

INT104

Artificial Intelligence

参考资料

西浦学长

X



联合出品

2021.05

INT104 期末资料

INT104 期末资料

前言

Lec 1: Introduction to Artificial Intelligence

什么是人工智能 (AI)

图灵测试 Turing Test

构建AI的难点

现实生活中AI的应用

CAPTCHA

Lec 2: python

python 基础语法

Lec 3: Probability and Linear Algebra

Probability

Probability Space

Random Variables

Density Functions

Distribution Functions

Common Density Functions

Joint Distributions

Conditional Distributions

Expected Value

Variance & Standard Deviation

Linear Algebra

Vectors

Vector Space

Linear Independence

Basis

Matrix transpose

Determinant

Eigenvectors and Eigenvalues

Matrix Inverse

Distances

Lec 4: Optimization

Introduction

Least-squares fitting 最小二乘拟合

Weber point

Classification of optimization problems 优化问题的分类

Convex optimization problems 凸优化问题

Property of Convex optimization problems 凸优化问题的性质

非凸优化问题

解决优化问题

Practically solving optimization problems

Lec 5: learning

Inductive Learning Hypothesis 归纳学习假说

评估分类结果 Evaluation of Classification Learning

什么是模式 (Pattern) 和类别 (Category)

Components of Learning System 一个学习系统的组成部分

Features 特征

Good/Bad Features & Classification

Feature Dimension

Overfitting and underfitting 过拟合与欠拟合

设计学习系统的工作流程 Cycle of Design

Type of Learning

Lec 6: Feature Selection

Features and Patterns

Data Reduction 数据缩减

Sampling 抽样

Dimensionality Reduction 降维

Feature Selection

Univariate Feature Selection

Other methods: Multivariate Feature Selection

Lec 7: Classification: Introduction & Quality Assessment

Training & Inference 训练和推理

Evaluating Classification Methods 评价分类结果的方法

Confusion Matrix 混淆矩阵

Receiver Operation Characteristics (ROC) 接收者操作特征曲线

Cross Validation 交叉验证

Data Partitioning Methods 数据集分割方法

Random Sampling

Bootstrap 自助法

k-fold Cross Validation k 折交叉验证

Leave-one-out Cross Validation 留一交叉验证

Three-way Data Splits

Lec 8 Discriminant Functions

Linearly Separable 线性可分

Logic Function by Linear Combination

Linear Discriminant Functions (LDF) 线性判别函数

Multiple-class Problem

Perceptrons 神经元

Deep Learning

Linear Functions as Features

LDA

PCA

ICA

NMF

Lec 9 Naive Bayesian 朴素贝叶斯

Bayes' Rule

Model Likelihood 模型似然性

Log-likelihood

Bayesian Estimation & Laplacian Correction 贝叶斯估计与拉普拉斯平滑

Naive Bayes Classifier

Semi-Naive Bayes Classifier

Bayesian Network

Lec 10: Clustering 聚类

Agglomerative Clustering 凝聚层次聚类

Divisive Clustering 分裂聚类

Model based Clustering

Expectation Maximisation 最大期望算法

Model Selection

Lec 11: Classification: Non-parametric Modeling

Kernel Density Estimation 核密度估计

K-Nearest Neighbour k近邻(KNN)

Density Estimation

Pros and Cons of Non-parametric Modeling

Lec 12: Propositional Logic & Prolog 命题逻辑

logic

Propositional Logic

Tautologies & Consistency 重言式与一致性

First-Order Logic 一阶逻辑

前言

Int104 这门课程刚刚经过了改革，因此笔者将尽可能全面的涵盖课程内容，仅供同学们参考之用，不构成复习建议。

Lec 1: Introduction to Artificial Intelligence

什么是人工智能 (AI)

AI is the study of complex information processing problems that often have their roots in some aspect of biological information processing. The goal of the subject is to identify solvable and interesting information processing problems, and solve them. (David Marr)

The intelligent connection of perception to action (Rodney Brooks)

Actions that are indistinguishable from a human's (Alan Turing)

人工智能并不是一个新兴事物。1956年人工智能就被确立为一门学科。其终极目标是让计算机拥有与人类一样的行为（而达到这个行为的过程可以不同）。

图灵测试 Turing Test

1950年，图灵发表了一篇划时代的论文(*Computing machinery and intelligence*)，文中预言了创造出具有真正智能的机器的可能性。由于注意到“智能”这一概念难以确切定义，他提出了著名的图灵测试：如果一台机器能够与人类展开对话（通过电传设备）而不能被辨别出其机器身份，那么称这台机器具有智能。这一简化使得图灵能够令人信服地说明“思考的机器”是可能的。论文中还回答了对这一假说的各种常见质疑。图灵测试是人工智能哲学方面第一个严肃的提案。

图灵在这篇论文中提出了以下问题与观点：

Can machines think? Can we tell if a conversation is by a machine and not a human?

- 计算机（机器）能思考吗？我们能否辨别一段对话是来自人类还是机器？

Operational test for intelligent behavior: aka the Imitation Game

- 对于智能行为的可操作性测试：模仿游戏。

Predicted that by 2000, a machine might have a 30% chance of fooling a lay person for 5 minutes

- （图灵）预测，在2000年之前，一台机器有30%的概率骗过一个外行5分钟。

Suggested major components of AI: knowledge rep., reasoning, natural language processing, learning

- 提出了AI的主要构成部分：知识表示，推理，自然语言处理，学习。

构建AI的难点

AI problems often involve large, complex data

- Speech, images, natural languages, genomic data, ...

- What are the right primitives to use?

- Data are often noisy, unstructured and have missing values

- 超大复杂数据集处理（语音，图像，文本）。数据集中常常有错误或者缺失的部分。

Computationally (NP-) hard

- 算力需求高 (NP 问题)

Very hard to define general, computational “competence theories” for specific tasks that say what is computed and why (what to compute)

- 如何将实际任务转化为纯数学计算任务

Need algorithms that use domain-specific knowledge and constraints with incomplete models, while being time and space constrained, stable, and robust (how to compute)

- 人工智能算法

现实生活中AI的应用

- 聊天机器人 (chat bot)
- 语音识别 (Speech Recognition)
- 机翻 (Machine Translation)
- 问答系统 (Question Answering Systems)
- 游戏人工智能: 深蓝, Alpha Go (Game Playing)
- 网页排名 (Web Page Ranking)
- 新闻概述 (News Aggregation and Summarization)
- 网页广告投送 (Web Advertising)
- 导航 (Navigation)
- 推荐系统 (Collaborative Filtering)
- 图像搜索 (Visual Search)
- 人脸识别 (Face Detection)
- 字迹识别 (Handwriting Recognition)
- 人体运动识别和追踪 (Body Part Detection and Tracking)

CAPTCHA

Completely Automated Public Turing test to tell Computers and Humans Apart

全自动区分计算机和人类的公开图灵测试, 俗称验证码, 是一种区分用户是机器或人类的公共全自动程序。在CAPTCHA测试中, 作为服务器的计算机会自动生成一个问题由用户来解答。这个问题可以由计算机生成并评判, 但是必须只有人类才能解答。由于机器无法解答CAPTCHA的问题, 回答出问题的用户即可视为人类。

这个其实在日常生活中应用很广, 我们在登录网站时输入的验证码就属于此范畴。



Lec 2: python

python 和 c++ 是现在最主流的两种AI应用开发语言。其中python因为语言特性简单，第三方支持库全被认为是适合初学者学习 AI 的语言。

另外，python作为现在主流的数据处理语言，能很好地帮助你完成准备数据集的工作。

Int104默认同学们有其他编程语言的基础 (prerequisite) 。

python 基础语法

1. 定义变量

python 是一门弱类型的动态语言，其变量类型在解释器在运行时推算而无需我们在编写代码时指定。

```
a = 2  
s = 'adsd'
```

2. 表达式

与其他主流语言不同的是，python不支持自增运算符和三元表达式。

算数表达式:

符号	名称	示例
+	加	a + b
-	减	a - b
*	乘以	a * b
\	除以	a / b
%	取余	a % b
**	幂方	a ** b (a^b)
-	取负	- a

自左向右结合，取负优先级最高，幂方次之，乘，除，取余再次之，加，减最低。

逻辑表达式:

符号	名称	示例
>	大于	a > b
\geq	大于等于	a \geq b
<	小于	a < b
\leq	小于等于	a \leq b
not	非	not a > b
$=$	判断是否相等	a == b
\neq	判断是否不相等	a \neq b
and	且	a < b and b < c
or	或	a < b or b < c

`and` , `or` , `not` 优先级相同且最低, 其余逻辑运算符自左向右结合, 优先级相等。

3. 控制流与缩进

条件判断:

```
if a == b:
    # do something
elif a > b:
    # do something
else:
    # do something
```

判断条件不需要括号包裹, 同时不要忘记冒号。

python 采用了独特的缩进法标识代码块, 一般来说使用tab键进行缩进。只要保持统一, tab键是两个空格(2 spaces)或是四个空格 (4 spaces) 没有区别, 甚至在同一代码块中, 只要缩进量统一, 任意缩进量都是合法的。

```
a = 2
b = 1

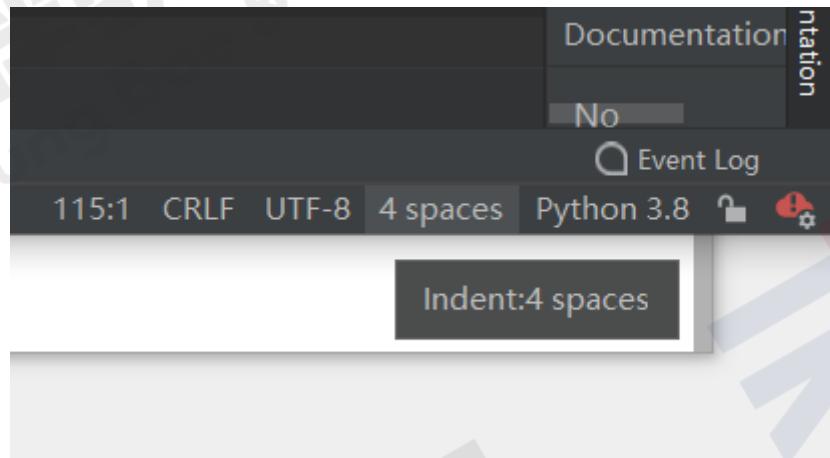
if a > b:
    print('hh')
else:
    print('not hh')

if a > b:
    print('hh')
else:
    print('not hh')
```

虽然不美观, 但是以上这段代码是合法的。

但是在一个代码块中混用不同量缩进会导致错误。因此在一份python文件中最好确保缩进统一为四空格(4 spaces)。如非必要，也不要混用tab键和空格键。

主流 ide 或者文本编辑器的右下角都可以修改tab的缩进量，这里以老师推荐的 pycharm 为例，保留4 spaces即可：



基于篇幅的考虑，剩余的python语法不再包含在正文中

Lec 3: Probability and Linear Algebra

Probability

Probability Space

当我们用数学语言描述一件事发生的概率时，就是在定义一个概率空间。概率空间由一个三元组表示(tripple)

$$(\Omega, F, P)$$

- Ω : represents a nonempty set, whose elements are sometimes known as outcomes or states of nature (Sample Space)

Ω 代表一个非空集合，代表样本集。

- F : represents a set, whose elements are called events. The events are subsets of Ω .

F 代表事件集合， F 是 Ω 的子集。

- P : represents the probability measure

P 则表示概率方程，这方程赋给了概率空间中每一个事件一个0到1之间的概率值

综上， (Ω, F, P) 这个三元组所表达的含义就是，总事件集合 Ω 的子集 F 发生的概率为 $P(F)$

因此我们有：

$$P(\Omega) = 1$$

Random Variables

Random variable is a function (or mapping) from a set of possible outcomes of the experiment to an interval of real (complex) numbers.

随机变量

想要用数学语言描述或者用计算机模拟概率，我们首先要将现实中可能出现的情况(事件)映射到实数集中。

例如一个最简单的离散量映射就是把一个6面骰子的正面的状态映射成计算机中一个变量的六个数值。

```
top = 1;
```

我使用python语言初始化一个变量 `top` 用于代表一个六面骰子的状态。

`top = 1` 即为骰子1朝上, `top = 2` 即为骰子2朝上, 依此类推。如此一来 $P(\text{骰子1朝上})$ 也可以用 $P(\text{top} = 1)$ 表示了。当我们需要模拟一个骰子的抛掷结果时, 只需要写如下语句就可以了:

```
import random  
top = random.randint(1,6)
```

random.randint(a,b) 会返回一个在 [a, b] 中的随机整数。

这里的随机变量可以写作下面这样一个映射:

骰子1面朝上	→	1
骰子2面朝上	→	2
...		
骰子6面朝上	→	6

现在这个映射看起来非常的理所当然, 因为骰子的每个面本身就有个数字属性。当我们处理更加复杂的概率时, 这样的映射就需要花一些时间去理解。

需要注意的是, **Random Variables 的定义是一个函数 (function) 或者是一个映射 (Mapping)**。在上面这个骰子的例子中, 随机变量是从样本集 (骰子顶面点数的情况集合) 到实数集的一个映射。

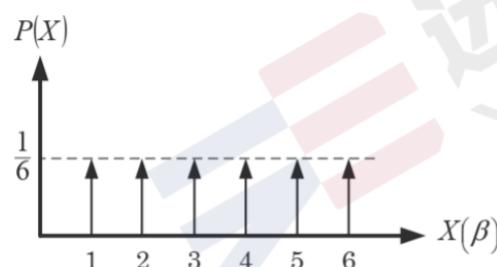
Density Functions

在上面一小节中, 我们将在这个映射的过程中, 出现了连续量 (continuous) 与离散量 (discrete) 的区别。现实事件中存在连续量, 而计算机只能表示离散量。

- Probability Mass Function

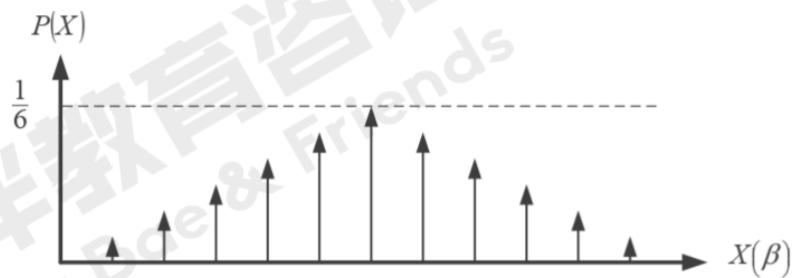
概率质量函数 (Probability Mass Function, 简写作PMF) 是**离散随机变量**在各特定取值上的概率。概率质量函数的函数的特点是: 函数值就是概率值。也就是说带入随机变量的值就能得到那个随机变量对应的事件发生的概率。

还是以骰子为例:



采取了前一小节所定义的一个骰子的Random Variables, 这是一个骰子的正面值的概率质量函数 X 。可以看到 $X(1) = \frac{1}{6}$, 而随机变量的值为1对应的事件是骰子的顶面点数为1。 $X(1) = \frac{1}{6}$ 就表示骰子1面朝上的概率是 $\frac{1}{6}$ 。

现在增加一个骰子, 概率空间中的样本集变为 **两个骰子朝上的面的点数的和的所有情况的集合**。同样采用点数与自然数对应的方式定义随机变量, 此时的概率质量函数为:



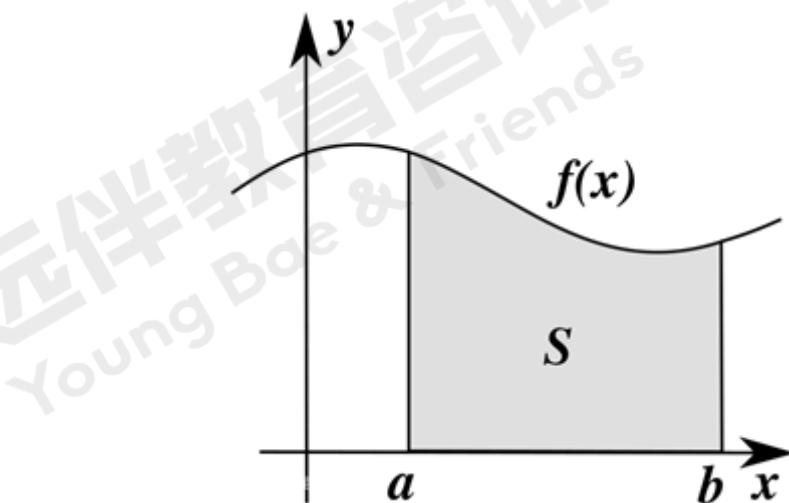
横轴上从左到右应依次为2到12.此时 $X(7) = 1/6$ 就表示两个骰子顶面点数的和为7的概率是 $\frac{1}{6}$ 。

- Probability Density Function

概率密度函数 (缩写 PDF) 可以看作概率质量函数的连续量版本。离散的随机变量 (Discrete random variables) 能描述的情况是有限的，随机变量应当也可以是连续量 (continuous random variables)。

请看这种情况：从一段木棍上随机选一个点切断，如何描述余下木棍长度的的概率分布？

长度是一个连续量，最后求得的概率分布函数应当是一条曲线（下图仅为示意图）：



这时我们引入微积分，随机变量落在内某个区间内的概率就是概率密度概率函数在这个区间内的定积分。

Distribution Functions

- Cumulative Distribution Function

将随机变量 X 的概率密度函数记作 f_X

在概率密度函数中引入从负无穷开始的定积分，就得到了连续量情况下的累积分布函数 (Cumulative Distribution Function, 缩写 CDF)

$$CDF(x) = \int_{-\infty}^x f_X(x) dx$$

因为随机变量落在内某个区间内的概率就是概率密度概率函数在这个区间内的定积分，我们有：

$$CDF(x) = P(X \leq x)$$

对于随机变量 X , CDF(x)也可记作 $F_X(x)$

综上，可得：

$$CDF(x) = F_X(x) = P(X \leq x) = \int_{-\infty}^x f_X(x) dx$$

当这个定积分的上界达到随机变量X值的上界时，CDF的值就应当等于1。

在CDF同样适用于概率质量函数，只不过此时是简单的相加而不需要积分了。

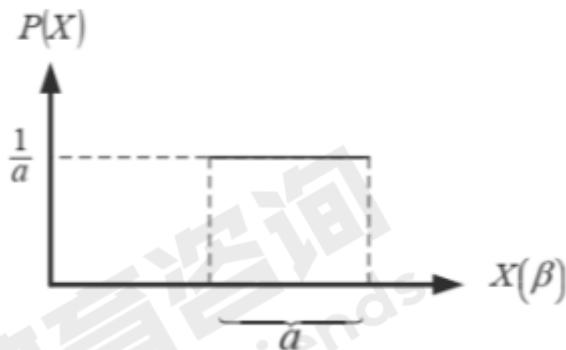
| 在本章余下的部分，随机变量 X 的密度函数就记作 f_X ，累积分布函数就记作 F_X

Common Density Functions

常见的密度函数有：

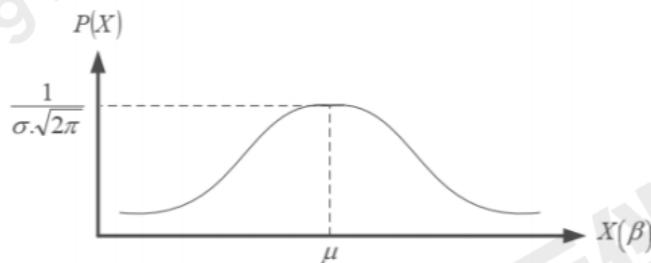
- 均匀密度 Uniform density

$$f(x) = \begin{cases} \frac{1}{a}, & b \leq x \leq a + b \\ 0, & \text{otherwise} \end{cases}$$



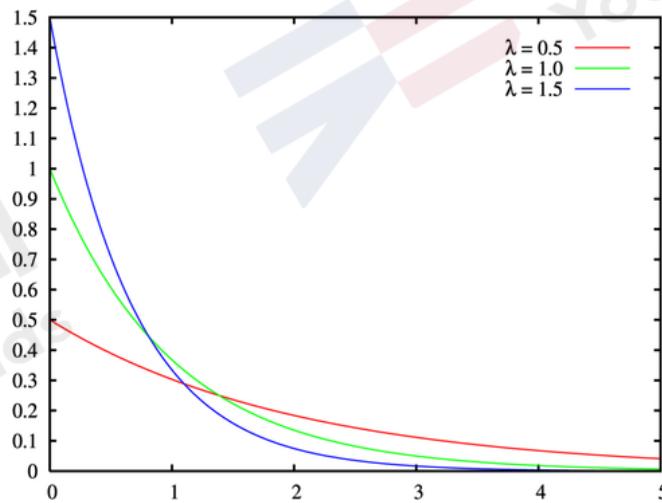
- Gaussian (or Normal) density 高斯（正态）密度

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} = N(\mu, \sigma)$$



- Exponential density 指数密度

$$f(x) = \lambda e^{-\lambda x}, x \geq 0$$



Joint Distributions

联合分布(Joint Distributions) 基于联合概率 (Joint Probability)

回到一开始的概率空间 (Probability Space) 中, 联合概率描述的就是样本集 Ω 中的两个事件 A, B 同时发生的概率。

事件 A, B 的联合概率记作:

$$P(AB) \text{ 或者 } P(A \cap B) \text{ 或者 } P(A, B)$$

现在指定事件A 中的随机变量为X, 事件B中的随机变量为Y, 我们就可以用以下这个函数表示 $P(A, B)$ 的概率密度方程:

$$f_{X,Y}(x, y)$$

对这个二元方程做二元积分就可以得到联合分布(Joint Distributions):

$$F_{X,Y}(x, y) = P(X \leq x, Y \leq y) = \int_{-\infty}^x \int_{-\infty}^y f_{X,Y}(X, Y) dy dx$$

随机变量 X, Y 的联合分布就记作 $F_{X,Y}(x, y)$

联合分布就是联合概率 (Joint Probability) 的概率密度方程(PDF)的累积分布函数(CDF)。

Conditional Distributions

条件分布 (Conditional Distributions) 基于条件概率 (Conditional Probability)

联合概率描述的是样本集 Ω 中的两个事件 A, B 中一个事件在另一个事件已经发生的前提下发生的概率。

事件 A, B 的条件概率记作:

$$P(A|B)$$

条件概率的计算基于联合概率:

$$P(A|B) = \frac{P(AB)}{P(B)}$$

继续指定事件A 中的随机变量为X, 事件B中的随机变量为Y, 相似的, 条件概率的概率密度函数的计算也基于来联合概率的概率密度函数:

$$f_{X|Y}(x|y) = \frac{f_{X,Y}(x, y)}{f_Y(y)}$$

当这个方程中的随机变量 Y 已知时, 我们就能得到随机变量 X 在 $Y = y$ 的前提下的概率密度函数。

Bayes' rule (optional),

对条件概率的概率密度函数积分就可以得到条件分布函数(Conditional Distribution Function), 但是这里的积分并不是二元积分, 其中的 y 被视作常数。

$$F_{X|Y}(x|y) = P(X \leq x|Y = y) = \int_{-\infty}^x f_{X|Y}(x|y) dx = \frac{\int_{-\infty}^x f_{X,Y}(x, y) dx}{\int_{-\infty}^x f_Y(y) dy}$$

当事件A与事件B统计独立时:

| 统计独立的两个事件中的任一个是否发生不会影响另一个发生的概率

$$P(AB) = P(A) \times P(B)$$
$$f_{X|Y}(x, y) = f_X(x) \times f_Y(y)$$

所以我们有:

$$P(A|B) = \frac{P(AB)}{P(B)} = \frac{P(A) \times P(B)}{P(B)} = P(A)$$
$$f_{X,Y}(x, y) = \frac{f_{X,Y}(x, y)}{f_Y(y)} = \frac{f_X(x) \times f_Y(y)}{f_Y(y)} = f_X(x)$$

Expected Value

一个离散性随机变量的期望值是试验中每次可能的结果乘以其结果概率的总和。

| x_k 指第 k 种实验的值, p_k 指第 k 种实验结果出现的概率

$$E(X) = \sum_{k=1}^{\infty} x_k p_k$$

设连续性随机变量 X 的概率密度函数为 $f(x)$, 则 X 的数学期望 $E(X)$ 为:

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

| 与离散随机变量的期望值的算法同出一辙, 由于输出值是连续的, 所以把求和改成了积分

当 $f(x)$ 是随机变量 X 的概率密度函数, 若另外一随机变量 Y 满足 $Y = g(x)$ (关于 x 的任意函数) 时:

如果 X 是离散性的:

$$E[g(X)] = \sum g(x) f(x)$$

如果 X 是连续性的:

$$E[g(X)] = \int_{-\infty}^{\infty} g(x) f(x) dx$$

期望值的其他一些数学性质:

- 常量的期望值是其本身
- 如果 a 与 b 是常量 $E(aX + b) = aE[X] + b$
- 如果 X 和 Y 是两个统计独立的随机变量, $E[XY] = E[X] \times E[Y]$

Variance & Standard Deviation

方差与标准差

设 X 是期望值为 μ 的随机变量:

$$E(X) = \mu$$

记 X 的方差为 $var(x)$, 标准差为 δ_x :

$$\begin{aligned} var(X) &= \delta_x^2 = E[(X - \mu)^2] \\ &= \sum (X - \mu)^2 f(x) \\ &= E[X^2] - \mu^2 \\ &= E[X^2] - [E(X)]^2 \end{aligned}$$

$f(x)$ 是随机变量 $(X - \mu)^2$ 的概率密度函数。

方差有以下特性:

- 常量的方差是 0。
- 如果 a 和 b 是常量, $\text{var}(aX + b) = a^2 \text{var}(X)$
- 如果 X 和 Y 是统计独立的随机变量:

$$\text{var}(X + Y) = \text{var}(X) + \text{var}(Y)$$
$$\text{var}(X - Y) = \text{var}(X) - \text{var}(Y)$$

Linear Algebra

鉴于所有同学在大一都学过线性代数, 推荐同学们自行复习线性代数, 此处只提一下后面课程需要用到的线代基本概念。

Vectors

- n 维向量 :

$$v = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \dots \\ x_n \end{bmatrix} = [x_1 \quad x_2 \quad x_3 \quad \dots \quad x_n]^T$$

- 单位向量 (模为1的向量)。

$$\|v\| = \sqrt{v^T v} = 1$$

- 向量内积 (Inner product), 也称为向量点乘:

$$x = (x_1, x_2, \dots, x_n), y = (y_1, y_2, \dots, y_n)$$

$$\text{dot}(x, y) = \sum_{i=1}^n x_i y_i$$

点乘的两个向量维度必须相同

- 向量点乘的几何表示:

$$\text{dot}(x, y) = \|x\| \|y\| \cos\theta$$

θ 是向量 x, y 之间的夹角。

- 正交向量组

一个向量集 x_1, x_2, \dots, x_n 如果满足:

$$x_i^T x_j = \text{dot}(x_i, x_j) = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$$

则被称作正交向量组。

正交向量组是一组非零的两两正交 (即内积为0) 的向量构成的向量组。

任何一个向量集 (x_1, x_2, \dots, x_n) 合可以被看作一个基底 (basis)。任何一个非正交的基底, 可以通过 Gram-Schmidt 正交化转化为正交基底 (o_1, o_2, \dots, o_n) 。

Vector Space

$$\lambda u + \mu v \in W$$

简单地讲，向量空间是一个对向量加法(vector addition)和标量乘法(scalar multiplication)封闭的向量集合，也就是说，此向量集中任意数量的向量的任意线性组合依然在此向量集合中。向量空间还有其他定义和限制，在此不再赘述。

除了只含有零向量的向量空间外，向量空间是一个无限集。

Linear Independence

一个向量集: $\{v_1, v_2, \dots, v_k\}$, 满足:

$$\sum_{i=1}^k c_i v_i = 0 \implies c_1 = c_2 = \dots = c_k = 0$$

也就是此向量集合不存在结果为零向量地线性组合，或者说向量集中的任何一个向量都不能用其余向量的线性组合表示。

则向量集和之间的向量彼此**线性无关(Linear Independence)**，这个向量集(组)也可称作线性无关的.

- 在二维(R^2)或者三维 (R^3)中，当两个向量不在一条线上，他们是线性无关的。
- 在三维 (R^3)中，当三个向量不在一个平面内，他们是线性无关的。

Basis

基底是一个满足以下特性的向量集 $S = v_1, v_2, \dots, v_k$:

- S 是线性无关集
- S 张开(span)了一个空间 W

如果 V 是一个 n 维的向量空间(向量集)， S 是一个含有 n 个元素的 V 的子集，那么若 S 是线性无关集，则 S 是 V 的一个基底，或者说 S 张开了空间 V 。

- 基底不一定是正交集
- 一个向量空间可以有无限个基底

Matrix transpose

将矩阵的行列互换得到的新矩阵称为转置矩阵

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}, \quad A^T = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{m1} \\ a_{21} & a_{22} & \cdots & a_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1n} & a_{2n} & \cdots & a_{mn} \end{bmatrix}$$

- $(AB)^T = B^T A^T$
- 如果一个矩阵是对称的(Symmetric)，那么 $A^T = A$

Determinant

一个 2×2 矩阵 M 的行列式计算方法为

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

$$\det(M) = \begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc$$

任意规模的矩阵的行列式都可以通过**拉普拉斯展开化归为** 2×2 矩阵行列式的计算。

- 只有方形矩阵 (Square matrix) 才有行列式
- $\det(AB) = \det(A)\det(B)$

Eigenvectors and Eigenvalues

对于一个给定的方阵，它的**特征向量 (eigenvector)** 经过以下这个线性变换之后，得到的新向量仍然与原来的 v 保持在同一條直線上，但其长度或方向也许会改变。即

$$Av = \lambda v$$

λ 是一个标量，也就是**特征值 (Eigenvalue)**，特征值和特征向量都可以通过求解以上这个方程获得。

n 维矩阵一定有n个特征值 (包括重根，也可能是复根)

| 重根值相同的多个解，复根指负数解。

- 特征方程 Characteristic equation

$$\det[A - \lambda I_n] = 0$$

| I_n 是 $n \times n$ 的单位矩阵

- 行列式等于特征值的乘积

$$\det(A) = \prod_{i=1}^n \lambda_i$$

Matrix Inverse

矩阵 A 的逆矩阵记作 A^{-1} ，并满足以下方程：

$$AA^{-1} = A^{-1}A = I$$

- 只有方形矩阵可求逆
- 当一个方形矩阵的行列式为0时，此据正被称作奇异矩阵，并不可求逆
- 矩阵的条件数(condition number):

矩阵 A 的条件数记作：

$$\kappa(A) = \|A\| \|A^{-1}\|$$

| $\|A\|$ 是矩阵范数 (Norm)，具体的计算细节本课程未作要求，了解概念即可。

- 病态矩阵(ill-conditioned):

condition number 是一个矩阵 (或者它所描述的线性系统) 的稳定性或者敏感度的度量，如果一个矩阵的 condition number 在1附近，那么它就是well-conditioned的，如果远大于1，那么它就是 ill-conditioned 的，如果一个系统是 ill-conditioned 的，它的输出结果就不要太相信了

奇异矩阵的条件数是无穷大，如果一个矩阵虽然不奇异，但是条件数很大(接近奇异)，那么其就是一个病态矩阵。

- 逆矩阵的一些性质:

$$\det(A^{-1}) = \frac{1}{\det(A)}$$

$$(AB)^{-1} = B^{-1}A^{-1}$$

$$(A^T)^{-1} = (A^{-1})^T$$

Distances

函数 d 当满足以下特性时，就可以称为是一种距离 (distance) 的度量方法：

$$\begin{aligned} d(x, y) &> 0 \\ d(x, y) &= 0, \text{ if } x = y \\ d(x, y) &= d(y, x) \quad (\text{symmetry 对称性}) \\ d(x, y) &< d(x, z) + d(z, y) \quad (\text{triangle inequality 三角不等性}) \end{aligned}$$

两个向量之间的距离也可以视作两个向量之间的相似度(similarity)。

出于聚类 (clustering) 的目的，距离有时并不一定是一种度量 (Metric)，而仅仅代表一种相似程度。此时距离方程 d 将不再满足三角不等性和对称性。

- **Minkowski Distance 闵可夫斯基距离**

p 阶闵可夫斯基距离：

$$L_p(x, y) = \left(\sum_{k=1}^n |x_i - y_i|^p \right)^{\frac{1}{p}}$$

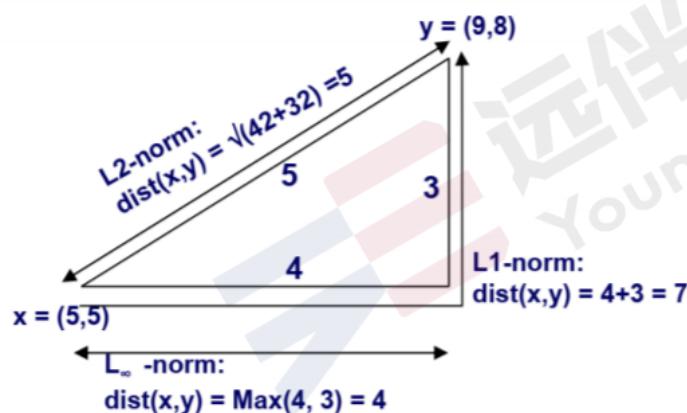
$p = 1$ 时 (L_1 norm)，闵可夫斯基距离也称曼哈顿距离 (Manhattan Distance)。

曼哈顿距离就是两个向量在各个维度上差值的和。

这里的维度指的应该是基底 (basis) 中每一个向量的方向。

$p = 2$ 时 (L_2 norm)，闵可夫斯基距离也称为欧式距离(Euclidean Distance)，即我们熟悉的距离公式。

$p = \infty$ 时 (L_∞ norm)，闵可夫斯基距离就是两点在差值最大的维度上的差值 (其他项在无限大指数下都可以忽略不计)。



- **Mahalanobis Distance 马氏距离**

马氏距离是一种相似性算法，其特点在于计算数据集中两向量相似度时同时考虑到了整个数据的分布情况。

$$D_M = (x - \mu)^T \sum^{-1} (x - \mu)$$

○ μ_i 是每个向量中第 i 维上的量 (表示同一种特性) 的平均值。

○ \sum 是整个样本集的协方差 (covariance)，马氏距离就是通过这个值将数据整体的分布情况纳入考量。

- **KL Divergence KL 散度**

KL 散度，也称相对熵，描述的是两个概率分布之间的“距离”

$$KL(p||q) = \sum_i p_i \log \frac{p_i}{q_i}$$

- o Measures the expected number of extra bits in case of using q rather than p
KL散度是用来度量使用基于 q 的分布来编码服从 p 的分布的样本所需的额外的平均比特数
- o Can be used as a distance measure when feature vectors form distributions
当特征向量形成了某种分布时，可以采用KL散度计算两个向量间的距离。
此处的feature vectors虽然也翻译为特征向量，但是其含义是描述实物或者抽象概念特征的向量，而非线性代数中的特征向量。
KL 散度就是一种非度量 (metric) 的距离，因为其不满足对称性和三角不等性。

Lec 4: Optimization

Introduction

最优化 (Mathematical optimization) 问题指的是形如:

$$\begin{aligned} & \underset{x}{\text{minimize}} f(x) \\ & \text{subject to } g_i(x) \leq 0 \quad i = 1, \dots, r \end{aligned}$$

的问题，其中:

- $x \in \mathbb{R}^n$, 优化变量 (optimization variable)
- $f : \mathbb{R}^n \rightarrow \mathbb{R}$ 目标方程 (objective function)
- $g_i : \mathbb{R}^n \rightarrow \mathbb{R}$ (不等) 约束方程

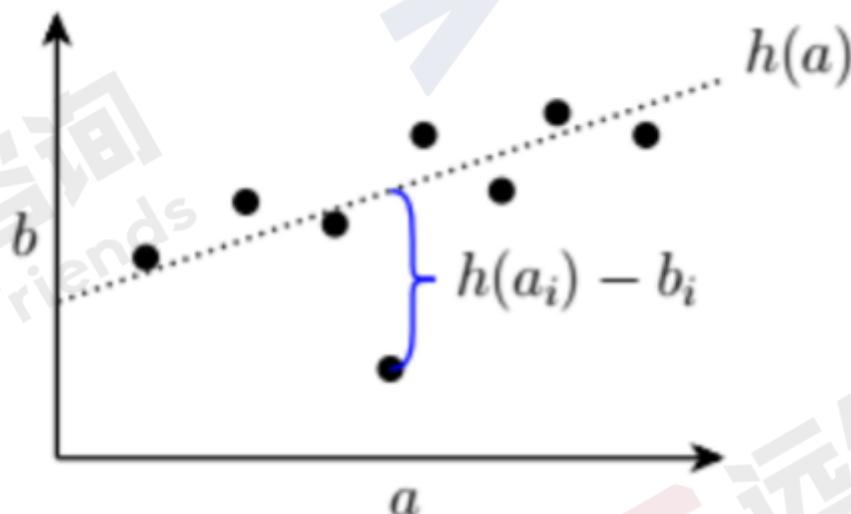
将可行域 (feasible region) 记作 C :

$$C = \{x : g_i(x) \leq 0, \forall i = 1, \dots, m\}$$

将最优解 (optimal solution) 记作 x^* :

$$\begin{aligned} & x^* \in C \\ & f(x^*) \leq f(x), \forall x \in C \end{aligned}$$

Least-squares fitting 最小二乘拟合

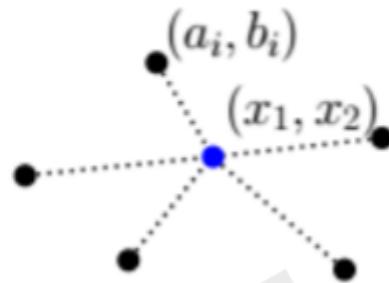


给定二维平面上的一系列点 $(a_i, b_i) i = 1, \dots, m$, 找到一条直线 $h(a) = x_1 a + x_2$ 能最优化:

$$\underset{x}{\text{minimize}} \sum_{i=1}^m (h(a_i) - b_i)^2$$

也就是使与每一个点在垂直方向上的差值的平方的和最小。

Weber point



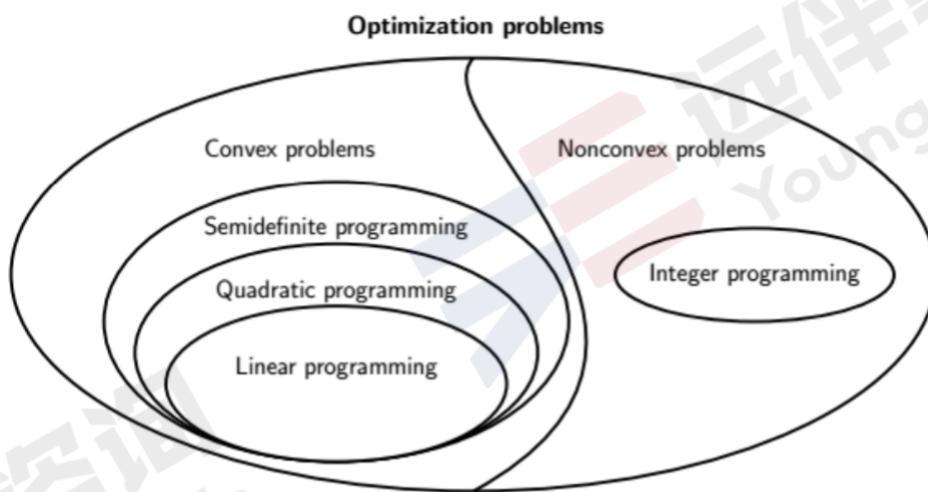
在二维平面是给定 m 个点 $(a_i, b_i) i = 1, \dots, m$, 找到一个点 (x_1, x_2) 能最优化此点和所有点的欧氏距离之和, 这个点就是 Weber point:

$$\underset{x}{\text{minimize}} \sum_{i=1}^m \sqrt{(x_1 - a_i)^2 + (x_2 - b_i)^2}$$

也可以给目标点添加限制条件, 例如目标点必须位于由 $(a_l, b_l), (a_u, b_u)$ 构成的线段作为对角线的矩形内:

$$\begin{aligned} & \underset{x}{\text{minimize}} \sum_{i=1}^m \sqrt{(x_1 - a_i)^2 + (x_2 - b_i)^2} \\ & \text{subject to } a_l \leq x_1 \leq a_u, b_l \leq x_2 \leq b_u \end{aligned}$$

Classification of optimization problems 优化问题的分类



Convex optimization problems 凸优化问题

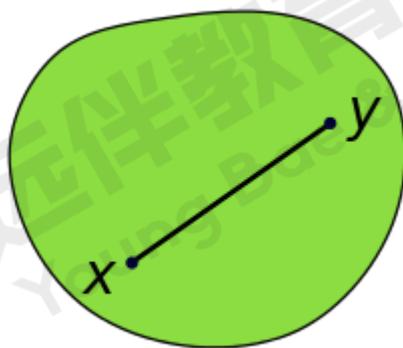
凸优化问题指的是突集 (convex set) 上的凸函数 (convex function) 的优化问题:

$$\begin{aligned} & \underset{x}{\text{minimize}} f(x) \\ & \text{subject to } x \in C \end{aligned}$$

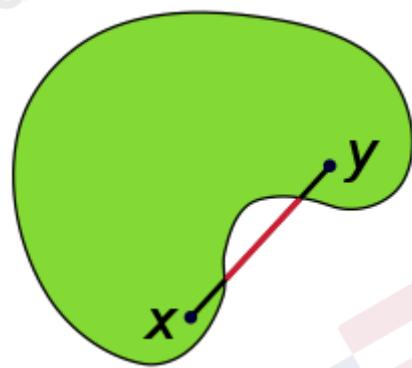
$x \in \mathbb{R}^n$, 为优化变量。 $f : \mathbb{R}^n \rightarrow \mathbb{R}$ 为凸函数, 可行域 C 为突集。

凸集

凸集 (Convex set) 是一个点集合，其中每两点之间直线上的每一点都落在该点集合中。



凸集



非凸集(凹集)

若一个集合 C 是凸集，则：

$$\forall x, y \in C \text{ and } \theta \in [0, 1] \quad \theta x + (1 - \theta)y \in C$$

由下面几种方式表示的集合 C 一定为凸集：

- 区间(Intervals): $C = \{x \in \mathbb{R}^n : l \leq x \leq u\}$ (矩阵对应元素间的不等 elementwise inequality)
- 线性不等关系: $C = \{x \in \mathbb{R}^n : Ax \leq b\}$, $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$

证明由线性不等关系形成的集合为凸集：

$$\begin{aligned} & \text{let } x, y \in \mathbb{R}^n \in C \\ \implies & Ax \leq b, Ay \leq b \\ \implies & \theta Ax \leq \theta b, (1 - \theta)Ay \leq (1 - \theta)b \\ \implies & \theta Ax \leq \theta b, (1 - \theta)Ay \leq (1 - \theta)b \\ \implies & \theta Ax + (1 - \theta)Ay \leq b \\ \implies & A(\theta x + (1 - \theta)y) \leq b \\ \implies & \theta x + (1 - \theta)y \in C \end{aligned}$$

- 线性相等关系: $C = \{x \in \mathbb{R}^n : Ax = b\}$, $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$
- 其他凸集的交集: $C = \cap_{i=1}^m C_i$, $i = 1, \dots, m$

凸函数



若一个函数 $f : \mathbb{R}^n \rightarrow \mathbb{R}$ 是凸函数，则：

$$\begin{aligned} & \forall x, y \in \mathbb{R}^n \text{ and } \theta \in [0, 1], \\ & f(\theta x + (1 - \theta)y) \leq \theta f(x) + (1 - \theta)f(y) \end{aligned}$$

- 如果 $-f$ 是凸函数, f 为凹函数。
- 如果 f 即使凸函数又是凹函数, 则为仿射函数。放射函数一定形如:

$$\begin{aligned} f(x) &= a^T x + b \\ \text{for } a &\in \mathbb{R}^n, b \in \mathbb{R} \end{aligned}$$

函数凸性的检验 (Testing for convexity)

凸函数一定处处上翘 (curve upwards)

对于输入值为标量的函数 $f : \mathbb{R} \rightarrow \mathbb{R}$, f 为凸函数等价于 $\forall x, f''(x) \geq 0$

对于输入值为向量的函数, f 为凸函数等价于:

$$\nabla_x^2 f(x) \succeq 0$$

\succeq 是正定符号 (positive definitiveness), 此处等价于;

$$\forall z \in \mathbb{R}^n \quad z^T (\nabla_x^2 f(x)) z \geq 0$$

- $\nabla_x^2 f(x)$ 是函数的黑塞矩阵 (Hessian Matrix), 记作 H .

黑塞矩阵是一个多元函数的二阶偏导数构成的方阵

$$H_{ij} = \frac{\partial^2 f(x)}{\partial x_i \partial x_j}$$

一些凸函数例子:

- 指数函数

$$f(x) = \exp(ax)$$

- 负对数函数

$$f(x) = -\log(x)$$

- 负平方根

$$f(x) = -\sqrt{x}$$

- 指数和的对数 (*Log – sum – exp*) :

$$f(x) = \log(e^{x_1} + e^{x_2} + \dots + e^{x_n})$$

- 欧式距离的平方: $f(x) = x^T x$ ($\nabla_x^2 f(x) = I$)

这里的 x 是两个向量的差

- 欧式距离: $f(x) = \|x\|_2 = \sqrt{x^T x}$
- 其他凸函数的带权和 (权非负)

$$f(x) = \sum_{i=1}^m w_i f_i(x)$$
$$w_i \geq 0, f_i \text{ convex}, \forall i = 1, \dots, m$$

- 其他凸函数中的最大函数:

$$f(x) = \max_{i=1}^m f_i(x)$$
$$f_i \text{ convex}, \forall i = 1, \dots, m$$

- 凸函数和仿射函数的复合函数

$$\text{if } f(y) \text{ convex in } y, , f(Ax - b) \text{ is convex in } x$$

一个性质:

子水平集 (sublevel sets) :

$$\text{for } f(y) \text{ convex, } C = \{x : f(x) \leq c\} \text{ is a convex set}$$

Convex optimization problems 凸优化问题

介绍完了凸优化问题中的基本概念, 现在开始介绍凸优化问题本身:

通过应用子水平集性质, 一个泛化的凸优化问题通常写成下面这种形式:

$$\begin{aligned}
& \underset{x}{\text{minimize}} \quad f(x) \\
& \text{subject to } g_i(x) \leq 0, i = 1, \dots, m \\
& \quad h_i(x) = 0, i = 1, \dots, p
\end{aligned}$$

即通过数个凸函数的子水平集来生成一个凸集作为可行域。

Property of Convex optimization problems 凸优化问题的性质

- 一个点 x , 如果其是可取的且不存在一个点 y 使得 $f(y) < f(x)$, 那么 x 就是全局最优的。
- 一个点 x , 如果其是可取的且存在 $R > 0$ 使得: 对于所有可取的 y , $\|x - y\|_2 \leq R$, $f(x) \leq f(y)$, 那么 x 就是局部最优的。

$\|A\|_2$ 是矩阵的谱范数。阵 A 的谱范数是 A 最大的奇异值或半正定矩阵 $A^* A$ 的最大特征值的平方根

$$\|A\|_2 = \sqrt{\lambda_{\max}(A^* A)}$$

其中 A^* 是 A 的共轭转置

Theorem: 对一个凸优化问题, 所有的局部最优点都是全局最优的。

简单来说, 可以理解为: 对于一个凸函数, 局部极小值就是最小值。

Example: least-squares 最小二乘

标准的最小二乘问题可以这样描述:

$$\underset{x}{\text{minimize}} \quad \|Ax - b\|_2^2$$

优化变量 $x \in \mathbb{R}^n$, 数据矩阵 $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $\|y\|_2^2$ 指的是 向量 $y \in \mathbb{R}^n$ 的 l_2 范数

$$\|y\|_2^2 = y^T y = \sum_{i=1}^m y_i^2$$

最小二乘问题是一个凸优化问题。

Example: linear programming 线性规划

标准的线性规划问题可以这样描述:

$$\begin{aligned}
& \underset{x}{\text{minimize}} \quad c^T x \\
& \text{subject to } Ax = b \\
& \quad Fx \leq g
\end{aligned}$$

优化变量 $x \in \mathbb{R}^n$, $c \in \mathbb{R}^n$ 是优化变量的系数。 $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $F \in \mathbb{R}^{p \times n}$, $g \in \mathbb{R}^p$ 是问题中的数据。

线性规划的约束条件可以是等式约束也可以是不等式约束, 只要是线性的即可。

线性规划问题是一个凸优化问题 (目标函数为仿射函数, 可行域为凸集)

Example: quadratic programming 二次规划

标准的线性规划问题可以这样描述:

$$\begin{aligned}
& \underset{x}{\text{minimize}} \quad x^T Qx + r^T x \\
& \text{subject to } Ax = b \\
& \quad Fx \leq g
\end{aligned}$$

优化变量 $x \in \mathbb{R}^n$ 。 $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $F \in \mathbb{R}^{p \times n}$, $g \in \mathbb{R}^p$ 是问题中的数据.

二次规划问题是一个凸优化问题。

非凸优化问题

标准的非凸优化问题可以这样描述:

$$\begin{aligned} & \underset{x}{\text{minimize}} \quad f(x) \\ & \text{subject to } g_i(x) \leq 0, i = 1, \dots, m \\ & \quad h_i(x) = 0, i = 1, \dots, p \end{aligned}$$

若 $f(x)$ 或者 任意一个 $g_i(x)$ 不是凸函数 或者任意一个 $h_i(x)$ 不是一个仿射函数, 以上这个问题就是一个非凸优化问题。

Example: Integer Programming 整数规划

$$\begin{aligned} & \underset{x}{\text{minimize}} \quad c^T x \\ & \text{subject to } Ax = b \\ & \quad Fx \leq g \\ & \quad x \in \mathbb{Z} \end{aligned}$$

整数规划是一个非凸优化问题, 因为可行域不是凸集.

整数规划是指规划中的变量 (全部或部分) 限制为整数, 若在线性模型中, 变量限制为整数, 则称为整数线性规划。

解决优化问题

有两类解决非凸优化问题的方法: 局部 (local) 和全局 (global)

- **Local methods:** Given some initial point x_0 .repeatedly search "nearby" points until finding a (feasible) solution \hat{x} that is better than its nearby points

- 以上这个方法解决凸优化问题和非凸优化问题的复杂度差距不大 (事实上就是把非凸优化问题当成凸优化问题解决) 。
- 但是这个方法在解决非凸优化问题时有可能无法找出最优解, 或者找到的不是最优解。

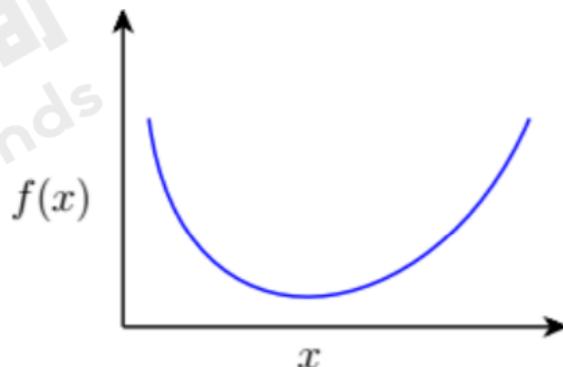
- **Global methods:** Find actual optimal solution x^* over the entire domain of feasible solution

- 以上这个方法为指数级时间复杂度, 其实就是穷举法.

这两个方法在实际就解决问题时都很常用

在二维中的标量输入函数, 最小的值的出现就是 $f'(x) = 0$, 也就是极值点。

如果函数为凸函数, 极值点就是最值点。



将这个规律泛化到多元函数中，但函数的梯度为0时，就出现(局部)最优解：

$$\nabla_x f(x) = 0$$

当函数为 $f : \mathbb{R}^n \rightarrow \mathbb{R}$ 时，梯度是一个 n 维的向量，包含了函数每个维度上的偏微分：

$$(\nabla_x f(x))_i = \frac{\partial f(x)}{\partial x_i}$$

以上这种解法仅限于无约束条件的整数规划。

如何计算 $\nabla_x f(x) = 0$ 的解：

1. 直接解法：令梯度向量的每个维度为 0，解方程。

例如二次规划：

$$f(x) = x^T Q x + r^T x$$

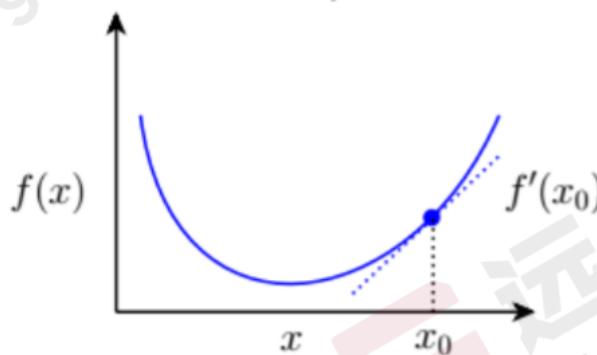
$$\Rightarrow \nabla_x f(x) = 2Qx + r$$

$$\Rightarrow x^* = -\frac{1}{2}Q^{-1}r$$

$$Q \in \mathbb{R}^{n \times n}, r \in \mathbb{R}^n$$

2. 梯度下降法 Gradient descent: Take small steps in the direction of the current (negative) gradient

- Intuition: negative gradient of a function always points "downhill" (actually points downhill in the steepest possible direction, for multivariate function)



梯度下降法是一种穷举法，在接近最优解的区间内向逼近最小值的方向不断微小步进计算梯度，直到找到梯度为0的点。

Repeat: $x \leftarrow x - \alpha \nabla_x f(x)$ where $\alpha \in \mathbb{R} > 0$ is a *step size*

3. Newton's method: Use root-finding algorithm to find solution to (no linear) equation

$$\nabla_x f(x) = 0$$

在标量输入函数中，使用下面这个式子作为步进，不断迭代直到 $f'(x) = 0$ ：

$$x \leftarrow x - \frac{f(x)}{f'(x)}$$

扩展到多元函数，使用下面这个式子作为步进，不断迭代直到 $\nabla_x f(x) = 0$ ：

$$x \leftarrow x - (\nabla_x^2 f(x))^{-1} \nabla_x f(x)$$

这样的算法每一步比梯度下降开销更大，但是能用更少的步数(更快)逼近最优解。

4. barrier method: Approximate problem via unconstrained optimization

$$\underset{x}{\text{minimize}} \quad f(x) - t \sum_{i=1}^m \log(-g_i(x))$$

当t趋向0时，这个式子就趋向于标准的优化问题。

- 输出不稳定，需要谨慎地使用
- 相等约束条件需要额外单独考虑

Practically solving optimization problems

在实际操作中使用计算机解决最优化问题只需要使用一些现成的库，例如：

- CVX (MATLAB)
- YALMIP (MATLAB)
- AMPL (custom language)
- GAMS (custom language)

Lec 5: learning

什么是学习 (learning) ?

Learning is any process by which a system improves performance from experience

- Herbert

Simon

什么是要通过学习 (learning) 解决的任务？

- Classification 分类

Assign object/event to one of a given finite set of categories.

将给定的对象/事件归入有限的类别集合中的一个

- Medical diagnosis
- Credit card applications or transactions
- Fraud detection in e-commerce
- Worm detection in network packets

- Problem solving / planning / control 问题解决，计划，(系统) 控制

Performing actions in an environment in order to achieve a goal

在特定环境中执行行为以 达到一个目标

- Solving calculus problems
- Playing checkers, chess, or backgammon
- Driving a car or a jeep

Inductive Learning Hypothesis 归纳学习假说

- Any function that is found to approximate the target concept well on a sufficiently large set of training examples will also approximate the target function well on unobserved examples.

一个函数，如果在一个足够大的训练集上良好地逼近目标概念，那么我们认为它在尚未被观测到的(未知)的样本上也能良好地逼近目标函数。

- Assumes that the training and test examples are drawn independently from the same underlying distribution

假设训练集和测试集合是被互不干扰地从同一个基础分布中抽取出来的。

以上的假设是无法被证明的，除非我们对目标概念(target concept)进行额外的假设并且“在未观察到的例子上很好地逼近目标函数”在具体问题中被恰当地定义。

评估分类结果 Evaluation of Classification Learning

- 分类准确率 (用百分比表示的正确率)
 - 这个准确率基于一个与训练集独立的测试集合
- 训练花费的时间 (efficiency of training algorithm)
- 测试花费的时间 (efficiency of subsequent classification)

什么是模式 (Pattern) 和类别 (Category)

Pattern: opposite to chaos; it is an entity, object, process or event, vaguely defined, that can be given a name of "label"

例如：

- A fingerprint image
- A human face
- A speech signal

每个模式可以被归到一个类别中，每个模式有自己的属性(attributes)，或者叫特征(features)。一个已经被分类的模式可以被打上一个标签 (labeled)，具有相同标签的模式就属于同一个类别。根据分类标准的不同，一个模式所属的类别也不同。

(图像)模式分类的目标：

Observing some labeled pixels, we wish to assign a label to each new (unlabeled) pixel

通过学习已经被打上标签的像素组合模式，将未被分类的区域中的像素打上标签。

Components of Learning System 一个学习系统的组成部分

- 传感器以及预处理 Sensors and preprocessing
- 特征提取 Feature extraction
- 分类器 Classifier
- 训练(Training): Provides some useful information for supervised learning
- 学习算法(Learning algorithm): Create classifier from training data (labeled samples)

Features 特征

Feature is any distinctive aspect, quality or characteristic

一个模式 (对象) 的特征是任意的具有独特性的方面，品质或特征

特征可以是象征性的 (symbolic)，例如颜色，也可以是数值 (numeric)，例如长度。

一些定义与概念：

- The combination of d-dimensional feature is represented as a d-dimensional column vector called feature vector.

一个模式的 d 个特征可以组合成一个 d 维的列向量，也就是特征向量。

- The d -dimensional space defined by the feature vector is called the feature space.

一个通过 d 维特征向量定义的 d 维空间被称作特征空间。

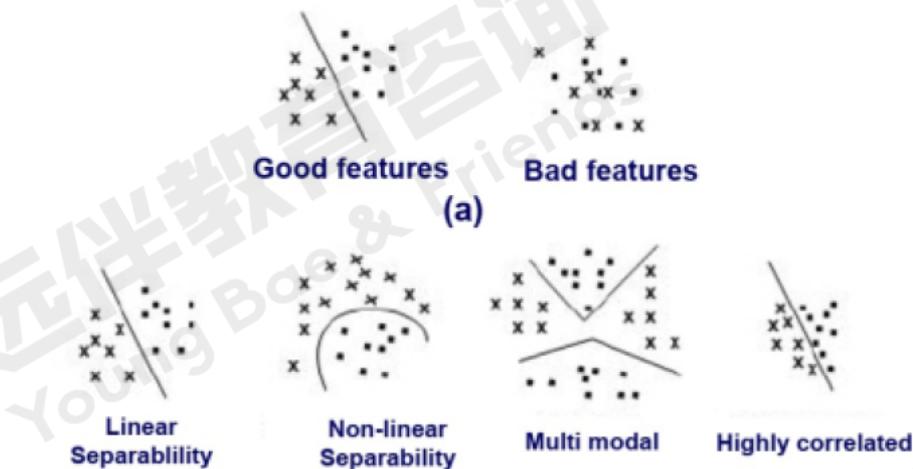
- Objects are represented as points in feature space. The representation is called a scatter plot

对象在特征空间中可以用一个特征向量表示。通过这样的表示法表示一组对象被称作一个散点图 (scatter plot)。

Good/Bad Features & Classification

一个特征向量优秀与否取决于其能否将来自不同类别的对象区分开

- Examples from the same class should have similar feature values
- Examples from the different classes have different feature values

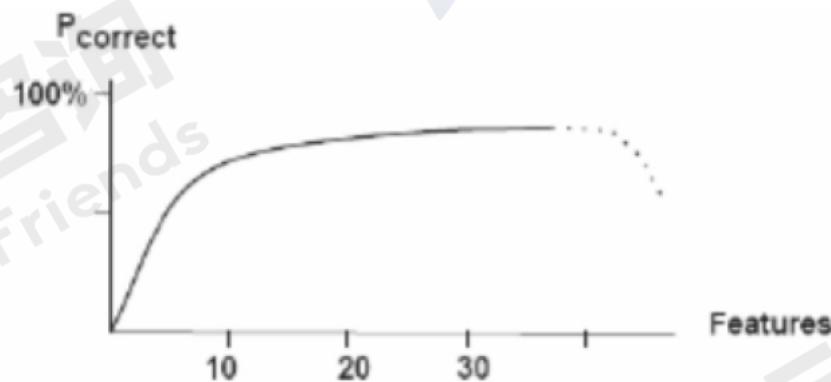


Feature Dimension

特征向量的维度数量就是从一个模式中提取的特征的数量，从常识出发，提取的特征越多，分类就越细。但是分类的细化会导致样本的分布发生变化（每个类别有多少样本），此时就需要收集更多样本让样本的分布符合预期。这就是 **The curse of dimensionality** 维度的诅咒：

The curse of dimensionality: more examples are required to keep the example density unchanged when the space is extended to the high dimension space.

当特征向量的维度不断增长，分类的准确率在超过某个临界值之后就不会再明显增长，甚至有可能下降，这种现象被称作 **Peaking phenomena**



| Adding features may actually degrade the performance of a classifier

Overfitting and underfitting 过拟合与欠拟合

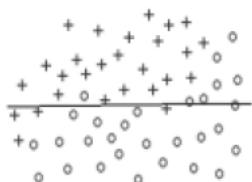
首先介绍几个概念:

- **Bias:** the expected predict error between the predict values and the true values

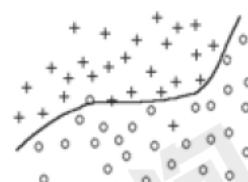
预测值与实际值之间的偏差，或者说允许最大偏差。

- **Variance:** 预测值的方差

Bias 是在拟合前给出的一个参数，表示拟合出的曲线上的点与实际点所允许的差值。如果这个参数设置的越大，拟合过程越简单，拟合结果越粗糙，但是太大会导致欠拟合 (underfitting)，即拟合出的曲线不能很好的表示点的分布趋势。反之，这个参数设置的越小，拟合的过程就越复杂，拟合的结果就越准确，但是拟合结果更容易被极小概率情况干扰，出现过拟合 (Overfitting)。



underfitting



good fit



overfitting

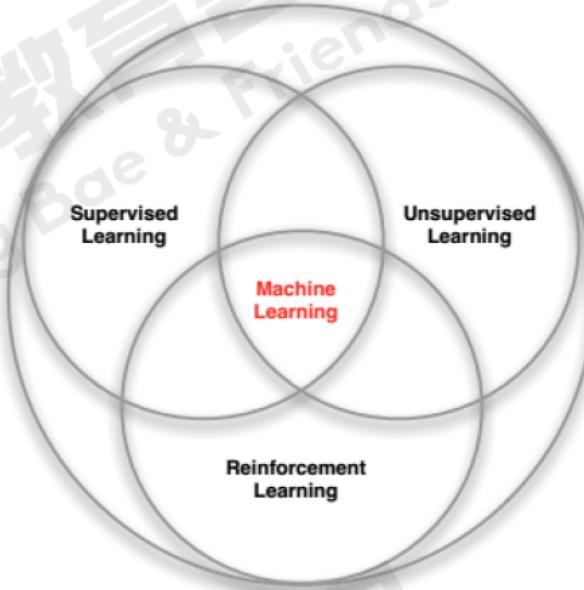
设计学习系统的工作流程 Cycle of Design

一次完整的模型迭代过程如下:

- Data collection 数据收集
数据集要兼顾充足性(sufficient)和代表性(representative)
- Feature Choice 选定特征
- Model Choice 选定模型
模型的选定会影响性能
- Training 训练
需要确定训练的流程
- Evaluation 评估准确率
- Computational Complexity 评估计算复杂度(性能)
权衡性能和准确度，通常二者不可兼得。

Type of Learning

- Supervised Learning 监督学习
- Unsupervised Learning 无监督学习
- Reinforcement Learning 强化学习



Characteristics of Reinforcement Learning 强化学习的特点

- There is no supervisor, only a reward signal 无监督，只有奖励信号。
- Feedback is delayed , not instantaneous 延迟反馈，非实时。
- Time really matters
- Agent's actions affect the subsequent data it receives

Lec 6: Feature Selection

Features and Patterns

Feature , Feature vector, Feature space, Scatter plot 等相关定义请参考上一章(Lec 5)的 Features 和 Feature Dimension 小节，此处不再赘述。

Pattern is a composite of traits or features corresponding to characteristics of an object or population

- In classification; a pattern is a pair of feature vector and label

一个模式就是一种特征的组合方式。例如我们现在从人身上选取身高，体重这两个两个象征性的特征(高与矮，胖与瘦)，这两个特征就组合出了四种模式：高瘦，高胖，矮瘦，矮胖。

Good features are:

- Representative: provide a concise description
- Characteristic: different values for different classes, and almost identical values for very similar objects
- Interpretable: easily translate into object characteristics used by human experts
- Suitable: natural choice for the task at hand
- Independent: dependent features are redundant

The performance of a classifier depends on the interrelationship between:

- sample sizes
- number of features
- classifier complexity

关于 curse of dimensionality 和 Peaking Phenomena, 请参考上一章 (Lec 5) 的 curse of dimensionality 小节，此处不再赘述。

一些样本和特征之间关系的实际例子:

- Face recognition application 人脸识别
 - For 1024*768 images, the number of features will be 786432 !
- Bio-informatics applications (gene and micro array data)
 - Few samples (about 100) with high dimension (6000 – 60000) (表示一种大分子需要的特征向量很多)
- Text categorization application
 - In a 50000 words vocabulary language, each document is represented by a 50000-dimensional vector

从上面这些例子可以看出，在解决实际问题时，特征向量的维度是相当高的，这会导致 **curse of dimensionality**

的出现，并且会极大的提高计算复杂度，甚至出现当前算力很难求解问题的情况。因此我们需要在尽量少的影响特征向量的分类效果的情况下减少特征向量的维度。这就是 **Dimensional reduction (selection or extraction)**

此外，样本集的容量也是影响计算复杂度的一个重要因素，有时我们也需要适当的削减样本集的容量，这就是 **Data reduction**

当问题的解决因为数据过于庞大而遇困难时，就需要 **Data reduction** 和 **Dimensional reduction** 缩小数据规模。

Data Reduction 数据缩减

Data Reduction goal: Obtain a reduced representation of the data set that is much smaller in volume but yet produce the same (or almost the same) analytical results.

数据缩减的目标就是在基本不影响数据集的分析结果的情况下缩小数据集的体积。

Data reduction methods:

- Regression 回归
 - | Data are modeled to fit a determined model (e.g. line or AR)
对数据建模以使其符合某个确定的数学模型，然后用这个数学模型取代数据集。
- Sufficient Statistics 充分统计量
 - | A function of the data that maintains all the statistical information of the original population
- Histograms 直方图
 - Divide data into buckets and store average (sum) for each bucket
 - Partitioning rules: equal-width, equal-frequency, equal-variance, etc
- Clustering 聚类
 - Partition data set into clusters based on similarity, and store cluster representation only |
◦ Clustering methods will be discussed later
- Sampling 抽样
 - | obtaining small samples to represent the whole data set D

Sampling 抽样

Simple Random Sampling 简单随机抽样

抽到每个总样本中每一项的概率相同。

Sampling without replacement 不放回抽样

一旦一项被抽取，其不会被放回到总样本中。因此同一个对象不会被选取超过一次。

Sampling with replacement 有放回抽样

一旦一项被抽取，其会被放回到总样本中。因此同一个对象可以被选取超过一次。

Stratified sampling 分层抽样

Grouping (split) population samples into relatively homogeneous subgroups. Then, draw random samples from each partition according to its size

首先将总样本集分为均匀子群，然后根据每个子群的大小决定从这个子群中抽取的样本数量。

Dimensionality Reduction 降维

为了模式表示和分类其设计的简化以及数据可视化的需要，我们需要将高维的特征向量降维（如果需要可视化需要降到二维或者三维）

降维有两种方法：

Feature Selection: Select the best subset from a given feature set

Feature Extraction: Create new features based on the original feature set and transforms are usually involved.

Feature Selection

Problem definition: Select the best subset from a given feature set:

$$X = [x_1, x_2, \dots, x_d] \rightarrow X' = [x_{i_1}, x_{i_2}, \dots, x_{i_m}] \\ m \leq d, \text{ usually}$$

从 d 维特征向量中选取 m 维 向量，一共有 C_d^m 种选取方法。

一种视角：

- Univariate method 单变量特征选择
一次考虑一个特征是否应该被选取。
- Multivariate method
将多个特征放在一个子集里面一同考虑。

另外一种视角：

- Filter method
Ranks features subsets independently of the classifier.
- Wrapper method
Uses a classifier to assess features subsets
- Embedded
Feature selection is part of the training procedure of a classifier

Univariate Feature Selection

想要决定某一个特征是否应该被选取，我们需要定义 Criterion of Feature Selection, 也就是衡量一个特征是否应该被选取的标准。

$$\text{Criteria} = \text{Significant difference} * \text{Independence}$$

- **Significant difference:** Pattern separability on individual candidate features.
- **Independence:** Non-correlation between candidate feature and already-selected features

衡量一个特征是否应该被选取，可以从信息论入手，一个特征包含的信息量越多，则越应该被选取：

Information Measurement

Information Measure: Consider a symbol x with an occurrence probability p , its info content is:

$$I = \log \frac{1}{p(x)} = -\log p(x)$$

| $p(x)$

- 在一个集合中，一个元素出现的概率越小，其包含的信息量越多。
- 在一个集合中，一个元素的出现概率某种程度上和它的不确定性相关。

Information Entropy 信息熵

The Entropy is defined as the average information content per symbol of the source. The Entropy, H , can be expressed as follows

$$H = E[-\log_2(P(X))] = -\sum_{i=1}^m p_i \log_2 p_i \text{ bits}$$

从中间这个表达式理解，熵实际是对随机变量的比特量的数学期望，代表的是整个系统的平均信息量。

$-\log_2(P(X))$ 表示的是用二进制编码代表一个事件所需的最少位数(想要用二进制编码n种情况需要 $\log_2 n$ 个二进制位)，这个算式就是上面提到的 **Information Measurement**。

例如 $P(x_i) = \frac{1}{4}$, $-\log_2(\frac{1}{4}) = 2$ bits, 概率为 $\frac{1}{4}$ 意味着样本集中可能有4个等可能事件，此事件是其中一个。想要用二进制编码代表四个事件至少需要两个二进制位(00 01 10 11)。

从这个算式可以看出，每种情况出现概率越小，这个数值越大。概率越小意味着分母越大，分子越小，需要用二进制编码代表的情况越多，编码所需的位数就越多，携带的信息量就越大。

- 信息熵是一个基于出现概率 (occurrence probability) 的方程。
 - P为X的概率质量函数
 - E为期望函数
 - p_i 是质量分布方程在 $x = x_i$ 处的值
- 信息熵在集合中的所有的元素出现的概率相同时达到最大。

Information Content 信息内容

信息的内容量就可以通过计算信息熵来衡量，下面是两个 8×8 矩阵：

0	0	0	0	0	0	0	0	0	1	1	0	1	0	1	0
0	0	0	0	0	0	0	1	1	0	1	0	1	0	0	1
0	0	0	0	0	0	0	0	1	1	0	1	0	1	1	0
0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0
0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	1
0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	1
0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	0
0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0

左边的矩阵是一个稀疏矩阵, $P(X = 0) = \frac{63}{64}$ $P(X = 1) = \frac{1}{64}$

其信息熵为 $-\frac{63}{64} \log_2 \frac{63}{64} + -\frac{1}{64} \log_2 \frac{1}{64} = 0.116 \text{ bits}$

也就是说左边的矩阵中平均每一位携带 0.116 比特信息, 一共携带 $64 \times 0.116 = 7.424$ 比特信息

右边的矩阵中0和1各占一半 $P(X = 0) = \frac{1}{2}$ $P(X = 1) = \frac{1}{2}$

其信息熵为 $-\frac{1}{2} \log_2 \frac{1}{2} + -\frac{1}{2} \log_2 \frac{1}{2} = 1 \text{ bits}$

也就是说右边的矩阵中平均每一位携带 1 比特信息, 一共携带 $64 \times 1 = 64$ 比特信息

Information Gain

令 p_i 为总样本集 D 中的一个样本属于类别 C_i 的概率

预期在总样本集 D 中分类一个样本所需的信息熵:

$$H(C) = - \sum_{c \in C} p(c) \log_2 p_c$$

Conditional entropy using the attribute A to split D into $|A|$ partitions:

$$H(C|A) = - \sum_{a \in A} p(a) \sum_{c \in C} p(a|c) \log p(a|c)$$

Information gained by attribute A:

$$IG(A) = H(C) - H(C|A)$$

Gain ratio

Information gain measure is biased towards attributes with a large number of unique values

Gain ration overcomes to this problem (normalization to information gain)

$$GR(A) = IG(A)/H(A)$$

where

$$H(A) = - \sum_{a \in A} p(a) \log p(a)$$

Other methods: Multivariate Feature Selection

Generation Methods

- Complete/exhaustive
 - Examine all combinations of feature subset (Too expensive if feature space is large).
 - Optimal subset is achievable.
- Heuristic

- Selection is directed under certain guideline
- Uses incremental generation of subsets, often.
- Possibility of miss out high importance features.
- Random
 - no pre-defined way to select feature candidate. pick feature at random.
 - optimal subset depend on the number of tries
 - require more user-defined input parameters.
 - result optimality will depend on how these parameters are defined.

Lec 7: Classification: Introduction & Quality Assessment

Classification: Predicts categorical class labels (discrete or nominal).

在这一章中，我们只讨论 binary classification



| Framework of classification

Training & Inference 训练和推理

人工智能模型的构建过程 (Model construction) 被称为训练 (Training):

- Each sample is assumed to belong to a predefined class, as determined by the class label
- The set of samples used for model construction is called training set (训练集)
- The model is represented as classification rules, decision trees, probabilistic model, mathematical formula and etc.

人工智能模型使用的过程 (Model usage) 被称为推理 (Inference)

- For classifying future or unknown objects
- Estimate accuracy of the model:
 - Accuracy rate(准确率): the percentage of test set samples that are correctly classified by the model
 - Test set is independent of training set, otherwise over-fitting will occur (测试集应该与训练集互不相关，否则会出现过拟合)

如果一个分类模型经过评估，其准确率是可接受的，我们就可以用其推理未知对象的类型了。

Evaluating Classification Methods 评价分类结果的方法

- Performance 性能
 - classifier performance: predicting class label compared with ground truth
 - **True Positive (TP), True Negative (TN), False Positive (FP), False Negative (FN)**
- Complexity 复杂度
 - Time to construct the model (training time)
 - Time to use the model (classification time)
 - operations to train / use the model
 - number of parameters in the model
- Robustness 鲁棒性

- handling noise and missing values
- handling incorrect training data

Confusion Matrix 混淆矩阵

混淆矩阵描述了一个分类器的测试结果。因为本章中只探讨 binary classification，即只分两类，混淆矩阵就是一个 2×2 的矩阵。

二元分类的两类可以用 negative 和 positive 抽象描述，测试的结果中会有如下四种类型的结果：(1) 模型预测某对象为 Positive，其实际也为 Positive (**TP**)。(2) 模型预测某对象为 Negative，其实际也为 Negative (**TN**)。(3) 模型预测某对象为 Positive，而其实际为 Negative (**FP**) (4) 模型预测某对象为 Negative，而其实际为 Positive (**FN**)

TP, FP, FN and TN form a matrix named confusion matrix. There are many parameters can be calculated according to the confusion matrix, which evaluate the resulting system from different perspectives.

		True condition		Prevalence = $\frac{\sum \text{Condition positive}}{\sum \text{Total population}}$	Accuracy (ACC) = $\frac{\sum \text{True positive} + \sum \text{True negative}}{\sum \text{Total population}}$
Total population	Condition positive	Condition negative			
Predicted condition	Predicted condition positive	True positive	False positive , Type I error	Positive predictive value (PPV), Precision = $\frac{\sum \text{True positive}}{\sum \text{Predicted condition positive}}$	False discovery rate (FDR) = $\frac{\sum \text{False positive}}{\sum \text{Predicted condition positive}}$
	Predicted condition negative	False negative , Type II error	True negative	False omission rate (FOR) = $\frac{\sum \text{False negative}}{\sum \text{Predicted condition negative}}$	Negative predictive value (NPV) = $\frac{\sum \text{True negative}}{\sum \text{Predicted condition negative}}$
		True positive rate (TPR), Recall, Sensitivity, probability of detection, Power = $\frac{\sum \text{True positive}}{\sum \text{Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm $= \frac{\sum \text{False positive}}{\sum \text{Condition negative}}$	Positive likelihood ratio (LR+) $= \frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) = $\frac{\text{LR+}}{\text{LR-}}$
		False negative rate (FNR), Miss rate $= \frac{\sum \text{False negative}}{\sum \text{Condition positive}}$	Specificity (SPC), Selectivity, True negative rate (TNR) $= \frac{\sum \text{True negative}}{\sum \text{Condition negative}}$	Negative likelihood ratio (LR-) $= \frac{\text{FNR}}{\text{TNR}}$	

TP, TN, FP, FN 也代指每种情况出现的数量，基于这四个数字，我们可以计算出一些用来评估一个模型的参数：

Recall (Sensitivity)	True Positive Rate	(1)
Fall-out	False Positive Rate	(3)
Miss rate	False Negative Rate	(2)
Specificity (Selectivity)	True Negative Rate	(4)

以上四种参数的计算方法如下：

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{\text{TP}}{n^+} \quad (1)$$

$$\text{fallout} = \frac{\text{FP}}{\text{TN} + \text{FP}} = \frac{\text{FP}}{n^-} \quad (3)$$

$$\text{miss} = \frac{\text{FN}}{\text{TP} + \text{FN}} = \frac{\text{FN}}{n^+} \quad (2)$$

$$\text{specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} = \frac{\text{TN}}{n^-} \quad (4)$$

除了以上4个参数，还有一些通用参数：

$$precision = \frac{TP}{TP + FP} \quad (5)$$

$$accuracy = \frac{TN + TP}{TP + TN + FP + FN} = \frac{TN + TP}{n} \quad (6)$$

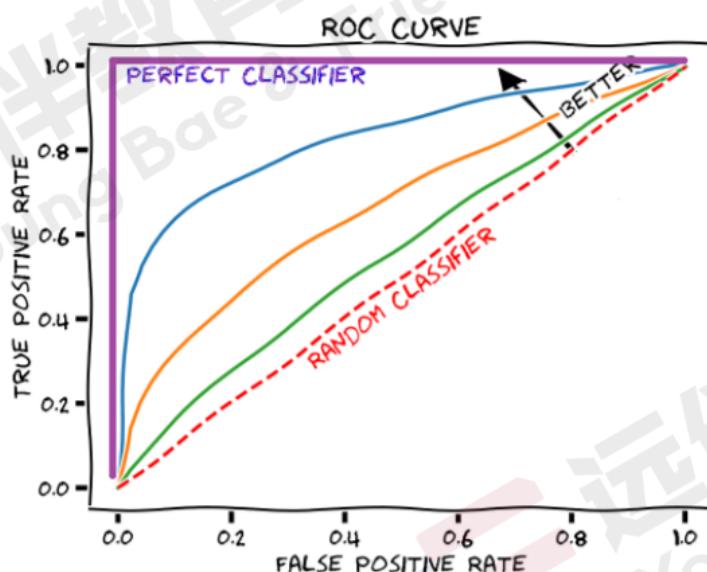
一个好的衡量模型的指标应当综合且平衡的考虑模型各个方面的指标, 例如 $F_1 Score$ 这个参数:

$$F_1 score = \frac{2 \cdot TP}{2 \cdot TP + FP + FN} = 2 \cdot \frac{precision \cdot recall}{precision + recall} \quad (7)$$

Receiver Operation Characteristics (ROC) 接收者操作特征曲线

ROC是信号检测理论中的一个概念, 用在模型训练中是为了寻找最佳阈值 (threshold values)。

| 在二元分类中, 决定negative 和 positive 的分界阈值会极大的影响模型的准确率。



One of the measurement for ROC test is Area under Curve (AUC)

| 在比较不同的分类模型时, 可以将每个模型的ROC曲线都画出来, 比较曲线下面积 (AUC) 作为模型优劣的指标, 简单说: AUC值越大的分类器, 正确率越高。

Cross Validation 交叉验证

在理想情况下, 我们使用的训练集中的数据分布应该和在实际应用场景中接收的数据有相同的分布, 但是这一点无法做到。

那么为了测试这种理想情况下的模型性能, 我们可以使用本次训练集的一部分作为另外一次训练结果的验证集(validation set), 这种验证模型性能的方法被称为交叉验证(Cross Validation)。

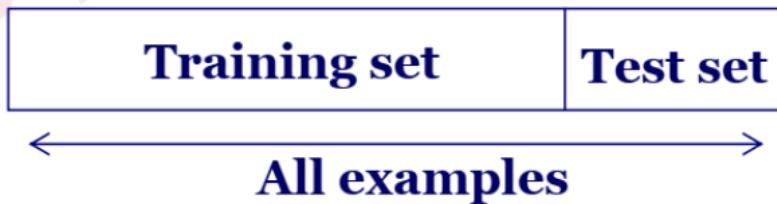
- By performing cross validation, we expect the validation set must be a good representative for the whole data

Data Partitioning Methods 数据集分割方法

由于在每次训练-测试的流程中，训练集和测试集必须互相独立，我们必须在开始训练模型前分割总数据集，一部分用作训练，一部分留作测试。

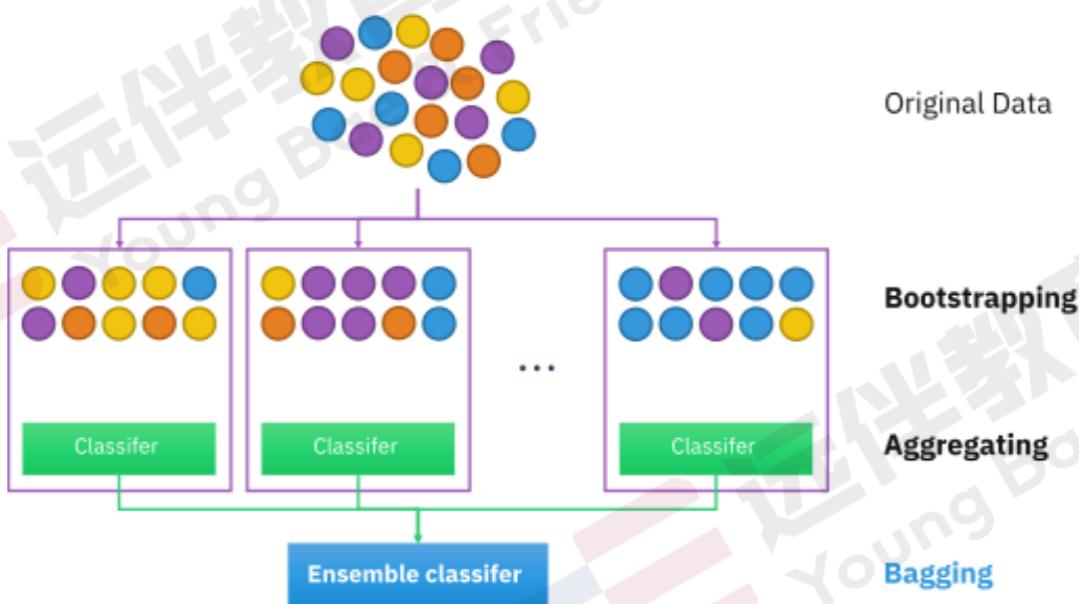
Random Sampling

最简单的分割训练集和测试集的办法就是将数据集随机抽样分割成指定比例的两份，此时我们假设绝对的随机选取可以使得测试机和训练集中的数据分布相同。



这种验证方法并不属于交叉验证，因为训练集和测试集并无交叉。

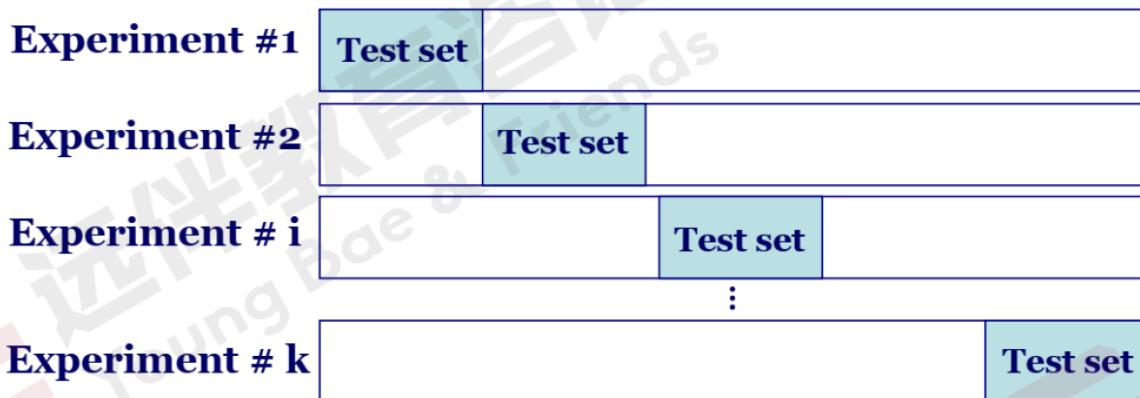
Bootstrap 自助法



Bootstrap Sampling 在统计学上是一种用于获取数据集分布的方法。例如我想要获取一万人的平均身高，普通的方法是测量全部人身高再计算平均值，但是如果使用自助法，则应该一次随机抽取 10 人，计算这 10 人的平均身高，重复 n 次，然后这 n 次结果的平均值就可以作为近似的一万人的平均身高。显而易见的，n 越大，结果越可信。

- resample with replacement n sample of original data as training set.
- Some numbers in the original sample may be included several times in the bootstrap sample
- Can be used to estimate data distribution

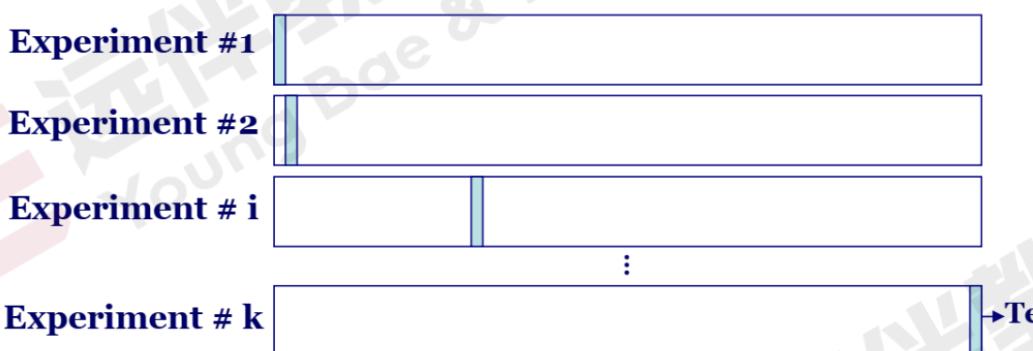
k-fold Cross Validation k 折交叉验证



k-fold (k is usually set to 5, 10 and other numbers)

- Randomly partition the data into k mutually exclusive subsets, each approximately equal size
将原始数据集分成等大的 k 份。 ($D \rightarrow D_1, D_2, \dots, D_k$)
- At i -th iteration, use D_i as test set and others as training set
在第 i 次迭代中，使用 D_i 最为测试集，其他的作为训练集。
- The mean of measures obtained in iterations used as the output of performance measure
每个子集都做过测试机后，用所有测试结果的平均值作为模型最终的性能指标。

Leave-one-out Cross Validation 留一交叉验证



留一交叉验证可以看作k 折交叉验证的变体，只不过每次选取的测试集是单个pattern (feature+label)，这种验证法适合在数据集很小的时候使用。

Three-way Data Splits

If model selection and true error estimates are to be computed simultaneously, the data needs to be divided into three disjoint sets.

- **Training set:** a set of examples used for learning: to fit the parameters of the classifier
- **Validation set:** a set of examples used to tune the parameters of a classifier
- **Test set:** a set of examples used only to assess the performance of a fully-trained classifier

测试集和验证集一般不用分开，甚至可以使用同一个子集，但是在 Three-way Data Splits 中，两者必须独立，原因如下：

- The error rate estimate of the final model on validation data will be biased(smaller than the true error rate) since the validation set is used to select the final model
- After assessing the final model with the test set, YOU MUST NOT tune the model any further

Lec 8 Discriminant Functions

Linearly Separable 线性可分

在理想情况下，特征足够好时（参考前文的 what is good feature），一个简单的线性方程就能够解决分类的问题。

但是样本之间的关系不一定是线性关系。也就是说，只使用线性方程有时不能完成分类的需求。

在这一小节中，我们讨论如何使用线性判别函数来分割特征空间，以及在特征向量的分布很难使用线性判别函数来分类时，如何使用特征提取来优化样本的分布。

Logic Function by Linear Combination

最基本的线性方程有三种 AND 且, OR 或, XOR 异或。

Linear Discriminant Functions (LDF) 线性判别函数

Definition: LDF is a function that is a linear combination of the components of x :

$$g(x) = w^T x + x_0$$

where w is the weight vector and w_0 the bias, or threshold weight

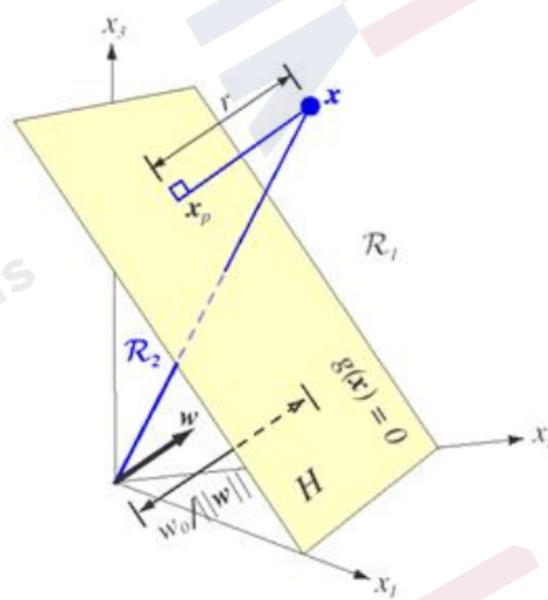
The formulae could also be determined by objective theories, in which case, the system could be considered as a rule-based system

对于不同数量的分类要求，线性判别函数有以下三种情况：



在二元分类中一个线性判别函数就足以分类，而在 general case 中，有多少种类别就（至少）需要多少个线性判别函数。

线性判别函数的本质就是代表特征向量的点之间的分界，在二维特征空间中，其是一条线，在三维特征空间中其是一个平面。如此向更高维度推广。



Multiple-class Problem

Suppose we have an n -classes classification problem, and we want to separate them with linear discriminant functions.

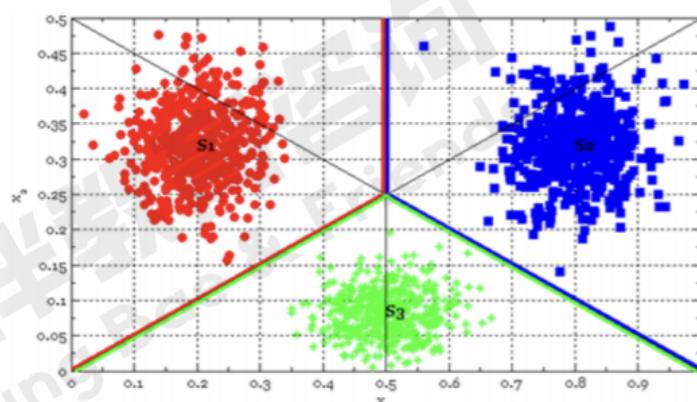
将多类问题分成 N 个二类分类问题，训练 N 个二类分类器，对第 i 个类来说，所有属于第 i 个类的样本为正 (positive) 样本，其他样本为负 (negative) 样本，每个二类分类器将属于 i 类的样本从其他类中分离出来。

how to use discriminant function in Multiple-class Problem?

- Linear machines (one versus one), Pairwise linearly separation.
- Completely linearly separation (one versus the rest).

Case 1: Linear Machine

假设我们现在需要将特征空间中的点分三类，我们可以先求出三个类别两两之间的线性判别函数。

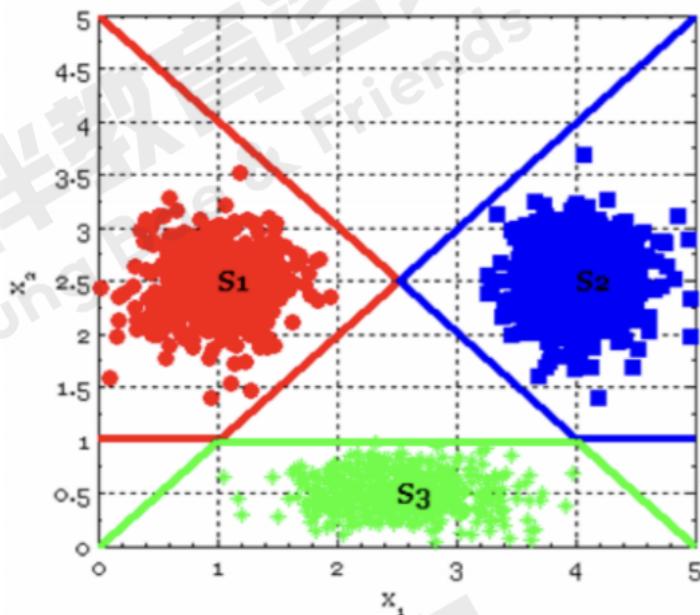


也就是每次求取方程是 OvO (one versus one) 一对一

假设一共有 n 个类别，OvO 需要 C_n^2 个分类器，也就是有 $\frac{n(n-1)}{2}$ 个线性方程。判断一个样本属于哪一类时，需要将特征向量其分别带入这 C_n^2 个线性方程，每次判断这个特征向量落在哪个线性方程所分出的两类中的哪一类，最后统计每个样本被分到每一类的次数，一个样本被分到哪一个类次数最多，这个样本就属于哪一类。

Case 2: Completely Linearly Separation

换一种方法，每次求取一个类别相对于其他所有类别的线性判别函数。OvM (one versus many)



OvM 需要 n 个分类器，理论上OvM法在判定一个点属于哪一类时最差需要判定 n 次。

容易看出，OvM 只需训练 N 个分类器，而OvO需训练 $\frac{N(N-1)}{2}$ 个分类器，因此，OvO的存储开销和测试时间开销通常比OvM 更大。但在训练时，OvM 的每个分类器均使用全部训练样例，而OvO 的每个分类器仅用到两个类的样例，因此，在类别很多时，OvO的训练时间开销通常比OvM更小。至于预测性能，则取决于具体的数据分布，在多数情形下两者差不多。

MvM Methods

Make use of propositional logic (命题逻辑)，classes can be re-defined into several groups where the determination of each class can be regarded as the results of combining classification results in each group.

在多对多分类时，我们可以各将一些类别划分进不同的组（group）里，每个类别的决定标准可以视作将在每个组中的分类结果组合的结果。

若一个组中至少有一个正例，则该包被标记为正（positive），若一个包中所有示例都是反例，则该包被标记为反（negative）。通过对训练包的学习，希望学习系统尽可能正确地对训练集之外的包的概念标记进行预测。

一种最常用的MvM技术就是纠错输出码（Error Correcting Output Codes，简称 ECOC）

The ECOC matrix could also be interpreted as a Directed Acyclic Graph (DAG 有向无环图).

Linear Discriminant Functions

回到求线性判别函数的问题，求解线性判别函数的关键其实就是求解权重向量 w

Key problem: How to create the discriminant functions for each class (how to obtain w)?

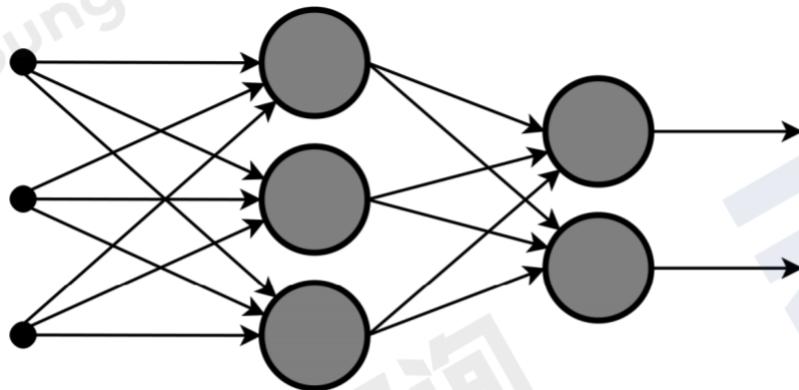
Many methods exist for this purpose, such as:

- Error Minimization Methods
- Least Mean Squared Error Method
- Sum of Squared Error Method
- Ho - Kashyap Method

Perceptrons 神经元

一个神经元的本质就是一个激活方程，输入数据集，输出一个0到1之间的数值。当这个输出值大于某个阈值时，这个神经元就会激活与之相连的下一个神经元（将输出传给与之相连的下一层神经元）。

The boundaries between classes are not necessary linear but can be approximate as a combination of linear functions.



- Could be single layer or multiple layer
- There is a threshold process after the output of each neuron, which is named as activation function

Deep Learning

- Perceptron is a basic unit of Neural Network.
- Neural Networks with multiple layers are considered as deep learning systems.
- The extension along space axis results in Convolutional Neural Network (with some simplification)

简单来说，将神经网络沿空间轴扩展就能得到卷积神经网络。

- The extension along time axis results in Recurrent Neural Network (with more controls between time slices)

简单来说，将神经网络沿时间扩展就能得到循环神经网络。

Linear Functions as Features

We expect the data can be linear separable direct which never happens. So we hope we could find features to do that. We now discuss about using linear transform as feature extractor.

All mentioned methods could be associated with the following formula:

$$X = W^T H + B$$

特征提取 (feature extraction) 就是一个降维的过程

使用线性组合方式 (给每个维度一个权重) 将特征向量的多个维度组合成一个维度放进新的特征向量是最常见的方法。

LDA

LDA (Linear Discriminant Analysis): Project the samples to a line such that the projection of samples are linearly separable.

LDA (线性判别式分析)

线性鉴别分析的基本思想是将高维的模式样本投影到最佳鉴别矢量空间，以达到抽取分类信息和压缩特征空间维数的效果，投影后保证模式样本在新的子空间有最大的类间距离和最小的类内距离，即模式在该空间中有最佳的可分离性。因此，它是一种有效的特征抽取方法。使用这种方法能够使投影后模式样本的类间散布矩阵最大，并且同时类内散布矩阵最小。就是说，它能够保证投影后模式样本在新的空间中有最小的类内距离和最大的类间距离，即模式在该空间中有最佳的可分离性。

PCA

PCA (Principal component analysis): Represent the sample with a set of basis (in linear algebra) such that the most significant differences between classes can be shown.

PCA (主成分分析) 是一种去相关的特征提取法，它将所有特征向量投影到低一维的向量空间中。

主成分分析的投影方向由以下步骤决定：

- 将坐标轴中心移到数据的中心，然后旋转坐标轴，使得数据在C1轴上的方差最大，即全部n个数据个体在该方向上的投影最为分散。意味着更多的信息被保留下来。C1成为**第一主成分**。
- C2**第二主成分**：找一个C2，使得C2与C1的协方差（相关系数）为0，以免与C1信息重叠，并且使数据在该方向的方差尽量最大。
- 以此类推，找到第三主成分，第四主成分……第p个主成分。p个随机变量可以有p个主成分

ICA

I Finding a set of basis to find most significant differences does not guarantee the success of classification. (Features may not be linearly separable)

The assumption of using a set of basis could be too strong to find essential differences among classes. (Orthogonal, Linear Independent, Span)

ICA (Independent Component Analysis): Represent the sample with a set of frame to show the most significant differences between classes.

NMF

Non-negative matrix factorization (NMF or NNMF), also **non-negative matrix approximation** is a group of algorithms in multivariate analysis and linear algebra where a matrix **V** is factorized into (usually) two matrices **W** and **H**, with the property that all three matrices have no negative elements. This non-negativity makes the resulting matrices easier to inspect. Also, in applications such as processing of audio spectrograms or muscular activity, non-negativity is inherent to the data being considered. Since the problem is not exactly solvable in general, it is commonly approximated numerically.

The regulation of basis does not guarantee the success of classification only the sparsity of features does.

The Non-negative Matrix Factorisation (NMF) factorise a matrix into two matrices: the weight matrix and the coefficient matrix.

The training process attempts to find a weight matrix such that resulting coefficient matrix is sparse enough for classification.

The weight matrix is also known as dictionary. I NMF leads to similar results with LDA but can reduce data dimension.

Lec 9 Naive Bayesian 朴素贝叶斯

The knowledge we have (or say the results we expect) is considered as prior probability (or prior knowledge) (先验概率)

The results we obtained in the experiment are considered as posterior probability (后验概率)

Bayes' Rule

The famous Bayes' Rule states the relationship between prior probability distribution and posterior probability distribution

通常，事件A在事件B已发生的条件下发生的概率，与事件B在事件A已发生的条件下发生的概率是不一样的。然而，这两者是有确定的关系的，贝叶斯定理就是这种关系的陈述。

$$P(c|x) = \frac{P(c)P(x|c)}{P(x)}$$

where c is considered as a class, x is considered as a set of samples with engaged attributes

- $P(c)$ is named as prior probability
- $P(x|c)$ is named as likelihood

其实就是在 c 发生的前提下 x 发生的概率，这个数值也被称为 c 的似然性 (likelihood)

- $P(c|x)$ is named as posterior probability
- $P(x)$ is considered as evidence factor (observation)

Bayes' Rule can be used for various purposes such as parameter estimation, classification and model selection

贝叶斯定理其实就是在说：后验概率与先验概率和似然性的乘积成正比

Model Likelihood 模型似然性

Model likelihood represents how likely a model will be observed according to the given posterior and prior probability hence can be used for model selection.

模型的反映了在给定的实验结果下，一个模型有多适合描述实验结果。

将贝叶斯公式变形后可以得到以下公式：

$$\text{Likelihood} = \frac{\text{Posterior Probability} \times \text{Observation}}{\text{Prior Probability}}$$

Given a dataset, the observation remain the same for different classes / samples.

极大似然估计，通俗理解来说，就是利用已知的样本结果信息，反推最具有可能（最大概率）导致这些样本结果出现的模型参数值。

换句话说，极大似然估计提供了一种给定观察数据来评估模型参数的方法，即：“模型已定，参数未知”。

例如我想用正态分布描述一个数据集，如何调整正态分布公式中的参数就可以使用MLE确定

Frequentists choose the model with highest likelihood as the best model, which is known as Maximum Likelihood Estimation.

Log-likelihood

When the number of samples increases, even computer may not be able to calculate correctly.

The Model likelihood of dataset D for a candidate model whose parameter sets are Θ has a likelihood of:

| 后验概率就是D中每个事件的在 Θ 条件下的联合概率

$$\mathcal{L} = p(D|\Theta) = \prod_{x \in D} p(x|\Theta)$$

For easier calculation, log-likelihood is commonly used which replace multiplications with additions

$$\mathcal{LL} = \log p(D|\Theta) = \sum_{x \in D} \log p(x|\Theta)$$

Occam's razor (size principle): the model favours the simplest (smallest) hypothesis consistent with the data

| 当多个模型都可以描述数据集时，选则简单的那个。

Bayesian Estimation & Laplacian Correction 贝叶斯估计与拉普拉斯平滑

- Considering the fact that if a case never appeared in the training dataset, how the model likelihood will be effected?

| 如果有一项频数为0，那么模型似然度将一直为0，这明显是不合理的，会导致分类的准确度受很大负面影响。

- This is known as a zero-point problem
- A smooth process can be performed by introducing prior probability distribution, which is known as Bayesian Estimation

| 一种解决 zero-point problem 的平滑化方法是贝叶斯估计

- More commonly, an uniform distribution is used as the prior probability distribution, in which case, it is called Laplacian Correction

| 更常用的方法是拉普拉斯平滑，拉普拉斯平滑的思想非常简单，就是对每个类别下所有划分的计数加1（也可以是其他正常数），这样如果训练样本集数量充分大时，并不会对结果产生影响，并且解决了上述模型似然性永远为0的尴尬局面。

Naive Bayes Classifier

朴素贝叶斯分类器其实非常简单，就是指定一个样本，为所有类别计算一个分数，那个类别得分最高，这个样本最后就分到哪一类。

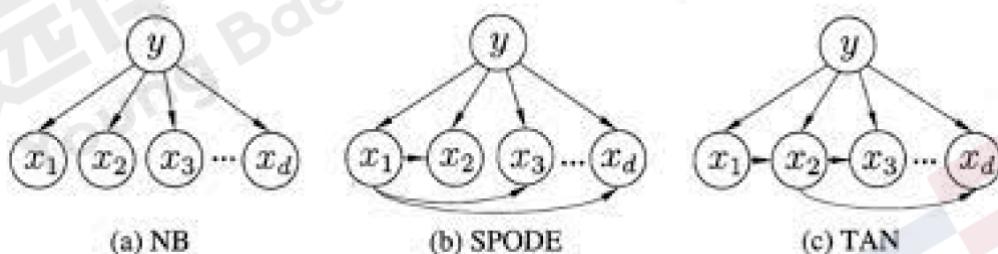
这个分数的计算方法为：

$$mark = p(c)p(x|c) = p(c)p(x_1|c)p(x_2|c)\dots p(x_l|c)$$

Semi-Naive Bayes Classifier

Naive Bayes assumes all conditions / features are independent to each other, which never happens

Semi-Naive Bayes allows dependencies between features



Bayesian Network

- The dependencies between features could be very complicated.
- Bayesian Network uses Directed Acyclic Graph (DAG) to show such dependencies, which is also known as Bayesian Graphical Model
- The idea of Bayesian Graphical model is to factorise the joint probability distribution to more readable conditional probability distribution, which benefits the modelling algorithm (especially for semantic modelling)
- The learning of structure, parameters and queries are all academic problems

Lec 10: Clustering 聚类

无监督学习 (unsupervised learning) 是机器学习的一种方法，没有给定事先标记过的训练示例，自动对输入的资料进行分类或分群。

聚类算法主要用于无监督学习。

- Unsupervised learning wants to learn the underlying structure of the unlabelled data and be able to explain it.
- Clustering is the assignment of a set of observations into subsets (which is called clusters) so that observations in the same cluster are similar in some sense.

Evaluation 如何评价聚类结果

The general idea of evaluating the performance of systems is to pursue a smaller intra-cluster distance but a larger inter-cluster distance.

评价聚类结果的基本方针是簇间间距尽可能大，簇内间距尽可能小

Distance Measurements 衡量距离

想要衡量簇间间距和簇内间距，就需要衡量距离的算法：

- Euclidean Distance
- City Block Distance
- Mahalanobis Distance
- Correlation Distance
- Cosine Distance
- Hamming Distance
- Chebyshev Distance
- Spearman Distance

- Jaccard Distance

Agglomerative Clustering 凝聚层次聚类

The general outline of agglomerative clustering is:

1. For n objects v_1, \dots, v_n , assign each to a singleton cluster $C_i = v_i$
2. Use any computable cluster similarity measure $D(C_i, C_j)$ e.g. Euclidean distance, cosine distance etc.
3. Repeat
 - identify the two most similar clusters C_j and C_k (could be ties - choose one pair)
 - delete C_j and C_k and add $(C_j \cup C_k)$ to the set of clusters
 until just one cluster remains
4. Use a dendrogram diagram to show the sequence of cluster mergers

简单来说，凝聚层次聚类就是从一个样本构成一族开始，每次选取最接近的两个簇合并为一个簇，直到只剩一个簇，最后得到的是一个簇数量一层层变少的合并过程记录，最后从中选择一层作为聚类的结果。

计算两个对象（特征向量）之间的距离很简单，套用所选的距离计算公式即可，但是计算簇之间的距离有多种办法。

There are multiple strategies for distance measurement between joint clusters (D):

- single linkage: D is taken as the minimum distance between samples in sub-clusters
 - | 簇之间的距离 D 是两簇中距离最近的样本间距离
- complete linkage: D is taken as the maximum distance between samples in sub-clusters
 - | 簇之间的距离 D 是两簇中距离最远的样本间距离
- average linkage: D is taken as the average distance between each pair of samples in sub-clusters
 - | 簇之间的距离 D 是两簇中样本两两计算距离的平均值

There are also other grouping strategies (such as centroid linkage).

Divisive Clustering 分裂聚类

k-means algorithms: an algorithm to classify or to group your objects based on attributes or features into k number of groups, where k is a positive integer

1. In the beginning, we determine the number of clusters (k) that we want and we assume the centroid or centre of these clusters
2. Then repeat the following steps until converge
 - Assign each training sample to the cluster with the nearest centroid.
 - Calculate the new centroid for each cluster

Stop until no samples changes the cluster belonged after an iteration

在开始时假设存在 k 簇，并设定这些簇的中心，每次将所有对象分到与其中心距离最近的簇中，然后重新计算每个簇的中心，重复上述过程直到每个样本所属的簇不再变化，

Model based Clustering

The mixture model could be used for clustering as well

The most classical model is Gaussian Mixture Model (GMM)

The definition of Gaussian Mixture Model is:

$$p(x) = \sum_k \pi \mathcal{N}(x | \mu_k, \text{Cov}_k)$$

where:

- \mathcal{N} represents Gaussian distribution (known as Gaussian component in GMM)
- π is the weight of Gaussian component
- μ is the mean of Gaussian component
- Cov is the co-variance matrix of Gaussian component

Each Gaussian component corresponds to a cluster

Expectation Maximisation 最大期望算法

1. When there are missing values exist among the data, or the model can be formulated more simply by assuming the existence of further unobserved data points, Expectation Maximisation (EM) can be used.
2. There are two steps in EM methods: Expectation step and Maximisation step:
 - Expectation step: find the posterior probability according to current model
 - Maximisation step: calculate the new modal parameters

EM是一个在已知部分相关变量的情况下，估计未知变量的迭代技术。EM的算法流程如下：

1. 初始化分布参数
2. 重复直到收敛：
 1. E步骤：根据参数的假设值，给出未知变量的期望估计，应用于缺失值。
 2. M步骤：根据未知变量的估计值，给出当前的参数的极大似然估计。

Model Selection

we evaluate the success of GMM by model likelihood

见上一章节模型的似然性

But model likelihood could lead to overfitting problem hence model selection criteria are introduced, which penalise model likelihood by the complexity of model:

- Akaike Information Criterion (AIC):

$$AIC = 2K - \mathcal{L}\mathcal{L}$$

$\mathcal{L}\mathcal{L}$ 见上一张 Log-likelihood

- Bayesian Information Criterion (BIC):

$$BIC = k \log N - 2\mathcal{L}\mathcal{L}$$

模型选取的结果可以用交叉验证法验证。

Lec 11: Classification: Non-parametric Modeling

Parametric model

在统计学中，参数模型或参数族或有限维模型是一类特殊的统计模型。具体来说，参数模型是一组具有有限数量参数的概率分布。

Non-Parametric Modeling

We presume that there exists a model, either a Bayesian model (e.g. Naive Bayes) or a mathematical model (e.g. GMM)

- We seldom obtain a “true” model due to the lack of prior knowledge
- For the same reason, we never know which model should be used

According to Central Limit Theorem (CLT), which roughly states that “when independent random variables are added, their properly normalised sum tends toward a normal distribution even if the original variables themselves are not normally distributed”

The non-parametric modelling depends on empirical data rather than the dependency on prior knowledge

Non-parametric models can be used with arbitrary distributions and without the assumption that the forms of the underlying densities are known

Moreover, they can be used with multimodal distributions which are much more common in practice than unimodal distributions

There are two types of non-parametric methods:

- Estimating $P(x|w_j)$
 - Kernel Density Estimation (Parzen window)
- go directly to a-posterior probability estimation (Estimating $P(x|w_j)$)
 - k-Nearest Neighbour (k-NN)

Kernel Density Estimation 核密度估计

核密度估计其实是对直方图的一个自然拓展, 用来估计未知的密度函数, 属于非参数检验方法之一, 又名Parzen窗 (Parzen window) 。

本质上就是采用平滑的峰值函数(“核”)来拟合观察到的数据点, 从而对真实的概率分布曲线进行模拟。

核密度估计对数据分布不附加任何假定, 是一种从数据样本本身出发研究数据分布特征的方法。

当我们想要了解每个数据集的概率分布情况时, 一般会画一个频率分布直方图 histogram, 这个直方图其实就是一种概率分布函数。

核密度估计就是一种将直方图转化为概率分布函数的方法。

Kernel 核

We could approximate the probability distribution function by the sum of probability density representations (which is known as a kernel function)

The kernel function follows three requirements

- Non-negative, real-valued and integrable
- Normality

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

- Symmetry

$$K(-x) = K(x)$$

一些比较常用的核函数是：

- 均匀核函数 $k(x)=1/2, -1 \leq x \leq 1$ 加入带宽 h 后： $kh(x) = 1/(2h), -h \leq x \leq h$
- 三角核函数 $k(x)=1-|x|, -1 \leq x \leq 1$ 加入带宽 h 后： $kh(x) = (h - |x|)/h^2, -h \leq x \leq h$

Bandwidth 带宽

虽然采用不同的核函数都可以获得一致性的结论（整体趋势和密度分布规律性基本一致），但核密度函数也不是完美的。除了核算法的选择外，带宽（bandwidth）也会影响密度估计，过大或过小的带宽值都会影响估计结果

The width of each bin affects the final probability mass distribution. Similarly we define the bandwidth h to regulate the “width” of kernel function i.e. $K(\frac{x}{h})$

For the observation at $x = x_i$, we have $K(\frac{x-x_i}{h})$

So the probability distribution function \hat{f} could be written as:

$$\hat{f} = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{x-x_i}{h}\right)$$

as:

$$\int_{-\infty}^{+\infty} K\left(\frac{x}{h}\right)dx = \frac{1}{h}$$

Parzen Window and Classification

In classifiers based on Parzen window estimation:

- We estimate the densities for each category and classify a test point by the label corresponding to the maximum posterior
 - Using the points of only category w_i , $P(x|w_i)$ can be estimated
 - Knowing $P(w_i)$, posterior probabilities can be found

The decision region for a Parzen window classifier depends upon the choice of window function.

K-Nearest Neighbour k近邻(KNN)

Goal: a solution for the problem of the unknown best window function

- Let the cell volume be a function of the training data
- Centre a cell about x and let it grows until it captures k_n samples
 - k_n samples are called k_n nearest-neighbours of x

Two possibilities can occur:

- Density is high near x ; therefore the cell will be small which provides a good resolution
- Density is low; therefore the cell will grow large and stop until higher density regions are reached

We can obtain a family of estimates by setting $k_n = \sqrt{n}$ and choosing different values for k_n .

KNN Algorithm

- As in the general problem of classification, we have a set of data points for which we know the correct class labels.
- When we get a new data point, we compare it to each of our existing data points and find similarity. I
- Take the most similar k data points (k nearest neighbours)
- From these k data points, take the majority vote of their labels. The winning label is the label / class of the new data point

Density Estimation

There are two different ways of obtaining sequences of regions that satisfy these conditions:

- Parzen-window estimation method: Shrink an initial region where $Vn = \frac{1}{\sqrt{n}}$ and show that

$$\lim_{n \rightarrow \infty} P_n(x) \rightarrow P(x)$$

- K-nearest neighbor estimation method: Specify kn as some function of n, such as $k_n = \sqrt{n}$; the volume V_n is grown until it encloses kn neighbors of x.

Pros and Cons of Non-parametric Modeling

- No assumptions are needed about the distributions ahead of time (generality).
- With enough samples, convergence to an arbitrarily complicated target density can be obtained.
- The number of samples needed may be very large (number grows exponentially with the dimensionality of the feature space).
- These methods are very sensitive to the choice of window size (if too small, most of the volume will be empty, if too large, important variations may be lost).
- There may be severe requirements for computation time and storage.

Lec12: Propositional Logic & Prolog 命题逻辑

logic

When most people say 'logic', they mean either *propositional logic* or *first-order predicate logic*

Any 'formal system' can be considered a logic if it has:

- a well-defined syntax;
 - The syntax of logic defines the syntactically acceptable objects of the language, which are properly called well-formed formulae
- a well-defined semantics;
 - The semantics of logic associate each formula with a meaning
- a well-defined proof-theory
 - The proof theory is concerned with manipulating formulae according to certain rules.

Propositional Logic

命题逻辑是最抽象的逻辑

Definition: A proposition is a statement that can be either true or false; it must be one or the other and it cannot be both.

Definition: An atomic (原子性) proposition is one whose truth or falsity does not depend on the truth or falsity of any other proposition

原子命题是不包含其他命题作为其组成部分的命题，即在结构上不能再分解出其他命题的命题。又称简单命题。原子命题不能带有非，或，且，如果，那么等联结词。

Connectives

逻辑连接词能够构造复杂的命题：

- AND: \wedge (& or .)
- OR: \vee (— or +)
- NOT: \neg (\sim)
- IMPLIES: \Rightarrow (\subset , \rightarrow)
- IFF (if and only if): \Leftrightarrow

Tautologies & Consistency 重言式与一致性

Definition: A valuation is a function which assigns a truth value to each primitive proposition

重言式 (Tautology) 又称为永真式。命题公式中有一类重言式。如果一个公式，对于它的任一解释下其真值都为真，就称为重言式 (永真式)

Definition:

- A formula is a tautology iff it is true under every valuation;
- A formula is consistent iff it is true under at least one valuation;
- A formula is inconsistent iff it is not made true under any valuation.

First-Order Logic 一阶逻辑

一阶逻辑 (FOL) 和命题逻辑的不同之处在于，一阶逻辑有使用量化变量

Terms

- The basic components of FOL are called terms.
- Essentially, a term is an object that denotes some object other than true or false.
- The simplest kind of term is a *constant*
- The second simplest kind of term is a variable.
- A variable can stand for anything in a set of objects.

Definition: A constant of type T is a name that denotes some particular object in the set T.

Definition: A variable of type T is a name that can denote any value in the set T.

a more complex class of term - functions:

- idea of functional terms in logic is similar to the idea of a function in programming: recall that in programming, a function is a procedure that takes some arguments and returns a

value

- In FOL, we have a set of function symbols; each symbol corresponds to a particular function.
(It denotes some function.)
- Each function symbol is associated with a natural number called its arity. This is just the number of arguments it takes.
- Each function symbol has a return-type associated with it and each function symbol has an argument type associated with it.
- A functional term is then built up by applying a function symbol to the appropriate number of terms, of the appropriate type

Definition: Let f be an arbitrary function symbol of type T , with arity $n \in N$, taking arguments of type T_1, \dots, T_n respectively. Also let τ_1, \dots, τ_n be terms of type T_1, \dots, T_n respectively. Then

$$f(\tau_1, \dots, \tau_n)$$

is a functional term.

Predicates 谓词

- In addition to having terms, FOL has relational operators, which capture relationships between objects.
- The language of FOL contains a stock of predicate symbols.
- These symbols stand for relationships between objects.
- Again, each predicate symbol has an associated arity and each argument has a type.

Definition: Let P be a predicate symbol of arity $n \in N$, which takes arguments of types T_1, \dots, T_n . Then if τ_1, \dots, τ_n are terms of type T_1, \dots, T_n respectively, then

$$P(\tau_1, \dots, \tau_n)$$

is a predicate, which will either be true or false under some interpretation.

Quantifiers

- We now come to the central part of first order logic: quantification.
- Consider trying to represent the following statements:
 - all men have a mother;
 - every natural number has a prime factor.
- We can't represent these using the apparatus we've got so far; we need quantifiers.
 1. \forall – the universal quantifier; is read 'for all ...'
 2. \exists – the existential quantifier; is read 'there exists ...'
 3. \exists_1 – the unique quantifier; is read 'there exists a unique ...'

The simplest form of quantified formula in Z is as follows:

$$\text{quantifier} \text{signature} \bullet \text{predicate}$$

where

- quantifier is one of $\forall, \exists, \exists_1$;
- signature is of the form:

$$\text{variable : type}$$

Decidability 可判定性

- In propositional logic, we saw that some formulae were tautologies
- they had the property of being true under all interpretations.
- We also saw that there was a procedure which could be used to tell whether any formula was a tautology – this procedure was the truth-table method.
- A formula of FOL that is true under all interpretations is said to be valid.
- Now we can't use truth tables to tell us whether a formula of FOL is valid.
- Is there any other procedure that we can use, that will be guaranteed to tell us, in a finite amount of time, whether a FOL formula is, or is not, valid?
- The answer is no.
- FOL is for this reason said to be undecidable.

| Prolog 是一门声明式的编程语言，同学们自行了解基本语法即可，这里就不再赘述了