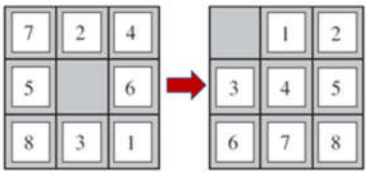
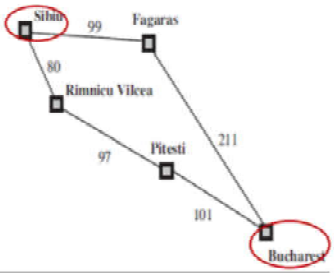
AI 期末笔记

2020年1月5日　2:49

0. 课程介绍
    a. assessment
        i. 期末考/project =4：6　to　6：4，取决于怎么调整能够让更多人及格。但是不调分
    b. data for 2018fall
        i. 188= 133pass + 48fail　fail rate：25.5%
        ii. 48个fail的人中，17个人一个project都没交。6个完成了一个，7个完成了2个。
        iii. 133个pass的人中，132个人完成了all 4 project，1个完成了3个project
        iv. 真正project都完成了的人很少fail，project过了就有60分
1. Solving problem by searching
    a. 问题的表示，即定义以下几个点
        - States: all allocations of the 9 tiles.
        - Initial state: the allocation of the Left.
        - Actions: the movement of the blank space $\{left, right, up, down\}$.
        - Transition: updated allocations of the tiles.
        - Goal test: the allocation of the Right?
        - Path cost: #moves.



    b. 一些通用搜索算法
        i. 记住并理解四个对搜索算法的评判标准
            1) Completeness：能不能保证找到解?
            2) Optimality：能不能保证找到最优解?
            3) 时间复杂度
            4) 空间复杂度
        ii. 能运用对树的几个分析量
            1) b：一个节点最多有几个子节点
            2) d: 最优解有几步
            3) m:最劣path有几步
        iii. Uniform-cost search (UCS)
            1)
                - Task: from Sibiu to Bucharest.
                - [0] {[Sibiu, 0]}
                - [1] {[Sibiu→Rimnicu, 80]; [Sibiu→Fagaras, 99]}
                - [2] {[Sibiu→Rimnicu→Pitesti, 177]; [Sibiu→Fagaras, 99]}
                - [3] {[Sibiu→Rimnicu→Pitesti, 177]; [Sibiu→Fagaras→Bucharest, 310]}
                - [4] {[Sibiu→Rimnicu→Pitesti→Bucharest, 278]; [Sibiu→Fagaras→Bucharest, 310]}



            2) 方法就是找一个没去过并且总cost最低的节点
            3) When all step costs are equal, UCS is similar to BFS.
        iv. Depth-first search (DFS)
            1) Complete? No, fail in infinite-depth space and space with loops.
            2) Optimal? No.
            3) Space complexity：bm, much lower than BFS.
            4) Terrible if m >>d
            5) 在解dense时，效果比BFS好

v. Depth-limited search (DLS)
   1) DFS with depth limit
vi. Iterative deepening search (IDS)
   1) Apply DLS with increasing limits
   2) 每遍历一整个树，就limit+1
vii. Bidirectional search
   1) 两边同时展开搜索
   2) complete、optimal
viii. 总结

Uninformed Search Methods

BFS：完整、最优

UCS：找一个没去过并且总cost最低的节点，完整、最优

DFS：不完整（死在无限循环上）、最优，但是空间复杂度小

DLS：不完整，不最优。给DFS增加一些深度约束

IDS（迭代深入）：每一次都同时访问这个节点的所有子节点，直到没了才回退

Uniformed Search: Use no domain knowledge.

- Breadth-first search (BFS): expand shallowest node
- Uniform-cost search (UCS): expand cheapest node
- Depth-first search (DFS): expand deepest node
- Depth-limited search (DLS): depth first with depth limit
- Iterative deepening search (IDS): DLS with increasing limit
- Bidirectional search: search from both directions

ix.

PF Metric	Breadth-first Search	Uniform-cost Search	Depth-first Search	Depth-limited Search	Iterative Deepening	Di-directional Search
Complete?	Yes*, if b is finite.	Yes*, if step costs$\geq \epsilon$.	No, infinite loops can occur.	No. (Eps. $l < d$)	Yes	Yes*, if BFS used for both search.
Optimal?	Yes*, if costs on the edge are non-negative.	Yes	No,	No	Yes*, if costs on the edge are non-negative.	Yes*, if BFS is used & paths have uniform cost.
Time?	$O(b^{d+1})$	$O(b^{1+\lfloor C^*/\epsilon \rfloor})$	$O(b^m)$	$O(b^l)$	$O(b^d)$	$O(b^{d/2})$
Space?	$O(b^{d+1})$	$O(b^{1+\lfloor C^*/\epsilon \rfloor})$	$O(bm)$	$O(bl)$	$O(bd)$	$O(b^{d/2})$

x. brief
   1) Memory + exponential time complexities are the biggest handicaps of BFS.
   2) When all step costs are the same, UCS is similar to BFS.
   3) Space complexity of DFS is much lower than BFS.
   4) IDS uses only linear space and NOT much more time than other uninformed methods： preferred

c. state elimination
   i. Cyclic State Repeats
      1) 有环的路径不可能是最优解

d. Heuristic (informed) Search
   i. PF metric of heuristic alg.
      1) Complete? No, can stuck in loops.
         Complete in finite space with repeated-state checking.
      2) Optimal? No.

3) Time? b^m .but a good heuristic can give drastic improvement.
4) Space? b^m

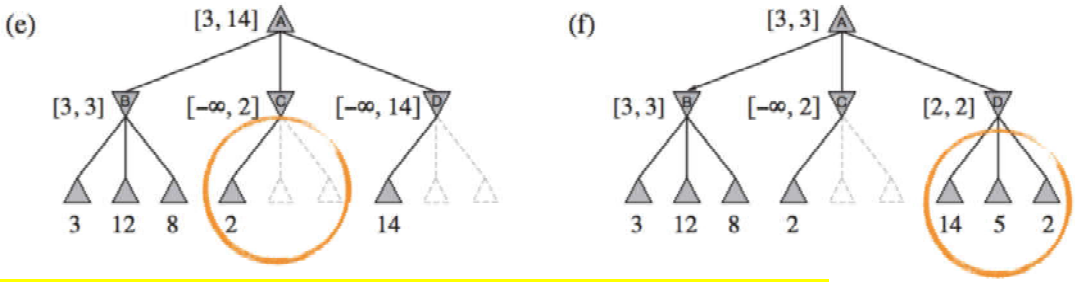
ii. A*核心就是best first search

1) PF of A*
   a) Complete & Optimal
2) Admissible heuristics are optimistic
   a) 就是我们对剩余工作量的估计是乐观估计，是一定不等于其真实工作量的，所以最后如果找到一个解，则这个解一定是最优的，因为我们能到这一步是因为我们的实际花费<其他解的实际花费+预期花费
3) 比如：华容道问题，我们就使用直接移动的步数来估计剩余步数
4) Effective Branching Factor b*
   a) 解方程：
      i) $N = b^* + (b^*)^2 + \cdots + (b^*)^d$
   b) 其中，N是解的节点数，d是解的深度，理想情况下N=d，即b=1
5) admissible：要保证h的这个代价要小于实际的代价，这样h 比较有意义
6) verification
   • Aim: Compare hT and hW on searching efficiency.
   • Setting: Generate 1200 random problems with

   Note: IDA – a baseline

2. lec6 10/16
   a. 对抗式搜索（adversarial search） game minimax algorithm
      i. Alpha-Beta Pruning

1)



2) 注意看这张图，右端点会优先出现，而左端点则出现得较晚，或者改变不改变
3) alpha：max所能获得的最好的值
4) beta：min所能获得的最好的值
5) 在最好的情况下，复杂度为
   a) $O(b^{m/2})$
   b) 相当于减少一层（max min层相当于一层）
6) 实践中
   a) 使用评估函数evaluation而非utility
   b) 使用剪枝cutoff-test而非终点判定terminal-test
7) Minimax Optimization：最大化下限

3. lec4 constraint satisfaction problem
   a. 只包含布尔变量和布尔运算的约束叫做 SAT 问题，SAT中只有命题演算，讨论命题公式的可满足性 CSP特征是变量的取值范围是有限的。
4. first-order logic （FOL） lec6
   a. 定义
      i. 概念
         1) 就是 predicate logic
         2) 关键是有quantifier：存在、任意
         3) 把世界想象成ER graph

ii. 用处
    1) 比较容易用来做inference或用来建立知识库（KB）
    2) 更符合人类的思维
iii. 语法

$$
\begin{aligned}
Sentence &\rightarrow AtomicSentence \mid ComplexSentence \\
AtomicSentence &\rightarrow Predicate \mid Predicate(Term, \ldots) \mid Term = Term \\
ComplexSentence &\rightarrow (\ Sentence\ ) \mid [\ Sentence\ ] \\
&\mid \neg\ Sentence \\
&\mid Sentence \wedge Sentence \\
&\mid Sentence \vee Sentence \\
&\mid Sentence \Rightarrow Sentence \\
&\mid Sentence \Leftrightarrow Sentence \\
&\mid Quantifier\ Variable, \ldots\ Sentence \\
Term &\rightarrow Function(Term, \ldots) \\
&\mid Constant \\
&\mid Variable \\
Quantifier &\rightarrow \forall \mid \exists \\
Constant &\rightarrow A \mid X_1 \mid John \mid \cdots \\
Variable &\rightarrow a \mid x \mid s \mid \cdots \\
Predicate &\rightarrow True \mid False \mid After \mid Loves \mid Raining \mid \cdots \\
Function &\rightarrow Mother \mid LeftLeg \mid \cdots \\
\text{OPERATOR PRECEDENCE} &: \neg, =, \wedge, \vee, \Rightarrow, \Leftrightarrow
\end{aligned}
$$

b. knowledge base engineering （KBE）
    i. 步骤
        1) 确定一个任务
        2) 组合相关的知识
        3) 确定相关的谓语、函数、常数之类的词汇
        4) 编码这个domain的通用知识、rule作为定理（axioms）

“A sibling is another child of one’s parents”

a) $\forall x, y\ Sibling(x, y) \Leftrightarrow x \neq y \wedge \exists Parent(p, x) \wedge Parent(p, y)$

“One’s mother is one’s female parent”

b) $\forall m, c\ Mother(c) = m \Leftrightarrow Female(m) \wedge Parent(m, c)$

        5) 编码这个问题的描述
        6) 对推理步骤进行查询并且获得答案
c. 通过命题化进行推理
d. 插入 lab10 逻辑推理
    i. 合取范式（conjunctive normal form）
        1) 使得SAT（satisfiablity problem）的问题搜索成为可能，通过回溯法搜索所有的子表达式，就像CSP那样
        2) 特征：子句间是conjunction，子句内是disjunction
        3) 转换示范

$B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

1. Eliminate $\Leftrightarrow$, replacing $\alpha \Leftrightarrow \beta$ with $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$.

   $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$

2. Eliminate $\Rightarrow$, replacing $\alpha \Rightarrow \beta$ with $\neg\alpha \vee \beta$.

   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg(P_{1,2} \vee P_{2,1}) \vee B_{1,1})$

3. Move $\neg$ inwards using de Morgan's rules and double-negation:

   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge ((\neg P_{1,2} \wedge \neg P_{2,1}) \vee B_{1,1})$

4. Apply distributivity law ($\vee$ over $\wedge$) and flatten:

   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$

4) 回溯中给符号取值的过程，就是在建立model（一个符号表达式的一组取值）
   a) DPLL算法（Davis-Putnam-Logemann-Loveland）
      i) 优先寻找纯符号，比如子句中只有A
      ii) 优先寻找单元子句，即子句中其他文字都为false只剩一个文字正确性未知

ii. 一阶逻辑的处理方向
   1) 退化成命题逻辑
   2) 一阶逻辑的推理
      a) 前向链接
         i) 根据条件，不断退出新的东西，并且加入KB，直到推出我们需要的东西
      b) 后向链接
         i) 根据需求，不断反推出条件，直到发现所需条件与已知矛盾，或者发现所需条件我们都有
   3) Unify
      a) x,y是变量，在这里被替换了
      b)

p	q	θ
$Knows(John, x)$	$Knows(John, Jane)$	$\{x/Jane\}$
$Knows(John, x)$	$Knows(y, OJ)$	$\{x/OJ, y/John\}$
$Knows(John, x)$	$Knows(y, Mother(y))$	$\{y/John, x/Mother(John)\}$
$Knows(John, x)$	$Knows(x, OJ)$	$fail$

   4) 合取范式
   5) 归结证明
      a) 即证明 KB^~α 为false
      b) 先把 ~α 转换成CNF，放入CNF标准化的KB
      c) 推导出一定为false

5. planning lec7 复习：12/7
   a. 目标：从动作集中找到goal state
   b. planning的定义
      i. 能让agent达到goal state的动作集
   c. planning可以formulated到Boolean SAT problem上
      i. 变量：动作（做或不做）
      ii. domain：做或不做
      iii. sentence：the goal
      iv. 这种的propositional representation（命题表征）效率低
         1) 搜索空间巨大

2) 这种问题是PSPACE，比NPC还难
3) 没有顺序?

d. Planning Domain Definition Language (PDDL，规划领域定义语言)
   i. 举例
      1) 从圣弗朗西斯科飞到肯尼迪机场
      2) $$Action(Fly(P_1, SFO, JFK),$$ $$\text{PRECOND:} At(P_1, SFO) \wedge Plane(P_1) \wedge Airport(SFO) \wedge Airport(JFK)$$ $$\text{EFFECT:} \neg At(P_1, SFO) \wedge At(P_1, JFK))$$
      3) 途径：
         a) 移除一些细节，进行一些约束
         b) 通过定义动作和状态之间的关系，从而得到一个abstract graph

e. 对planning的通用启发式（general heuristics）
   i. 方法：加边或者减点
      1) 为什么不减边：因为可能会把最优解减了，这个比较难判断
      2) 为什么不加点：没有意义，只会让抽象图更加复杂
   ii. 加边
      1) 忽略了前提（preconditions）导致了集合覆盖问题（set-cover problem，属于NC hard）
         a) 前提：任意两个state之间都有边
         b) 事实：每个顶点（state）都可以表示为一个命题（logic sentence）
         c) 表述：从2n种状态中选择k个，从而覆盖整个goal state
      2) Ignore delete lists: hill climbing is applicable since the goal state can be monotonically satisfied
      3) ？？？
   iii. 删点
      1) 有些状态我们不关心（irrelevant）

6. representing uncertainty lec8 复习：12/7
   a. 不确定性和推理判断
      i. 世界是不确定的，agent很多时候无法全部观测
      ii. 解决办法
         1) 效用理论（Utility theory）
            a) 给每一个action或state赋予一个utility
         2) 概率论（probability theory）
            a) 计算每一个状态的不确定性
         3) 理性决策（rational decisions）
            a) 最大化expected utility（probability + utility）
   b. 基础的概率论及其应用
      i. 联合概率分布（Joint probability distribution）
      ii. 查询（query）
      iii. 非条件概率
      iv. 概率分布
      v. 每一个关于domain的问题都可以使用联合概率分布来回答，因为每一件事情都是一些sample points之和
      vi. 条件概率/后验概率（posterior probability）
         1) 可以简化条件，而结果保持不变
         2) 这种推导依赖于domain knowledge，很重要
         3) 定义

a) $P(a|b) = \frac{P(a \wedge b)}{P(b)}$ if $P(b) \neq 0$

4) product rule

a) $P(a \wedge b) = P(a|b)P(b) = P(b|a)P(a)$

5) 链式法则

a) $$\begin{aligned}\mathbf{P}(X_1, \ldots, X_n) &= \mathbf{P}(X_1, \ldots, X_{n-1})\ \mathbf{P}(X_n|X_1, \ldots, X_{n-1}) \\ &= \mathbf{P}(X_1, \ldots, X_{n-2})\ \mathbf{P}(X_{n_1}|X_1, \ldots, X_{n-2})\ \mathbf{P}(X_n|X_1, \ldots, X_{n-1}) \\ &= \ldots \\ &= \prod_{i=1}^{n} \mathbf{P}(X_i|X_1, \ldots, X_{i-1})\end{aligned}$$

6) 独立性证明

a) $\mathbf{P}(A|B) = \mathbf{P}(A)$ or $\mathbf{P}(B|A) = \mathbf{P}(B)$ or $\mathbf{P}(A, B) = \mathbf{P}(A)\mathbf{P}(B)$

b) $$\begin{aligned}&\mathbf{P}(Toothache, Catch, Cavity, Weather) \\ &= \mathbf{P}(Toothache, Catch, Cavity)\mathbf{P}(Weather)\end{aligned}$$

7) 贝叶斯rule

a) Product rule $P(a \wedge b) = P(a|b)P(b) = P(b|a)P(a)$

$\Rightarrow$ Bayes' rule $P(a|b) = \frac{P(b|a)P(a)}{P(b)}$

b) 在由casual probability （随机概率）计算diagnostic probability（诊断概率）时有用：

i) $P(Cause|Effect) = \frac{P(Effect|Cause)P(Cause)}{P(Effect)}$

7. Inference in Bayesian Networks 贝叶斯网络（bayesian network，BN） lecture 9

a. 没学好，记得看chrome AI文件夹中的知乎文章

b. 核心就是根据已有的这些东西（先验概率），来推测一些东西的概率（后验概率）

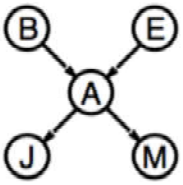
i. 例子

1) 就是一些基本的贝叶斯变换，套公式就行

Simple query on the burglary network:

$$\begin{aligned}&\mathbf{P}(B|j, m) \\ &= \mathbf{P}(B, j, m)/P(j, m) \\ &= \alpha\mathbf{P}(B, j, m) \\ &= \alpha \sum_e \sum_a \mathbf{P}(B, e, a, j, m)\end{aligned}$$

Need to consider all values of "hidden variables", e.g., *alarm*=true, *alarm*=false



ii. 根据图中的连接关系可知 B、E互相独立，只有A能决定J、M，所以可以将上式重写为

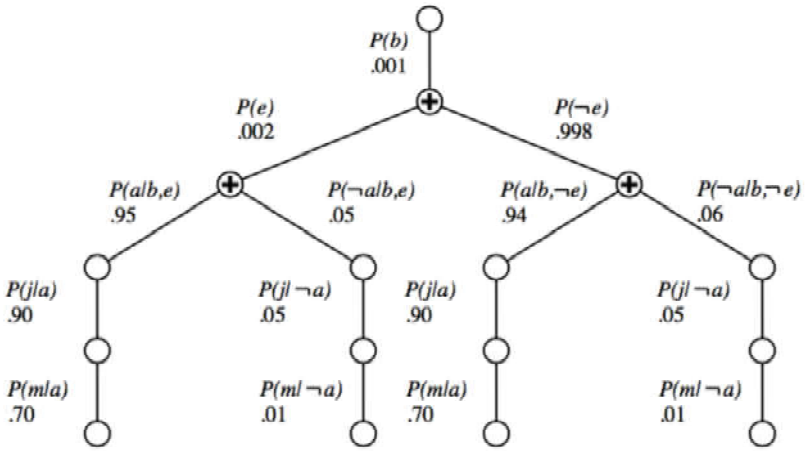
$$\begin{aligned}&\mathbf{P}(B|j, m) \\ &= \alpha \sum_e \sum_a \mathbf{P}(B)P(e)\mathbf{P}(a|B, e)P(j|a)P(m|a) \\ &= \alpha\mathbf{P}(B) \sum_e P(e) \sum_a \mathbf{P}(a|B, e)P(j|a)P(m|a)\end{aligned}$$

iii. 复杂度

1) 不断地分解各个项，然后计算，这样的话就要遍历所有的可能，d 表示变量domain的大小（比如如果变量是布尔类型，则 d 2），n表示变量的数量

Recursive depth-first enumeration: $O(n)$ space, $O(d^n)$ time

iv. 这样就可以生成这种 Evaluation Tree了



v. 但是如果直接这样算会有很多东西要重复计算，所以不妨把计算出的一些东西记下来

c. 优势

i. 表示更加紧致（比2^n好）

ii. 如果每个节点的父节点都不超过k个，这个图的大小只有n*2^k

d. 建立方式

i. For $i = 1$ to n
add X_i to the network
select parents from $X_1, \ldots, X_{i-1}$ such that
$\mathbf{P}(X_i|Parents(X_i)) = \mathbf{P}(X_i|X_1, \ldots, X_{i-1})$

ii. 建立时要保证

iii. $$\begin{aligned}\mathbf{P}(X_1, \ldots, X_n) &= \prod_{i=1}^{n} \mathbf{P}(X_i|X_1, \ldots, X_{i-1}) \quad \text{(chain rule)} \\ &= \prod_{i=1}^{n} \mathbf{P}(X_i|Parents(X_i)) \quad \text{(by construction)}\end{aligned}$$

e. Markov Chain Monte Carlo (MCMC)

f. Enumeration by Variable Elimination

i. 指将一些要计算的项组合在一起，生成已知变量，从而避免重复计算

g. Rejection Sampling 拒绝采样

function REJECTION-SAMPLING($X, \mathbf{e}, bn, N$) **returns** an estimate of $P(X|\mathbf{e})$
local variables: $\mathbf{N}$, a vector of counts over X, initially zero

for $j = 1$ to N **do**
$\mathbf{x} \leftarrow$ PRIOR-SAMPLE(bn)
if $\mathbf{x}$ is consistent with $\mathbf{e}$ **then**
$\mathbf{N}[x] \leftarrow \mathbf{N}[x]+1$ where x is the value of X in $\mathbf{x}$
return NORMALIZE($\mathbf{N}[X]$)

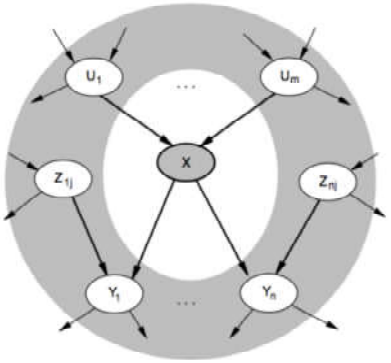
i. 采样n次，记录下这个时候的如果有e时x发生的数量

h. likelihood weighting（LW）

i. 固定evidence变量，对其他非evidence的变量进行取样，然后将样本和evidence对比

i. Markov blanket

i. 与已知变量直接相关的变量，包括父、子、子的其他父母



j. 总结
    i. 通过消除变量来推导
        1) 在无环图中，耗时多项式时间；在普通图中，耗时NP-hard
    ii. 通过LW（Likelihood Weighting）或MCMC近似推导
        1) LW在有很多evidence时效果很差
        2) LW、MCMC对拓扑结构不敏感
        3) 当概率接近0或1时，收敛非常慢
        4) 可以处理任意的离散或者连续的变量的组合

8. learning principle lecture10 复习：12/7
    a. learning的关键
        i. 数据的格式是什么？（data representation）
        ii. 有先验知识吗？
        iii. agent function 长什么样？（model representation）
        iv. 如何测量提高？（objective function）
        v. 学习算法是什么？（为了获得一个好的agent function）
        vi. 总结：Representation + Algorithm + Evaluation
    b. 归纳（inductive）和推导（deductive）
        i. 归纳：没有先验知识，纯粹凭借数据
            1) 如：手写数字识别
        ii. 推理：从先验知识和数据中学习
    c. 监督与无监督
        i. supervised learning：有标准答案
        ii. unsupervised learning：没有标准答案
        iii. reinforce learning：没有标准答案，只有正确与否
    d. learning principle
        i. generalization：学到的这个agent function 能够handle 之前没见过的情况
            1) 要实现这一点，关键在于设计一个良好的目标函数
            2) Occam s razor：当多个模型都可以时，挑选那个最简单的
            3) overfitting：模型对于已经见过的数据fit得很好，但是不能generalization
            4) Occam ‘ razor和over fitting关系紧密

9. supervised learning I lecture 11 复习：12/7
    a. learn啥
        i. 理论上，我们希望整一个全能的超级AI，但这目前显然不实际，所以我们就希望整一个有特定功能的AI
        ii. 比如：分类。分类很难用传统的数值编码的方式完成coding
    b. 目标：最大化后验概率

$$P(w_j|x) = \frac{p(x|w_j)P(w_j)}{p(x)}$$

        1) 因为 是一个常数，而我们只是希望找到概率最大的那个事件而非计算出其概论，所以这一项可以省略
        2) 再取个 对数，就得到我们的简化版目标函数：

$$g_i(x) = \ln p(x|w_i) + \ln P(w_i)$$

后验概率

1 最后算出的P(X=玩lol|Y=男性)称之为X的后验概率，即它获得是在观察到事件Y发生后得到的

2 后验概率是在考虑和给出相关证据或数据后所得到的条件概率

先验概率

1 P(X=玩lol)=0.6；P(X=不玩lol)=0.4，这个概率是统计得到的，即X的概率分

布已知，我们称其为先验概率(prior probability)

parametric method

i d：独立同分布

步骤

1 先预设一个模型

2 然后统计先验知识和似然

通过先验知识计算模型中的一些参数，不用手调这些参数

非参数化方法才会手调参数

d Linear Discriminant Analysis（线性判别分析）

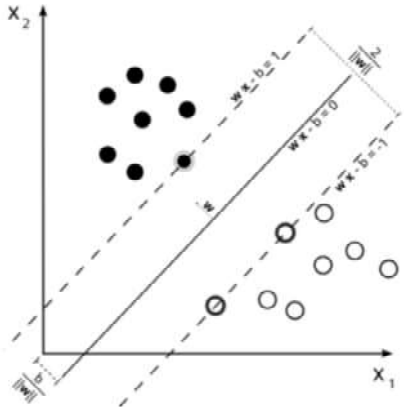
在Euclidean space中，找一个线性函数（直线），将所有case分成两堆

$$g(x) = w^t x + b$$

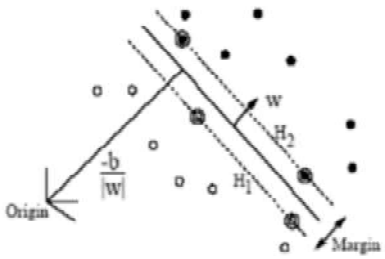
ii. SVM就是这样构建的

e. SVM（support vector machine）

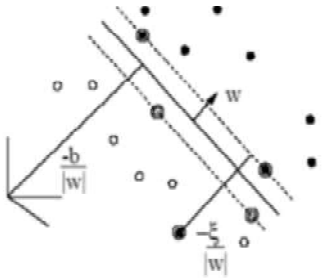
i. basic idea：让margin 最大化



soft margin 更常用



Hard Margin（硬间隔）



Soft Margin（软间隔）

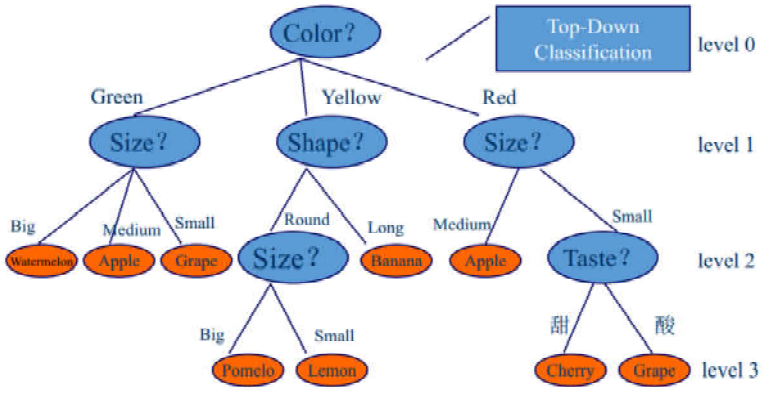
f. Universal Approximation Theory

i. 可以使用神经网络逼近任何连续函数，正如 多项式函数 或 三角函数做的那样

ii. 原则上，一个隐藏层就可以拟合绝大多数问题了

iii. 超过一个隐藏层的 Fully connected NN (MLP) 非常难训练

g. 决策树（decision tree， DT）



i. 训练目标：

1) 让叶子节点的 instance 尽可能纯粹，尽可能来自同一个class

ii. 训练方法

1) constructive heuristic

a) 从跟节点开始，对一个节点都找一个能够让impurity下降得最快的rule

b) 当整个过程结束了，就给叶子节点贴标签，通常的处理方式为：标签内容为 该节点中的majority class

iii. 必须控制一下树的复杂程度，不然会overfitting（比如每一个叶子节点都只有一个instance）

1) 给树的高度一个penalty（甚至直接规定最大高度

2) 或者先生成，再剪枝

10. Supervise learning II lecture 12　12/11

a. Metric

i. 性能指标（ Performance metrics ）

1) accuracy：猜对的阴阳性的数量 / 所有测试用例的数量

2) precision：真正猜对的阳性的数量 / 所有被你声称是阳性的数量

	Predicted Positive	Predicted Negative
Positive	True Positive rate	False Negative rate
Negative	False Positive rate	True Negative rate

$$accuracy = \frac{TPR \times N^+ + TNR \times N^-}{N^+ + N^-}$$

$$precision = \frac{TPR \times N^+}{TPR \times N^+ + FPR \times N^-}$$

$$recall = \frac{TPR \times N^+}{TPR \times N^+ + FNR \times N^+}$$

$$F - measure = \frac{2 \times precision \times recall}{precision + recl}$$

i. Generalization metrics

1) 步骤

a) 将数据随机分成两部分

b) 计算结果的均值和方差

c) 做出进一步的统计分析

11. Unsupervised learning

a. 概论

i. 和监督学习的关键词一样：representation + algorithm + evaluation

ii. 对比于监督学习

1) evaluation没那么准确，也更加主观

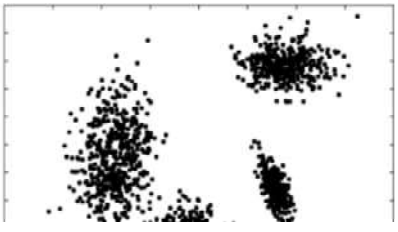
2) 非监督学习的问题是ill-defined（不清楚的），学得的东西也很可能来自于我们的intuition

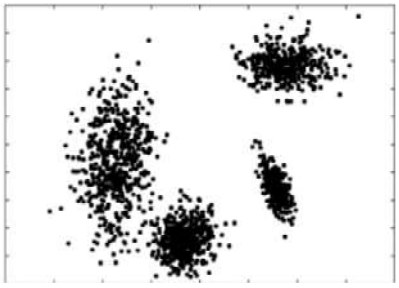
b. Clustering

i. 问题描述

1) 是非监督学习中的一个基本难点

2) 比如：把一堆数据分成若干个cluster使得cluster内的相似度（intra-clustering similarity）高，而cluster之间的相似度低





3) 在任何目标函数（objective function）下，clustering都是NP-hard的（解空间大小为K

ii. 方法

1) Naïve 方法：使用决策树
2) K-mean算法
   a) 先随机初始化若干中心
   b) 把点分入其最近的中心中
   c) 更新中心为以其中心的group中所有点的均值
   d) 重复1-3步直到没有任何改变

c. Learning Low-Dimensional Representations

i. 降维

1) 并不必要，但是在非监督学习中很有趣，因为其对人类观测这些数据很有帮助
2) Principal Component Analysis（主成分分析）
   a) 首先，应该使用更低维度的向量表示这些数据
   b) 其次，要从各个维度来分析数据之间的方差
   c) 这两个情况使得主成分分析变成了Eigenvalue decomposition problem
3) Local linear embedding
   a) 能否使用低维的向量来map这些高维的例子
   b) 直觉上来说，似乎不可能