

# Contents

<b>1</b>	<b>概述</b>	<b>2</b>
1.1	统计学习三要素	2
1.1.1	模型	2
1.1.1.1	模型的假设空间	2
1.1.1.1.1	定义 1	2
1.1.1.1.2	定义 2	3
1.1.2	策略	3
1.1.2.1	损失函数与风险函数	3
1.1.2.2	经验风险最小化与结构风险最小化	4
1.1.3	算法	4
1.2	模型评估与模型选择	4
1.2.1	训练误差与测试误差	4
1.2.2	过拟合与模型选择	5
1.3	正则化与交叉验证	5
1.3.1	正则化	5
1.3.2	交叉验证	5
1.4	泛化能力	5
1.4.1	泛化误差	5
1.4.2	泛化误差上界	5
1.5	生成模型与判别模型	5
1.6	分类问题	5
1.7	标注问题	5
1.8	回归问题	5
<b>2</b>	<b>感知机</b>	<b>6</b>
<b>3</b>	<b>k 近邻法</b>	<b>6</b>
<b>4</b>	<b>朴素贝叶斯法</b>	<b>6</b>
4.1	贝叶斯公式	6
4.2	极大似然估计 (MLE)	6
4.3	最大后验估计 (MAP)	7
4.4	贝叶斯估计	7
<b>5</b>	<b>决策树</b>	<b>7</b>
<b>6</b>	<b>logistic 回归与最大熵模型</b>	<b>7</b>
<b>7</b>	<b>支持向量机</b>	<b>7</b>
<b>8</b>	<b>提升方法</b>	<b>7</b>
8.1	gbdt	7
8.1.1	gbdt 概述	7
8.1.2	gbdt 的负梯度拟合	8
<b>9</b>	<b>EM 算法及其推广</b>	<b>8</b>
<b>10</b>	<b>隐马尔可夫模型</b>	<b>8</b>
<b>11</b>	<b>条件随机场</b>	<b>8</b>
<b>12</b>	<b>附录</b>	<b>8</b>
12.1	矩阵	8
12.2	优化	8

12.2.1	拉格朗日乘子法 . . . . .	8
12.2.1.1	等式约束 . . . . .	8
12.2.1.2	不等式约束 . . . . .	9
12.2.1.3	带等式和不等式约束的拉格朗日乘子法 . . . . .	10
12.2.2	梯度下降 . . . . .	11
12.2.2.1	《统计学习方法》的视角 . . . . .	11
12.2.2.2	《机器学习》的视角 . . . . .	12
12.2.3	牛顿法 . . . . .	12
12.2.3.1	二阶导基本性质 . . . . .	12
12.2.3.2	牛顿法 . . . . .	13
12.2.4	拟牛顿法的思路 . . . . .	13
12.2.5	DFP(Davidon-Fletcher-Powell) . . . . .	14
12.2.6	BFGS(Broydon-Fletcher-Goldfarb-Shanno) . . . . .	14
12.3	拉格朗日对偶性 . . . . .	14
12.4	信息论相关 . . . . .	14
12.4.1	凸集 . . . . .	14
12.4.2	凸函数 . . . . .	14
12.4.3	KL 散度 . . . . .	14
12.5	采样 . . . . .	16
12.5.1	拒绝采样 . . . . .	16
12.5.2	重要性采样 . . . . .	16
12.5.3	马尔可夫蒙特卡洛采样 (MCMC) . . . . .	17
12.5.3.1	Metropolis-Hastings 采样 . . . . .	17
12.5.3.2	吉布斯采样 . . . . .	17

下载地址: <https://github.com/daiwk/collections/blob/master/pdfs/int-ml.pdf>

本文参考自李航的《统计学习方法》、周志华的《机器学习》、Hulu 的《百面机器学习》等。

## 1 概述

### 1.1 统计学习三要素

#### 1.1.1 模型

监督学习中,模型是要学习的条件概率分布或决策函数。

##### 1.1.1.1 模型的假设空间

假设空间是所有可能的条件概率分布或决策函数

###### 1.1.1.1.1 定义 1

可以定义为决策函数的集合:

$$\mathcal{F} = \{f|Y = f(X)\}$$

- $X$  和  $Y$  是定义在  $\mathcal{X}$  和  $\mathcal{Y}$  上的变量
- $\mathcal{F}$  是一个参数向量决定的函数族:

$$\mathcal{F} = \{f|Y = f_{\theta}(X), \theta \in R^n\}$$

参数向量  $\theta$  取值于  $n$  维欧式空间  $R^n$ , 称为参数空间

### 1.1.1.1.2 定义 2

也可以定义为条件概率的集合：

$$\mathcal{F} = \{P|P(Y|X)\}$$

- $X$  和  $Y$  是定义在  $\mathcal{X}$  和  $\mathcal{Y}$  上的随机变量
- $\mathcal{F}$  是一个参数向量决定的条件概率分布族：

$$\mathcal{F} = \{P|P_{\theta}(Y|X), \theta \in R^n\}$$

## 1.1.2 策略

### 1.1.2.1 损失函数与风险函数

**损失函数 (loss function)** 或 **代价函数 (cost function)**：度量预测值  $f(X)$  与真实值  $Y$  的误差程度，记为  $L(Y, f(X))$ ，是个非负实值函数。损失函数越小，模型越好。

- 0-1 损失函数：

$$L(Y, f(X)) = \begin{cases} 0 & Y \neq f(X) \\ 1 & Y = f(X) \end{cases}$$

- 平方损失函数：

$$L(Y, f(X)) = (Y - f(X))^2$$

- 绝对损失函数：

$$L(Y, f(x)) = |Y - f(X)|$$

- 对数损失函数 (logarithmic loss function)/对数似然损失函数 (log-likelihood loss function)：

$$L(Y, P(Y|X)) = -\log P(Y|X)$$

**风险函数 (risk function)** 或 **期望损失 (expected loss)**： $X$  和  $Y$  服从联合分布  $P(X, Y)$ ，理论上模型  $f(X)$  关于联合分布  $P(X, Y)$  的平均意义下的损失：

$$R_{exp}(f) = E_P[L(Y, f(X))] = \int_{\mathcal{X} \times \mathcal{Y}} L(y, f(x)) P(x, y) dx dy$$

学习的目标：选择期望风险最小的模型。但联合分布  $P(X, Y)$  是未知的，所以无法直接计算  $R_{exp}(f)$ 。所以监督学习是病态问题 (ill-formed problem)：一方面需要联合分布，另一方面联合分布是未知的。

给定训练集：

$$T = \{(x_1, y_1), \dots, (x_N, y_N)\}$$

**经验风险 (empirical risk)/经验损失 (empirical loss)**：模型  $f(X)$  关于训练集的平均损失

$$R_{emp}(f) = \frac{1}{N} \sum_{i=1}^N L(y_i, f(x_i))$$

根据**大数定律**，当样本容量  $N$  趋向无穷时，经验风险  $R_{emp}$  趋于期望风险  $R_{exp}(f)$ 。

### 1.1.2.2 经验风险最小化与结构风险最小化

经验风险最小化 (empirical risk minimization, ERM): 经验风险最小的模型就是最优模型。所以需要求解的最优化问题是:

$$\min_{f \in \mathcal{F}} R_{erm} = \min_{f \in \mathcal{F}} \frac{1}{N} L(y_i, f(x_i))$$

当满足以下两个条件时, 经验风险最小化就等价于极大似然估计 (maximum likelihood estimation):

- 模型是条件概率分布
- 损失函数是对数损失函数

当样本量足够大时, ERM 能有很好的效果, 但样本量不够多时, 为了防止过拟合, 需要用下面的方法。

结构风险最小化 (structural risk minimization, SRM): 结构风险 = 经验风险 + 表示模型复杂度的正则化项 (regularizer) 或罚项 (penalty term)。结构风险定义如下:

$$R_{srm}(f) = \frac{1}{N} \sum_{i=1}^N L(y_i, f(x_i)) + \lambda J(f)$$

其中,  $J(f)$  是模型的复杂度, 模型越复杂,  $J(f)$  越大。  $\lambda \geq 0$  是用于权衡经验风险和模型复杂度的系数。

当满足以下 3 个条件时, 结构化风险最小化等价于) 贝叶斯估计中的最大后验概率估计 (maximum posterior probability estimation, MAP):

- 模型是条件概率分布
- 损失函数是对数损失函数, 对应后验估计中的似然函数
- 模型复杂度由模型的先验概率表示

似然函数和先验概率的乘积即对应贝叶斯最大后验估计的形式

参考<https://www.zhihu.com/question/23536142>

所以结构风险最小化就是求解优化问题:

$$\min_{f \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^N L(y_i, f(x_i)) + \lambda J(f)$$

### 1.1.3 算法

算法指的是学习模型的具体方法, 即使用什么计算方法求解最优模型。

因为统计学习问题归结为最优化问题, 所以统计学习的算法就是求解最优化问题的算法。

- 如果有显式的解析解, 此最优化问题就比较简单
- 如果没有, 需要用数值计算方法求解, 需要考虑如何保证找到全局最优解, 并使求解过程高效

## 1.2 模型评估与模型选择

### 1.2.1 训练误差与测试误差

假设学习到的模型是  $Y = \hat{f}(X)$ , 训练误差是模型  $Y = \hat{f}(X)$  关于训练数据集的平均损失 ( $N$  是样本容量):

$$R_{emp}(\hat{f}) = \frac{1}{N} \sum_{i=1}^N L(y_i, \hat{f}(x_i))$$

测试误差是模型  $Y = \hat{f}(X)$  关于测试数据集的平均损失 ( $N'$  是测试样本容量):

$$e_{test}(\hat{f}) = \frac{1}{N'} \sum_{i=1}^{N'} L(y_i, \hat{f}(x_i))$$

- 训练误差的大小，对判断给定的问题是不是一个容易学习的问题是有意义的
- 测试误差反映了学习方法对未知数据的预测能力，即泛化能力（generalization ability）

### 1.2.2 过拟合与模型选择

当模型复杂度增加时，训练误差会逐渐减小并趋向于 0；测试误差会先减小，达到最小值后又会增大。

当模型复杂度过大时，就会出现过拟合。所以需要在学習时防止过拟合，选择复杂度适当的模型。

## 1.3 正则化与交叉验证

### 1.3.1 正则化

模型选择的典型方法是正则化（regularization）。正则化是结构风险最小化策略的实现，即在经验上加一个正则化项（regularizer）或罚项（penalty term）。

正则化项一般是模型复杂度的单调递增函数，模型越复杂，正则化值越大。比如，正则化项可以是模型参数向量的范数。

$$\min_{f \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^N L(y_i, f(x_i)) + \lambda J(f)$$

其中  $\lambda \geq 0$

正则化符合奥卡姆剃刀（Occam's razor）原理：在所有可能选择的模型中，能够很好地解释已知数据并且十分简单才是最好的模型，也就是应该选择的模型。

从贝叶斯估计的角度来看，正则化项对应于模型的先验概率。可以假设复杂的模型有较小的先验概率，简的模型有较大的先验概率。

### 1.3.2 交叉验证

## 1.4 泛化能力

### 1.4.1 泛化误差

### 1.4.2 泛化误差上界

## 1.5 生成模型与判别模型

## 1.6 分类问题

## 1.7 标注问题

## 1.8 回归问题

一个常问的问题：平方损失在做分类问题的损失函数时，有什么缺陷？

一方面，直观上看，平方损失函数对每一个输出结果都十分看重，而交叉熵只看重正确分类的结果。例如三分类问题，如果预测的是  $(a, b, c)$ ，而实际结果是  $(1, 0, 0)$ ，那么：

$$mse = (a - 1)^2 + b^2 + c^2$$

$$crossentropy = -1 \times \log a - 0 \times \log b - 0 \times \log c = -\log a$$

所以，交叉熵损失的梯度只和正确分类有关。而平方损失的梯度和错误分类有关，除了让正确分类尽可能变大，还会让错误分类都变得更平均。实际中后面这个调整在分类问题中是不必要的，而回归问题上这就很重要。

另一方面，从理论角度分析。两个损失的源头不同。平方损失函数假设最终结果都服从高斯分布，而高斯分布的随机变量实际是一个连续变量，而非离散变量。如果假设结果变量服从均值  $t$ ，方差  $\sigma$  的高斯分布，那么利用最大似然法可以优化其负对数似然，最终公式变为：

$$\max \sum_i^N \left[ -\frac{1}{2} \log(2\pi\sigma^2) - \frac{(t_i - y)^2}{2\sigma^2} \right]$$

除去与  $y$  无关的项，剩下的就是平方损失函数。

## 2 感知机

## 3 k 近邻法

## 4 朴素贝叶斯法

参考<https://www.cnblogs.com/jiangxinyang/p/9378535.html>

### 4.1 贝叶斯公式

$$p(\theta|X) = \frac{p(X|\theta)p(\theta)}{p(X)}$$

又可以写为：

$$posterior = \frac{likelihood * prior}{evidence}$$

其中：

- *posterior*: 通过样本  $X$  得到参数  $\theta$  的概率，即后验概率
- *likelihood*: 通过参数  $\theta$  得到样本  $X$  的概率，即似然函数。
- *prior*: 参数  $\theta$  的先验概率
- *evidence*:  $p(X) = \int p(X|\theta)p(\theta)d\theta$ 。样本  $X$  发生的概率，是各种  $\theta$  条件下发生的概率的积分。

### 4.2 极大似然估计 (MLE)

极大似然估计的核心思想是：认为当前发生的事件是概率最大的事件。因此就可以给定的数据集，使得该数据集发生的概率最大来求得模型中的参数。

似然函数如下：

$$p(X|\theta) = \prod_{i=1}^n p(x_i|\theta)$$

为便于计算，对似然函数两边取对数，得到对数似然函数：

$$\begin{aligned} LL(\theta) &= \log p(X|\theta) \\ &= \log \prod_{i=1}^n p(x_i|\theta) \\ &= \sum_{i=1}^n \log p(x_i|\theta) \end{aligned}$$

极大似然估计只关注当前的样本，也就是只关注当前发生的事情，不考虑事情的先验情况。由于计算简单，而且不需要关注先验知识，因此在机器学习中的应用非常广，最常见的就是逻辑回归。

### 4.3 最大后验估计 (MAP)

最大后验估计中引入了先验概率（先验分布属于贝叶斯学派引入的，像 L1, L2 正则化就是对参数引入了拉普拉斯先验分布和高斯先验分布），而且最大后验估计要求的是  $p(\theta|X)$

$$\begin{aligned} f(x) &= \arg \max_{\theta} p(\theta|X) \\ &= \arg \max_{\theta} \frac{p(X|\theta)p(\theta)}{p(X)} \\ &= \arg \max_{\theta} p(X|\theta)p(\theta) \end{aligned}$$

其中因为分母  $p(X)$  与  $\theta$  无关，所以可以去掉，同样地，取  $\log$ ：

$$\begin{aligned} f(x) &= \arg \max_{\theta} \log p(X|\theta)p(\theta) \\ &= \arg \max_{\theta} \left\{ \sum_{i=1}^n \log p(x_i|\theta) + \log p(\theta) \right\} \end{aligned}$$

最大后验估计不只是关注当前的样本的情况，还关注已经发生过的先验知识。

最大后验估计和最大似然估计的区别：

最大后验估计允许我们把先验知识加入到估计模型中，这在样本很少的时候是很有用的（因此朴素贝叶斯在较少的样本下就能有很好的表现），因为样本很少的时候我们的观测结果很可能出现偏差，此时先验知识会把估计的结果“拉”向先验，实际的预估结果将会在先验结果的两侧形成一个顶峰。通过调节先验分布的参数，比如 beta 分布的  $\alpha, \beta$ ，我们还可以调节把估计的结果“拉”向先验的幅度， $\alpha, \beta$  越大，这个顶峰越尖锐。这样的参数，我们叫做预估模型的“超参数”。

### 4.4 贝叶斯估计

贝叶斯估计和极大后验估计有点相似，都是以最大化后验概率为目的。区别在于：

- 极大似然估计和极大后验估计都是只返回了的预估值。
- 极大后验估计在计算后验概率的时候，把分母  $p(X)$  忽略了，在进行贝叶斯估计的时候则不能忽略
- 贝叶斯估计要计算整个后验概率的概率分布

## 5 决策树

## 6 logistic 回归与最大熵模型

## 7 支持向量机

## 8 提升方法

### 8.1 gbd

参考<https://www.cnblogs.com/pinard/p/6140514.html>

梯度提升树 (Gradient Boosting Decision Tree, 以下简称 GBDT) 有很多简称，有 GBT (Gradient Boosting Tree), GTB (Gradient Tree Boosting), GBRT (Gradient Boosting Regression Tree), MART (Multiple Additive Regression Tree) 等等。

#### 8.1.1 gbd 概述

在 Adaboost 中，我们是利用前一轮迭代弱学习器的误差率来更新训练集的权重，这样一轮轮的迭代下去。

GBDT 也是迭代，但是弱学习器限定了只能使用 CART 回归树模型，同时迭代思路和 Adaboost 也有所不同。

假设我们前一轮迭代得到的强学习器是  $f_{t-1}(x)$ ，损失函数是  $L(y, f_{t-1}(x))$ ，本轮迭代的目标是找到一个 CART 回归树模型的弱学习器  $h_t(x)$ ，使得本轮的损失函数  $L(y, f_t(x)) = L(y, f_{t-1}(x) + h_t(x))$  最小。也就是说，要找到一个决策树，使得样本的损失  $L(y - f_{t-1}(x), h_t(x))$  最小。

### 8.1.2 gbdt 的负梯度拟合

第  $t$  轮的第  $i$  个样本的损失函数的负梯度：

$$r_{ti} = - \left[ \frac{\partial L(y_i, f(x_i))}{\partial f(x_i)} \right]_{f(x)=f_{t-1}(x)}$$

利用

## 9 EM 算法及其推广

## 10 隐马尔可夫模型

## 11 条件随机场

## 12 附录

### 12.1 矩阵

### 12.2 优化

#### 12.2.1 拉格朗日乘子法

拉格朗日乘子法 (Lagrange multipliers) 是一种寻找多元函数在一组约束下的极值的方法。通过引入拉格朗日乘子，将  $d$  个变量和  $k$  个约束条件的最优化问题转化为具有  $d + k$  个变量的无约束优化问题求解。

##### 12.2.1.1 等式约束

假设  $x$  是  $d$  维向量，要寻找  $x$  的某个取值  $x^*$ ，使目标函数  $f(x)$  最小且同时满足  $g(x) = 0$  的约束。

从几何角度看，目标是在由方程  $g(x) = 0$  确定的  $d - 1$  维曲面上，寻找能使目标函数  $f(x)$  最小化的点。

1. 对于约束曲面  $g(x) = 0$  上的任意点  $x$ ，该点的梯度  $\nabla g(x)$  正交于约束曲面
2. 在最优点  $x^*$ ，目标函数  $f(x)$  在该点的梯度  $\nabla f(x^*)$  正交于约束曲面

对于第 1 条，梯度本身就与曲面的切向量垂直，是曲面的法向量，并且指向数值更高的等值线。

证明：

参考<http://xuxzmail.blog.163.com/blog/static/251319162010328103227654/>

假设  $z = f(x, y)$  的等值线：  $\Gamma : f(x, y) = c$ ，两边求微分：

$$df(x, y) = dc$$

$$\frac{\partial f}{\partial x} dx + \frac{\partial f}{\partial y} dy = 0$$

看成两个向量的内积：

$$\frac{\partial f}{\partial x} dx + \frac{\partial f}{\partial y} dy = \left\{ \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right\} \cdot \{dx, dy\} = 0$$



而内积  $a \cdot b = |a||b|\cos\theta$  为 0 说明夹角是 90 度，而  $\{\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\}$  是梯度向量， $\{dx, dy\}$  是等值线的切向量，所以梯度向量和切向量是垂直的。

对于第 2 条，可以用反证法，如下图，蓝色是  $g(x) = 0$ ，橙色是  $f(x)$  的等值线（图里假设  $f(x) = x^2 + y^2$ ），交点的  $\nabla f(x^*)$  的梯度和  $g(x)$  的切面不垂直，那么，可以找到更小的等值线，使夹角更接近 90 度，也就是说，这个点不是真正的最优点  $x^*$ 。



所以，在最优点  $x^*$  处，梯度  $\nabla g(x)$  和  $\nabla f(x)$  的方向必然相同或相反，也就是存在  $\lambda \neq 0$ ，使得：

$$\nabla f(x^*) + \lambda \nabla g(x^*) = 0$$

其中  $\lambda$  是拉格朗日乘子，定义拉格朗日函数

$$L(x, \lambda) = f(x) + \lambda g(x)$$

其中  $L(x, \lambda)$  对  $x$  的偏导  $\nabla_x L(x, \lambda)$  置 0，就得到：

$$\nabla f(x) + \lambda \nabla g(x) = 0$$

而  $L(x, \lambda)$  对  $\lambda$  的偏导  $\nabla_\lambda L(x, \lambda)$  置 0，就得到

$$g(x) = 0$$

所以，原约束问题可以转化为对  $L(x, \lambda)$  的无约束优化问题。

#### 12.2.1.2 不等式约束

考虑不等式约束  $g(x) \leq 0$ ，最优点或者在边界  $g(x) = 0$  上，或者在区域  $g(x) < 0$  中。



- 对于  $g(x) < 0$

相当于使  $f(x)$  取得最小值的点落在可行域内，所以约束条件相当于没有用，所以，直接对  $f(x)$  求极小值即可。因为  $L(x, \lambda) = f(x) + \lambda g(x)$ ，所以

$$\nabla_x L(x, \lambda) = \nabla f(x) + \lambda \nabla g(x)$$

因为  $g(x) < 0$ ，想要只让  $\nabla f(x) = 0$ ，那么令  $\lambda = 0$  即可。

- 对于  $g(x) = 0$

这就变成了等式约束，且此时  $\nabla f(x^*)$  和  $\nabla g(x^*)$  反向相反。因为在  $g(x) = 0$  越往里值是越小的，而梯度是指向等值线高的方向，所以梯度是指向外的。而  $f(x)$  的可行域又在  $g(x)$  的里面和边界上，我们要找的是  $f(x)$  的最小值，所以  $f(x)$  的梯度是指向内部的。

而  $\nabla f(x) + \lambda \nabla g(x) = 0$ ，两个又是反向的，所以  $\lambda > 0$ 。

结合  $g(x) \leq 0$  和  $g(x) = 0$  两种情况的结论，就得到了 KKT (Karush-Kuhn-Tucker) 条件

$$\left. \begin{array}{l} g(x) = 0, \lambda > 0 \\ g(x) < 0, \lambda = 0 \end{array} \right\} \Rightarrow \begin{cases} g(x) \leq 0 \\ \lambda \geq 0 \\ \lambda g(x) = 0 \end{cases}$$

其中  $\lambda g(x) = 0$  是因为  $\lambda$  和  $g(x)$  至少一个是 0，而且不能都不是 0。

以上三个条件有各自的名字：

- Primal feasibility(原始可行性):  $g(x) \leq 0$
- Dual feasibility(对偶可行性):  $\lambda \geq 0$
- Complementary slackness:  $\lambda g(x) = 0$

### 12.2.1.3 带等式和不等式约束的拉格朗日乘子法

对于多个约束的情形， $m$  个等式约束和  $n$  个不等式约束，可行域  $\mathbb{D} \subset \mathbb{R}^d$  非空的优化问题：

$$\min_x f(x)$$

$$\begin{aligned} s.t \quad & h_i(x) = 0, i = 1, \dots, m \\ & g_j(x) \leq 0, j = 1, \dots, n \end{aligned}$$

引入拉格朗日乘子  $\lambda = (\lambda_1, \lambda_2, \dots, \lambda_m)^T$  和  $\mu = (\mu_1, \mu_2, \dots, \mu_n)^T$ ，相应的拉格朗日函数为：

$$L(x, \lambda, \mu) = f(x) + \sum_{i=1}^m \lambda_i h_i(x) + \sum_{j=1}^n \mu_j g_j(x)$$

由不等式约束引入的 KKT 条件 ( $j = 1, 2, \dots, n$ ) 为

$$\begin{cases} g_j(x) \leq 0 \\ \mu_j \geq 0 \\ \mu_j g_j(x) = 0 \end{cases}$$

## 12.2.2 梯度下降

### 12.2.2.1 《统计学习方法》的视角

假设  $f(x)$  有一阶连续偏导, 对于无约束的最优化问题而言:  $\min_{x \in R^n} f(x)$

而  $f(x)$  在  $x^{(k)}$  附近的一阶泰勒展开如下, 其中  $g_k = g(x^{(k)}) = \nabla f(x^{(k)})$  是  $f(x)$  在  $x^{(k)}$  的梯度:

$$f(x) = f(x^{(k)}) + g_k^T (x - x^{(k)})$$

所以对于  $x = x^{(k+1)}$ :

$$f(x^{(k+1)}) = f(x^{(k)}) + g_k^T (x^{(k+1)} - x^{(k)})$$

令  $x^{(k+1)} = x^{(k)} + \lambda_k p_k$ ,  $p_k$  是搜索方向,  $\lambda_k$  是步长, 代入上式, 有

$$\begin{aligned} f(x^{(k+1)}) &= f(x^{(k)}) + g_k^T (x^{(k)} + \lambda_k p_k - x^{(k)}) \\ &= f(x^{(k)}) + g_k^T \lambda_k p_k \end{aligned}$$

为了让每次迭代的函数值变小, 可以取  $p_k = -\nabla f(x^{(k)})$

把  $\lambda_k$  看成是可变化的, 所以需要搜索  $\lambda_k$  使得

$$f(x^{(k)} + \lambda_k p_k) = \min_{\lambda \geq 0} f(x^{(k)} + \lambda p_k)$$

梯度下降法:

输入: 目标函数  $f(x)$ , 梯度  $g(x) = \nabla f(x)$ , 精度要求  $\varepsilon$ 。

输出:  $f(x)$  的极小点  $x^*$ 。

1. 取初始值  $x^{(0)} \in R^n$ , 置  $k = 0$
2. 计算  $f(x^{(k)})$
3. 计算梯度  $g_k = g(x^{(k)})$ , 当  $\|g_k\| < \varepsilon$ , 则停止计算, 得到近似解  $x^* = x^{(k)}$ ; 否则, 令  $p_k = -g(x^{(k)})$ , 求  $\lambda_k$  使得  $f(x^{(k)} + \lambda_k p_k) = \min_{\lambda \geq 0} f(x^{(k)} + \lambda p_k)$
4. 置  $x^{(k+1)} = x^{(k)} + \lambda_k p_k$ , 计算  $f(x^{(k+1)})$  当  $\|f(x^{(k+1)}) - f(x^{(k)})\| < \varepsilon$  或  $\|x^{(k+1)} - x^{(k)}\| < \varepsilon$  时, 停止迭代, 令  $x^* = x^{(k+1)}$
5. 否则, 置  $k = k + 1$ , 转第 3 步

只有当目标函数是凸函数时, 梯度下降得到的才是全局最优解。

### 12.2.2.2 《机器学习》的视角

梯度下降是一阶 (first order) (只用一阶导, 不用高阶导数) 优化方法, 是求解无约束优化问题最简单、最经典的方法之一。

考虑无约束优化问题  $\min_x f(x)$ ,  $f(x)$  是连续可微函数, 如果能构造一个序列  $x^0, x^1, x^2, \dots$  满足

$$f(x^{t+1}) < f(x^t), t = 0, 1, 2, \dots$$

那么不断执行这个过程, 就可以收敛到局部极小点, 根据泰勒展开有:

$$\begin{aligned} f(x) &= f(x^{(k)}) + \nabla f(x^{(k)})^T (x - x^{(k)}) \\ f(x + \Delta x) &= f(x^{(k)}) + \nabla f(x^{(k)})^T (x + \Delta x - x^{(k)}) \\ &= f(x^{(k)}) + \nabla f(x^{(k)})^T (x - x^{(k)}) + \nabla f(x^{(k)})^T \Delta x \\ &= f(x) + \nabla f(x^{(k)})^T \Delta x \end{aligned}$$

而  $\nabla f(x^{(k)})^T \Delta x$  是一个标量, 其转置等于自己, 所以

$$f(x + \Delta x) = f(x) + \Delta x^T \nabla f(x^{(k)})$$

想要让  $f(x + \Delta x) < f(x)$ , 只需要令:

$$\Delta x = -\gamma \nabla f(x)$$

其中的步长  $\gamma$  是一个小常数

如果  $f(x)$  满足  $L$ -Lipschitz 条件, 也就是说对于任意的  $x$ , 存在常数  $L$ , 使得  $\|\nabla f(x)\| \leq L$  成立, 那么设置步长为  $\frac{1}{2L}$  就可以确保收敛到局部极小点。

同样地, 当目标函数是凸函数时, 局部极小点就对应全局最小点, 此时梯度下降可以确保收敛到全局最优解。

### 12.2.3 牛顿法

#### 12.2.3.1 二阶导基本性质

对于点  $x = x_0$ ,

- 一阶导  $f'(x_0) = 0$  时, 如果二阶导  $f''(x_0) > 0$ , 那么  $f(x_0)$  是极小值,  $x_0$  是极小点
- 一阶导  $f'(x_0) = 0$ , 如果二阶导  $f''(x_0) < 0$ , 那么  $f(x_0)$  是极大值,  $x_0$  是极大点
- 一阶导  $f'(x_0) = 0$ , 如果二阶导  $f''(x_0) = 0$ , 那么  $x_0$  是鞍点

证明:

对于任意  $x_1$ , 根据二阶泰勒展开, 有

$$f(x_1) = f(x_0) + f'(x_0)(x_1 - x_0) + \frac{1}{2}f''(x_0)(x_1 - x_0)^2 + \dots + R_n(x_1)$$

因为  $f''(x_0) > 0$  且  $f'(x_0) = 0$ , 所以, 不论  $x_1 > x_0$  还是  $x_1 < x_0$ , 总有  $f(x_1) > f(x_0)$ , 也就是周围的函数值都比  $f(x_0)$  大, 而  $x_0$  又是极值点, 所以是极小点。

### 12.2.3.2 牛顿法

对于矩阵形式,  $x$  是一个  $n \times 1$  的列向量,  $H(x)$  是  $f(x)$  的海赛矩阵, 即二阶导,  $\text{shape}$  是  $n \times n$ :

$$f(x) = f(x^{(x)}) + g_k^T(x - x^{(k)}) + \frac{1}{2}(x - x^{(k)})^T H(x^{(k)})(x - x^{(k)})$$

函数  $f(x)$  有极值的必要条件是在极值点处一阶导为 0, 特别地, 当  $H(x^{(k)})$  是正定矩阵时 (二阶导大于 0), 是极小值。

牛顿法利用极小点的必要条件  $\nabla f(x) = 0$ , 每次迭代从点  $x^{(k)}$  开始, 求目标函数极小点, 作为第  $k + 1$  次迭代值  $x^{(k+1)}$ , 具体地, 假设  $\nabla f(x^{(k+1)}) = 0$ , 有

$$\begin{aligned} f(x) &= f(x^{(x)}) + g_k^T(x - x^{(k)}) + \frac{1}{2}(x - x^{(k)})^T H(x^{(k)})(x - x^{(k)}) \\ &= f(x^{(x)}) + [g_k^T + \frac{1}{2}(x - x^{(k)})^T H(x^{(k)})](x - x^{(k)}) \\ &= f(x^{(x)}) + [g_k + \frac{1}{2}H(x^{(k)})(x - x^{(k)})]^T(x - x^{(k)}) \end{aligned}$$

把其中的  $g_k + \frac{1}{2}H(x^{(k)})(x - x^{(k)})$  看成一阶导, 则上式就是一阶泰勒展开。记  $H^k = H(x^{(k)})$ , 令  $x = x^{(k+1)}$ , 令一阶导为 0:

$$\begin{aligned} g_k + \frac{1}{2}H^k(x^{(k+1)} - x^{(k)}) &= 0 \\ g_k &= -\frac{1}{2}H^k(x^{(k+1)} - x^{(k)}) \\ -2H_k^{-1}g_k &= x^{(k+1)} - x^{(k)} \\ x^{(k+1)} &= -2H_k^{-1}g_k + x^{(k)} \end{aligned}$$

可以无视这个 2, 变成:

$$x^{(k+1)} = x^{(k)} - H_k^{-1}g_k$$

或者

$$x^{(k+1)} = x^{(k)} + p_k$$

其中,

$$H_k p_k = -g_k$$

牛顿法:

输入: 目标函数  $f(x)$ , 梯度  $g(x) = \nabla f(x)$ , 海赛矩阵  $H(x)$ , 精度要求  $\varepsilon$ 。

输出:  $f(x)$  的极小点  $x^*$ 。

1. 取初始点  $x^{(0)}$ , 置  $k = 0$
2. 计算  $g_k = g(x^{(k)})$
3. 若  $\|g_k\| < \varepsilon$ , 则停止计算, 得到近似解  $x^* = x^{(k)}$
4. 计算  $H_k = H(x^{(k)})$ , 并求  $p_k$ , 满足  $H_k p_k = -g_k$
5. 置  $x^{(k+1)} = x^{(k)} + p_k$
6. 置  $k = k + 1$ , 转到第 2 步

其中的步骤 4, 求  $p_k$  时,  $p_k = -H_k^{-1}g_k$  需要求解  $H_k^{-1}$  很复杂。

### 12.2.4 拟牛顿法的思路

基本想法就是通过一个  $n$  阶矩阵  $G_k = G(x^{(k)})$  来近似代替  $H^{-1}(x^{(k)})$ 。

### 12.2.5 DFP(Davidon-Fletcher-Powell)

### 12.2.6 BFGS(Broydon-Fletcher-Goldfarb-Shanno)

## 12.3 拉格朗日对偶性

## 12.4 信息论相关

### 12.4.1 凸集

假设  $S$  为在实或复向量空间的集。若对于所有  $x, y \in S$  和所有的  $t \in [0, 1]$  都有  $tx + (1 - t)y \in S$ , 则  $S$  称为凸集。

也就是说,  $S$  中任意两点间的线段都属于  $S$

性质:

如果  $S$  是凸集, 对于任意的  $u_1, u_2, \dots, u_r \in S$ , 以及所有的非负  $\lambda_1, \lambda_2, \dots, \lambda_r$  满足  $\lambda_1 + \lambda_2 + \dots + \lambda_r = 1$ , 都有  $\sum_{k=1}^r \lambda_k u_k \in S$ 。这个组合称为  $u_1, u_2, \dots, u_r$  的凸组合。

### 12.4.2 凸函数

函数是凸函数: 曲线上任意两点  $x$  和  $y$  所作割线 (与函数图像有两个不同交点的直线, 如果只有一个交点, 那就是切线) 一定在这两点间的函数图像的上方:

$$tf(x) + (1 - t)f(y) \geq f(tx + (1 - t)y), 0 \leq t \leq 1$$

有如下几个常用性质:

- 一元可微函数在某个区间上是凸的, 当且仅当它的导数在该区间上单调不减。
- 一元连续可微函数在区间上是凸的, 当且仅当函数位于所有它的切线的上方: 对于区间内的所有  $x$  和  $y$ , 都有  $f(y) \geq f(x) + f'(x)(y - x)$  (右边就是一阶泰勒展开)。特别地, 如果  $f'(c) = 0$ , 那么  $f(c)$  是  $f(x)$  的最小值。
- 一元二阶可微的函数在区间上是凸的, 当且仅当它的二阶导数是非负的; 这可以用来判断某个函数是不是凸函数。如果它的二阶导数是正数, 那么函数就是严格凸的, 但反过来不成立。
- 多元二次可微的连续函数在凸集上是凸的, 当且仅当它的黑塞矩阵在凸集的内部是半正定的

### 12.4.3 KL 散度

熵的小结: <https://blog.csdn.net/haolexiao/article/details/70142571>

相对熵 (relative entropy) 又称为 **KL 散度** (Kullback-Leibler divergence, 简称 KLD), 信息散度 (**information divergence**), 信息增益 (**information gain**)。

KL 散度是两个概率分布  $P$  和  $Q$  差别的非对称性的度量。KL 散度是用来度量使用基于  $Q$  的编码来编码来自  $P$  的样本 平均所需的额外的位元数。典型情况下,  $P$  表示数据的真实分布,  $Q$  表示数据的理论分布, 模型分布, 或  $P$  的近似分布。

注意:  $D_{KL}(P||Q)$  是指的用分布  $Q$  来近似数据的真实分布  $P$ , 先写  $P$  再写  $Q$ , 公式里没有  $-\ln$  的时候, 就是  $p/q$

对于离散随机变量:

$$D_{KL}(P||Q) = \sum_i P(i) \ln \frac{P(i)}{Q(i)} = - \sum_i P(i) \ln \frac{Q(i)}{P(i)}$$

KL 散度仅当概率  $P$  和  $Q$  各自总和均为 1, 且对于任何  $i$  皆满足  $Q(i) > 0$  及  $P(i) > 0$  时, 才有定义。如果出现  $0 \ln 0$ , 当做 0

对于连续随机变量:

$$D_{KL}(P||Q) = \int_{-\infty}^{\infty} p(x) \ln \frac{p(x)}{q(x)} dx$$

性质:

KL 散度大于等于 0

证明:

先了解一下 Jensen 不等式:

如果  $\varphi$  是一个凸函数, 那么有:

$$\varphi(E(x)) \leq E(\varphi(x))$$

对于离散随机变量,  $\sum_{i=1}^n \lambda_i = 1$ :

$$\varphi(\sum_{i=1}^n g(x_i) \lambda_i) \leq \sum_{i=1}^n \varphi(g(x_i)) \lambda_i$$

当我们取  $g(x) = x$ ,  $\lambda_i = 1/n$ ,  $\varphi(x) = \log(x)$  时, 就有

$$\log(\sum_{i=1}^n \frac{x_i}{n}) \geq \sum_{i=1}^n \frac{\log(x_i)}{n}$$

对于连续随机变量, 如果  $f(x)$  是非负函数, 且满足 ( $f(x)$  是概率密度函数):

$$\int_{-\infty}^{\infty} f(x) dx = 1$$

如果  $\varphi$  在  $g(x)$  的值域中是凸函数, 那么

$$\varphi(\int_{-\infty}^{\infty} g(x) f(x) dx) \leq \int_{-\infty}^{\infty} \varphi(g(x)) f(x) dx$$

特别地, 当  $g(x) = x$  时, 有

$$\varphi(\int_{-\infty}^{\infty} x f(x) dx) \leq \int_{-\infty}^{\infty} \varphi(x) f(x) dx$$

回到这个问题中,  $g(x) = \frac{q(x)}{p(x)}$ ,  $\varphi(x) = -\log x$  是一个严格凸函数, 那么  $\varphi(g(x)) = -\log \frac{q(x)}{p(x)}$ , 所以

$$\begin{aligned} D_{KL}(P||Q) &= \int_{-\infty}^{\infty} p(x) \ln \frac{p(x)}{q(x)} dx \\ &= \int_{-\infty}^{\infty} p(x) (-\ln \frac{q(x)}{p(x)}) dx \\ &= \int_{-\infty}^{\infty} (-\ln \frac{q(x)}{p(x)}) p(x) dx \\ &\geq -\ln(\int_{-\infty}^{\infty} \frac{q(x)}{p(x)} p(x) dx) \\ &\geq -\ln(\int_{-\infty}^{\infty} q(x) dx) \\ &= -\ln 1 = 0 \end{aligned}$$

## 12.5 采样

假设有一个很复杂的概率密度函数  $p(x)$ ，求解随机变量基于此概率下的某个函数期望，即：

$$E_{x \sim p(x)}[f(x)]$$

求解有两种方法：

解析法：

将上式展开成积分，并通过积分求解：

$$\int_x p(x) f(x) dx$$

对于简单的分布，可以直接这么做。但  $\text{dnn}$  就不可行了。

蒙特卡洛法：

根据大数定理，当采样数量足够大时，采样的样本就可以无限近似地表示原分布：

$$\frac{1}{N} \sum_{x_i \sim p(x), i=1}^N f(x_i)$$

### 12.5.1 拒绝采样

拒绝采样又叫接受/拒绝采样 (Accept-Reject Sampling)，对于目标分布  $p(x)$ ，选取一个容易采样的参考分布  $q(x)$ ，使得对于任意  $x$  都有  $p(x) \leq M \cdot q(x)$ ，则可以按如下过程采样：

- 从参考分布  $q(x)$  中随机抽取一个样本  $x_i$
- 从均匀分布  $U(0, 1)$  产生一个随机数  $u_i$
- 如果  $u_i \leq \frac{p(x_i)}{Mq(x_i)}$ ，则接受样本  $x_i$ ，否则拒绝。

重新进行如上 3 个步骤，直到新产生的样本  $x_i$  被接受

其中的第三步是因为  $p(x) \leq M \cdot q(x)$ ，所以  $\frac{p(x_i)}{Mq(x_i)} \leq 1$ ，说明只有函数值在  $p(x_i)$  下方的  $x_i$  才接受，所以  $x_i \sim p(x)$ 。相当于为目标分布  $p(x)$  选一个包络函数  $M \cdot q(x)$ ，包络函数紧，每次采样时样本被接受的概率越大，采样效率越高。实际应用中还有自适应拒绝采样等。

### 12.5.2 重要性采样

强化学习经常使用重要性采样。重要性采样主要应用在一些难以直接采样的数据分布上。

我们令待采样分布为  $p(x)$ ，有另一个简单可采样且定义域和  $p(x)$  相同的概率密度函数为  $\tilde{p}(x)$ ，可以得到：

$$\begin{aligned} E_{x \sim p(x)}[f(x)] &= \int_x p(x) f(x) dx \\ &= \int_x \tilde{p}(x) \frac{p(x)}{\tilde{p}(x)} f(x) dx \\ &= E_{x \sim \tilde{p}(x)}\left[\frac{p(x)}{\tilde{p}(x)} f(x)\right] \\ &\simeq \frac{1}{N} \sum_{x_i \sim \tilde{p}(x), i=1}^N \frac{p(x)}{\tilde{p}(x)} f(x) \end{aligned}$$

因此，只需要从简单分布  $\tilde{p}(x)$  中采样，然后分别计算  $p(x)$ 、 $\tilde{p}(x)$  和  $f(x)$  就可以了。

最好选择一个和原始分布尽量接近的近似分布进行采样。



例如，要对一个均值 1，方差 1 的正态分布进行采样，有两个可选的分布：均值 1 方差 0.5 和均值 1 方差 2。

从图像上可以看到方差为 0.5 的过分集中在均值附近，而且方差为 2 的与原分布重合度较高，所以应该选方差为 2 的。

### 12.5.3 马尔可夫蒙特卡洛采样 (MCMC)

如果是高维空间的随机向量，拒绝采样和重要性采样经常难以找到合适的参考分布，采样效率低下（样本的接受概率低或者重要性权重低），此时可以考虑马尔可夫蒙特卡洛采样法。

蒙特卡洛法指基于采样的数值近似求解方法，马尔可夫链用于进行采样。

基本思想是：

- 针对待采样的目标分布，构造一个马尔可夫链，使得该马尔可夫链的平稳分布就是目标分布；
- 然后从任何一个初始状态出发，沿着马尔可夫链进行状态转移，最终得到的状态转移序列会收敛到目标分布
- 由此可以得到目标分布的一系列样本

核心点是如何构造合适的马尔可夫链，即确定马尔可夫链的状态转移概率。

#### 12.5.3.1 Metropolis-Hastings 采样

对于目标分布  $p(x)$ ，首先选择一个容易采样的参考条件分布  $q(x^*|x)$ ，并令

$$A(x, x^*) = \min\{1, \frac{p(x^*)q(x|x^*)}{p(x)q(x^*|x)}\}$$

然后根据如下过程采样：

- 随机选一个初始样本  $x^{(0)}$
- For  $t = 1, 2, 3, \dots$  :
  - 根据参考条件分布  $q(x^*|x^{(t-1)})$  抽取一个样本  $x^*$ ；
  - 根据均匀分布  $U(0, 1)$  产生随机数  $u$ ；
  - 若  $u < A(x^{(t-1)}, x^*)$ ，则令  $x^{(t)} = x^*$  【接受新样本】，否则令  $x^{(t)} = x^{(t-1)}$  【拒绝新样本，维持旧样本】

可以证明，上述过程得到的样本序列  $\{\dots, x^{(t-1)}, x^{(t)}, \dots\}$  最终会收敛到目标分布  $p(x)$

#### 12.5.3.2 吉布斯采样

吉布斯采样是 Metropolis-Hastings 算法的一个特例，核心思想是只对样本的一个维度进行采样和更新。对于目标分布  $p(x)$ ，其中  $x = (x_1, x_2, \dots, x_d)$  是一个多维向量，按如下过程采样：

- 随机选择初始状态  $x^{(0)} = (x_1^{(0)}, x_2^{(0)}, \dots, x_d^{(0)})$
- For  $t = 1, 2, 3, \dots$  :
  - 对于前一步产生的样本  $x^{(t-1)} = (x_1^{(t-1)}, x_2^{(t-1)}, \dots, x_d^{(t-1)})$ ，依次采样和更新每个维度的值，即依次抽取分量  $x_1^{(t)} \sim p(x_1|x_2^{(t-1)}, x_3^{(t-1)}, \dots, x_d^{(t-1)})$ ， $x_2^{(t)} \sim p(x_2|x_1^{(t-1)}, x_3^{(t-1)}, \dots, x_d^{(t-1)})$ ， $\dots$ ， $x_d^{(t)} \sim p(x_d|x_1^{(t-1)}, x_2^{(t-1)}, \dots, x_{d-1}^{(t-1)})$
  - 形成新的样本  $x^{(t)} = (x_1^{(t)}, x_2^{(t)}, \dots, x_d^{(t)})$

同样可以证明，上述过程得到的样本序列  $\{\dots, x^{(t-1)}, x^{(t)}, \dots\}$  会收敛到目标分布  $p(x)$ 。另外，步骤 2 中对样本每个维度的抽样和更新操作，不是必须按下标顺序进行的，可以是随机顺序。

注意点：

- 拒绝采样中，如果某一步中采样被拒绝，则该步不会产生新样本，需要重新采样。但 MCMC 采样每一步都会产生一个样本，只是有时候这个样本与之前的样本一样而已。
- MCMC 采样是在不断迭代过程中逐渐收敛到平稳分布的。实际应用中一般会对得到的样本序列进行“burn-in”处理，即截除掉序列中最开始的一部分样本，只保留后面的样本。

MCMC 得到的样本序列中相邻的样本是不独立的，因为后一个样本是由前一个样本根据特定的转移概率得到的。如果仅仅是采样，并不要求样本之间相互独立。

如果确实需要产生独立同分布的样本，可以：

- 同行运行多条马尔可夫链，这样不同链上的样本是独立的
- 或者在同一条马尔可夫链上每隔若干个样本才选取一个，这样选出来的样本也是近似独立的。