Chapter 1 概述

2018年12月28日 18:59

1.总误差 Total – Error = Bias + Variance + Noise
Bias 模型本身精度，模型对数据分布的表达能力
Variance 泛化能力
Noise 观察数据本身的不确定性
2.奥卡姆剃刀：保证好的表达能力，选择简单的模型。
No free lunch：一个模型在一个条件和数据下好，则在另一个条件和数据下不好。
3.监督学习 无监督学习 半监督学习 强化学习
4.参数与模型：有较强的模型假设，训练数据少；非参数模型：训练数据多
5.生成式模型：先对每类数据分布建模p(x|Ck)，在按照贝叶斯公式建模。判别式模型：直接对后验概率建模。

Chapter 2 线性模型

2018年12月28日 19:00

一.线性预测模型

线性回归、Logistic回归，贝叶斯方法。有监督学习。
x和t都是可见变量，由x对t预测。

0.多项式拟合

预测函数: M次多项式，误差函数: 平方误差，优化误差函数，有唯一解。
一般线性拟合：对x做非线性映射，x扩展到多维变量。

1.线性回归

1）假设t是随机的，这个随机性用一个随机变量表示，符合高斯分布。epsilon~N（0，1/β）
生成模型。过程：x -- 非线性映射 – Φ – 线性映射 – y – 加入高斯噪声 -- t
2）似然函数：模型生成某一数据集的概率。
最大似然估计：
对w的最大似然估计等价于平方误差最小化。说明最大似然估计等价于线性拟合。线性拟合中的误差源于噪声高斯分布。当噪声不符合高斯分布时，应设计其他误差函数。
对方差1/β的最大似然估计等价于回归模型对训练数据进行预测得到的预测残差。
预测x'的目标变量t'：求预测的期望E(t'|x')。和线性预测一致。

2.Logistic回归

处理离散噪声。目标t离散，不能假设高斯分布。

1）Fisher准则

分类问题三种方法：区分函数（不考虑概率意义，直接涉及分类函数），生成式概率模型（对每个类建立模型，用贝叶斯公式得到后验概率），区分性概率模型（直接对后验概率建模，不对数据分布做显示假设）
Fisher准则：区分性度量，区分性越大越好。类间距离越大、类内分散程度越小，区分性越强。最有区分性的方向w是两类采样点中心连线的方向，基于类内方差矩阵调整。
问题：Fisher准则在二分类中得到的映射函数和线性拟合的映射函数是等价的，说明做了类别是高斯分布的假设，不合理。当出现奇异点时，区分性下降。

2）Logistics回归

修正线性拟合，防止奇异点影响过大，方法是1.进行非线性压缩，预测函数Logistic函数；2.t的高斯分布假设不合理，设为伯努利分布
生成过程：x – 非线性映射Φ生成特征 -- 线性映射投影到标量空间 – Logistic压缩到0至1之间 – 符合伯努利分布的t

3）Softmax回归

多分类问题：伯努利分布→多项式分布，0-1表示→one-hot表示
概率预测函数：Softmax函数，归一化到(0,1)之间。

二.线性概率模型

主成分分析、概率主成分分析、概率线性判别分析。无监督学习。
推理任务，由t推出x。t是可见变量，x是隐变量。对p(x)和p(t|x)的分布进行假设，用贝

叶斯公式计算后验概率p(x|t)。

1.PCA

希望找到若干正交方向，观察数据在这些方向上代表元数据的分布性质，每个方向是一个主成分。可用于降维。

主成分对数据的代表性准则：数据在映射空间里方差最大，映射恢复成原始数据的损失最小。

2.PPCA

过程：基于先验概率p(x)生成隐变量采样点x，线性变换Wx，移位μ，加入高斯噪音epsilon。都是高斯的，t也是高斯的，称为线性高斯模型。

p(x)设了，p(t|x)建模了，可求出p(t)，再用贝叶斯公式求出后验概率p(x|t)。

3.PLDA

生成过程：从一个一维正太分布采样生成中心类μk，线性映射到二维空间生成μk'。高斯随机变量生成随机数x，线性变换Wx，加入高斯随机变量Wx+epsilon，再加中心类，得到采样数据t。

是线性高斯模型。

三、贝叶斯方法

将模型参数看作随机变量，参数取值范围的先验知识转化为参数的先验概率。模型优化不再是寻找最优参数，而是对参数进行最大后验估计。数据量大时最大后验趋近于最大似然。

chapter 3 神经模型

2018年12月28日 19:02

一、概述

- 受人类神经结构启发。信息表现在连接结构上，而不是神经元。
- 定义：wikipedia/工程化定义。特点：同质，连接，可学习。
- 与符号方法：符号方法是“白箱”，但限制较多：任务不通用、泛化性差、难以根据新数据修正。但是，神经模型的困难：符号表达是离散的，解决方法：enbedding。更多困难：小概率的符号有价值，神经模型会忽略。

二、基于映射的神经模型

输入特征向量，输出预测目标。是线性模型的扩展。

1.线性模型开始

1）logistic模型中的sigmoid函数替换为阶跃函数，可以实现二分类。二分类预测：【公式】是早期的感知器模型。logistic模型被称为“近线性模型”。

2）模型优化。

线性回归模型存在闭式解，近线性模型或其他模型一般不存在闭式解，采用数值法，迭代逼近。常用SGD。

SGD用在感知器模型里。误差函数：【公式】问题：SGD要求目标函数连续可导，但是错误预测集M会变，目标函数不连续。解决：1.不看阶跃函数2.假设M在邻域内不变，强行求梯度。迭代后M变化会使L收敛性不直观，但能保证收敛。

感知器收敛定理：在**线性可分**（重要条件。举例，XOR线性不可分）数据集上经过有限迭代之后，确保收敛到对该数据集完美可分的分类器。（问题：完美可分指的是针对每一个数据都能正确分类？）

3）非线性扩展

解决线性不可分的数据集。1.阶跃函数→sigmoid，调整学习率保证收敛，即logistic回归。（问题：调整学习率就能保证收敛?）2.非线性变换，在变换空间建立线性模型。多层感知器，径向基函数，核函数。

2.多层感知器

1）结构。一层→多层。一般近似定理保证包含一个隐层的MLP可以模拟一切函数。

2）训练。

SGD，BP：SGD收敛快、随机性强→消除不合理初始化影响。BP顺序求导，避免重复计算。（问题：哪里重复计算了？）

问题：梯度爆炸/消失；过拟合；局部最优。

技巧：

- 输入输出正规化和特征变换。优点是防止取值分散→权重初始化在0附近，需迭代多轮才能适应数据值域；饱和区。方法：最大最小值归一、均值方差归一、高斯化。其他：隐层正规化；降维。
- 激发函数。sigmoid易在初始阶段陷入饱和区；ReLU无界易导致发散。深度网络一般使用分段线性函数（ReLU/Max-out），梯度传导更容易。
- 初始化。均匀分布。【公式】第四章详细讲了p164。
- 二阶信息。第四章讲了。优点是能利用曲率信息调整学习率。各种牛顿法，缺点是Hessian矩阵不好计算。故用一阶信息的统计量来近似二阶信息，AdaGrad，AdaDelta，Adam，Natural SGD等。（问题：为啥能近似？）
- 动量。另一种更新参数的公式，考虑了上一个mini-batch的梯度。优点是：当目标函数在不同方向曲率相差较大时，可补偿不同曲率对学习率的不同要求，是以较低计算代

价获得二阶信息的方法。（问题：为啥就补偿了？哪个是二阶信息？）

- 课程学习。队训练样本分组，先学简单的再学难的。（问题：难易的分组标准？）
- 迁移学习。用以有网络作为基础，用其中知识学习新的任务。（问题：意思是使用别的数据集训练出来的模型参数？一个是把原始模型的前几层拿来作为特征提取层，另一个是原始模型对新任务进行指导。咋指导？）
- 正则化。作用：防止过拟合。因为神经网络完全是数据驱动，缺少先验知识。方法：加入l1/l2范数；early stop；对参数/梯度取值范围进行限制（问题：梯度截断？）；带噪训练（问题：我不觉得dropout是带噪训练，它没有在隐藏节点上加噪声，它只是部分地训练网络）；剪裁模型。

3.径向基函数网络

1）在映射空间设置一系列标识点作为基，采样点到标识点的举距离函数是径向基函数。用无监督方法来选择标识点。

2）意义

插值分析：插值中的映射函数如果是训练出来的话则变成RBF网络。

核回归：核函数定义为径向基函数，tn视为权重wn，则得到RBF形式。

分类任务：

3）多层感知器和径向基函数网络比较

4.神经网络模型与先验知识

特征、目标函数、超参数等可以利用。

1）结构化模型，eg CNN

2）混合密度网络

3）贝叶斯方法：希望引入“稀疏”的结构限制。网络权重上引入Laplace分布，相当于目标函数上引入l1正则项，鼓励小参数和连接权重置零，能达到稀疏性要求。

三、基于记忆的神经模型

针对的任务：没有标记t，希望描述x分布。

1.Kohonen网络（自组织映射 Self Organization Map）

将连续空间数据点映射到离散空间。高维到低维。

2.Hopfield网络

全连接，二值神经元节点，所有节点的取值作为一个模式。由Hebbian学习权重。

3.Boltzmann machine

引入隐藏节点和节点的取值随机性。

4.RBM (Restricted BM)

限制只有隐藏节点和非隐藏节点可以连接。

5.Auto Encoder

四、基于过程的模型

映射模型和记忆模型是静态模型，描述了数据分布特征。序列问题的模型是动态模型/过程模型。RNN。

1.Elman RNN

2.门网络LSTM/GRU

3.Seq2Seq

4.Seq2Seq + Attention诗词生成

五、神经图灵机

定义一系列元操作，通过组合学习更复杂的动态性。模拟传统图灵机，内存单元-纸带，神经网络-控制器，记忆单元-寄存器，寻址和读写操作-读写头。

Chapter4 深度学习

2018年12月28日 19:02

一.浅层学习-深层学习

1. 表达能力指数级提高：嵌套减少参数量
2. 层次表示：越高层越具有不变性
3. 非监督学习：DNN不是以优化为目标的近似，而是层次化学习。非监督学习提取特征，监督学习选择和建模。特征和模型视为一个整体，越底层越偏向特征，高层偏向建模。
4. 数据驱动。

二、训练

1.基础训练算法

1）GD。BGD，SGD，mini-batch SGD。minibatch SGD实用，且有正则化效果。

2）二阶方法。利用曲率调整学习率。Newton开销大，Quasi-Newton, Truncated Newton作为近似二阶方法。

3）NSGD。对SGD进行二阶约束。二阶信息可以由一阶信息的二阶统计量得到，减小计算量。（？）

2.训练困难

1）局部极值

2）梯度奇异

3）饱和区

4）梯度爆炸/消失

5）马鞍点

在某些方向局部最小，在另一些方向是局部最大或非极值的点。用Hessian矩阵可以判断多元函数极值点。原理是：Hessian矩阵的特征值的符号可以判断驻点在该特征向量方向上的极值属性，因此看Hessian矩阵里的特征值可以知道该点在各个方向上的属性，因此判断该点属性（因为马鞍点的定义：在某些方向局部最小，在另一些方向是局部最大或非极值）。正定-极小值；负定-极大值；有正数也有负数-严格马鞍点；奇异的（特征值包括0）-不定，驻点/猴状马鞍点。

目标函数维度高时，马鞍点数量增加，导致很多马鞍点都不是局部最优。对于SGD，该点附近梯度消失，难以跳出。解决办法是参数更新时加入扰动，使之跳出马鞍点，但是扰动不好加。对于二阶方法，有几个方法。

负奇异值越多，模型越不充分，即马鞍点性质越显著，越不好。

3.训练技巧

1）参数初始化

- 最好的是同时考虑前向传导和梯度传导，使参数符合一个均匀分布。【公式】
  稀疏初始化：防止数值上太小，对某些权重置零。
- 随机正交矩阵。
- 实验时应该观察初始几个mini-batch的前向计算和梯度回传是否合理。

2）学习率

- 第t次迭代除以t。
- 动量。

- 学习率自适应：AdaGrad，RMSProp，AdaDelta。
- 学习率自适应+动量：Adam

3）batch norm

4）dropout：

训练时随机扔掉一些节点，减轻计算负担，预测时是考虑所有节点的。这是一种高效的群体决策，相当于每个mini-batch训练一个子网络，共享一个父网络的结构和参数，训练后得到平均网络。和群体决策相比，没有很多数量的模型，计算量小。

随机生成子网络：伯努利分布采样决定节点去留。

也有随机扔边的dropconnect方法。

三、正则化

1.结构化网络与参数共享

CNN

RNN

NADE

2.范式约束与稀疏网络

3.加噪训练与数据扩增

4.联合训练

5.知识迁移

四、生成模型下的深度学习

1.神经网络的简单概率表达

2.后验拟合和Variational AE

3.Variational RNN

五、计算图与复杂神经网络

1.Chain ReLU到计算图

2.基于计算图的参数优化

3.计算图的模块化

4.计算图与深度神经网络

六、计算平台与方法

1.GPU与TPU

GPU提高矩阵乘法速度，适合大量简单计算。数据规模必须大。数据训练阶段：成批快大规模计算，计算精度高。

在解码阶段，数据是流式的，精度低，GPU效率低，用TPU。

2.并行计算

模型并行/数据并行，同步更新/异步更新……

3.模型压缩

减少解码阶段计算量。DNN都是过度参数化的，结构和参数上有冗余。

- 网络剪裁：去掉节点/连接。
- 对矩阵进行结构化处理。（不懂）
- 参数量化：参数在0附近，量化可减少存储空间。
- 参数共享。

注意：剪裁可以节约空间但未必提速，因为计算设备对稠密矩阵运算有优化，对稀疏矩阵没有。

七、应用

1.语音信号处理

2.自然语言处理

word embedding很重要。用连续向量代表词。word2vec是学习磁向量的工具。

机器翻译简史：

- 统计概率模型SMT，分为翻译模型和语言模型。缺点是只能“句译”不能“意译”，没有语义关系的度量。
- 带有Attention的NMT：词向量解决语义度量的问题。Attention提高长句翻译性能。
- Memory机制：对于出现频率低的词保存在一个内存结构中，类似于生僻词查词典。

3.计算机视觉

Chapter 5 核方法

2018年10月11日 17:31

推出核函数

1. 由线性回归引出。线性回归：
2. 化为对偶问题。将参数w中拿出一个ϕ，和ϕ的转置凑成对偶。

因为Φ是向量，乘以转置后得到方阵，元素是向量元素的两两组合。每个矩阵元素涉及两个x，可以表示为x之间的关系（距离）。

$y = \phi(\vec{x})^T\vec{w}$ 线性

$y = \phi^T\phi\alpha$ $\omega = \phi \cdot \alpha$

$y = \phi^T\phi\alpha = K\alpha$

$K = \phi^T\phi$ 核函数

kij=Φ(xi)TΦ(yj)

定义

k是核函数的充要条件：对称半正定

理论支撑：Cholesky decomposition

定理：

若

$A \in R^{n\times n}$

对称正定，则存在一个对角元为正数的下三角矩阵

$L \in R^{n\times n}$

使得

$A = LL^T$

成立。

分解条件：

1. Hermitian matrix：矩阵中的元素共轭对称（实数则为对称）
2. Positive-definite：正定，即(x^T)Ax>0

★ 如果A是半正定也能分解，但结果不唯一

核函数性质

再生希尔伯特空间(RKHS)

常用核函数

线性核 高斯核 概率核

集合/序列/图上的核 基于局部距离度量的核

Kernel PCA

高斯过程

支持向量机

线性 寻找**最大**边界分类面

边界样本集到分类面的距离为边界，样本为支持向量（只保留这些样本）

Chapter 6 概率图模型

2018年10月18日 14:46

Intro

神经模型：缺点：数据驱动，参数多。
核方法：优点：参数少，需要的数据少。缺点：需要较强先验知识来设计核函数。
概率模型：优点：将先验知识转化为变量相关性。缺点：变量太多时很复杂。
概率图模型：优点：简化了概率模型。

概率图模型：
变量-节点，变量间的相关性-边
有向图模型（信任网络/贝叶斯网络）：因果关系
无向图模型（马尔科夫随机场）：概率独立性

有向图模型

所有变量的联合概率分布：

$$p(X)=\prod_{x_i\in X} p(x_i|Pa(x_i)),$$

- 每个点的关于父节点的条件概率的乘积。
- 如果不存在父节点，条件概率退化为先验概率。

任务：

预测 - 原因推结果
推理：结果推原因
分析：判断是原因还是结果

特点：数据x和标签t都是观测变量。打破了监督学习和非监督学习的边界。

变量相关性判断

已知有向图，判断任意两个变量是否有相关性。判断标准：两个节点间是否有通路。

- Explain Away现象
- 贝叶斯球

无向图模型

不描述因果关系，仅描述相关性。
联合概率： 归一化的极大团的乘积 （离散、连续）

$$p(X)=1/Z\prod_c \psi_c(X_c)$$

$$Z=\int \prod_c \psi_c(X_c)\mathrm{d}X.$$

问题：连续变量X能表示为图模型？

变量相关性判断

两个变量间是否有路径连接。注意，如果路径中的某个节点是观测变量，则路径是阻断的。

有向图无向图相互转化

有向图无向图对比

常用概率图模型

高斯混合模型

若干高斯分布加权平均来描述一个复杂分布。

$$p(x)=\sum_{k=1}^{K}\pi_k N(x;\mu_k,\sigma_k^2)$$

单高斯分布估计参数

最大似然准则。

似然函数的变量是模型参数，函数表示生成样本集合的概率。

$$L(\mu,\sigma)=\prod_{n=1}^{N}p(x_n;\mu,\sigma^2).$$

令似然函数最大化，表示所选参数使生成样本集合的概率最大化。即令μ，σ偏导等于零，得到参数：

$$\tilde{\mu}=\frac{1}{N}\sum_{n=1}^{N}x_n,$$

$$\tilde{\sigma}^2=\frac{1}{N}\sum_{n=1}^{N}\{x_n-\tilde{\mu}\}^2.$$

参数估计

难点：引入了隐变量，即生成一个样本点时不知道哪个高斯源被激活。（假设一次只有一个高斯源被激发，z_n取值在1至k之间，表示哪个被激发。每个被激发的概率是pi）

思路：如果知道隐变量z，则可以将数据按照不同分步进行划分，分别求参数。

解决办法：x_n不再属于一个高斯成分，而是在每个高斯成分中都占一定比例，比例为后验概率$p(z_n|x_n)$.然后每个高斯成分按单高斯模型训练参数。

步骤：

1. 求后验概率

$$p(z_n|x_n)=\frac{p(x_n|z_n)p(z_n)}{p(x_n)}=\frac{p(x_n|z_n)p(z_n)}{\sum_k p(x_n|z_n=k)p(z_n=k)},$$

   $p(z_n=k)=\pi_k$是先验概率。每个样本对第k个高斯成分的贡献的比例为$p(z_n=k|x_n)$，记为r_{nk}。

2. 求参数

隐马尔科夫模型

特点是存在隐变量，即状态变量。

有向图。生成模型（根据图的方向生成）。由马尔可夫链（随机）生成状态序列，每个状态（随机）生成一个观察值。一般生成离散值用多项式分布，生成连续值用混合高斯分布。最终模型是p(x|pi, A, b)

[pi, A是马氏链参数，b是生成观察值概率分布的参数]

预测任务：x观察序列，y预测序列=状态序列。

本质是对p(x, y)建模，在推理出p(x|y)。

线性条件随机场

无向图。不对条件概率做定义。用“团”的势函数来计算概率。

本质是直接对p(x|y)建模。

★ EM算法

精确推理算法

加加-乘积算法

树状图的加加-乘积算法

联合树算法

近似推理算法

采样法

基于采样来估计概率分布。

- MCMC
- Gibbs：需要知道样本中一个属性在其它所有属性下的条件概率，然后利用这个条件概率来分布产生各个属性的样本值。（由条件概率求联合概率）

变分法

对比

Chapter 8 非参数模型

2018年10月11日 11:27

❖ 简单非参数模型

1. 直方图模型

对分布不做先验假设，统计数据分布

2. K近邻（KNN）

原理：假设空间是连续的，所以一个测试点与周围样本属于一个类别。

方法：由最近的k个样本的类别来决定测试点的类别。

评价：不对数据分布做先验假设。数据多到充满整个空间时，KNN超过所有参数模型。数据少时，参数模型的先验假设是有价值的；数据多时成为模型表达能力的限制。

3. 支持向量机（SVM）

- 非参数模型特点：
  a. 不对数据分布做假设。
  b. 保留众多数据。模型随数据增长。

- 非参数模型并非没有参数。
  例如SVM是参数/非参数混合模型 - 核函数是参数的，支持向量选择是非参数的。

❖ 高斯过程

1. 高斯过程

概念：

高斯变量X→扩展到n维（无限维）→高斯过程（每个xi是一维）

- n为无限维：说明是连续域上的随机变量
- 每一个维度上的随机变量服从高斯分布
- 这些随机变量的任意有限集合都服从多元高斯分布（一致性）

一致性的意义：
用有限点的性质代替无穷序列的性质。函数的形式没有限制，而是用点去限制的函数。

形式：

$$\begin{bmatrix} y(\mathbf{x}_1) \\ \vdots \\ y(\mathbf{x}_N) \end{bmatrix} \sim N\left(\begin{bmatrix} m(\mathbf{x}_1) \\ \vdots \\ m(\mathbf{x}_N) \end{bmatrix}, \begin{bmatrix} k(\mathbf{x}_1,\mathbf{x}_1) & \cdots & k(\mathbf{x}_1,\mathbf{x}_N) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}_N,\mathbf{x}_1) & \cdots & k(\mathbf{x}_N,\mathbf{x}_N) \end{bmatrix} \right)$$

高斯分布由mean, variance决定；
多元高斯分布由mean vector和covariance vector决定；
高斯过程由mean function和covariance function决定。

mean function：函数的基准线。一般取0。

covariance function：协方差矩阵，是不同维度的x的相关性的度量。

★ covariance function = Kernel

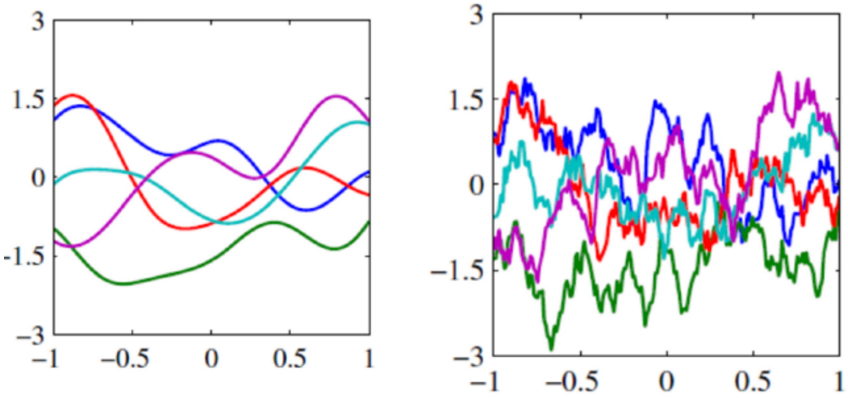
根据定义，Cov(y1,y2)=E{[y1-E(y1)][y2-E(y2)]}。k具体形式是由y的模型的形式决定的。

线性模型：$y=\mathbf{w}^T\mathbf{x}$，$k_{ij}=\alpha^{-1}\mathbf{x}_i\mathbf{x}_j$

非线性模型：$y=\mathbf{w}^T\Phi(\mathbf{x})$, $kij=\alpha^{-1}\Phi(\mathbf{x}_i)\Phi(\mathbf{x}_j)$, 发现k符合Kernel的形式。

- 因为mean和covariance决定了高斯过程，又因为covariance等于Kernel且mean一般取0，所以直接定义Kernel便能决定高斯过程。
- 根据核方法，不同的Kernel描述不同的模型，因此不同Kernel生成的高斯过程对应的随机函数性质不同（样子不同）。协方差越大，曲线越平坦；反之起伏大。

下图是不同Kernel生成的高斯过程采样的函数簇。



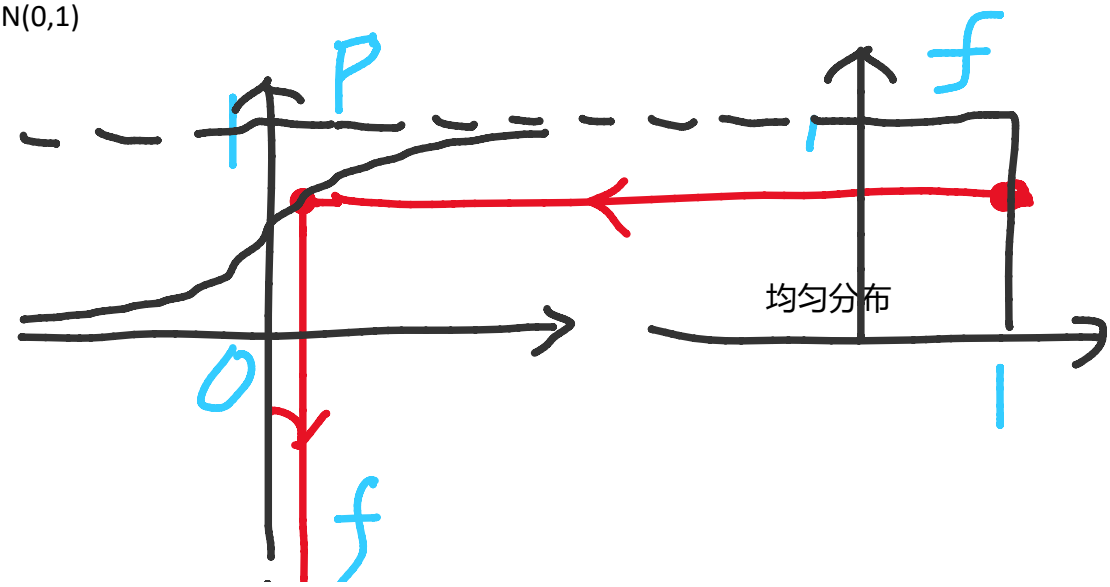
采样过程：

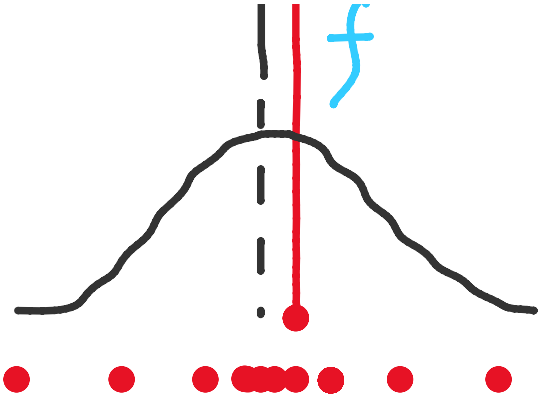
1）为什么要采样?

因为高斯过程是关于函数的随机分布，就像随机变量需要采样样本点，随机过程采样出的是随机函数，无限维的样本点。理解为把无限维的样本点的维度视为序列，把每一维的样本点连起来，则成了函数。

2）采样过程

- 一维 x~N(0,1)

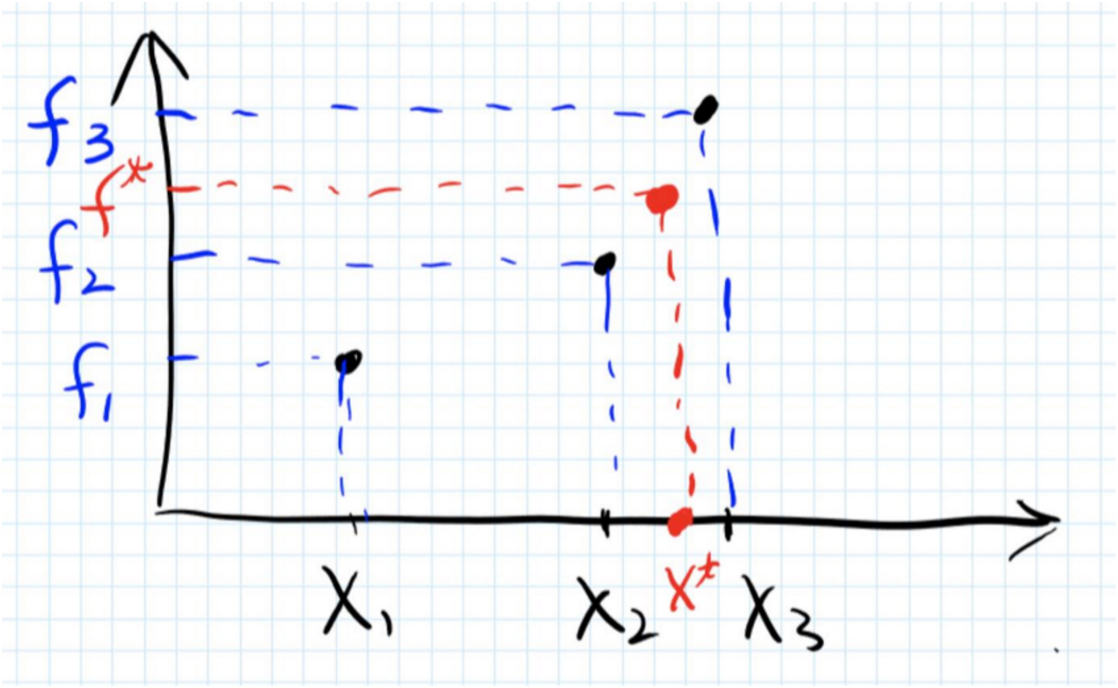


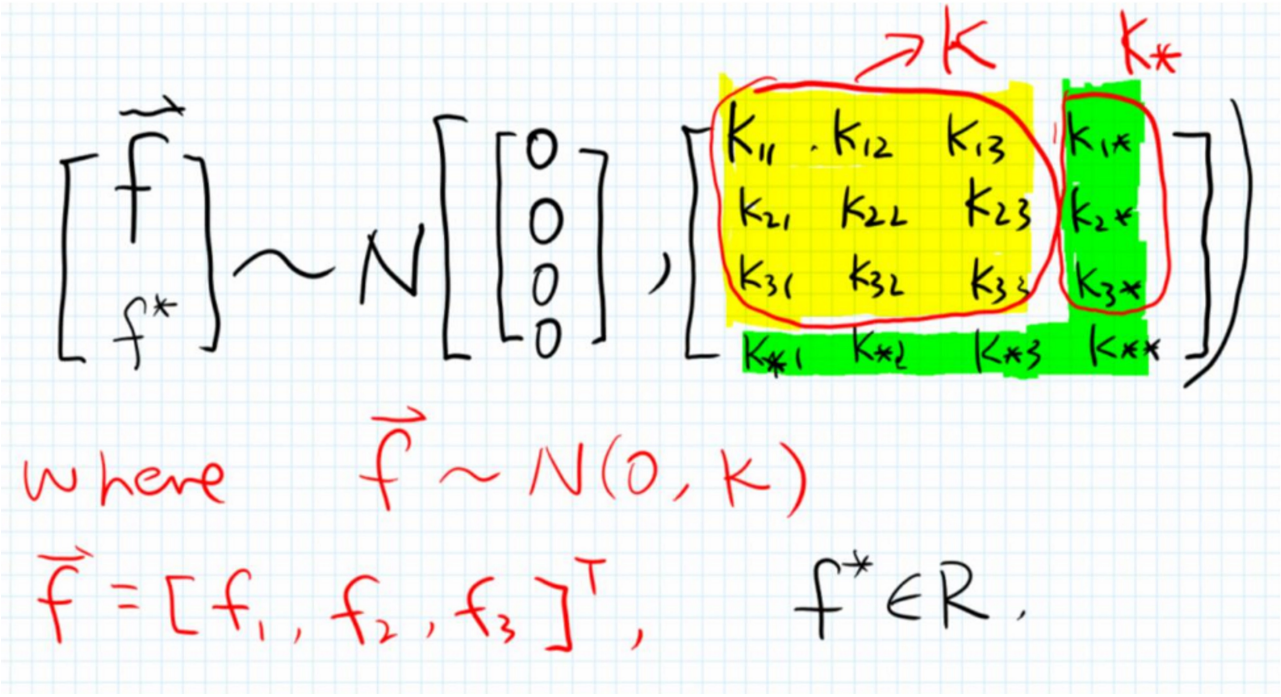


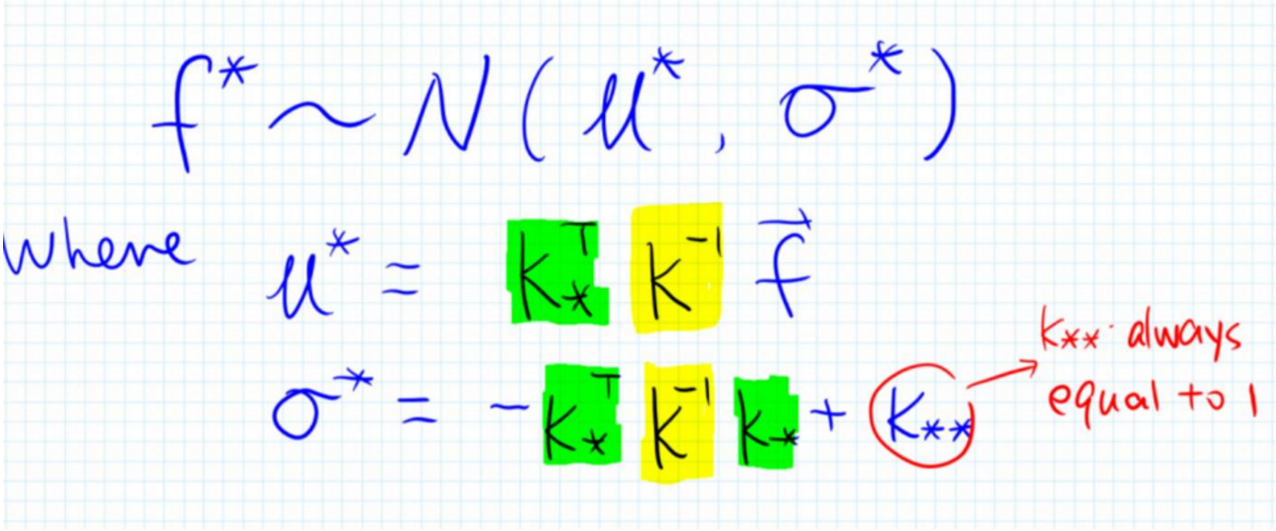
- 一维 $x \sim N(\mu,\sigma^2)=\mu+\sigma N(0,1)$
- 多维 $x \sim N(\mu,\Sigma)=\mu+LN(0,1)$ 在多维空间里采样。例如二维则在一个面上采样，通过一个钟形蛋糕上投影到另一个面上。

2. 高斯过程回归

预测：







多元高斯分布的条件分布

$$\mathbf{X1} = \begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ X_R \end{pmatrix} \quad \mathbf{X2} = \begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_S \end{pmatrix}$$

和

$$\mathbf{X} = \begin{pmatrix} \mathbf{X_1} \\ \mathbf{X_2} \end{pmatrix}$$

核方法是一种生成映射函数的方法。一个和核函数k对应着至少一种映射函数Φ。

核函数满足形式：k(x,x')=Φ(x)TΦ(x')
这个形式是天生的，之所以引入核函数，就是为了构造对偶问题。（对偶问题可以将参数模型化为非参数模型。）

推导：
依据Cholesky Decomposition定理：
若$A \in R^{n\times n}$,则存在一个对角元为正数的下三角矩阵L $\in R^{n\times n}$使A=LL^{T}成立。
分解条件：1）矩阵对称 2）正定
*半正定也可分解，但结果不唯一
结论：
任何一个对称半正定函数都可以作为核函数。核函数是对称半正定矩阵。

Chapter 9 演化学习

2018年10月18日 14:45

基于采样的优化方法

推理法：基于观测数据推测出模型参数优化值。

采样法：随机生成新解并优化。好处：不受模型复杂度限制。

演化学习

对空间的随机搜索。

GA：模型参数所有取值作为解空间，搜索最优点。

GP：所有可能的程序中寻找最优程序。

群体学习和随机优化

群体学习：趋同

随机优化：变异

遗传算法GA

算法框架

种群初始化

遗传算法概念	目标任务概念
种群G_0	所有解法集合
基因（个体）	一个具体解法
基因编码	

模型参数、路径

初始化：根据随机/根据先验知识

编码方式：二值串，每位代表一个基因位；浮点值串；离散值串……

个体选择

适应函数：目标函数

通过适应函数选出好的个体，得到中间种群G_0'

种群繁衍

交叉繁衍：随机选两个个体交换基因片段

个体变异：随机选一个个体改变某些基因位

得到新种群G1（不包括父母辈成员）

算法细节

编码方式

二进制：交换操作 - 按位操作/其他二进制操作；变异操作 - 基于伯努利分布的按位随机采样

浮点数串：变异操作需要基于连续分布（如高斯）

任意数据结构的指针编码：操作需要设计

对操作编码：GP

个体选择策略

选择概率正比于适应函数 （余量随机采样）

适应函数应形式简单

繁衍策略

交叉：单点交叉；两点交叉（交换两点间片段）

变异：二进制 - 每个基因位独立随机替换；浮点值 - 加入高斯噪声

控制随机性：随机性过强 - 收敛速度低；随机性过弱 - 陷入局部最优

其他策略：多父本、精英策略等

结束条件

GA收敛性不能保证。

结束条件包括：迭代次数、个体达到满意、没有更好个体、运行时间/资源消耗等

参数调节

种群大小

个体选择的概率形式

交叉和变异的随机性

迭代次数

进化理论

简单情况下可以理论证明GA的有效性。（演化真的能实现优化？）

例如：Holland的超平面随机采样(Hyperplane Sampling)。

遗传编程GP

算法基础

基因编码：语法树编码

初始化

个体选择：适应函数/适应性排序

交叉与变异

GP高级话题

其他演化学习方法

群体学习方法

蚁群优化算法

人工蜂群算法

粒子群算法

捕猎者搜索

萤火虫算法

其他随机优化方法

模拟退火算法

杜鹃搜索

和声搜索

紧急搜索

Chapter 10
强化学习

刘逸博

2018/10/17

1

本章概览

- 概念
- 基本元素
- DP
- MC
- TD
- 模型学习
- 函数近似
- 深度强化学习

2

10.1 强化学习

1. 什么是强化学习

- 通过和环境交互，利用反馈信息学习。
- 观察状态 - 选择动作 - 获得奖励 ---- 选择策略（优化对象），总体收益最大化（目标函数Q或V）

2. 与其他学习的区别

- 目标驱动（奖励；对样本来说是弱标记，对结果而言是强的）
- 主动学习（和环境交互产生样本。探索-利用问题。）
- 与监督学习、非监督学习并列/更通用（交互性，时序性）

3. 真正的人工智能？优点

- 有针对性，标注成本低。
- 只关注结果。适用于过程复杂的时序任务。
- 主动探索
- 不需要环境的随机性、动态性建模
- 在线学习

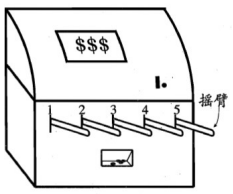
- **4. 应用**

机器人，商业，金融，媒体，医疗，人机对战

3

探索-利用困境

1. **模型：k-摇臂赌博机**



2. 假设

- 操作：每次摇一个摇杆
- 环境：每个摇杆每次被摇有一个概率会吐币
- 目的：最大化单步奖赏（简化问题，一般只知道多步的奖赏）
- 策略：估计奖赏的期望，选择奖赏最大的摇杆

3. 困境

1）仅探索：为求每个摇臂的期望。策略：平均按下所有摇臂。

2）仅利用：获得最大奖赏。策略：按下目前最优摇臂。

两个单独使用都不能使积累奖赏最大化。二者矛盾，互相削弱。

4. 解决：ε-贪心

4

10.2 基本元素

1. 三元素

- 状态 动作 收益
- 策略 $\pi(a|s) = P[A_t = a|S_t = s]$

（在s状态下选a，时不变）

2. 长期收益Gt（对应一个时刻）

- 加和收益：多轮任务
- 折扣收益：连续任务
- 平均收益

3. 值函数与策略优化

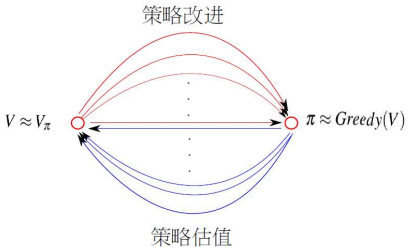
- 动作值函数 Q，状态值函数 V，二者关系（全概率展开）
- 最优策略：策略的偏序关系，优化值函数
- 策略优化：贪心（升级版：ε-贪心）

4. 通用策略迭代 GPI

5. 算法分类

5

4. 通用策略迭代 GPI



- 两个步骤交替迭代
- **策略估值：**计算值函数V或Q

$V_\pi(s) = \mathbb{E}_\pi[G_t|S_t = s]$　　$Q_\pi(s,a) = \mathbb{E}_\pi[G_t|S_t = s, A_t = a]$

- **策略改进：**贪心

$$\pi'(a|s) = \begin{cases} 1 & \text{if } a = \arg\max_{a \in \mathcal{A}} Q_\pi(s,a) \\ 0 & otherwise \end{cases}$$

当前动作收益　后续所有收益

$$\pi'(a|s) = \begin{cases} 1 & \text{if } a = \arg\max_{a \in \mathcal{A}} \pi(a|s)P(r|s,a) + \sum_{s'} P(s'|s,a)V_\pi(s') \\ 0 & otherwise \end{cases}$$

6

5. 算法分类

- 控制任务
- 规划任务
- 学习任务

学习任务类别			算法
环境模型学习	最间接	模型方法（例如可以画成图）	DP
值函数学习	其次	无模型方法-基于采样	MC TD
策略学习	最直接	（也没有模型）	函数近似
混合学习			Dyna

7

DP概览

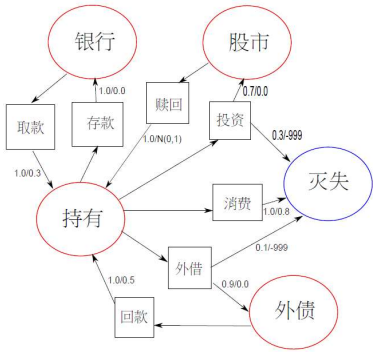
- 假设模型：
  MDP马尔可夫决策过程
- 值函数计算方法（由定义加上MDP得到）
- 策略估值：
  迭代（回溯） 近似求解值函数
- 策略优化（包括策略估值、策略改进）：
  策略迭代、值迭代

8

10.3 全回溯与动态规划算法

1.假设

- 已知模型。环境有马尔可夫性，交互过程是马尔可夫决策过程（可表示为有限自动机）。
- 目标：学习值函数。



2. 值函数

$$V=\sum_a \pi(s,a)\sum_r rP(r|s,a)+\gamma\sum_a \pi(s,a)\sum_{s'} P(s'|s,a)V_\pi(s').$$

$$Q=\sum_r rP(r|s,a)+\gamma\sum_{s'} P(s'|s,a)\sum_{a'} \pi(s',a')Q_\pi(s',a')$$

- 贝尔曼公式。理论上可以解出给定策略pi之下每个状态/状态-策略对的V/Q值，再求期望可求出V/Q。
- 实际上不可行。利用其递归性质用迭代法近似求解。

9

3. 策略估值

$$V(s)\leftarrow\sum_a \pi(s,a)\sum_r rP(r|s,a)+\gamma\sum_a \pi(s,a)\sum_{s'} P(s'|s,a)V_\pi(s').$$

$$Q(s,a)\leftarrow\sum_r rP(r|s,a)+\gamma\sum_{s'} P(s'|s,a)\sum_{a'} \pi(s',a')Q_\pi(s',a').$$

- **关于回溯：**由下一个状态计算当前状态

一步回溯：一次迭代中只回溯一步 DP

多步回溯：MC

深度

全回溯：考虑所有状态和动作

广度

采样回溯：只考虑采样的状态和动作

- **为什么叫“动态规划”**

10

4.策略优化

1）策略迭代

① 初始化

② 【策略估值】迭代多次直至收敛：批更新所有状态的V(s)

③ 【策略改进】一次：批更新所有状态的Π(s) (V(s)->Π(s))

④ 检查Π是否满意了，若没有则重复2、3。

2）值迭代

改进：策略估值迭代次数减为1次。

公式：将策略改进和策略估值写在一个式子里了。

好处：效率高。

```
Initialize V(s), π(s,a)  ∀s,a;
while True do
    Policy Evaluation:
    while True do
        Δ = 0;
        for s ∈ S do
            v = V(s);
            V(s) = Σ_{s',r} P(s',r|s,π(s))[r + γV(s')];
            Δ = max(Δ, |v − V(s)|);
        end
        if Δ < δ then
            Break while;
        end
    end
    Policy Improvement:
    Conv = True;
    for s ∈ S do
        Act = π(s);
        π(s) = arg max_a Σ_{s',r} P(s',r|s,a)[r + γV(s')];
        if Act ≠ π(s) then
            Conv = False;
        end
    end
    if Conv=True then
        Break while;
    end
end
```

11

- **策略估值的迭代：可同步/异步更新。**

异步：一轮迭代中，更新一个V(s)后立即生效，其他状态立即可用更新后的V，不需要等到下一轮迭代才用。

异步更新状态的顺序有多种选法。

12

MC概览

- 假设：
  环境未知，通过采样学习值函数
- 采样
- 策略估值：
  - 样本的平均奖赏近似V/Q的期望
  - ε-贪心增加采样覆盖度
- 策略优化：
  同策略(On-policy)、异策略(Off-policy)
  （区别是策略估计和策略改进是否使用同分布的采样）

13

10.4 蒙特卡罗方法

1. 假设：环境未知，通过采样学习值函数

采样：是一种通用方法，用于模型未知/复杂

好处：不用考虑所有状态，只考虑重要的。

坏处：采样有偏差，导致偏离最优策略

DP：每个状态单独估值。（由模型结构（链的长度）保证的准确性。）

MC：（由采样点数量（数据的广度）保证的准确性）

① 从一个状态开始，用某种策略采样（每个采样点是[(St, At), Rt+1]），执行T步（采样T次），获得一个轨迹（一个完整交互过程）

② 多次采样获得多条轨迹

③ 策略估值：对一个状态-动作对，将多次得到的R进行平均。得到Q。

14

2. 策略估值

- 目的：求一个状态-动作对多次采样得到的值函数平均。
- 方法：增量法

状态值函数：$$V(S_t) \leftarrow V(S_t) + \frac{1}{N(S_t)}(G_t - V(S_t)),$$

动作值函数：$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{\rho_t^T}{C(s)}(G_t - Q(S_t, A_t))$$

有修正因子

15

3. 策略优化

- **值函数选择**：环境模型未知，选动作值函数Q。
- **增加采样覆盖度**（防止每次最优都选的一样，多次采样的序列相同，采不到足够样本）

方法：ε-贪心

ε-贪心有两种描述方式：一种策略角度，一种动作角度。

$\Pi^{\varepsilon}(x)=$ Π(x) (贪心)，p=1−ε；A(动作集)中均匀选取，p=ε

$$\pi(a|s) = \begin{cases} 1 - \varepsilon + \frac{\varepsilon}{|\mathscr{A}(f)|} & \text{if } a = \arg\max_{a \in \mathscr{A}} Q_\pi(s,a) \\ \frac{\varepsilon}{|\mathscr{A}(f)|} & otherwise \end{cases}$$

可以证明，基于GPI的ε-贪心可以收敛到最优策略。 ε越小，越接近贪心策略。即，逐渐减小ε可以接近全局最优。但是上述收敛性以Q的充分估计为前提，需要大量采样。（这里又体现了探索利用困境。）

16

On-policy同策略

- **概念：**策略估值、策略改进用的是同一个策略Π。

① 初始策略：均匀概率

② 采样多条轨迹，每一条轨迹：

① 用策略Π采样得<S0, A0, R1, S1, A1, R2, … S(T-1), A(T-1), RT>

② 【策略估值】对序列中每个（S， A）计算Q，增量法：

$$Q(S_t,A_t) \leftarrow Q(S_t,A_t) + \frac{\rho_t^T}{C(s)}(G_t - Q(S_t,A_t))$$

③ 【策略改进】对所有已见状态，ε-贪心更新策略Π。

- **缺点：**策略改进也使用的ε-贪心，不利于寻找最优解。
- **解决方法：**异策略。

输入： 环境 E;
动作空间 A;
起始状态 x_0;
策略执行步数 T.

过程：

$Q(x,a)=0$, $\text{count}(x,a)=0$, $\pi(x,a)=\frac{1}{|A(x)|}$;
for $s=1,2,\ldots$ **do**
  在 E 中执行策略 π 产生轨迹
  $< x_0, a_0, r_1, x_1, a_1, r_2, \ldots, x_{T-1}, a_{T-1}, r_T, x_T >$;
  for $t=0,1,\ldots,T-1$ **do**
    $R=\frac{1}{T-t}\sum_{i=t+1}^{T} r_i$;
    $Q(x_t,a_t)=\frac{Q(x_t,a_t)\times \text{count}(x_t,a_t)+R}{\text{count}(x_t,a_t)+1}$;
    $\text{count}(x_t,a_t)=\text{count}(x_t,a_t)+1$
  end for
  对所有已见状态 x：

$$\pi(x,a)=\begin{cases}\arg\max_{a'} Q(x,a'), & \text{以概率 } 1-\epsilon;\\ \text{以均匀概率从 } A \text{ 中选取动作}, & \text{以概率 } \epsilon.\end{cases}$$

end for
输出： 策略 π

17

Off-policy异策略

- **概念：**策略估值、策略改进用的不是同一个策略。

例如，策略估值时，采样的策略称为动作策略，使用ε-贪心法（为了提高样本覆盖率）；策略改进时，优化的策略称为目标策略，使用贪心法（为了接近最优解）。

- **问题：**不同策略的每个状态-采样对被采样的概率不同，即不同策略的采样点服从不同的分布。（策略估值是根据动作策略得到的样本进行的估值，无法直接由目标策略得到的样本进行估值（不可见，没被采样出来）。但是希望优化目标策略，就得知道目标策略的值函数。）
- **解决问题的理论基础：重要性采样**(Importance-sampling)：基于分布π的期望可以由基于分布μ的期望得到。

$$\mathbb{E}_\pi f(x) = \mathbb{E}_\mu \frac{\pi(x)}{\mu(x)} f(x).$$

重要性因子

18

- **解决问题的具体过程：**
- 已知动作策略μ、目标策略π，可得交互路径概率比：

$$\rho_t^T(\lambda)=\prod_{k=t}^{T}\frac{\pi(A_k|S_k)}{\mu(A_k|S_k)}$$ （针对一个路径）

- 动作值函数更新公式中引入上述修正因子：

$$Q(S_t,A_t)\leftarrow Q(S_t,A_t)+\frac{\rho_t^T}{C(s)}(G_t-Q(S_t,A_t))$$

- C的更新、W的更新

Output: $\pi(s,a)$

; Initialize $Q(s,a)$; $\pi(s,a)=Greedy(Q)$; $C(s,a)=0$;
for *Each Iteration* **do**
  Sample $S_0,A_0,R_1,\ldots,R_t,S_T$ by μ;
  W = 1;
  G = 0;
  for *t:=T-1 to 0* **do**
    $G\leftarrow R_{t+1}+\gamma G$;
    $C(S_t,A_t)\leftarrow C(S_t,A_t)+W$;
    $Q(S_t,A_t)\leftarrow Q(S_t,A_t)+\frac{W}{C(S_t,A_t)}[G-Q(S_t,A_t)]$;
    $\pi(S_t,a)=\begin{cases}1 & a=\arg\max_a Q(S_t,a)\\ 0 & otherwise\end{cases}$
    if $\pi(S_t,a)\neq 1$ **then**
      Break;
    end
    $W\leftarrow W\frac{1}{\mu(A_t|S_t)}$;
  end
end

19

4. MC搜索

- **启发式搜索：** $f(s,s',\tilde{s})=g(s,s',\tilde{s})+h(\tilde{s})$ h(s)是人为指定的估计值
- **MC搜索** ：V(s)代替h(s)，通过采样估计V(s)

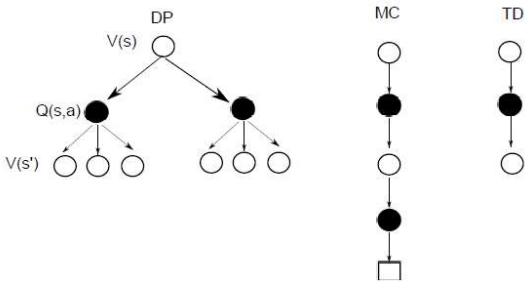
20

TD概览

- 假设
- 思路
  三种方法比较
- 策略估值
- 策略优化
  Sarsa (On-policy)
  Q-learning (Off-policy)
- N-step TD, TD(λ)

21

10.5 时序差分法



1. 假设：MDP

2. 思路

- 和DP相比，全回溯->采样回溯
- 和MC相比，全过程采样->一步采样

3. 策略估值

$$V(S_t) \leftarrow V(S_t) + \alpha(R_{t+1} + \gamma V_\pi(S_{t+1}) - V(S_t))$$

- 一步采样，余下步骤bootstrapping估计
- 长期收益：和DP一样，有Bootstrap性质（利用了其他状态）。

$$G_t = R_{t+1} + \gamma V_\pi(S_{t+1})$$

- 增量法：和MC类似，是基于采样的估计。

Input: π
Output: $V(s)$
Initialize $V(s)$;
for *Each Episode* **do**
  Init S;
  while S *not terminal* **do**
    Sample A by π for S;
    Take A, observe S', R;
    $V(S) \leftarrow V(S) + \alpha[R + \gamma V(S') - V(S)]$;
    S = S';
  end
end

22

4. 策略优化

- 不依赖环境，需要估计Q

$$Q(S_t,A_t) \leftarrow Q(S_t,A_t) + \alpha[R_{t+1} + \gamma Q(S_{t+1},A_{t+1}) - Q(S_t,A_t)]$$

- **Sarsa、Q-learning**

5. TD优点：

- 高效，充分利用了MDP结构。
- MC是批处理的，采样完一个序列才求平均（尽管公式上是增量式的，假的）；而TD真正是增量式。

23

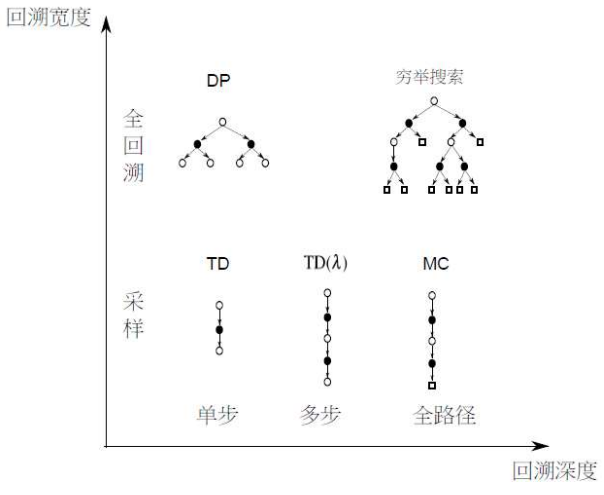
DP MC TD比较

DP与MC、TD：

有/无模型本质：是否采样

TD与MC：

- 采样深度；
- TD依赖MDP假设，有较强结构化先验知识，趋向最大似然估计；MC没有假设趋向最小均方差估计



24

模型学习概览

- 值函数学习与模型学习
- 模型学习方法
- 混合学习：Dyna

25

10.6 模型学习

1. **值函数学习与模型学习**

值函数学习缺点：数据不一定充足，缺少先验知识，容易过拟合

2. 模型学习方法

定义概率模型，通过交互学习参数。得到模型后，通过采样来优化值函数，优化策略。一般假设有限离散MDP，收益高斯。

3. 混合学习Dyna

回溯Q。基于环境采样学习，基于模型采样学习，交替进行。

优点：适合环境复杂、数据量大

26

函数近似概览

- 为什么用函数近似
- 值函数近似
- 基于梯度的参数优化
- 基于函数近似的策略学习
- Actor-Critic方法

27

10.7 函数近似

1. 为什么用函数近似

离散方法局限性。连续函数更接近真实情况。

2. 值函数近似

函数近似：离散模型是精确函数，实际上不精确。函数近似用连续函数拟合离散函数。

可度量性：原本离散的进行连续空间嵌入。好处：不同状态促进；任何新状态都可以被估计；参数数量少，泛化能力提高。

3. 基于梯度的参数优化

4. 基于函数近似的策略学习

5. Actor-Critic方法：值函数近似和策略近似的结合。

28

深度强化学习概览

- Atari
- AlphaGo

29

10.8 深度强化学习

- **函数近似法：**空间映射是深度学习的要义。
- **结构：**深度学习：利用DNN对值函数或策略做近似；
  强化学习：利用强化学习方法更新DNN参数。

Atari游戏

Alpha Go

30


END