一、概论

1. 概念

- 分类和回归（输出为离散值/连续值）
- 维数灾难：随着输入维数增加，算法将需要更多数据
- 欠拟合（train和test都很低）、过拟合（train很高，test很低）

2.基本评价指标

- 均值、方差、协方差
- 距离度量函数：
  - 欧氏距离：平方（求乘积）开根号
  - 余弦相似性：矩阵乘除以各自模
  - 曼哈顿距离：各维度距离求和
  - 切比雪夫距离：各维度距离求Max
  - 马氏距离：带M矩阵的欧氏距离
- 高斯分布（正态分布）
- 概率定义：
  - 类的先验概率 $p(y=i)$
  - 样本的先验概率 $p(\mathbf{x})$
  - 类条件概率（似然） $p(\mathbf{x}|y=i)$
  - 后验概率 $p(y=i|\mathbf{x})$
- 生成/判别式模型
  - 生成模型：

    估计类条件概率和类的先验概率，用贝叶斯定理求后验概率

- 判别式模型：

  找到判别函数，不假设概率模型，直接求后验概率

二、KNN

1. 流程

- 计算测试样本x和D_{train}中所有训练样本$\mathbf{x}_i$的距离
- 对所有距离（相似度）升序（降序）排列
- 选择k个最近（距离最小，相似度最大）样本
- 采用投票法将这k个邻居中数量最多的类别标签分配给x

2. k取值影响

- 取奇数以免平局
- k太小：对噪声敏感，易过拟合
- k太大：对噪声不敏感，易欠拟合

3. 区别

- 懒惰学习：仅保存样本，训练时间开销为0
- 急切学习：（SVM、CNN等）训练阶段对样本进行学习处理，尝试构造一个通用的与输入无关的目标函数

优点：精度高，对异常值不敏感，输入无要求

缺点：计算、空间复杂度都很高

4. 优化

- 特征维度2-5：维诺图（根据一组给定目标，将一个平面划分成靠近每一个目标的多个区块）
- 6-30：KD-Tree（在最大方差维度中位数处split，递归直至每个区域只有一个样本）

- 高维特征：PCA降维、近似最近邻、哈希

三、无监督学习

1. 聚类

- 将相似的数据对象归为同一类
- 聚类算法：根据给定的相似性评价标准，将一个数据集分组/划分成几个聚类
- 有效性：分布情况和选取的特征是否有参考价值

2. 依据

- 簇内相似度高，簇间相似度低

3. 聚类方法

- 基于试探的搜索算法

  选初始中心、定阈值T，对每个点要么归类，要么开辟新中心

- 系统聚类法

  初始：各为一类 递归：最近类合并 条件：类数

- 动态聚类法

  初始：选择若干样本为中心 按距离得到初始聚类

  合理：结束；不合理：改变聚类

4. 动态聚类法--Kmeans

- 选择聚类数量k
- 随机初始化样本
- 对每个样本点计算到中心距离并分配到最近
- 聚类中心更改为该类的位置均值

- 重复前两步直到没有样本所属的聚类改变

5. Kmeans++

- 选择初始中心点的方式不同

初始选法：随机选择第一个；

剩下的每个样本被选中的概率为$\frac{D^2_{min}(x_i,x)}{\Sigma_{x\in X}D^2_{min}(x_i,x)}$，即到所有已有中心的距离最小值的平方作为被选作下一个中心的概率（越近越不容易被选）

四、树学习

1. 概念学习

- 实例集合X：属性矩阵
- 目标概念c：定义在实例集合上的布尔函数$c:X\rightarrow\{0,1\}$
- 训练样例：正例$c(x)=1$，反例$c(x)=0$
- 假设集H：每个假设h上表示X上定义的布尔函数$h:X\rightarrow\{0,1\}$
- 概念学习：寻找一个假设h，使得所有$x\in X$，有$h(x)=c(x)$

2. 决策树

2.1 决策树学习

- 实例：kv对表示，应用最广的归纳推理算法之一
- 目标函数输出值离散
- 健壮性很好
- 能够学习析取表达式
- 归纳偏置：优先选择较小的树

2.2 假设空间搜索

- 从一个假设空间搜索一个正确拟合训练样例的假设

- 搜索的假设空间：可能的决策树集合
- 爬山算法遍历假设空间，从空的树开始，逐步考虑更加复杂的假设，评估函数是信息增益度量

2.3 信息增益

- 熵定义：

  $Entropy(S) = -\sum_{i=1}^{n} p_i log p_i$

  例：

  $Entropy([9_+, 5_-]) = -\frac{9}{14} log \frac{9}{14} - \frac{5}{14} log \frac{5}{14} = 0.940$

- 信息增益：

  使用属性分割样例，导致的期望熵降低

  $$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

  Values(A)：A属性所有可能值的集合

  Sv：属性A等于v的所有样例的集合

  后一项即熵的期望（按属性值比例权重的单个熵求和）

2.4 ID3算法

- 假设空间：所有决策树
- 遍历过程：仅维持单一的当前假设
- 回溯：不回溯（局部最优）
- 基于统计：对错误样例不敏感；不适用于增量处理
- 选择信息熵最大的属性作为节点

2.5 算法对比

代表性树算法对比

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
ID3	分类	多叉树	信息增益	不支持	不支持	不支持
C4.5	分类	多叉树	信息增益比	支持	支持	支持
CART	分类/回归	二叉树	基尼系数 均方差	支持	支持	支持

- 信息增益比：

$$GainRate(S,A)=\frac{Gain(S,A)}{Entropy_A(S)},$$
$$where\ Entropy_A(S)=-\sum_{v\in Values(A)}\frac{|S_v|}{|S|}log\frac{|S_v|}{|S|}$$

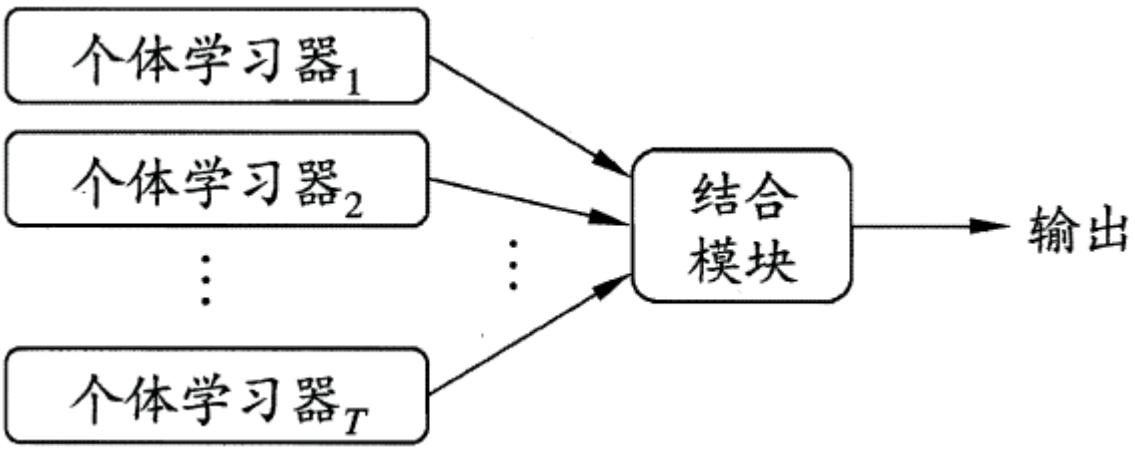
- 基尼指数（Gini）：(K为类数量)

$$Gini(p)=\sum_{k=1}^{K}(1-p_k)p_k$$
$$=1-\sum_{k=1}^{K}p_k^2$$

五、集成学习

1. 原理

预测模型的元方法

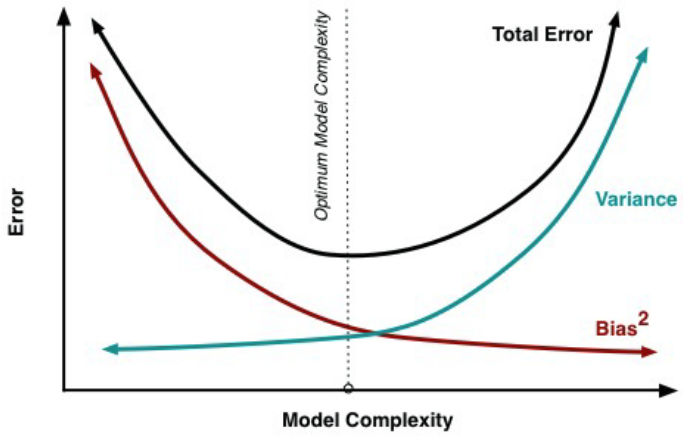


2. 特点

- 本身不是分类器，而是分类器的结合方法
- 通常性能会好于单个分类器

3. Bias-Variance tradeoff

- Bias：学习结果的期望与真实规律的差距（训练期望减真实）
- Variance：学习结果自身的不稳定性（训练的方差）
- 总误差：Err=Bias方+Variance+Random Error



- 序列集成（基学习器）法减少偏差

- 并行集成（基学习器）法减少方差
- 结合策略：平均法、投票法、学习法

4. Bagging

Bootstrap aggregating

- 并行式集成学习，降低分类器方差，改善泛化
- 性能依赖于基分类器的稳定性，误差由基分类器的bias决定
- 代表方法：随机森林

5. Boosting

概率近似正确（PAC）理论

- 强弱可学习：存在一个多项式的学习算法能够学习概念（类）且正确率高于随机猜测。依据正确率分为强/弱可学习的（本质等价）
- Boosting：通过改变一系列弱分类器的权值分布，组合构成一个强分类器
- 代表方法：Ada（ptive） Boost

六、概率与学习

- 张量（tensor）：一个泛化的实数构成的n-维数组

1. 带约束的数学优化问题

$$\begin{aligned} \text{minimize} \quad & f_0(x) \\ \text{subject to} \quad & f_i(x) \leq b_i, \quad i = 1, \ldots, m. \end{aligned}$$

- 优化变量：$\boldsymbol{x} = (x_1, \ldots, x_n)^T \in \mathbb{R}^n$
- 目标函数：$f_0 : \mathbb{R}^n \to \mathbb{R}$
- 约束函数：$f_i : \mathbb{R}^n \to \mathbb{R}, i = 1, \ldots, m$
- 最优解　：$\boldsymbol{x}^* = (x_1, \ldots, x_n)^T \in \mathbb{R}^n$

2. 不带约束的数学优化问题

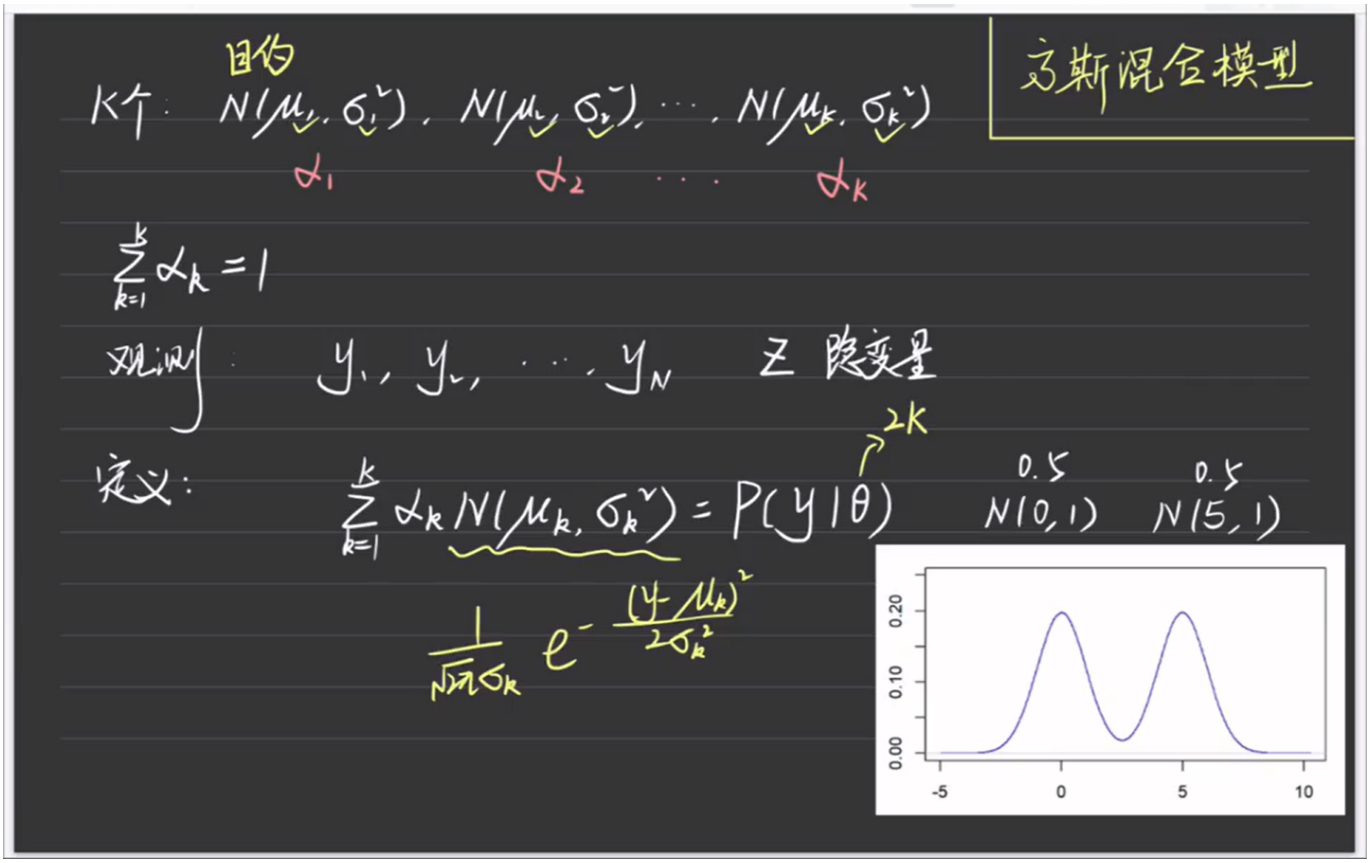
最小二乘问题

$$\text{minimize} \quad f_0(x) = \|Ax - b\|_2^2 = \sum_{i=1}^{k} (a_i^T x - b_i)^2$$

- 优化变量：$\boldsymbol{x} = (x_1, \ldots, x_n)^T \in \mathbb{R}^n$
- 系数矩阵：$A \in \mathbb{R}^{k*n}$
- A 的第i行：$a_i^T \in \mathbb{R}^{1*n}$
- 最优解　：$\boldsymbol{x}^* = (x_1, \ldots, x_n)^T \in \mathbb{R}^n$

3.GMM 高斯混合模型

多个高斯分布线性加权，也是一个高斯分布，称作高斯混合模型


高斯混合模型
K个 N(μ1, σ1²), N(μ2, σ2²), ..., N(μk, σk²)
α1 α2 ... αk
Σ_{k=1}^{k} αk = 1
观测 y1, y2, ..., yN 关于隐变量
定义: Σ_{k=1}^{k} αk N(μk, σk²) = P(y|θ)
2k
0.5 N(0,1) 0.5 N(5,1)
1/√(2π)σk e^{-(y-μk)²/2σk²}
0.00 0.10 0.20
-5 0 5 10

4. MLE最大似然估计：

最大似然估计

- 最大似然估计 (Maximum likelihood estimation，MLE)
  - ✓ **高斯混合模型**：

最大对数似然估计MLE：$\hat{\theta} = \underset{\theta \in \Theta}{\operatorname{argmax}} \ln \mathcal{L}(\theta|X)$

$$\ln \mathcal{L}(\theta|X) = \sum_{j=1}^{M} \ln p(\boldsymbol{x}_j|\theta) = \sum_{j=1}^{M} \ln \sum_{i=1}^{N} \alpha_i N(\boldsymbol{x}_j; \boldsymbol{\mu}_i, \Sigma_i)$$

$$= \sum_{X} \ln \sum_{\mathcal{Z}} p(\mathcal{Z}|\alpha)\, p(X|\mathcal{Z}; \boldsymbol{\mu}, \Sigma)$$

$$= \sum_{X} \ln \sum_{\mathcal{Z}} p(X, \mathcal{Z}|\theta)$$

$Pr(\mathcal{Z} = i) = \alpha_i$

$\theta = \{\alpha, \boldsymbol{\mu}, \Sigma\}$

5.期望最大化算法

期望最大化（EM）算法

- EM算法（Expectation-Maximization algorithm）

E-Step: 利用可观测数据$\mathcal{X}$和当前估计的参数为$\boldsymbol{\theta}^{(t)}$，估计更好的隐藏变量$\mathcal{Z}$

M-Step: 利用可观测数据$\mathcal{X}$和当前估计的隐藏变量$\mathcal{Z}$，估计更好的参数$\boldsymbol{\theta}^{(t+1)}$

Repeat: 重复上述两个步骤，直至收敛

2023/02/16 41

七、支持向量机SVM

1. 线性支持向量机

- 一个样例的间隔（margin）是其到分界超平面的垂直距离
- SVM最大化最小间隔
- 具有最小间隔的点称为支持向量
- 拉格朗日乘子法在对偶空间优化后得到最优a，得到原始空间的最优解w
- Soft margin：
  - 允许少数点margin比1小
  - 松弛变量ξ_i：允许犯的错误

- 惩罚：C代价函数

2. 非线性支持向量机

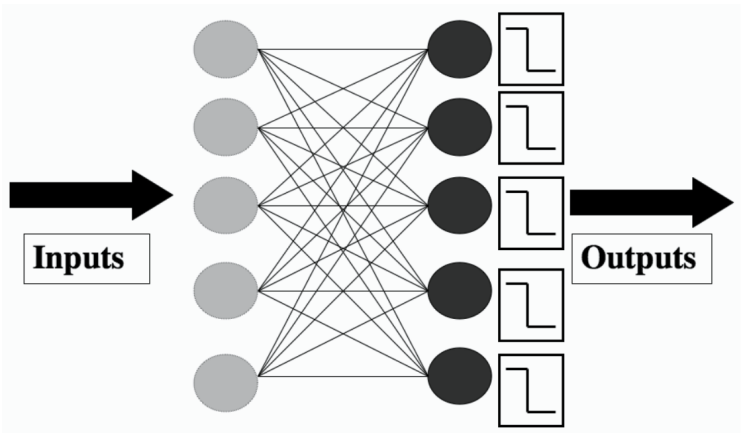
- 核函数：将低维线性不可分的样本升维，构建高维超平面后，降低回低维

八、神经元和感知机

1. 神经元

- MP神经元：输入向量X和权值W，输出激活函数$f(net)$决定是否激活
- 激励函数：非线性，阶跃函数、Sigmoid函数、ReLU、LeakyReLU等

2. 感知机

- 最简单的前馈式人工神经网络
- 二元线性分类器，输入特征向量，映射到二元输出值上
- 

- 算法
  - 权值初始化
  - 输入样本对
  - 计算输出
  - 根据学习规则调整权重
  - 回到第二步直到所有实际输出都等于期望输出

九、神经网络

1. 向前计算

分阶段逐层计算输入和输出

2. 反向传播

- BP：通过比较输出与期望输出，产生误差信号；反向传播误差信号以修正突触的权值
- 计算每一层神经元的局域梯度delta，然后修正deltaw修正突触权值

十、深度学习

1.CNN 卷积神经网络

一个卷积神经网络主要由以下5层组成：

- 数据输入层/ Input layer
- 卷积计算层/ CONV layer
- ReLU激励层 / ReLU layer
- 池化层 / Pooling layer
- 全连接层 / FC layer

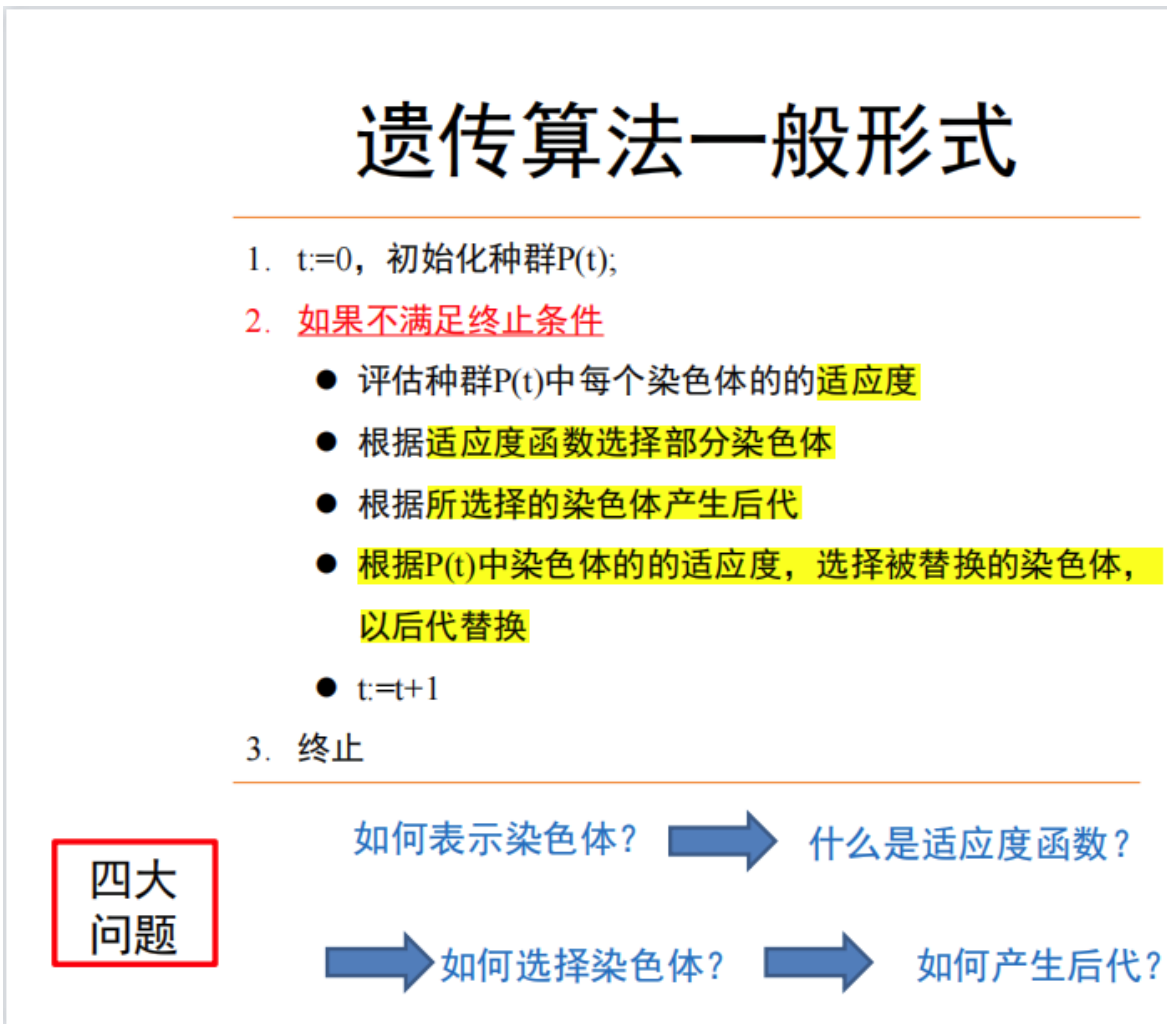
输入后求卷积变为更小的矩阵；池化减小大小（下采样）；ReLU激活；FC将矩阵延展为一个tensor；最终由softmax完成分类

2. 深度学习技巧

数据增广、预处理、初始化、过滤器、池化大小、学习率

十一、演化学习

1. 遗传算法



- 染色体：0，1，#（无所谓）的串
- 适应度函数：依据实际情况
- 选择染色体：
  - 锦标赛选择：每次放回抽样选最优，直到达到原来的种群规模
  - 截断选择：根据适应度排序，前f个进入下一代种群并复制
  - 轮盘赌选择：按适应度概率随机抽取
- 产生后代：染色体重组
  - 遗传算子：按交叉点交换后续"染色体"片段
  - 变异：极低概率某个位取非操作

2. 模式定理

- 01#,#表示0或1，如##10包含4个，0010等
- o(H)阶：确定位置的个数
- d(H)长度：第一个确定位置到最后一个确定位置的距离
- m(s,t)表示t代种群中模式s的实例数量

十二、维度约简

1. LDA

有监督降维：尽可能按类别区分接近

- 计算每个类的均值μ_i，全局样本均值μ
- 计算类内散度矩阵$S_W = \sum_{classes\ c} \sum_{j \in c} p_c(x_j - \mu_c)(x_j - \mu_c)^T$，($p_c$：概率)

  类间散度矩阵$S_B = \sum_{classes\ c} (\mu_c - \mu)(\mu_c - \mu)^T$
- 对矩阵$S_W^{-1} S_B$做特征值分解
- 取最大的数个特征值对应的特征向量
- 计算投影矩阵

2. PCA

无监督：将数据尽可能散地降到低维

- 样本去中心化（所有样本减1/N均值，将均值调到0）
- 计算协方差矩阵$Cov = \frac{1}{n} \sum_{i,j=1}^{n} x_i^T \cdot x_j$
- 对协方差矩阵做特征值分解
- 取最大的数个特征值所对应的特征向量
- 计算投影矩阵（前k个特征向量按行排列）

3. ICA（开摆）

十三、强化学习

思路：根据先验得到初始认知，（伴随随机性）选择动作，获得经验，修改认知，回退修改历史认知

1. MDP马尔可夫过程

- S 状态集合
- A 动作集合
- δ 状态转移概率
- R 即时奖赏函数
- 返回函数：R的线性组合
- 动作选择
    - 目标：期望返回值
    - 对象：策略
    - 潜在的公理：最优策略

2. 动态规划

给定一个完全已知的MDP模型：

- 策略评估：给定策略，给出评估值
- 最优控制：寻找一个策略，使其从任状态出发，返回值都最大
- 计算：值迭代或策略迭代

值函数：

- $V^{\pi}(s)$从状态s出发，采用π策略所获得的期望返回值
- $Q^{\pi}(s,a)$从状态s出发采用a动作和π策略所获得的期望返回值
- 最优:$V^{*}(s)=max_{\pi}V^{\pi}(s),\ V^{\pi}(s)=max_{a}Q^{\pi}(s,a)$

贪心策略和ε-贪心策略：

100%/1-ε概率选择$\pi(s) = argmax_a Q^{\pi}(s,a)$

3. 强化学习要素

- 策略：选择动作的规则
- 奖赏/返回：学习系统试图最大化的函数
- 值函数：评估策略
- 模型：环境（问题）演变遵循的法则

4. Monte Carlo策略：

- 目标：学习$V^{\pi}(s)$
- 给定：在访问状态s采用策略π获得的若干经验
- 思路：访问状态s后对所获得的返回进行平均
- MC策略迭代：使用MC方法对策略进行评估，计算值函数
- MC策略修正：根据值函数采用贪心策略进行策略修正

$$V(s_t) \leftarrow V(s_t) + \alpha\lfloor R_t - V(s_t)\rfloor$$

4.1 时差学习

- 若干次平均后得到真实的返回值
- 时间差分方法（TD）：在每一次经验后，都对返回值进行估计

$$V(s_t) \leftarrow V(s_t) + \alpha\lfloor r_t + \gamma V(s_{t+1}) - V(s_t)\rfloor$$

5. Bootstraps 和 Sampling

Bootstraps：通过一个估计值进行更新，动态规划/时差学习中采用

Sampling：根据经验进行更新，蒙特卡洛/时差学习中采用

6. Q学习和SARSA

Q学习：

$$Q(s,a) \leftarrow Q(s,a) + \mu(r + \gamma max_{a'} Q(s',a') - Q(s,a)$$

SARSA:

$$Q(s,a) \leftarrow Q(s,a) + \mu(r + \gamma Q(s',a') - Q(s,a)$$

7. N步TD预测，TD(λ)算法

TD(λ)算法

1. 初始化V(s), e(s)=0
2. 对每一个episode，重复

   初始化s

   对episode中的每一步

   根据ε-贪心策略选择动作a

   执行动作a，获得r和s'

   $\Delta \leftarrow r + \gamma V(s') - V(s)$

   $e(s) \leftarrow e(s)+1$

   对于所有s

   $V(s) \leftarrow V(s)+\alpha\Delta e(s)$

   $e(s) \leftarrow \gamma\lambda e(s)$

   $s \leftarrow s'$

   直到s为终止状态