内样本与质心的距离，逐步找到最优分配
  - **算法步骤**
    - 初始化：随机选择K个样本作为初始质心
    - 分配样本：将每个样本分配给离其最近的质心，形成K个簇
    - 更新质心：计算每个簇的均值，将均值作为新的质心
    - 迭代：重复分配样本和更新质心的过程，直到质心不再变化或达到最大迭代次数
- K-Means++
  - K-Means++是对K均值的改进，通过优化初始质心选择来提高聚类效果
    - 初始化：优先选择离已选质心较远的点作为下一个质心，增加了初始质心的多样性，提升了最终聚类的效果和稳定性

降维方法

- 降维算法是一类将高维数据转换为低维数据的技术
  - 目的是在减少特征数量的同时尽可能保留原始数据的有用信息
  - 降维有助于数据可视化、降低计算成本、减小模型复杂度、去噪以及防止过拟合
- 基本原理
  - 降维算法通过减少数据特征的数量，将高维空间中的数据投影到低维空间中
    - 线性降维：通过线性变换提取数据的主要特征，适合处理线性可分的数据
    - 非线性降维：通过非线性映射将数据从高维空间压缩到低维空间，以揭示复杂的特征结构，适合处理非线性分布的数据



(a) 三维空间中观察到的样本点　(b) 二维空间中的曲面

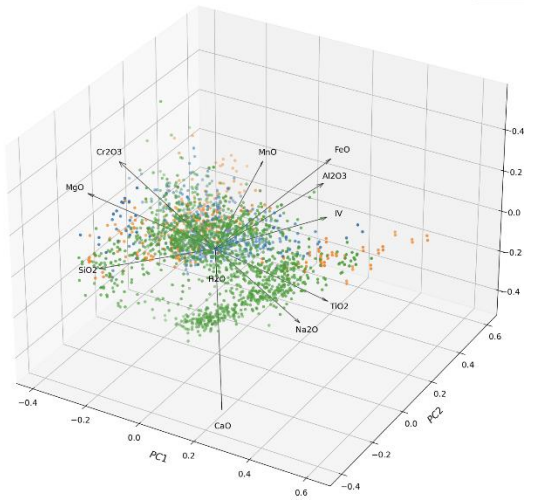


(a) 三维空间中的观察　(b) 本真二维结构　(c) PCA 降维结果

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.

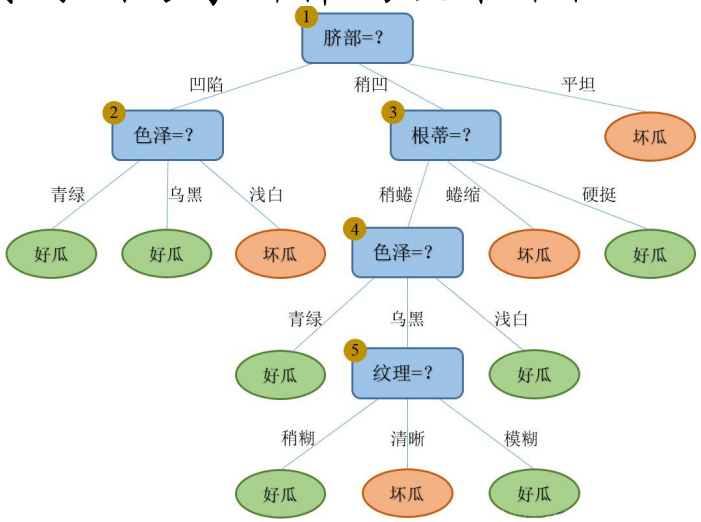
经典算法

- 降维的两种方式
  - 特征选择：从原始特征中选出一部分重要的特征
  - 特征提取：通过组合原始特征生成新的低维特征
- 主成分分析（Principal Component Analysis，PCA）
  - PCA是一种线性降维算法，用于最大化数据在低维空间中的方差
  - **基本思想**
    - 将特征从n维映射为k维($k < n$)，得到新的k维正交特征，即主成分
    - 只保留包含大部分方差的维度特征，忽略包含方差几乎为0的特征维度



决策树方法

- 决策树是一种树状结构的监督学习算法，用于分类和回归任务
  - 通过不断地根据特征将数据集分割成多个子集，从而构建树状结构
  - 模型的每个节点代表一个特征分裂决策，叶节点给出最终的分类或回归结果，易于解释和理解
- 基本原理
  - 它通过递归地将数据集划分为多个子集，使每个子集中的样本尽可能属于同一类（分类）或具有相似值（回归），从而建立一系列决策规则，最终得到易于解释的决策路径
- 基本概念
  - 根结点：包含样本全集
  - 叶节点（Leaf Node）：对应于决策结果
  - 非叶节点（Node）：对应于一个属性测试，每个结点包含的样本集合根据属性测试的结果被划分到子结点中
  - 从根结点到每个叶结点的路径对应了一个判定测试序列



决策树方法

- 构建过程
  - **选择划分特征**：通过信息增益、基尼不纯度等标准选择“划分属性”进行划分
    - 信息增益：衡量划分前后的信息熵变化

$$\mathrm{Ent}(D) = -\sum_{k=1}^{|\mathcal{Y}|} p_k \log_2 p_k$$

$$\mathrm{Gain}(D, a) = \mathrm{Ent}(D) - \sum_{v=1}^{V} \frac{|D^v|}{|D|} \mathrm{Ent}(D^v)$$

    - 基尼指数：衡量数据集的纯度

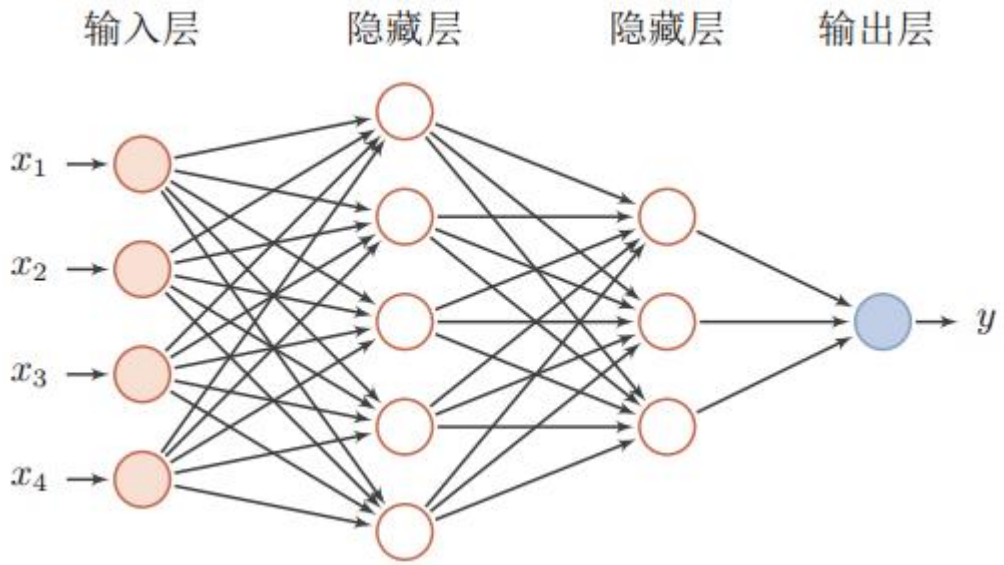
$$\mathrm{Gini}(D) = 1 - \sum_{k=1}^{|\mathcal{Y}|} p_k^2 \qquad \mathrm{Gini\_index}(D, a) = \sum_{v=1}^{V} \frac{|D^v|}{|D|} \mathrm{Gini}(D^v)$$

  - **递归划分**：对每个节点的数据集进行递归划分，直到达到终止条件（属于同一类别/当前属性集为空/所有样本在属性上取值相同或为空）
  - **（可选）剪枝**：通过减少决策树的节点数来防止过拟合，包括预剪枝（提前停止）和后剪枝（树构建完成后再修剪）

目录

前馈神经网络

- 前馈神经网络（Feedforward Neural Network，FNN）是神经网络最基本类型
  - FNN模拟生物神经元的工作机制，并用于各种预测和分类任务
  - 主要特点是信息在网络中单向流动，不存在反馈回路



  - 前馈神经网络的结构通常包括输入层、隐藏层和输出层，节点间的连接由权重和偏置构成
  - FNN是一种全连接网络，其中每一层的每个神经元都与下一层的所有神经元相连

网络结构

- 输入层
  - 输入层直接接受原始特征，例如图像的像素值、文本的词向量、传感器的读数等
  - 输入层的节点数量等于输入数据的特征维度。例如，如果输入是一个包含784个像素值的28x28灰度图像，输入层将有784个节点
- 隐藏层
  - 隐藏层是前馈神经网络的核心，负责对输入数据进行特征提取和非线性变换
  - 隐藏层的数量和每层的节点数量决定了网络的复杂度。每个节点计算一个加权和，并通过激活函数进行非线性变换 $y = \sigma(Wx + b)$
- 输出层
  - 输出层的结构取决于任务类型：分类问题的输出节点数量等于类别数，而回归问题的输出节点数通常为1
  - 输出结果通过激活函数映射到特定的范围，比如用Softmax函数将输出转换为概率分布，或用线性激活函数直接输出数值

网络结构

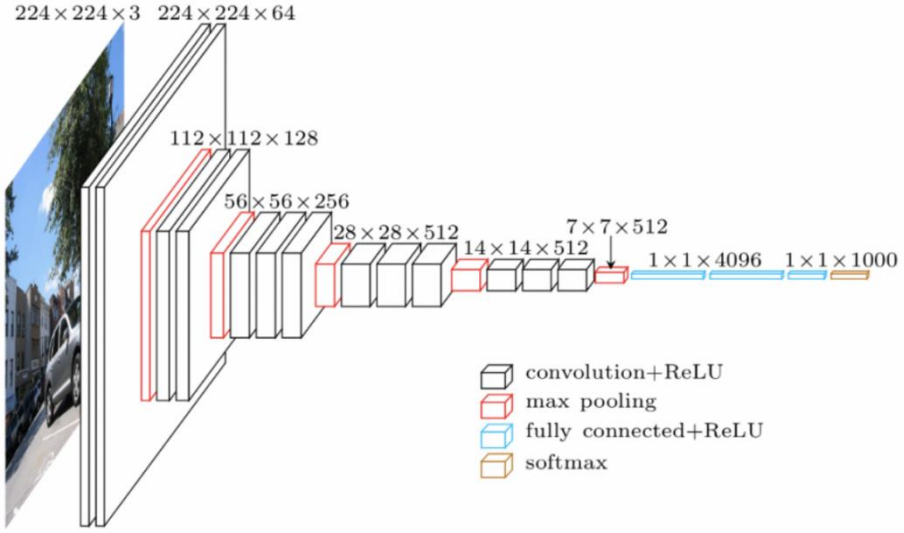
- FNN的训练过程包括前向传播、损失计算、反向传播和优化
  - **前向传播**
    - 输入数据逐层传播，每层的输出作为下一层的输入，直至输出层
  - **损失计算**
    - 计算模型的损失函数，衡量预测值与真实值之间的误差
  - **反向传播**
    - 通过最小化损失函数来计算损失对每个参数的梯度，从而更新和调整网络的参数
  - **参数更新**
    - 通过优化算法利用计算得到的梯度更新网络的权重和偏置
    - 常用的优化算法
      - 随机梯度下降（SGD）：根据每个批次的数据更新参数
      - 动量优化（Momentum）：在梯度下降的基础上增加动量，能够加速收敛
      - Adam：结合动量和RMSProp，适合处理稀疏梯度和非平稳目标

适用场景

- 适用于多种任务和领域，特别是在处理结构化数据、简单的图像或文本数据时效果较好
  - **分类任务**
    - 图像分类：可以用于基本的图像分类任务，比如手写数字识别（MNIST数据集）
    - 文本分类：通过词向量输入，FNN可用于情感分析、垃圾邮件检测等任务
    - 信号处理：用于语音识别、音频信号分类
  - **回归任务**
    - 时间序列预测：用于预测连续的数值数据，如股市价格、销售量等
    - 函数拟合：在物理实验、经济建模中，用于拟合复杂的非线性函数关系
  - **特征提取和降维**
    - 特征提取：在无监督学习任务中，可将隐藏层的输出用作数据的特征表示
    - 降维：可以用作数据降维或压缩的工具，将高维数据压缩为低维表示

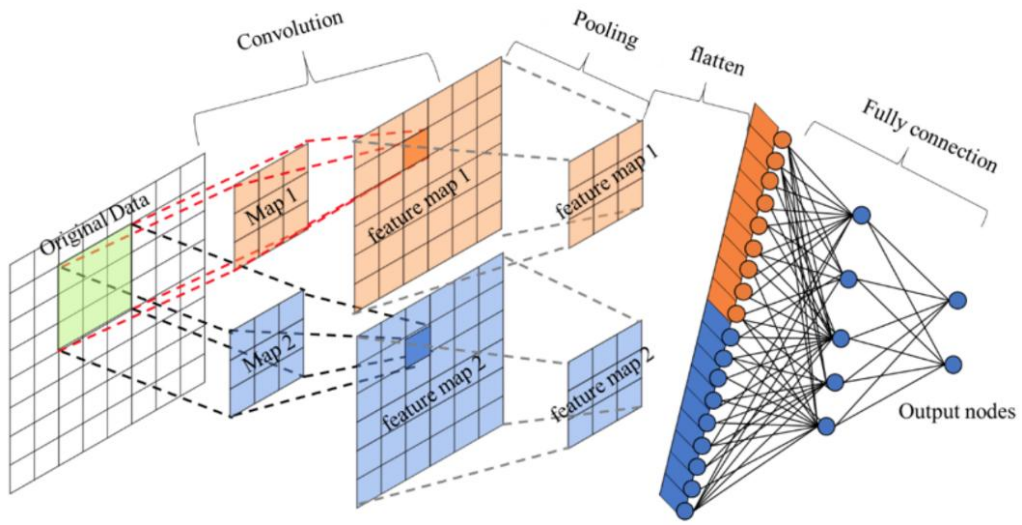
卷积神经网络

- **卷积神经网络是一类包含卷积计算且具有深度结构的前馈神经网络**
  - 卷积神经网络（Convolutional Neural Networks, CNN）是一种专门用于处理具有网格结构数据的深度神经网络模型，尤其在图像识别和计算机视觉任务中表现出色
  - CNN的核心特性是通过卷积操作提取局部特征，并利用权重共享减少参数量
- **基本思想**
  - 利用卷积操作和池化操作逐步提取输入数据的局部特征，通过多层次特征提取，最终完成对数据的全局理解
- **主要组件**
  - **卷积层**：执行卷积操作，提取输入的局部特征
  - **激活函数**：引入非线性，通常使用ReLU
  - **池化层**：减少特征图的维度，降低计算复杂度
  - **全连接层**：将提取的特征用于分类或回归



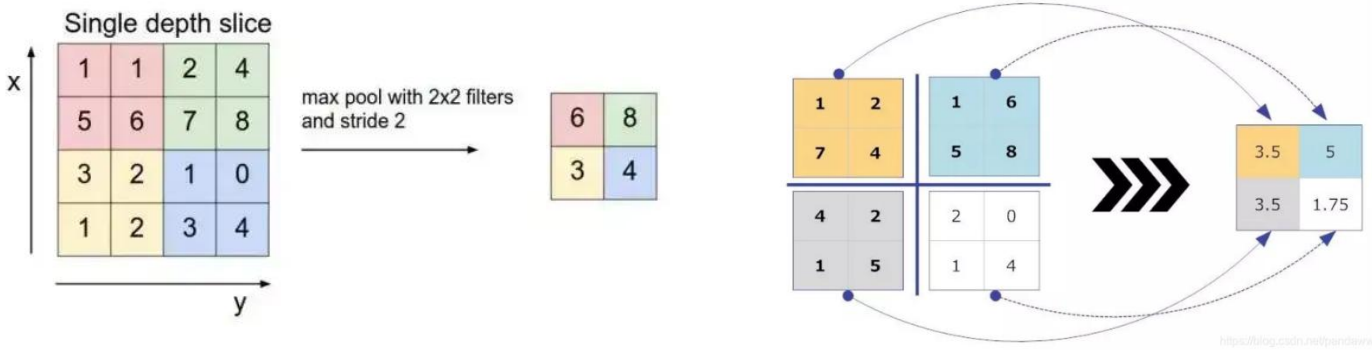
网络结构

- CNN主要由卷积层、池化层和全连接层组成
  - 通常按照顺序堆叠多个卷积和池化层，最后接全连接层进行分类或回归
- 卷积层
  - 卷积层是CNN的核心部分。通过卷积核（滤波器）扫描输入数据的局部区域，提取特征
  - 卷积核在输入的图像或特征图上滑动，进行卷积运算，生成一个特征图（Feature Map）
  - 卷积操作通过共享卷积核参数，大大减少了需要训练的参数量，同时保留输入的空间信息
  - 卷积层的参数
    - 卷积核大小：通常为3x3或5x5，其大小决定了特征提取的局部范围
    - 步长（Stride）：卷积核每次移动的步幅，步长越大，生成的特征图越小
    - 填充（Padding）：在输入数据边界添加额外的0，以保留输入的空间维度



网络结构

- 池化层
  - 池化层用于对特征图进行下采样，从而减少计算量和特征图的维度，保留主要特征
    - 最大池化（Max Pooling）是最常用的池化方式，提取局部区域的最大值，增强特征的显著性
    - 平均池化（Average Pooling）计算局部区域的平均值，更适合提取平滑特征



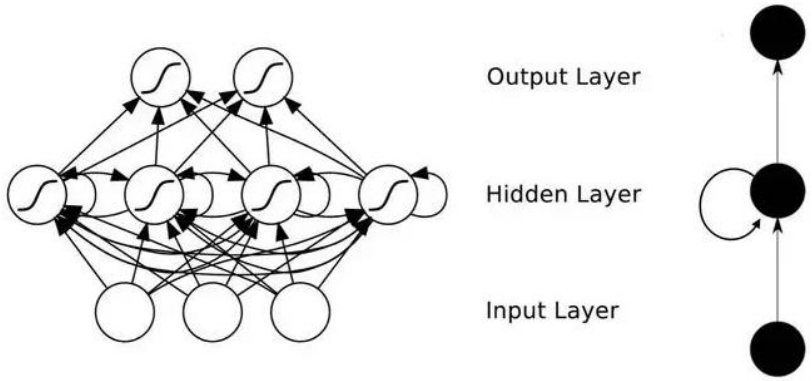
- 全连接层
  - 全连接层与前馈神经网络类似，将卷积和池化层提取的高维特征映射到目标类别
  - 在全连接层之前，通常需要将特征图展平（Flatten），然后输入到全连接层
  - 输出层根据任务类型选择激活函数：分类任务常用Softmax，回归任务使用线性激活函数

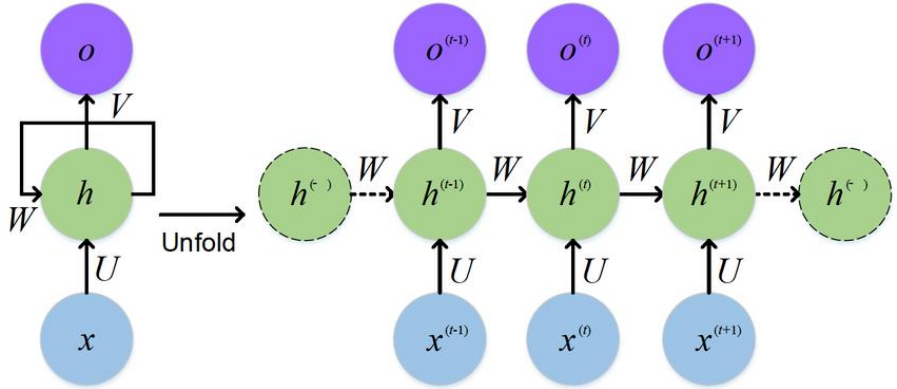
适用场景

- CNN广泛应用于计算机视觉和其他领域，特别是对图像、视频、语音等网格结构数据的处理
  - **图像分类**
    - 自然图像分类（如ImageNet）：利用深层卷积网络，可将自然图像分类为上千种类别
  - **目标检测**
    - CNN结合区域建议网络（如R-CNN、Faster R-CNN）可在图像中定位并标注目标物体
    - 应用于人脸检测、车辆检测、行人检测等任务
  - **图像分割**
    - CNN与全卷积网络（FCN）、U-Net等结合，可以在图像中进行像素级的目标分割
    - 用于医学图像中的病变区域检测、卫星图像中的地物分割等
  - **自然语言处理**
    - 在文本分类任务中，将文本嵌入为词向量后，通过1D卷积提取上下文信息，进行情感分析、垃圾邮件检测等任务

循环神经网络

- 循环神经网络是一种用于处理序列数据的神经网络模型
  - 循环神经网络（Recurrent Neural Network，简称RNN）能记忆序列中的上下文信息，因此在需要时间相关性和上下文理解的任务中表现出色
  - RNN擅长处理时间序列、自然语言和其他序列相关的任务



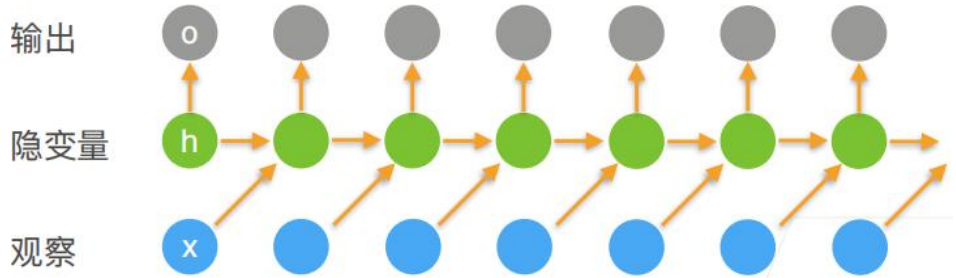


- 基本思想
  - RNN中的神经元在时间步之间相互连接，将前一步的隐藏状态传递到当前时间步，形成循环结构
  - 网络中的信息循环传递，使得网络能够从输入序列中捕获和记忆上下文信息

网络结构

- 工作原理
  - 时间步的循环计算
    - 对于输入序列$x_1, x_2, \ldots, x_t$，在每个时间步t，隐状态h_t由当前输入x_t和前一隐状态h_{t-1}共同决定：

$$h_t = \sigma(W_{xh}x_t + W_{hh}h_{t-1} + b_h)$$

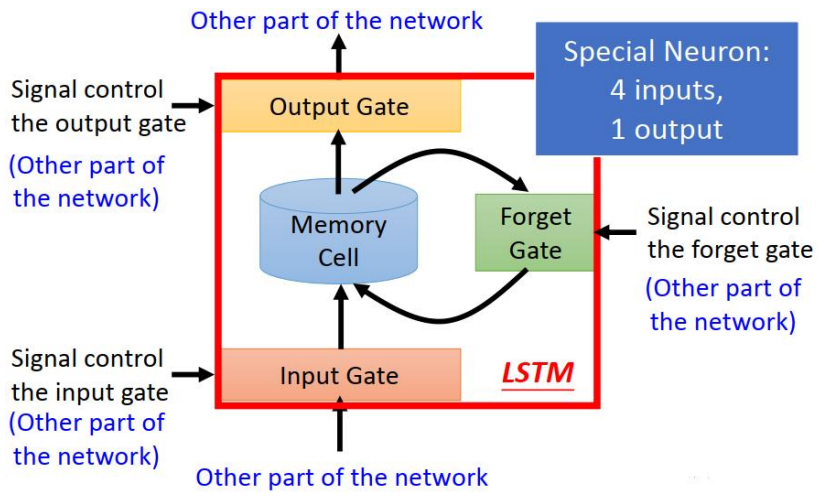


  - 隐状态的传递
    - 隐藏状态h_t在时间步之间传递，存储着从序列开头到当前时间步的所有信息
  - 输出层
    - 根据需求，可以在每一个时间步生成输出，也可以在最后一个时间步生成输出

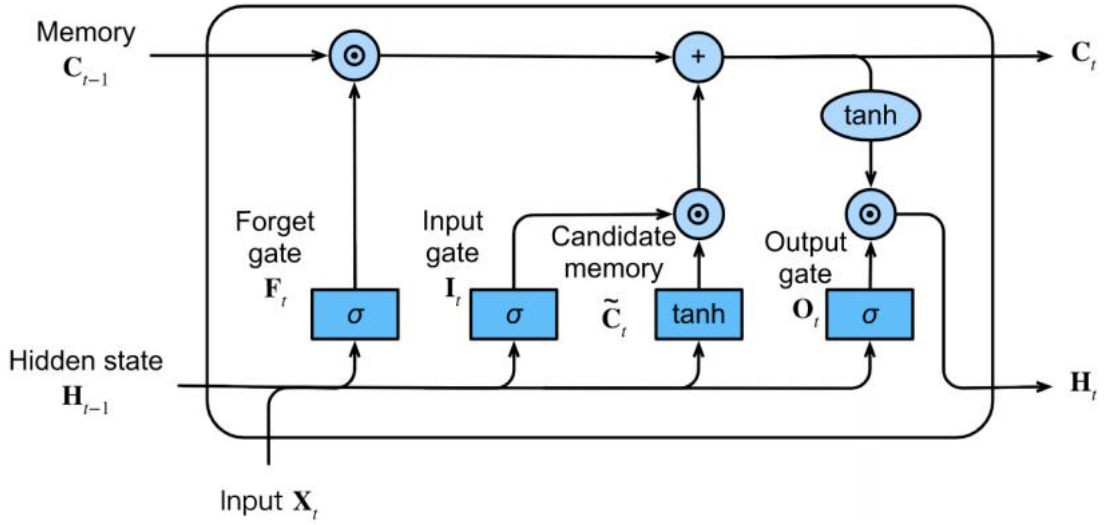
$$o_t = f(W_{ho}h_t + b_o)$$

常见变种

- 长短期记忆网络（LSTM）
  - LSTM通过引入记忆单元（Cell State）和门控机制来控制信息的存储和更新，从而有效解决长距离依赖问题
  - 记忆单元可以在序列中长期保留重要信息

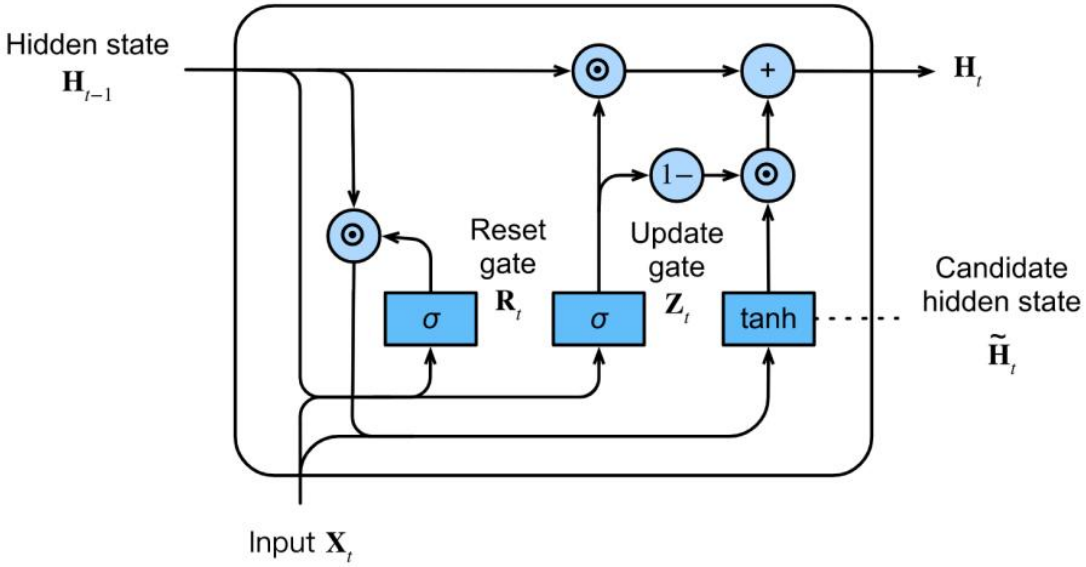


- LSTM中的门控装置
  - **输入门**（Input Gate）：决定当前时刻有多少信息进入记忆单元
  - **输出门**（Output Gate）：决定当前时刻有多少信息从记忆单元输出
  - **遗忘门**（Forget Gate）：决定遗忘前一时间步信息的比例



常见变种

- 门控循环单元（GRU）
  - 与LSTM类似，通过门控单元解决RNN中不能长期记忆和反向传播中的梯度爆炸、梯度消失等问题
  - 与LSTM相比，GRU内部的网络架构较为简单



- GRU中的门控装置
  - **更新门**（update gate）：类似于LSTM中的遗忘门和输入门，决定要丢弃哪些信息和要添加哪些新信息
  - **重置门**（reset gate）：用于决定丢弃先前信息的程度

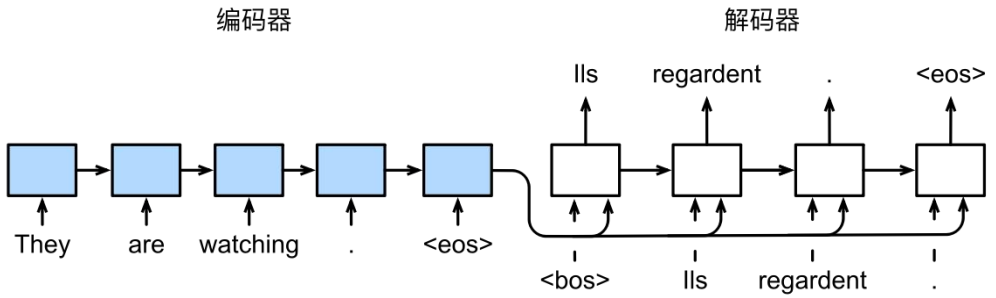
$$\boldsymbol{R}_t = \sigma(\boldsymbol{X}_t \boldsymbol{W}_{xr} + \boldsymbol{H}_{t-1} \boldsymbol{W}_{hr} + \boldsymbol{b}_r),$$
$$\boldsymbol{Z}_t = \sigma(\boldsymbol{X}_t \boldsymbol{W}_{xz} + \boldsymbol{H}_{t-1} \boldsymbol{W}_{hz} + \boldsymbol{b}_z)$$
$$\tilde{\boldsymbol{H}}_t = \tanh(\boldsymbol{X}_t \boldsymbol{W}_{xh} + (\boldsymbol{R}_t \odot \boldsymbol{H}_{t-1}) \boldsymbol{W}_{hh} + \boldsymbol{b}_h)$$
$$\boldsymbol{H}_t = \boldsymbol{Z}_t \odot \boldsymbol{H}_{t-1} + (1 - \boldsymbol{Z}_t) \odot \tilde{\boldsymbol{H}}_t$$

适用场景

- RNNs在处理具有时间依赖性或顺序关系的任务时表现较为突出
  - 自然语言处理（NLP）
    - 语言模型：RNN可以用作语言模型，预测句子中的下一个单词或短语
    - 机器翻译：RNN能够将源语言的句子转换为目标语言的句子
    - 文本生成：通过学习文本的语言模式，RNN可用于生成具有连续性的句子或段落，如自动写作、新闻生成等
  - 时间序列预测
    - 金融预测：RNN可用于股价、外汇等金融时间序列数据的预测
    - 天气预报：RNN通过对过去的天气数据进行建模，预测未来的气象情况
    - 需求预测：在供应链管理中，RNN可用于预测产品的市场需求，优化库存管理
  - 语音处理
    - 语音识别：RNN可以对语音信号建模，提取时间序列特征，用于语音到文本的转换
    - 语音合成：RNN也可以用于语音合成，通过学习真实语音数据生成流畅的语音输出

序列生成模型

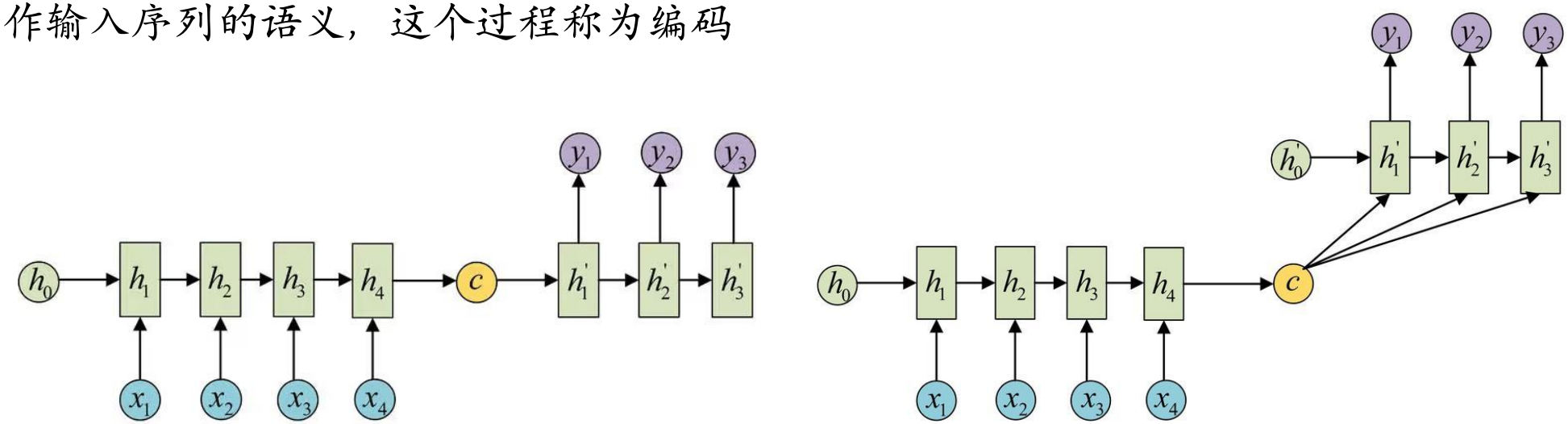
- 序列生成模型是一类输入和输出均为序列数据的深度学习模型
  - 序列到序列模型（sequence to sequence，seq2seq）能够根据给定的序列，通过特定的生成方法生成另一个序列，同时这两个序列可以不等长
  - 是Encoder-Decoder架构在序列到序列任务上的具体应用



- seq2seq模型的工作原理
  - 通过学习输入序列的统计特性来预测或生成新的序列数据
  - 编码器负责将输入序列映射到一个高维空间的隐状态，这个隐状态捕捉了序列的关键信息
  - 解码器利用这个隐状态来逐步生成目标序列，每一步生成的输出又作为下一步的输入，从而实现序列的自回归生成

网络结构

- seq2seq模型的结构（以两个RNN分别作为Encoder和Decoder为例）
  - Encoder负责将不定长的输入序列变换为指定长度的向量，即语义向量c，这个向量可以看作输入序列的语义，这个过程称为编码



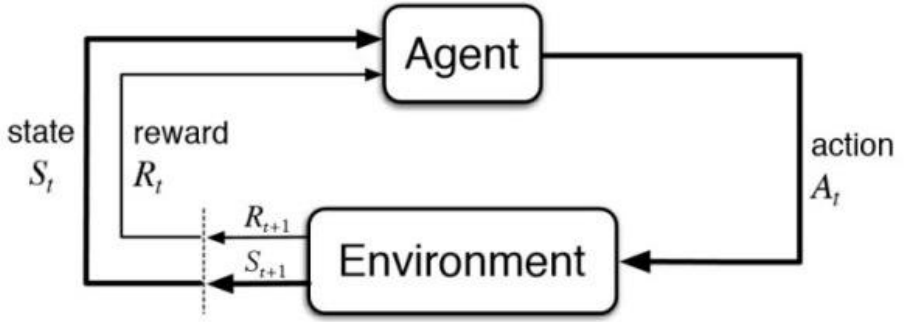
  - Decoder负责根据语义向量生成指定的序列，这个过程称为解码
    - 语义向量C只作为初始状态输入到Decoder中，上一时刻的输出会成为当前时刻的输入
    - 语义向量C参与序列所有时刻的运算，上一时刻的输出仍然作为当前时刻的输入

适用场景

- 序列生成模型在多个领域有着广泛的应用
  - 自然语言处理(NLP):在机器翻译、文本摘要、问答系统、对话系统和文本生成等领域，序列生成模型能够理解和生成自然语言文本
  - 语音合成:将文本转换为口语化的语音，使机器能够“说话”，在语音助手、自动语音识别和语音合成器中应用广泛
  - 音乐和艺术创作:生成新的音乐作品或艺术图案，为创意产业提供灵感和工具
  - 生物信息学:在基因序列分析中，预测蛋白质结构或识别有潜力的药物分子
  - 时间序列预测:在金融领域，预测股票价格、交易量等经济指标;在气象学中，预测天气变化
  - 游戏开发:用于生成游戏内的故事、对话和角色行为，提高游戏的互动性和沉浸感
  - 文本校正和语言学习:辅助语言学习者进行语法和拼写校正，提供语言学习建议

深度强化学习

- 深度强化学习是将深度学习与强化学习相结合的一种方法，旨在解决具有高维状态空间和复杂决策问题的任务
  - 深度强化学习（Deep Reinforcement Learning，DRL）利用深度神经网络来近似表示策略函数或值函数，从而在复杂的环境中进行决策



- 基本原理
  - 深度强化学习的工作原理基于智能体（Agent）与环境（Environment）的交互过程。在过程中的每个时间步t，智能体观察环境状态s_t，根据策略$\pi(a_t, s_t)$选择并执行一个动作a_t，环境随之转移到下一状态s_{t+1}并给予智能体相应的奖励r_t
  - 智能体的目标是学习一个最优策略π^*，使得累积奖励期望最大化 $\max_{\pi} \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t r_t\right]$

经典算法

- Q-Learning
  - Q-Learning是一种基于值函数的强化学习算法，用于训练智能体在不同状态下选择最优动作以最大化累计奖励
  - Q-Learning的核心思想是构建Q函数$Q(s, a)$用于表示在状态s下采取动作a时，能获得的累计奖励的期望值。通过不断更新Q函数的值，智能体可以找到一个最优策略，使得在每个状态下的动作选择都能够最大化未来的回报
  - **算法步骤**
    - 初始化Q-table，将所有状态-动作对的Q值设为一个初始值（通常为零）
    - 选择、执行动作：在每个状态下，根据某种策略选择动作，通常是以一定概率选择最优动作或随机动作；执行动作并观察奖励和新状态
    - 更新Q值：使用Q-Learning公式更新当前状态和动作的Q值

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left(R + \gamma \max_{a'} Q(s',a') - Q(s,a) \right)$$

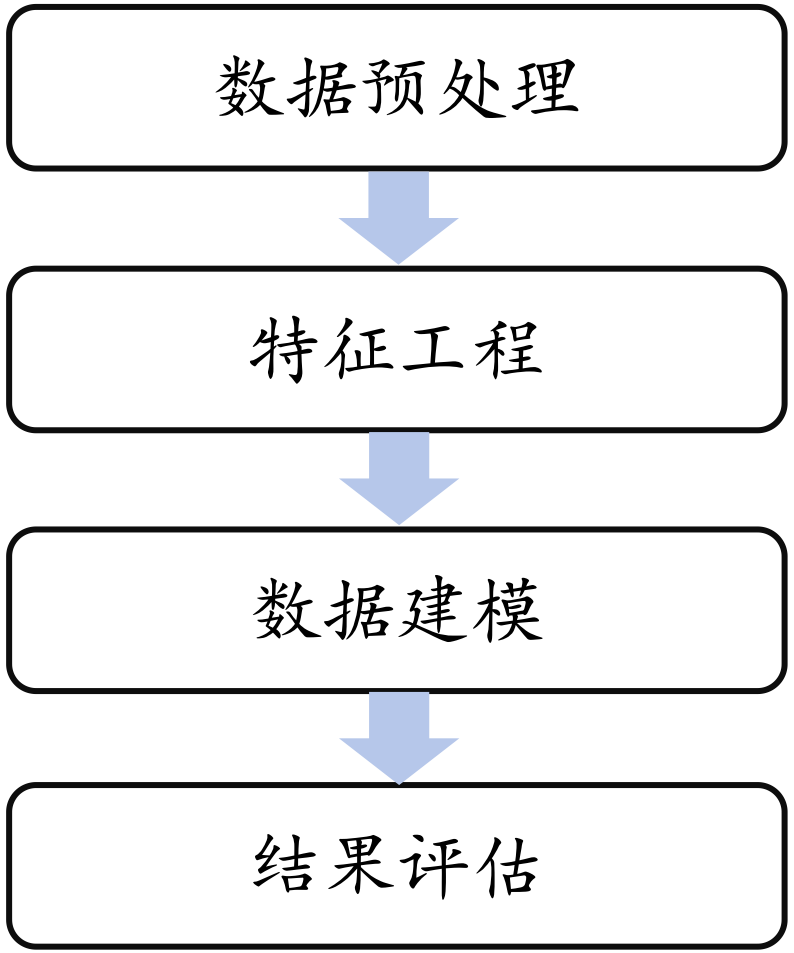
    - 重复以上步骤，直到达到停止条件（如Q值收敛或到达最大迭代次数）

适用场景

- DRL适用于需要智能体在复杂、动态环境中通过试错来学习策略的任务，特别是那些具有高维状态空间和连续动作空间的场景
  - **游戏AI**
    - DQN和AlphaGo等算法在游戏领域表现出色，能够在复杂策略游戏（如围棋、国际象棋、Atari游戏）中学习到超过人类水平的策略
    - 在《Dota2》、《星际争霸》等游戏中，DRL模型学习到的策略可以击败顶级人类选手
  - **金融领域**
    - 算法交易：通过学习历史市场数据，DRL可以生成自动交易策略，实时预测市场趋势并执行交易决策
  - **自动驾驶**
    - 在自动驾驶系统中，DRL通过摄像头、雷达等传感器输入进行环境感知，并学习车辆的最优驾驶策略，包括加速、刹车和转弯等动作
    - DRL能够适应复杂的道路环境，如应对行人、交通信号、其他车辆的突发情况等

目录

工作流程



数据清洗、数据采样、数据拆分

特征编码、特征选择、特征降维

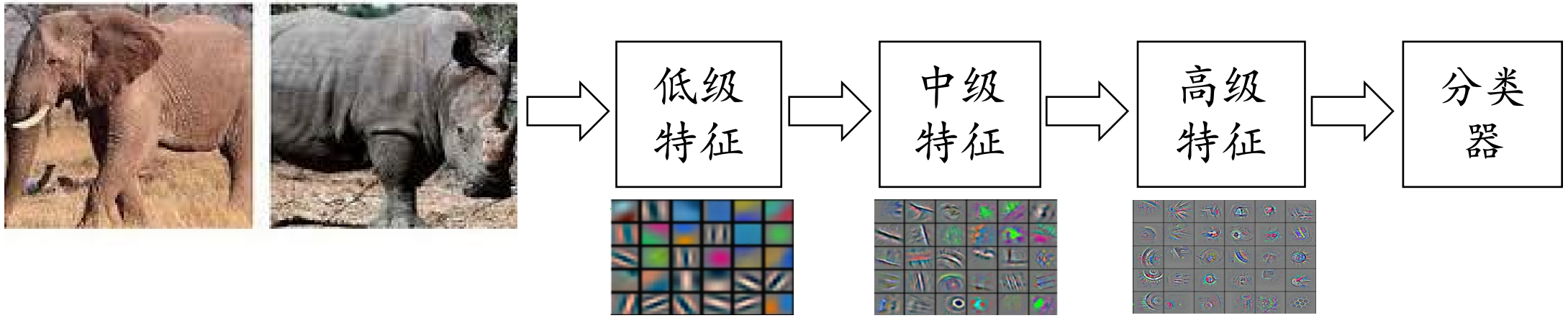
回归问题、分类问题、聚类问题

准确率、召回率、F1值、PR曲线

传统ML与深度ML

- 传统机器学习：人工特征工程+分类器

- 深度学习：自动学习多尺度的特征表示



经典编程与机器学习

计算模式	信息处理的流程架构（算法/模型）				基础组件	需否训练	适用场景
	整体流程架构	抽象环节（特征/表示）	计算环节（操作/参数）	输出环节（直出/搜索）			
经典编程	手工编写算法（搜索、排序、优化、规划等）	手工设计数据结构，自动填充数值	手工设计操作，手工设置参数（通常很少参数）	手工设计输出逻辑，通常直接输出	算术、逻辑、分支、循环、递归、顺序等	人工设计所有逻辑，无需训练	输入信息不复杂，计算逻辑不复杂
传统机器学习	手工编写模型（SVM、CRF、DT等）	手工设计特征模板，自动抽取特征数值	手工设计操作，自动学习参数	手工设计输出逻辑，通常评估+搜索	核函数、损失函数、决策节点等	人工设计架构，需要训练确定参数，参数量小	输入信息不复杂，计算逻辑复杂
深度学习	手工编写模型（RNN、CNN、TRM等）	手工设计表示向量，自动学习向量数值	手工设计操作，自动学习参数	手工设计输出逻辑，通常评估+搜索	神经元、层、损失函数、优化器等	人工设计架构，需要训练确定参数，参数量大	输入信息复杂，计算逻辑复杂

趋势与思考

- 传统机器学习
  - 手工设计特征表示，手工设计模型结构
- 一般深度学习
  - 自动学习特征表示，收工设计模型结构
- 端到端深度学习
  - 表示决策全流程建模，模型结构统一化
- 自动机器学习
  - 自动搜索模型结构，自动完成模型学习

谢谢大家！