Machine Learning 10-701

Tom M. Mitchell
Machine Learning Department
Carnegie Mellon University

January 11, 2011

Today:
- What is machine learning?
- Decision tree learning
- Course logistics

Readings:
- "The Discipline of ML"
- Mitchell, Chapter 3
- Bishop, Chapter 14.4

Machine Learning 10-701

Tom M. Mitchell
Machine Learning Department
Carnegie Mellon University

January 13, 2011

Today:
- The Big Picture
- Overfitting
- Review: probability

Readings:
Decision trees, overfitting
- Mitchell, Chapter 3

Probability review
- Bishop Ch. 1 thru 1.2.3
- Bishop, Ch. 2 thru 2.2
- Andrew Moore's online tutorial

Machine Learning 10-701

Tom M. Mitchell
Machine Learning Department
Carnegie Mellon University

January 18, 2011

Today:
- Bayes Rule
- Estimating parameters
  - maximum likelihood
  - max a posteriori

many of these slides are derived from William Cohen, Andrew Moore, Aarti Singh, Eric Xing, Carlos Guestrin. - Thanks!

Readings:

Probability review
- Bishop Ch. 1 thru 1.2.3
- Bishop, Ch. 2 thru 2.2
- Andrew Moore's online tutorial

Machine Learning 10-701

Tom M. Mitchell
Machine Learning Department
Carnegie Mellon University

January 20, 2011

Today:
- Bayes Classifiers
- Naïve Bayes
- Gaussian Naïve Bayes

Readings:
Mitchell:
"Naïve Bayes and Logistic Regression"
(available on class website)

Machine Learning 10-701

Tom M. Mitchell
Machine Learning Department
Carnegie Mellon University

January 25, 2011

Today:
- Naïve Bayes
  - discrete-valued X_i's
  - Document classification
- Gaussian Naïve Bayes
  - real-valued X_i's
  - Brain image classification
- Form of decision surfaces

Readings:

Required:
- Mitchell: "Naïve Bayes and Logistic Regression" (available on class website)

Optional
- Bishop 1.2.4
- Bishop 4.2

Machine Learning 10-701

Tom M. Mitchell
Machine Learning Department
Carnegie Mellon University

January 27, 2011

Today:
- Naïve Bayes – Big Picture
- Logistic regression
- Gradient ascent
- Generative – discriminative classifiers

Readings:

Required:
- Mitchell: "Naïve Bayes and Logistic Regression" (see class website)

Optional
- Ng and Jordan paper (class website)

Four Fundamentals for ML

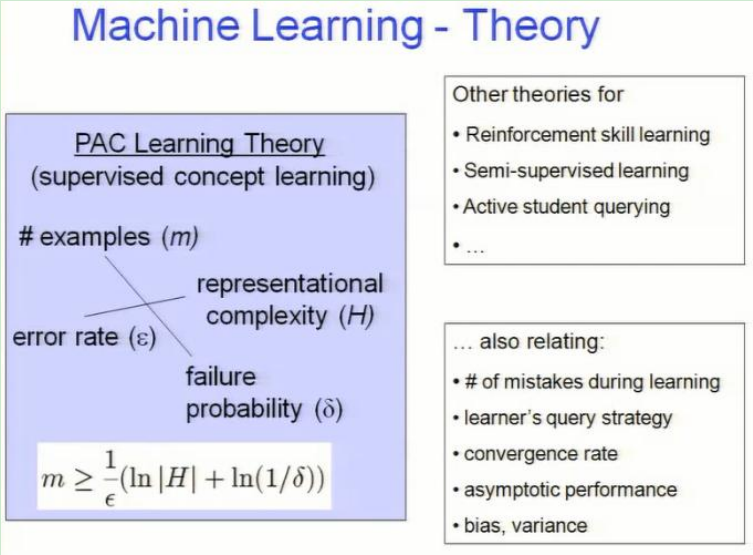
1. Learning is an optimization problem
   - many algorithms are best understood as optimization algs
   - what objective do they optimize, and how?
2. Learning is a parameter estimation problem
   - the more training data, the more accurate the estimates
   - MLE, MAP, M(Conditional)LE, …
   - to measure accuracy of learned model, we must use test (not train) data
3. Error arises from three sources
   - unavoidable error, bias, variance
4. Practical learning requires making assumptions
   - Why?
   - form of the f:X → Y, or P(Y|X) to be learned
   - priors on parameters: MAP, regularization
   - Conditional independence: Naive Bayes, Bayes nets, HMM's

ML 三要素+Function approximation

1Task 2performance(为了 improve) 3experience

等同于 Function approximation
Task ; function map x to Y
performance: accuracy of output
experience: train example input and output of function

PAC learn Theory



PAC learn Theory
#train example 和 error rate 的关系，moderate by 失败的概率，和假设的复杂性(模型复杂性)

Function approximation and decision tree learning

Function approximation

Problem Setting:
- Set of possible instances X
- Unknown target function $f: X \rightarrow Y$
- Set of function hypotheses $H=\{ h \mid h: X \rightarrow Y \}$

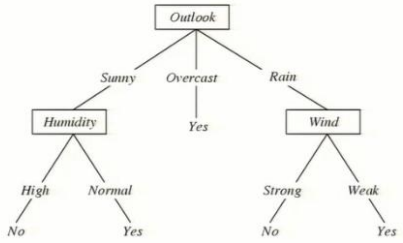
Input:
- Training examples $\{<x^{(i)}, y^{(i)}>\}$ of unknown target function f

Output:
- Hypothesis $h \in H$ that best approximates target function f

set of possible hypothesis H 可以选
然后通过 train example 选择最佳的 h,来 best aproximate the real function f

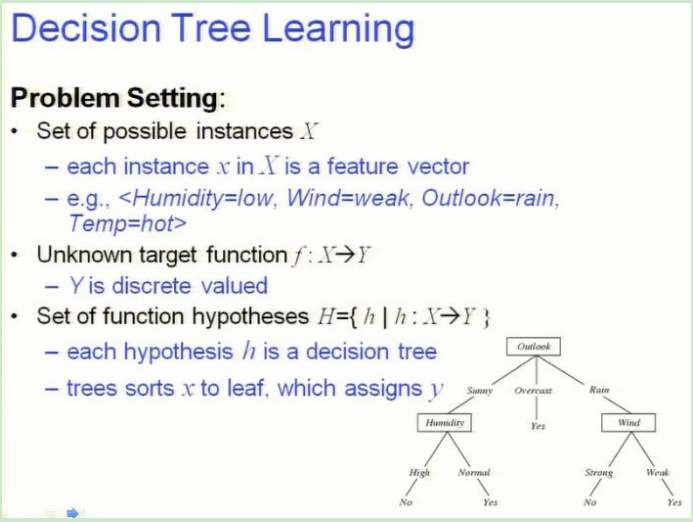
Decision Tree
我们的 H 是所有的 DT

A Decision tree for
F: <Outlook, Humidity, Wind, Temp> → PlayTennis?

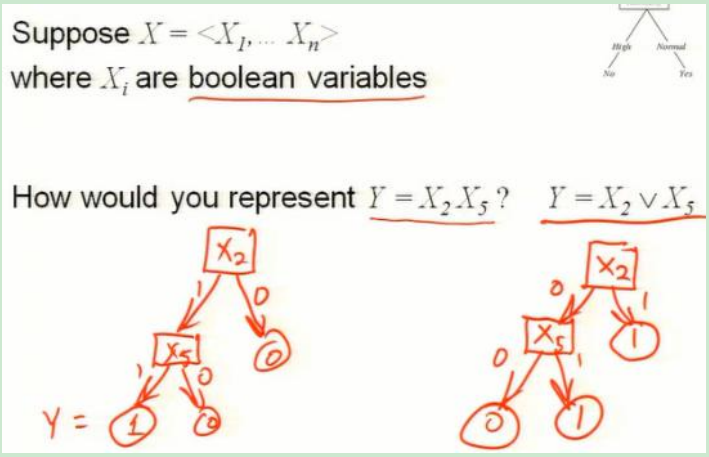


Each internal node: test one attribute X_i
Each branch from a node: selects one value for X_i
Each leaf node: predict Y (or P(Y|X ∈ leaf))

F 是 feature，天气来决定是否运动(label)



问题定义：
instance feature vctor
unknown function output 是离散值
hypothesis：全部 DT



假设 X 是 boolean variable Y 也是 boolean
如何表示 and 和 or 逻辑关系
用树来表示
DT 可以表示全部的逻辑函数吗？ Yes
要经常问，我们的模型可以表示任意函数吗!!

解读：833 正样本 167 负样本
FP root 节点有三个取值：123，但只有 1 有分支节点 PC

```
[833+,167-] .83+ .17-
Fetal_Presentation = 1: [822+,116-] .88+ .12-
| Previous_Csection = 0: [767+,81-] .90+ .10-
| | Primiparous = 0: [399+,13-] .97+ .03-
| | Primiparous = 1: [368+,68-] .84+ .16-
| | | Fetal_Distress = 0: [334+,47-] .88+ .12-
| | | | Birth_Weight < 3349: [201+,10.6-] .95+ .
| | | | Birth_Weight >= 3349: [133+,36.4-] .78+
| | | Fetal_Distress = 1: [34+,21-] .62+ .38-
| Previous_Csection = 1: [55+,35-] .61+ .39-
Fetal_Presentation = 2: [3+,29-] .11+ .89-
Fetal_Presentation = 3: [8+,22-] .27+ .73-
```

最后是比例(或概率)

a algorithm 根据 train set to generate DT
fit the best tree to fit train examples

Top-Down Induction of Decision Trees [ID3, C4.5, Quinlan]

node = Root

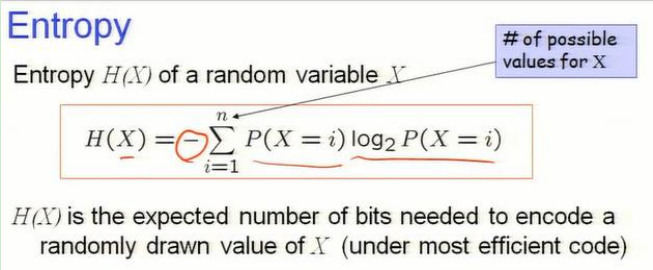
Main loop:

1. $A \leftarrow$ the "best" decision attribute for next *node*
2. Assign A as decision attribute for *node*
3. For each value of A, create new descendant of *node*
4. Sort training examples to leaf nodes
5. If training examples perfectly classified, Then STOP, Else iterate over new leaf nodes

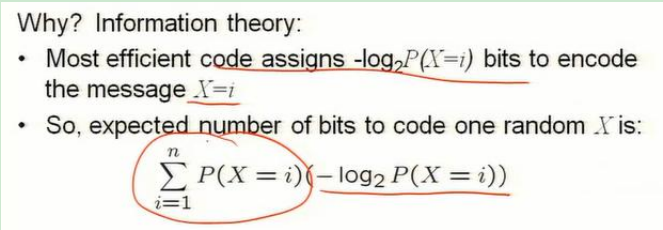
top-down 生成 tree，越往上重要性越大！
1 root best decision attribute
2 create 这个 node
3 将这个 node 的所有可能取值变成每个分支节点
4 把所有 example 按照节点的取值排序 sort(就是按这个节点划分样本)，看一些此时每个叶节点,example 的分布,如果分的比较纯就结束,如果还有些 mixed,可以继续添加 attribute，重复上面循环，增加子分支
所以上面每个 attribute，可以有多个取值
如何定义第一步的 best？？

Which attribute is best?

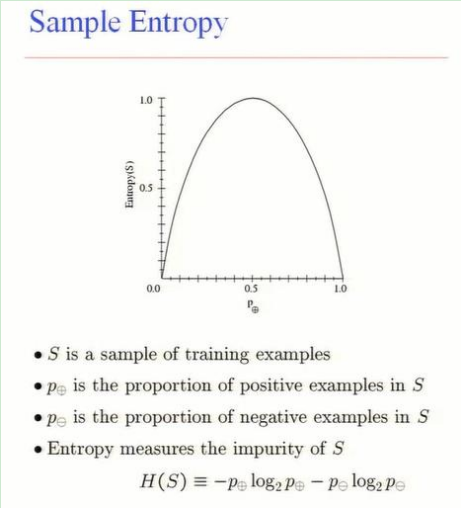
entropy:



1 来描述 homogeny 同质性或 skew 偏态
2 H 是为了 encode X(随机变量)平均需要多少 bits,X 可以取很多值,有 category 分布，每个值都有不同的概率，熵越大代表 X 越随机，越不确定，也就是实际 sample X distibution 时，得到的 X 的取值种类越杂!!!



每个不同 Xi 取值信息长度不同，-log 是每个 Xi 的 bit 长度，乘以概率就是所有 X 可能取值期望



Sample entropy 很重要的概念，将上应用在 DT 的关键
假设 S 是一个 sample examples，包含+ -样本，此时 X 就是 label+-,此时对于该 sample 的 H 计算如上式，代表 S 的纯度，0.5 最高，越不纯！
Sample entropy 是为了选择最初的 root best 节点

conditional entropy
特定条件概率，也就是经过这个 Y 节点 sort 后，Y 的特定一个分支所划分的 sample 的 H!!

Specific conditional entropy $H(X|Y=v)$ of X given $Y=v$:

$$H(X|Y=v) = -\sum_{i=1}^{n} P(X=i|Y=v)\log_2 P(X=i|Y=v)$$

给定 Y=v 下 X 需要的平均信息长度，熵的两边都加条件不变！

或者我们想知道对**所有 Y 的 v 取平均条件下**：得到最终的平均的**平均长度**是下面期望：
也就是经过这个 Y 节点 sort 后，Y 的每个分支所划分的 sample 的 H 的均值!!

Conditional entropy $H(X|Y)$ of X given Y :

$$H(X|Y) = \sum_{v \in values(Y)} P(Y=v)H(X|Y=v)$$

p(Y=v)就是每个分支数量比例

mutual information
information gain:
也就是经过 X 节点 sort 之后，对于 **H(Y)减少的信息量,我们目标是想让 H 减少!! 纯度增加!!**

Mututal information (aka Information Gain) of X and Y :

$$I(X,Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$$

Information Gain is the mutual information between input attribute A and target variable Y

Information Gain is the expected reduction in entropy of target variable Y for data sample S, due to sorting on variable A

$$Gain(S,A) = I_S(A,Y) = H_S(Y) - H_S(Y|A)$$

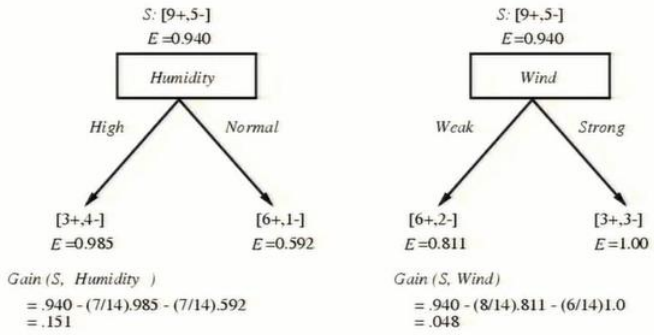
A1 和 A2 是两个不同的 attribute Y 是 label(+-) 求的是 Y 的 H
H(Y) 是原本的[29+,35-]
H(Y|A)是**条件熵**，应该是算 A1 两种结果熵的均值！
左边 sort on A1
右边 sort on A2
分别计算 **information Gain** : entropy of Y from data S,due to sort on A
在 A 排过序后，S 的信息熵的**减少**

Training Examples

Day	Outlook	Temperature	Humidity	Wind	PlayTenn
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Selecting the Next Attribute

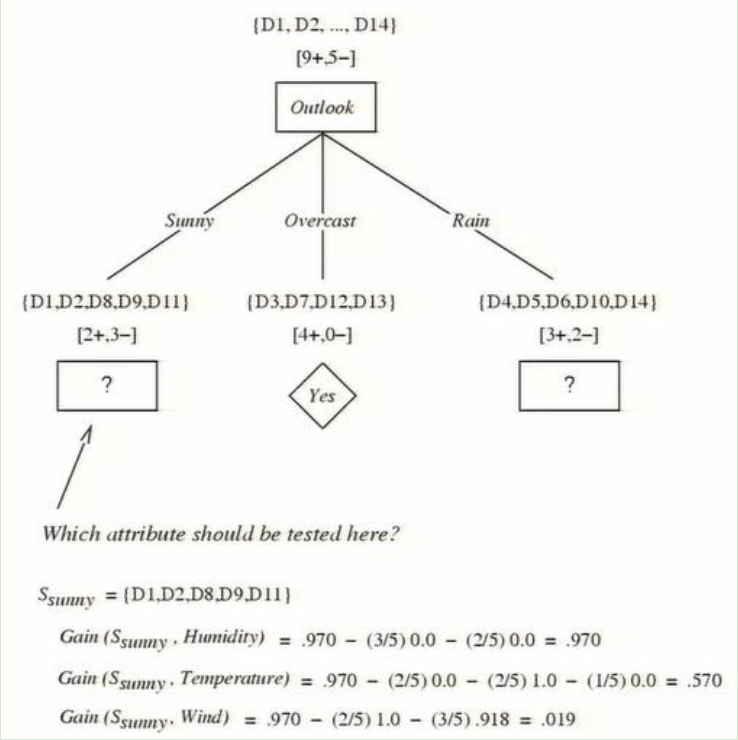
Which attribute is the best classifier?



Gain (S, Humidity)
= .940 - (7/14).985 - (7/14).592
= .151

Gain (S, Wind)
= .940 - (8/14).811 - (6/14)1.0
= .048

因为 A 取多个值 A1 A1，求的是 H(Y|A)
H(Y)-H(Y|A)
可以看出左边减少的更多，左边的 attribute 是 best
information gain 和 conditional entropy;是为了在确定根节点的条件下，确定下一个 best 节点



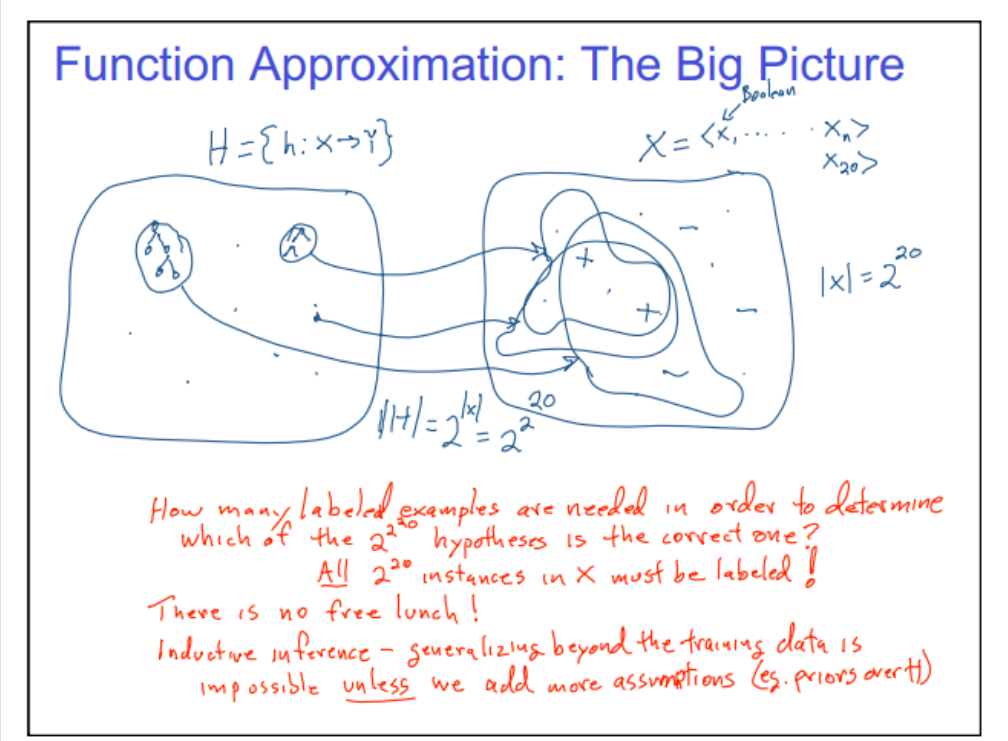
按照上面方法我们选取 best 是 outlook attribute(有三个取值)
其中一个 overcast 节点已经 pure，可以结束，其他节点任然 iteration，再选 best

我们的最终目标就是**每个叶节点都是 pure!!!** 也就是有这些 attribute 所有取值的组合，最终都会告诉你+-的概率!!! **每个特定取值的 attribute 都对应一个不同的分类概率**
再回头看这个就看懂了：

```
[833+,167-] .83+ .17-
Fetal_Presentation = 1: [822+,116-] .88+ .12-
| Previous_Csection = 0: [767+,81-] .90+ .10-
| | Primiparous = 0: [399+,13-] .97+ .03-
| | Primiparous = 1: [368+,68-] .84+ .16-
| | | Fetal_Distress = 0: [334+,47-] .88+ .12-
| | | | Birth_Weight < 3349: [201+,10.6-] .95+ .
| | | | Birth_Weight >= 3349: [133+,36.4-] .78+
| | | Fetal_Distress = 1: [34+,21-] .62+ .38-
| Previous_Csection = 1: [55+,35-] .61+ .39-
Fetal_Presentation = 2: [3+,29-] .11+ .89-
Fetal_Presentation = 3: [8+,22-] .27+ .73-
```

我们最终 approximate 一个 function
X 是 category feature Y 是 1 or 0
此时我们输出的是概率(不是纯的)

1 big pic



右边是全部的 data set，左边是全部的可能的 hypothesis

左边是 hypothesis space，右边是 example space(每个都有自己的 label Y)

1 需要多少 example 来决定哪个 tree 是正确的那个？需要全部的 example

正确的 tree 是将全部 example 分类正确！如果少的 example，会有很多 tree 将这些少量的 example 分类正确

2 用少量 example 得到的 tree 不可能 generalize to future data，因为不是 true function，除非我们增加一些 prior of H

function approximation!!!

假设都生成 complete tree（全部 feature 都用上!），而且 DT 可以 represent 任意 boolean function

1 如果 20 个 boolean feature，共 2 的 20 次方 examples，记为 |X|

2 会生成 dep=20 的 complete tree，每一层的 leaf：2,4,8

最后一层 2 的 20 次(X)的 leaf，也就是每个 example 都有一个自己的 leave

每个 leaves 要么是 true 要么是 false，因此{h|是 2 的|X|次方

假如只有 4 个 example，一定有很多 DT 可以正确的分类这些 example

只有 X example 时,我们才能学到唯一一个 DT （查看每个 xample 的 label，每个 example 都分对），也就是我们的 true function

如果只用 X-1 个 example，会有 2 的 1 次方个 tree(对这一个 example label True 或 False)

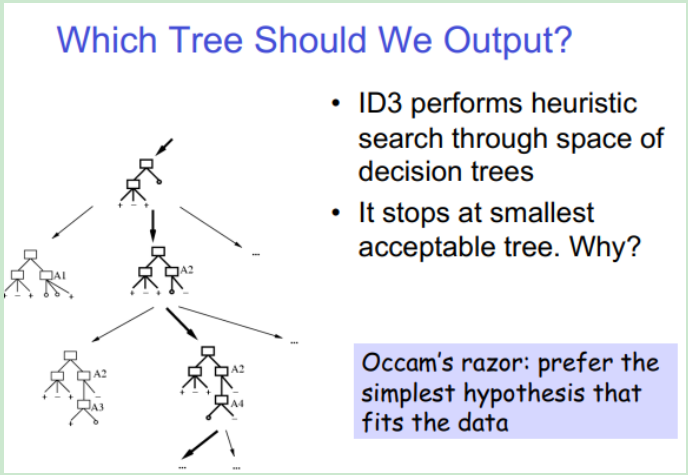
即使用多个的 DT model 投票也是 0.5:0.5 没用

Approximize function，我们只有 subset of example，得到的 DT 只是能近似，不能对全部的 example 分类正确

其实就是，serach from the h space，hoping to classify the train，and hopefully for the residual data
hypothsis 作为一个备选方案，为了近似 function

解决：
prior 先验，constrain：
hard：只选 dep=3 的 DT,会 miss 掉很多更合适的 h

如果是对于 assign 每个分类概率值的算法模型，可以对不同的 h 赋予不同的 prior 概率
soft prior：dep=3 找不到可以 try dep=4
更常见是给每个 h 一个概率 distribution(先验 p(theta))



ID3 会在 DT space 里 search
只需最 simplest hypothesis fit the data
有个缺点：只用小的 model，会 miss 掉我们 fit 的 hypothesis 恰好是 true function 的机会!!
why we use this idea?
假如我设计机器人，我们只给他们有限的几个 sensor，这些 sensor 只能构成有限的 example(得不到全部的 example)，所以只需简单的 model

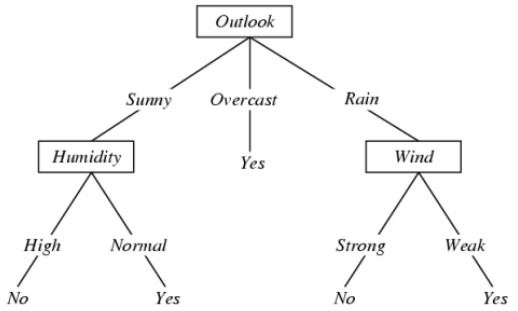
Overfitting：

Overfitting in Decision Trees

Consider adding noisy training example #15:

$Sunny,\ Hot,\ Normal,\ Strong,\ PlayTennis = No$

What effect on earlier tree?



增加几个 noise 错误的 example，可能会得到其他的 tree，但是错误的 tree

overfitting: pick the h do well in train, do terriblely on test

Overfitting

Consider error of hypothesis h over

- training data: $error_{train}(h)$
- entire distribution $\mathcal{D}$ of data: $error_{\mathcal{D}}(h)$

Hypothesis $h \in H$ **overfits** training data if there is an alternative hypothesis $h' \in H$ such that

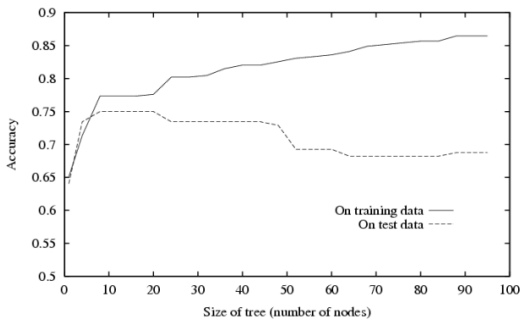
$$error_{train}(h) < error_{train}(h')$$

and

$$error_{\mathcal{D}}(h) > error_{\mathcal{D}}(h')$$

如果存在一个 h’，在 test 上更好，在 train 上更差

Overfitting in Decision Tree Learning



size 越大，越 overfitting

Avoiding Overfitting

How can we avoid overfitting?

- stop growing when data split not statistically significant
- grow full tree, then post-prune

1 set termination: 叶节点的数量限制，或者假设检验通过了**假设检验**再 split
2 grow full tree 然后剪枝(**实际效果最好!!**) 用 hold-out data 来 prune the tree(validate)

Reduced-Error Pruning

Split data into *training* and *validation* set

Create tree that classifies *training* set correctly

Do until further pruning is harmful:

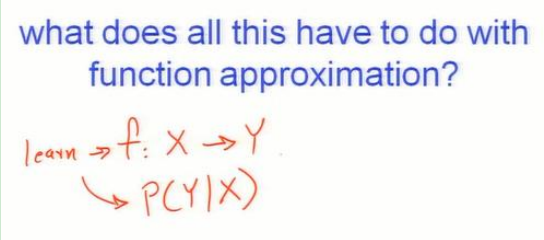
1. Evaluate impact on *validation* set of pruning each possible node (plus those below it)
2. Greedily remove the one that most improves *validation* set accuracy

- produces smallest version of most accurate subtree
- What if data is limited?

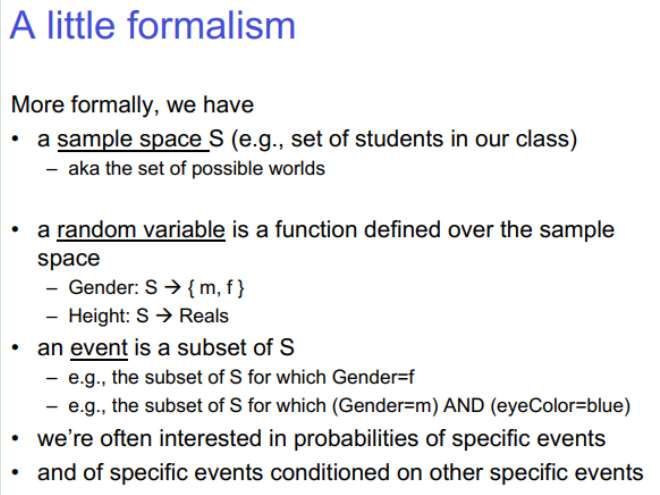
是否剪枝，根据 validation

3 将 train 分成不同的部分，分别 train 不同的 tree，然后 vote!!

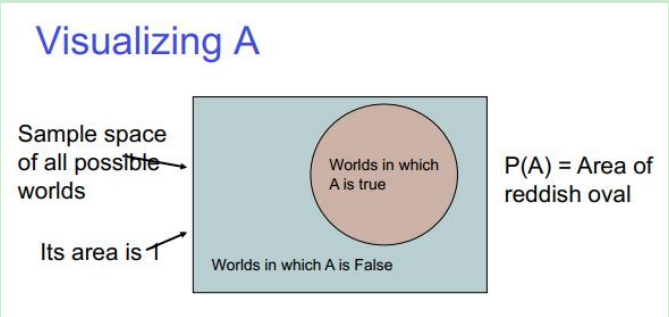
why probability？and function approx

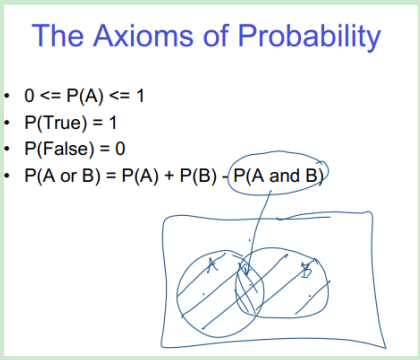


之前是 learn: function X->Y **determinate function**
现在可以 learn ： p(Y|X) distribution，现实生活中更常见!!!
是等价的



experiment 和 random variable 对应，而 random variable 是 sample space 的函数，函数值是实验结果
event 是 S 的子集，一般是定义在实验结果的基础上





sum rule 全概率：

p(B)=P(B,A)+P(B,not A)

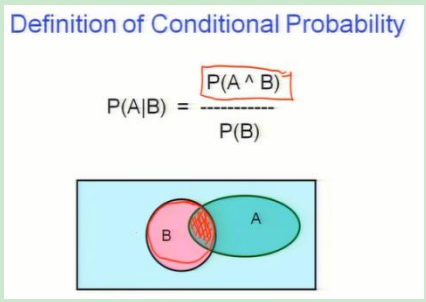
p(B)=P(B,A1)+ P(B,A2)+ P(B,A3)+…

The sum rule also holds if we split into three or more mutually exclusive and exhaustive events: if $B_i \cap B_j = \emptyset$ and $\sum_i P(B_i) = 1$ then

$$P(A) = \sum_i P(A, B_i).$$

p(B)=P(B|A)P(A)+P(B|not A)P(not A)

可以进一步分解



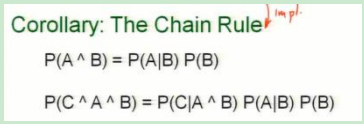
VN 图解释条件概率：

其实就是 A 与 B 相交的部分比上 B 发生的概率

这就是 B 发生的概率下 A 发生的概率

B 发生的情况下，A 要么发生，要么不发生

进而推出：chain rule，**通用的联合概率分解方式**



chain rule 两边加条件仍成立！

p(A,B|C)=P(A|C)P(B|AC)

independent event

P(A,B)=P(A)P(B)

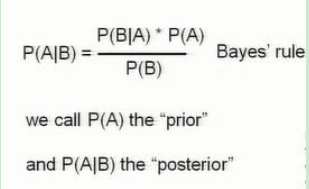
P(A|B)=P(A)　知道 B 对 P(A)没有影响

bayes rule

其实就是 chain rule 的两种分解

P(B)P(A|B)=P(A,B)=P(A)P(B|A)

P(A|B)= P(A)P(B|A)/P(B)



由先验得到后验

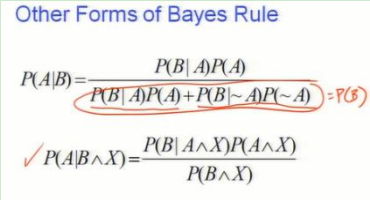
A 是 prior(hidden state) B 是 obs (evident)

后验就是根据观测来推测隐状态的概率!!!

直接计算后验比较复杂，转化成计算其他的概率

之前是 learn: function X->Y determinate function

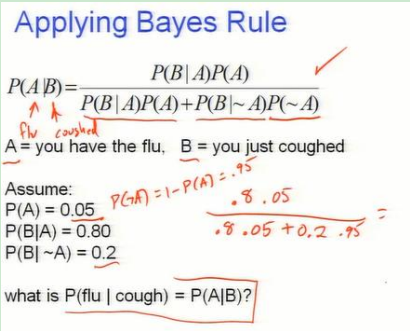
现在可以 learn ： p(Y|X) distribution



第一行用 sum rule，和 chain rule，进一步将 P(B)分解

第二行可以有多个条件，所有项加条件不变!!

example

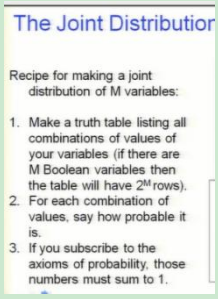


求后验就是，根据观测来推测隐状态的概率

joint distribution

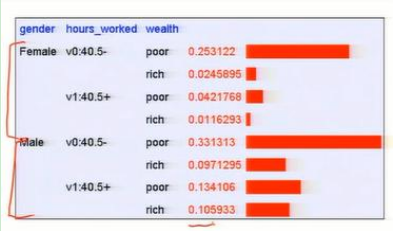
JD 是所有概率变化的来源

一旦有困惑，就回到 JD,回到本源!!!

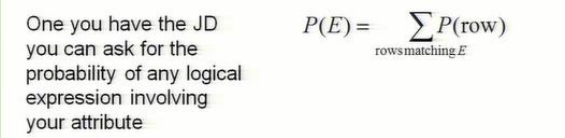


有了 joint distribution 可以计算 marginal/condition distribution 和 join 是基础!!!
有了 joint distribution，就足够了！不需要其它额外的信息了！
全部 join 概率值和为 1

另外我们记联合概率都是不考虑顺序的，任意一个顺序概率都相等



sample space 是 people
是对每个人的多个 random variable(不一定 independent)



marginal pro

有了 JD 可以回答任意问题：

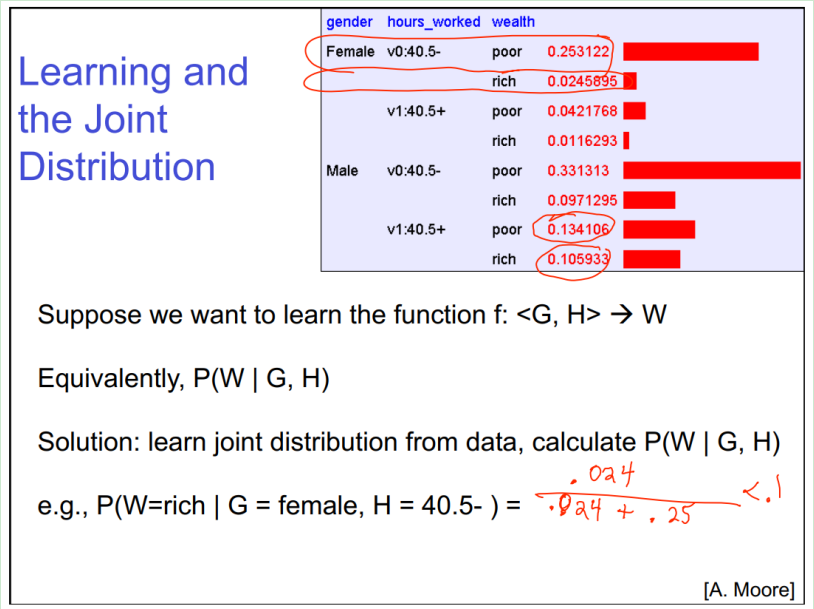
A	B	C	
1	2	1	0.05
1	2	2	0.3
2	3	1	0.2
2	2	1	0.04

可以问 marginal pro：P(C)=SUM(C) P(A,B,C)
其实就是 sum 所有 A，B 的行，如 P(C=1)

conditional pro

有了 JD 和 marginal，可以计算任意 conditional pro
P(A|B)=P(A,B)/p(B)

learn JD from data
target function=condition probability

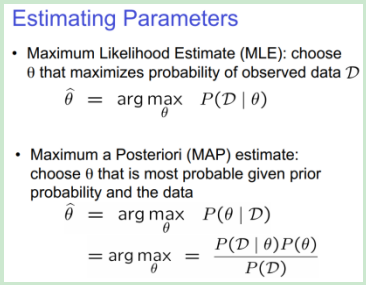


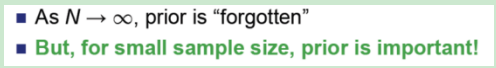
假设 我们的 function 2 个 feature 1 个 output
其实等价于 learn 条件概率：P(W|GH) 可以通过 JD 来求:P(WGH)/P(GH)
这样将 learning target function reduce 转化成 learning the JD!!!
其实就是拿到 data，我们生成一个经验的 JD,通过 count!!

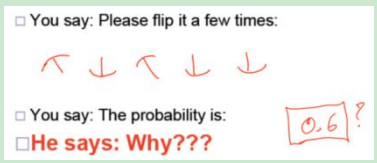
问题：
1 如果 exmaple 不够，这个 JD 表无法填满，无法继续推断!!
2 variable 太多时，JD 表太大，不现实!!!
所以直接 learn JD 不太好，最好是知道 data 服从的分布!!! 去 estimate 分布的参数!!!
上面联合概率的 8 个值是该分布的参数，但我们只需 7 个就可以!

learning and 分布的参数估计
也就是拿到一个分布的 sample，来估计这个分布的参数!!!
MLE 是我们标准的方法，来制定一个 training rule，learn some function，或 esitimate 参数 of 概率分布
find the 参数，来 max 概率 of observed train data

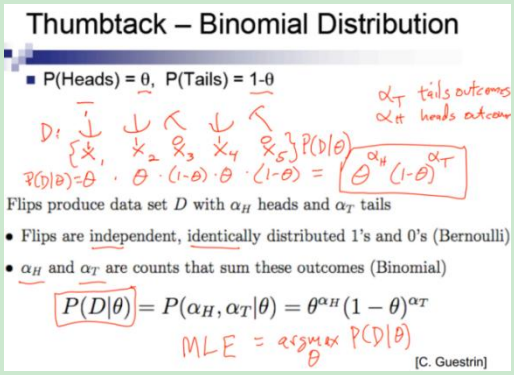
learning 是从 training data 里 estimate 参数
频率学派 - Frequentist - Maximum Likelihood Estimation (MLE，最大似然估计) P(D|theta) Likelihood
贝叶斯学派 - Bayesian - Maximum A Posteriori (MAP，最大后验估计)
posterior:P(theta|D)
MAP 比 MLE 的结果就是多了 imaginary example，防止 train 的特征值，test 未出现为 0 的情况







想知道 硬币正反的概率（参数）category 分布的参数，也就是假设我们的 data 服从 category 分布!! 需要估计该分布的参数!!



data likelihood，其实就是所有 example 的概率的乘积(data likelihood 不考虑顺序，因此就直接是概率乘积，其实也就是没有顺序的多项分布)
每个 example 都是一次实验的结果，data 是多次同分布实验的结果
example 的概率，就是联合概率!!!
如果只有一个变量，如抛硬币，联合概率是 P(X)
如果只有多个变量，如抛硬币再掷筛子，联合概率是 P(X,Y)

写 data likelihood，我们会默认概率分布的参数已存在，只是是变量 P(D|theta)，也就是给定参数

例如 p(X)是伯努利，参数是 theta，P(D|theta)， theta 也可以是其他分布的参数

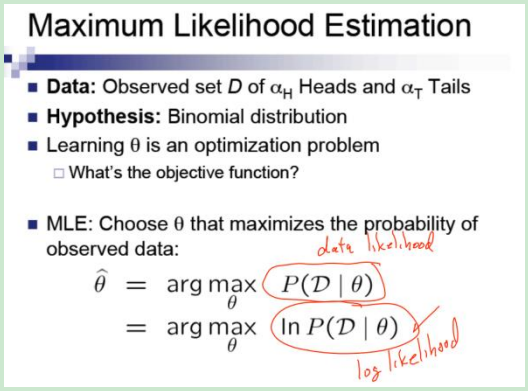
data likelihood 是不考虑顺序的 binomial distribution(二项分布)
也就是 P(D)服从是多项分布!!!

就是上面相乘 $\theta^{\alpha_H}(1-\theta)^{\alpha_T}$ 不考虑顺序，只考虑发生

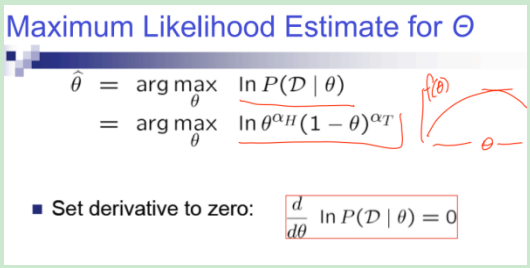
MLE: $MLE = \arg\max_\theta P(D|\theta)$

使得式子 $\theta^{\alpha_H}(1-\theta)^{\alpha_T}$ 值最大的 theta(也就是我们假定 theta 一定存在!!! 只

是 unknown)就是我们对 theta 的估计值
MLE 也就是当 theta 等于几时，该分布的 sample 更接近我们的 data

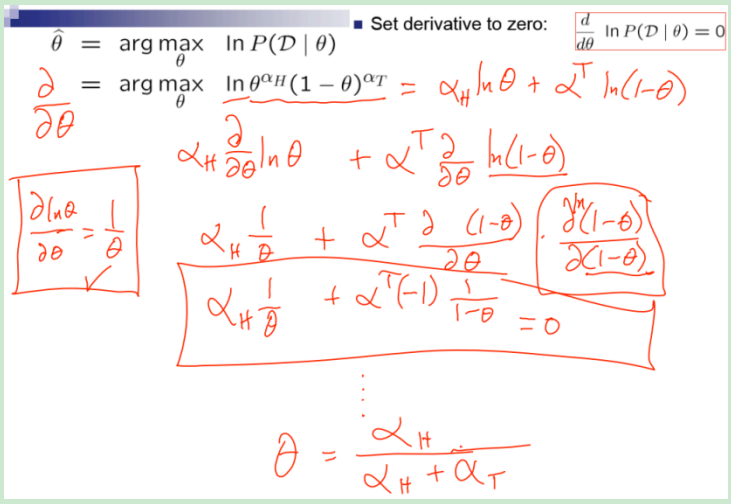


log 化 转成 log likelihood，转成 sum



求导为 0 求 theta

推导：



MLE estimate solution:
theta=head count/head + tail head
也就是我们观测到的 data 经验结果
也就是说通过多次的重复分类分布得到观测，用多项分布来反推分类分布的参数
多项分布可以作为我们的实验，得到 train example，每个 example 都服从同一个分类分布

同理，假设 data 服从高斯分布，我们想估计高斯分布的均值和方差，MLE 的结果也是经验均值和方差!!

多项分布补充知识：

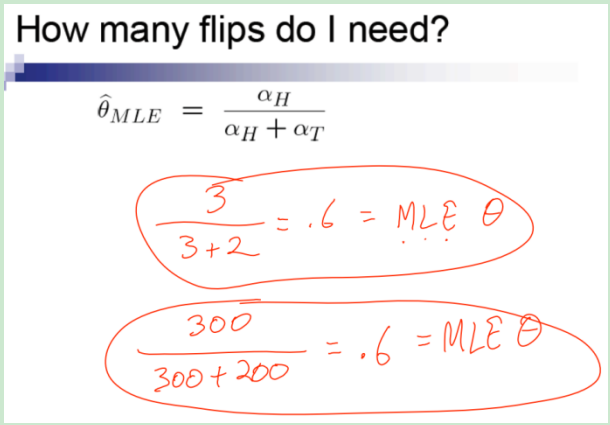
$$f(x) = P(X = x) = P(X = x|n, p) = C_n^x p^x (1-p)^{n-x}$$

二项分布是重复 n 次伯努利实验，2 分类变量分别出现次数的分布!! X 是次数，y 是概率。 因为次数包含很多种顺序结果，因此需要 sum

多项分布：是对于 n 个 catogory ，每一类有个出现的概率，然后实验多次，每一类出现的次数的概率也就是 p(X) X 是向量元素是每个类别出现的次数

x 项分分布其实就是次数的分布!!!

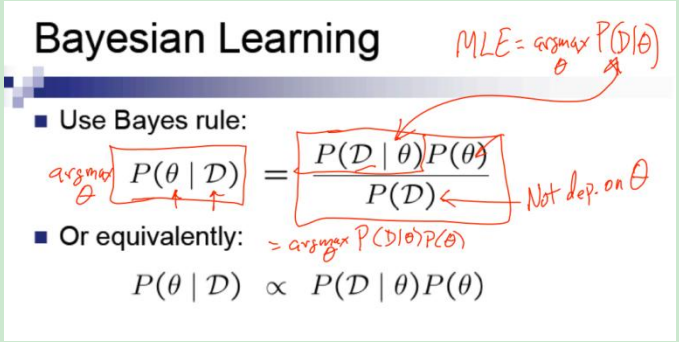
问题：



重复 5 次和重复 500 次结果一致，但第二个更可信
重复次数少会有 variance 大，得到的 estimate 有 variance
不一定每次重复 5 次都 0.6

1 数据量会影响结果
当数据量很小，为了减少 estimate 的 variance，可以给他一个 prior(MAP)
2 prior 也会影响，estimate 结果

MAP：最大后验估计 利用 bayes rule

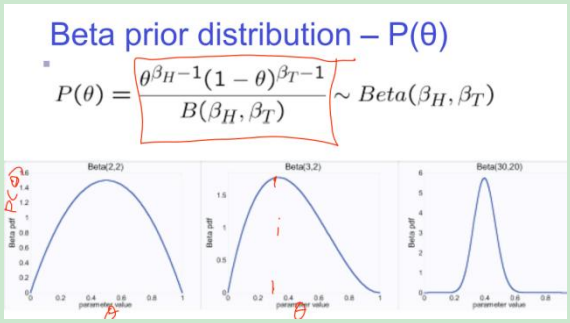


bayse rule:告诉我们如何利用 **prior** 和 **P(D|theta) data likelihood** to get **posterior 概率**
我们的目标是 **max p(theta|data)** 的 theta **跟右边分母无关**

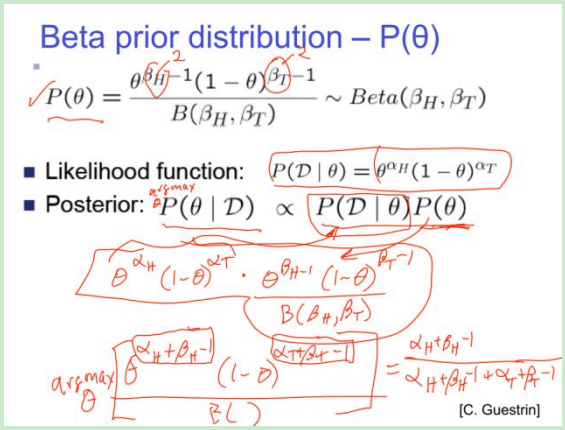
$$\arg\max_{\theta} P(\theta \mid D) = \arg\max_{\theta} P(D \mid \theta) P(\theta)$$

只需 max 分子，**比 MLE 增加了 prior**
p(thate)是 theta 的分布，如果是 uniform，p(theta)是常数，可以约掉，**MLE=MAP**
因此我们需要给参数选择一个 proir distribution(此时参数是概率!!)，不同概率的概率

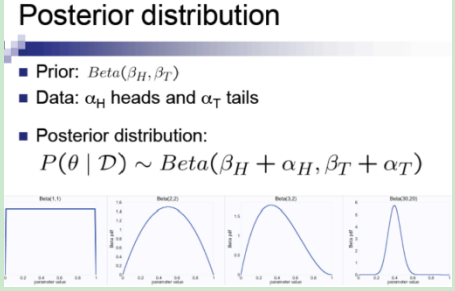
beta prior distribution
参数的分布，也是概率的概率分布



不同的分布，代表我们不同的 **prior**
分子就是我们充分 n 次正反的 count，count 是该分布的**参数**
分子就是我们的 **P(D|theta)likelihood 只是 count-1，是成比例的！(分母只是归一化的 constant)**

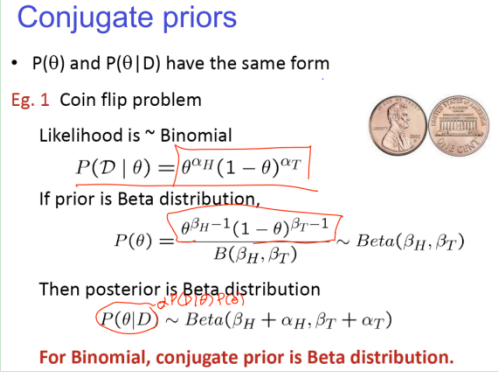


后验最终的结果：prior 和 likelihood 相乘后，还是 beta 分布
也就是当我们的 prior 是 Beta distribution，后验也是 beta 分布
然后 max 得到的 theta 其实就是**经验参数!!**



也就是：P(theta)和 P(theta|D) have same form(同一种分布!!!)

共轭先验：



P(theta)和 P(theta|D)have same form

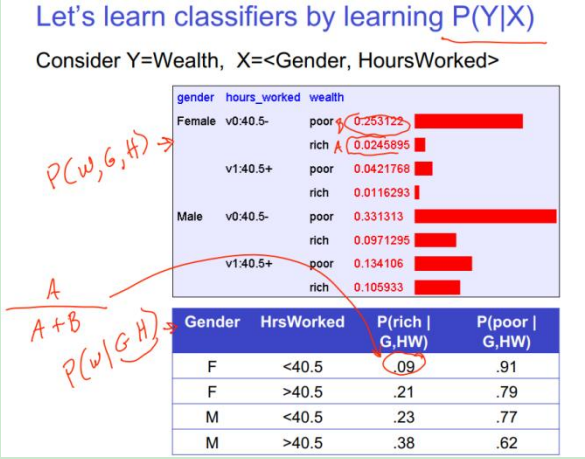
因为：Beta distribution 是共轭先验 of binomial distribution

从上面式子看出:p(D|theta) 和 p(theta) form 很像，当相乘时可以分子合并，分母不变,但仍然是 beta 分布,只是乘一个 scalar,后验=p(D|theta) *p(theta)，后验和 prior 一样也是 beta 分布

共轭先验是 choose the prior 分布，使得后验也服从该分布

learn condition function

learn condition function = learn approximate function

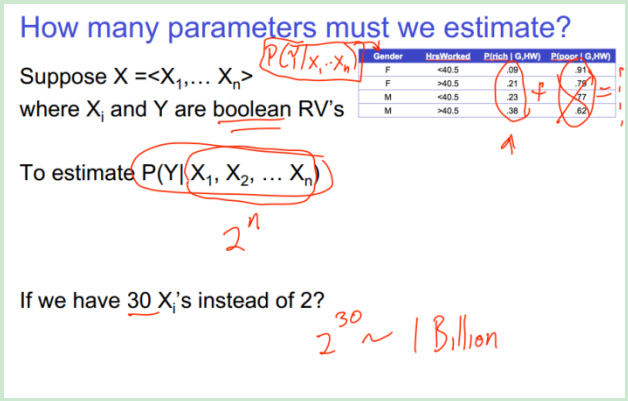


用 JD 来计算 conditional dsitribution

根据 defination: p(rich|G,HW)=P(rich,G,HW)/ P(G,HW)

conditional table 就告诉我们给定 G 和 HW，得到 W 的概率

P(Y=yi|X)也就是给定一个 X 值，得到每个 Y=yi 的概率，我们只需取概率最大的那个 yi，就是在 learn approximate function X->Y

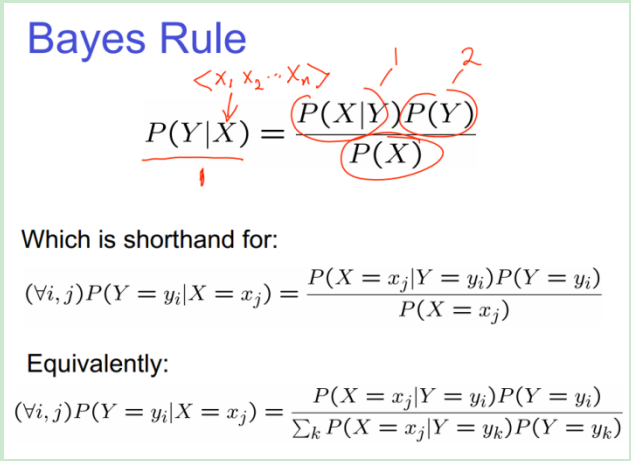


X 有多个特征 X1 X2..
想学习 P(Y|X1X2X3..),需要多少参数?
首先 X1X2X3..Xn(假设 boolean)有 2 的 n 方取值，每个取值对应不同的 Y(k 个值)，又因为给定 X 知道 Y k-1 个值的概率就知道第 k 个，因此需要 **2 的 n 次方*(k-1)**

因此，如果 X 是 boolean，Y 也是 boolean，如果有 n 个 X，需要学 **2 的 n 次方参数**
如果 30 个变量，参数太多了!!! 百万个
实际就是 JD table(X,Y)所需要的参数数量，我们需要至少**百万个数据**来填满 JD 才能计算 conditional table，不现实!

我们想通过 data 直接 learn conditional table(而不是先学习 JD)

bayes rule 帮助我们:



不需要直接 estimate p(Y|X)

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)} = \frac{P(X|Y)P(Y)}{\sum P(X,Y_i)} = \frac{P(X|Y)P(Y)}{\sum P(X|Y_i)P(Y_i)}$$

可以看出分子分母形式一样
我们可以 estimate 分子，有了分子，可以计算分母! 就可以计算 p(Y|X)，但需**要用到全部的 P(X|Y)这是参数**
可以用 **bayes rule reduce 参数?**

Can we reduce params using Bayes Rule?

Suppose X =<X_1,… X_n>
where X_i and Y are boolean RV's

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

how many params for $P(X_1 \cdots X_n|Y)$ $(2^n-1)\cdot 2$

how many for $P(Y) = 1$

P(Y)只有 1 参数
大 Xi 是 X 的第 i 个特征!!!
P(X|Y) 需要多少参数? P(X1, X2..Xn|Y)

Y 有 2 个取值假设，X 有 2 的 n 次方取值，但 sum 为 1 因此需要 $(2^n-1)2$

所有参数共有还是很多!!
bayes rule cannot help!!!

Naïve Bayes(是为了 learn conditional distribution)

Naïve Bayes 也就是在 bayes rule 上加了条件独立的假设!!
将参数个数从 exponential of n reduce to linear of n
assumption : 条件独立!! ML 非常重要的假设，make estimating feaiable

Naïve Bayes

Naïve Bayes assumes

$$P(X_1 \ldots X_n|Y) = \prod_i P(X_i|Y)$$

i.e., that X_i and X_j are conditionally independent given Y, for all i≠j

因此我们不再需要直接 estimate P(X|Y)
分解的每个因子是独立分布，有独立的参数个数，
假设 boolean 每个 p(Xi|Y) 2 个参数
共 2*n 个参数

Conditional Independence

Definition: X is conditionally independent of Y given Z, if the probability distribution governing X is independent of the value of Y, given the value of Z

$$(\forall i,j,k) P(X=x_i|Y=y_j,Z=z_k) = P(X=x_i|Z=z_k)$$

Which we often write

$$P(X|Y,Z) = P(X|Z)$$

E.g.,

$$P(Thunder|Rain, Lightning) = P(Thunder|Lightning)$$

light cause thunder!!! thunder happen together with rain
也就是 indirect interaction between Thunder 和 rain，可以被 lighting 解释
但 Thunder 和 lighting 是直接影响关系

Naïve Bayes uses assumption that the X_i are conditionally independent, given Y

Given this assumption, then:

$$P(X_1, X_2|Y) = P(X_1|X_2,Y)P(X_2|Y) = P(X_1|Y)P(X_2|Y)$$

Chain rule
Cond. Indep.

in general: $P(X_1 \ldots X_n|Y) = \prod_i P(X_i|Y)$

How many parameters to describe $P(X_1 \ldots X_n|Y)$? $P(Y)$?

- Without conditional indep assumption? $2(2^n-1)+1$
- With conditional indep assumption? $2n + 1$

Naïve Bayes 假设 X 的各个特征 Xi 对于条件 Y 独立!!
Naive bayes rule learn P(Y)和 P(X|Y)，其实就是 learn 了 P(XY)JD
这是 Naive bayes 的 learning 目标

Naïve Bayes in a Nutshell

Bayes rule:

$$P(Y=y_k|X_1 \ldots X_n) = \frac{P(Y=y_k)P(X_1 \ldots X_n|Y=y_k)}{\sum_j P(Y=y_j)P(X_1 \ldots X_n|Y=y_j)}$$

Assuming conditional independence among X_i's:

$$P(Y=y_k|X_1 \ldots X_n) = \frac{P(Y=y_k)\prod_i P(X_i|Y=y_k)}{\sum_j P(Y=y_j)\prod_i P(X_i|Y=y_j)}$$

index for X_i

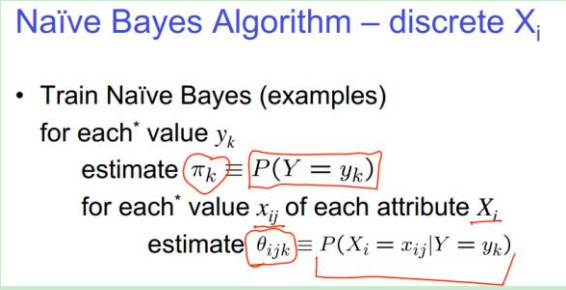
So, classification rule for $X^{new} = < X_1, \ldots, X_n >$ is:

$$Y^{new} \leftarrow \arg\max_{y_k} P(Y=y_k)\prod_i P(X_i^{new}|Y=y_k)$$

$$P(Y=y_k|X_1, X_2..X_n) = \frac{P(Y=y_k)P(X_1, X_2..X_n|Y=y_k)}{P(X_1, X_2..X_n)}$$

我们想 max condition proba，参数 Y， 分母 P(x)与参数 y 无关
条件概率跟分子成比例

最后得到 classification rule：
也就是给定 X 选择让 p(yi|X)最大的 yi，也就是让分子最大，也就带入我们从 data 里 learn 到的参数 P(X|Y)和 P(Y)来决定最后的 classification 结果!!!
如何 train and learn 参数？

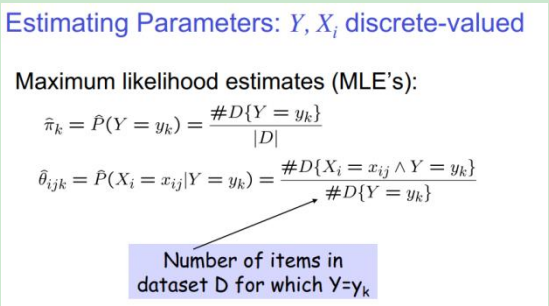


train：
共 2 层的循环
其实还是参数估计的方法：
对于 P(y) y 也是 category 分布，y 出现的频率服从 multinomial,可以写出 likelihood，然后 MLE
对于 P(Xi|y)不同的 y，Xi 也是 category 分布.....同上

1 遍历 yi
estimate 每个 y 取值的概率，就是用 MLE ,假设 Y 只取 0,1,就假设 data 服从 binomial 分布，得到的 esimate 的结果其实就是比例!!

2 再遍历每个特征 Xi
得出对于每个 y 值，对应的每个特征 Xi 的取值(j 个取值)的概率估计
因为对应每个特征都有一个独立分布 p(Xi|y)

同样对于 MLE 也是直接求比例!!
证明：当 Y=1 时，假设第 1 个特征 X1 取某一个值 x1j，假设 x1j=1 的概率，也就是求这些数据出现 1 的概率
Y=1 假设有 m 个 data(这个就是我们下面公式的分布，因为我们的观察数据限定在 Y=1)，可能会有多个 data 第 1 个特征取 x1j(0 或 1)，假设有 3 个取该特征值，因为每个 data X1 特征都是条件独立的，同服从伯努利分布，其实就是求重复 m 次，出现 0 或 1 的次数服从 binomial，可以写成 MLE，得到结果



上面 1,2 步， 分别为下面的 P(Y=yk),P(X new|Y)准备值
doing classify

- Classify (X^{new})

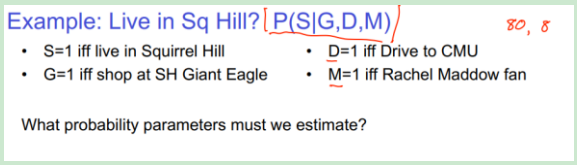
$$Y^{new} \leftarrow \arg\max_{y_k} P(Y = y_k) \prod_i P(X_i^{new} | Y = y_k)$$

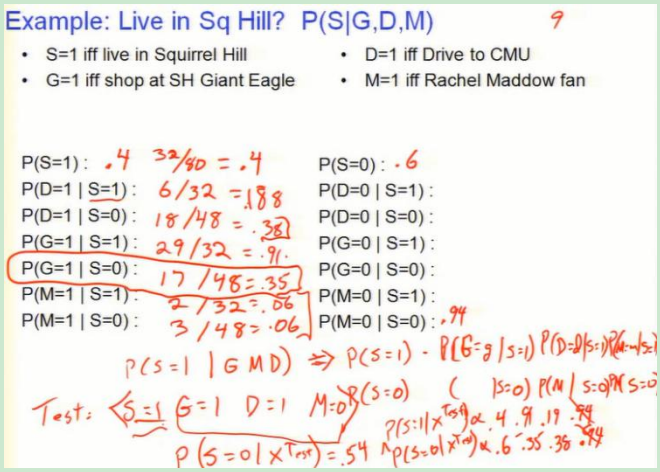
$$Y^{new} \leftarrow \arg\max_{y_k} \pi_k \prod_i \theta_{ijk}$$

probabilities must sum to 1, so need estimate only n-1 of these...

对于一个新的 X，获取所有需要参与计算的参数，然后选取 Y 最大的概率!!

example：





我们需要 P(Y)和 P(X|Y)

以上是我们的参数

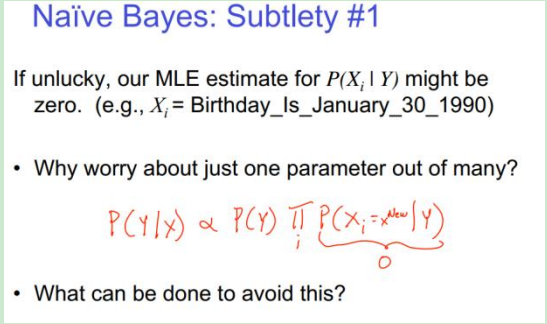
对于一个 test： G=1，D=1，M=0 S=1

分别带入 P(s=1|GMD test)和 P(s=0|GMD test)，哪个大就取哪一个

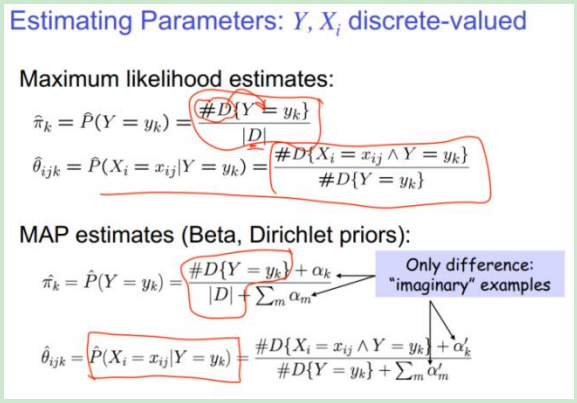
question 万一数据还是不够呢，也就是 X new 的特征值没有该参数？是需要 EM 吗？见下面的问题 1

问题 1：

如果一个 feature 是出生年 year，但 train 里没有出现，test 出现了，train 在 MLE 时会将 test 的出生日期条件概率 MLE estimate 为 0，此时计算条件概率时也是 0



product 为 0，这样不合理，可以用 MAP，给参数一个先验，在估计参数时增加个 prior，也就是给参数一个 prior
MAP 的结果，只是在 MLE 的基础上增加了 hallucinated examples：
MAP 的 prior 一般是 Dirchelet，该分布的参数是 beta，就是每个类别 hallucinated examples 的数量 question 带考证!!!

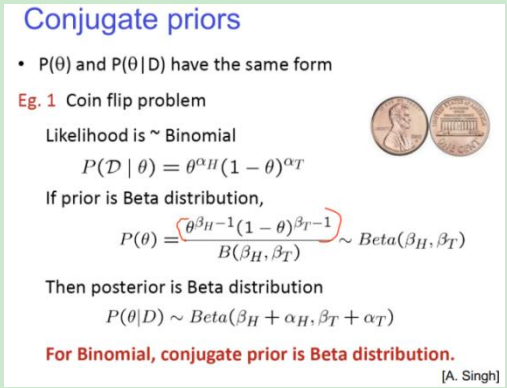


也就是为所有可能的 feature，增加一个 prior 概率，每个 year 对应的不同概率？
MAP 的结果：相当于对 feature 的所有可能每个取值增加一个 imaginary example 加入对于出生年. 需要加 365 个 imaginary examples alpha m，也就是假的每个日期的数量，这个数量是预先设定的，可以是 uniform，至少每个日期都是 1

以上是 MAP 的 uniform prior
question 这一块不是完全清楚，怎样得到上面的结果
其实就是给 y 的每个分类都默认为 1，统计的时候，无论有没有，都要加上这个 1!

以上只是对于离散变量的处理，可以这样统计，如果对于连续，就不能这样统计频率统计了，需要概率密度 p 代替了，例如 p(X|Y)是高斯分布，此时只需估计高斯分布的参数 u 和 sigma 就好!!!

Dirchelet conjugate prior



prior 会影响 posterior

prior p(参数)和 posterior p(参数|data)形式一样!!

Conjugate priors

- P(θ) and P(θ|D) have the same form

Eg. 2 Dice roll problem (6 outcomes instead of 2)

Likelihood is ~ Multinomial($\theta = \{\theta_1, \theta_2, \ldots, \theta_k\}$)

$$P(\mathcal{D} \mid \theta) = \theta_1^{\alpha_1}\theta_2^{\alpha_2}\ldots\theta_k^{\alpha_k}$$

If prior is Dirichlet distribution,

$$P(\theta) = \frac{\prod_{i=1}^{k}\theta_i^{\beta_i - 1}}{B(\beta_1, \ldots, \beta_k)} \sim \text{Dirichlet}(\beta_1, \ldots, \beta_k)$$

Then posterior is Dirichlet distribution

$$P(\theta|D) \sim \text{Dirichlet}(\beta_1 + \alpha_1, \ldots, \beta_k + \alpha_k)$$

For Multinomial, conjugate prior is Dirichlet distribution.

但此时 birthday 不是 boolean 是 category variable
datalikehood 需要其他 distribution Dirchelet (dieshelet)
Dirchelet 作为 prior，对多个概率值的概率分布
p(theta)是 Dirchelet
首先是 category 分布，每种结果对于一个概率这是我们的参数
重复 n 次，得到观测，每种出现的次数服从 multinomial，也就是我们的 likehood
p(D|theta)是 Multinomial

后验 p(theta)p(D|theta) 还是 Dirchelet

问题 2:

Naïve Bayes: Subtlety #2

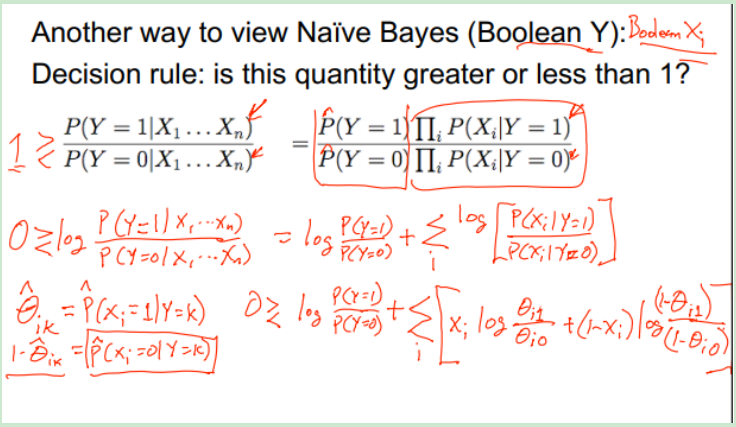
Often the X_i are not really conditionally independent

- We use Naïve Bayes in many cases anyway, and it often works pretty well
  - often the right classification, even when not the right probability (see [Domingos&Pazzani, 1996])
- What is effect on estimated P(Y|X)?
  - Special case: what if we add two copies: $X_i = X_k$

$$P(Y=y|X) \propto P(Y=y)\prod P(X_i=x|Y=y)$$

假设不一定成立
但实际应用效果好!!
假设有两个一样的 examples，一定不 condition independent
会使得右边乘以两次

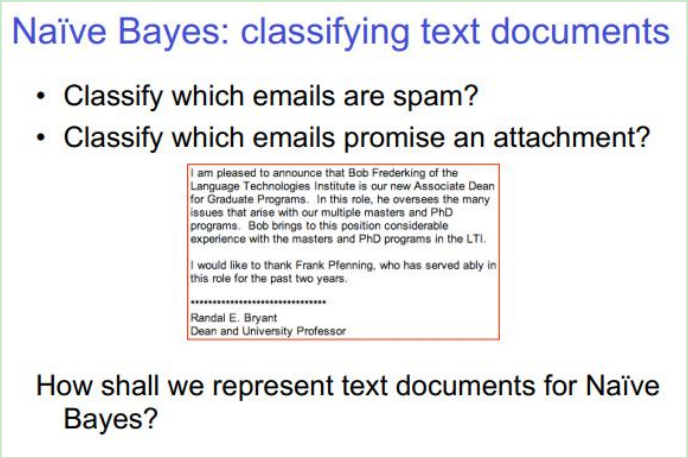
Naïve Bayes 另一种解释
假设 X Y 都是 boolean 时， Naïve Bayes 是 X 的 linear 函数！
可以将概率比值作为 decision rule，看看是否大于小于 1，来决定 Y
加个 log 就变成大于或小于 0,来决定 Y



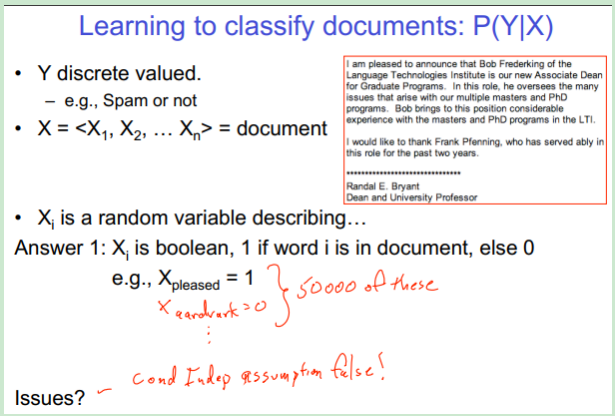
式子变成 X 的**线性组合和**的形式
左下是我们 **estimate 的参数**
右边处理时：因为 xi 可以取 0 或 1，因此可以作为 indicate 变量
将两种情况都写出来：左边是 Xi=1，右边 xi=0 1-theta
可以看出：decision rule 是 Xi 的 linear function！ X 和参数 theta 的线性组合(前提是 X Y 是 boolean) 最终分类变成了 linear function 大于小于 0
此时是 linear decision boundary

如果 X Y 是其他类型变量，结果就不是线性分类器，可能是非线性分类器！
question 先试着写出如果 X Y 是多分类，可以先 onehot encoding！

classify text documents



首先保证不同长度的 text，有相同的长度的 feature
这就需要**固定长度的 vocabulary！！**

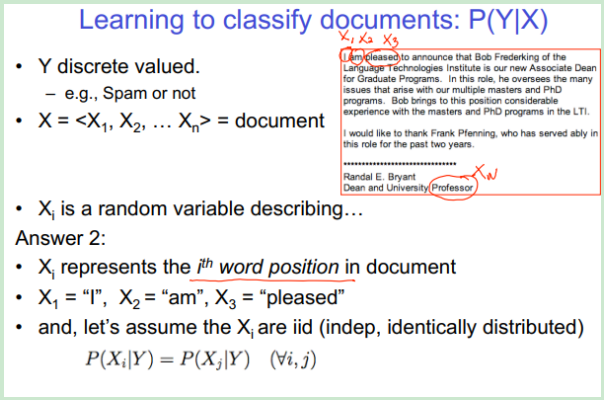


如何为文本定义**特征 X**

1 假设共有 50000 词汇，**每个文本定义 50000 个 Boolean 特征 X**，也就是文本的每个 word **是否出现在** dict 里

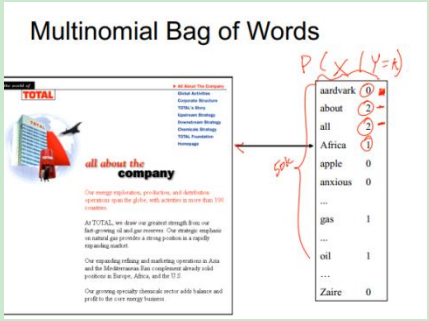
同样需要 estimate p(Spam) p(word|Spam)

问题：1 word 不一定 conditional independent **2 忽略了每个词汇的出现频率**



2 每个 word 都是 dict 的一个 sample，follow **category distribution**

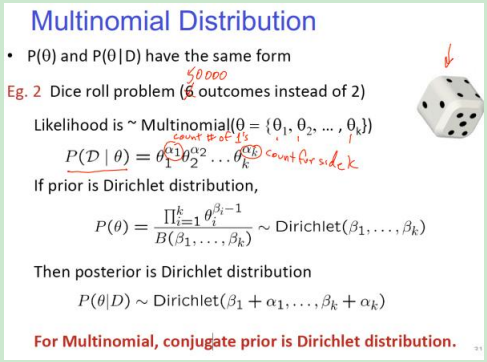
因此每个 document 都是一个 dict 长度的，记录每个 word 的频率(bag of word，统计**词频**)，**就是 document 的 feature！服从多项分布**



假设每个 word 都是 iid 相当于每次掷 50000 的面的骰子(Category distribution)

每个 word 都服从 category 分布，每个 document 相当于是多次重复 category 实验，我们只需统计出所有 word 出现的次数，每个文档所有 word(word 词频)**服从 multinomial distribution ，所有文档的词频也服从 multinomial distribution，也就是 datalikehood p(D|theta)**

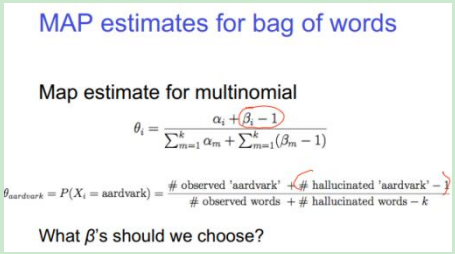
所有负样本的 data likelihood p(D|not spam)
所有正样本的 data likelihood p(D|spam)
MLE MAP 以后得到 p(Xi|not spam) p(Xi|spam) 概率分布参数



还需要个一个 prior p(theta)，应该是 Dirchelet prior，参数是 beta

$$\arg\max_{\theta} P(\theta \mid D) = \arg\max_{\theta} P(D \mid \theta) P(\theta)$$

MAP 得到的 theta 就是给定 Y 下 X category 分布的概率 P(Xi|Y)
（一个 document 一个 data，用到了全部的 document，其实也就是所有 example 里某个 word 的频率）



beta 就是我们 Dirchelet 分布的参数!! 代表先验的 imaginary example 数量

prior 的参数 beta 如何选？可以从其他的大量的 web 文本里面获取 word 的频率，然后乘以 5000 得到我们的实际 count beta!!

MLE 也可以得到 P(Y)

最终可以将 theta 带入条件概率公式
P(Y|X)=P(Y)P(X|Y)=P(Y) π P(Xi|Y) 进行分类

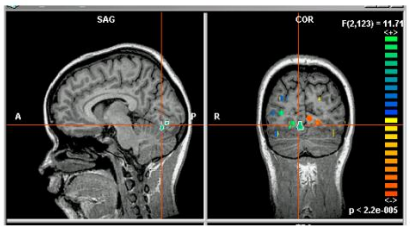
Gaussain Naive Bayes

when X is continue value

MRI 例子

What if we have continuous X_i ?

Eg., image classification: X_i is real-valued ith pixel



每个图片一个 example，每个像素一个特征 X 都是 continue value

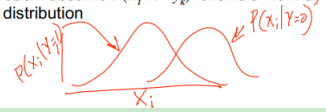
What if we have continuous X_i ?

Eg., image classification: X_i is real-valued ith pixel

Naïve Bayes requires $P(X_i \mid Y=y_k)$, but X_i is real (continuous)

$$P(Y = y_k|X_1 \dots X_n) = \frac{P(Y = y_k) \prod_i P(X_i|Y = y_k)}{\sum_j P(Y = y_j) \prod_i P(X_i|Y = y_j)}$$

Common approach: assume $P(X_i \mid Y=y_k)$ follows a Normal (Gaussian) distribution

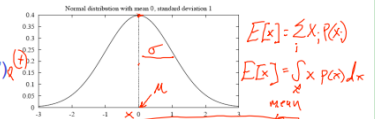


假设 p(X|Y=0 or 1)服从高斯分布，**不同的条件 yi 和不同的 Xi 特征**服从不同的高斯分布

Gaussian Distribution

(also called "Normal")

p(x) is a *probability density function*, whose integral (not sum) is 1



$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

The probability that X will fall into the interval (a, b) is given by $\int_a^b p(x)dx$

- Expected, or mean value of X, $E[X]$, is $E[X] = \mu$
- Variance of X is $Var(X) = \sigma^2$
- Standard deviation of X, σ_X, is $\sigma_X = \sigma$

是概率 density!!! 需要**积分!!**

What if we have continuous X_i ?

Gaussian Naïve Bayes (GNB): assume

$$p(X_i = x|Y = y_k) = \frac{1}{\sqrt{2\pi\sigma_{ik}^2}} e^{-\frac{1}{2}\left(\frac{x-\mu_{ik}}{\sigma_{ik}}\right)^2}$$

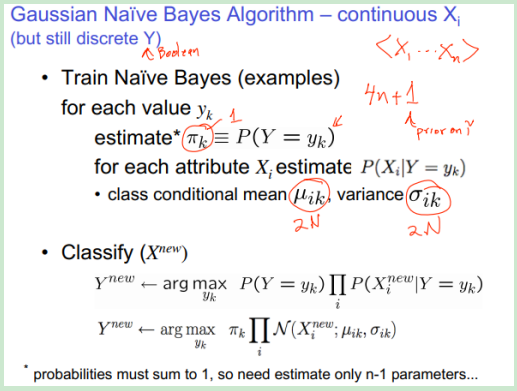
Sometimes assume variance

- is independent of Y (i.e., σ_i),
- or independent of X_i (i.e., σ_k)
- or both (i.e., σ)

假设：**不同的 Xi 特征** 和不同的 Yk 有不同的高斯分布!! **有独立的方差和均值**
如果假设 X 有 n 个特征，Y 是 boolean，需要多少参数？为了得到后验 p(Y|X)
prior 需要一个 p(Y)
conditional probability P(X|Y)：X 此时需要参数：均值和方差 2*2*n

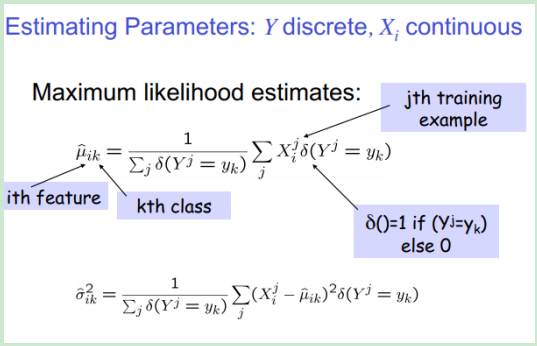
共 4n+1

如何 train?



p(y) 还是 MLE

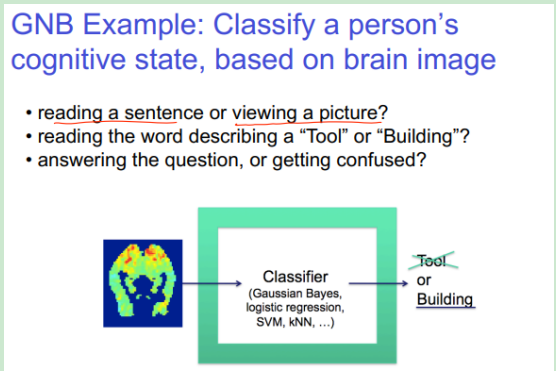
对每一个**高斯 conditional probability** 参数，用 MLE,也就是假设 data 服从高斯分布，**写出 likelihood** 用 MLE,结果是**经验参数(count)**

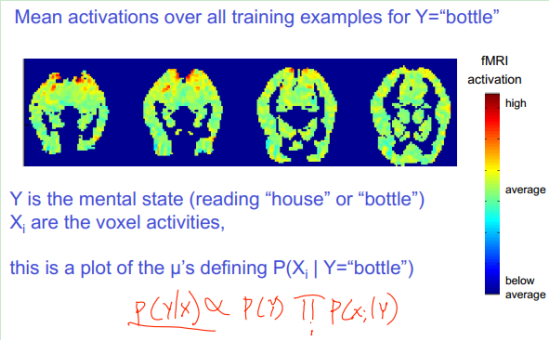


δ 是 indicate 函数

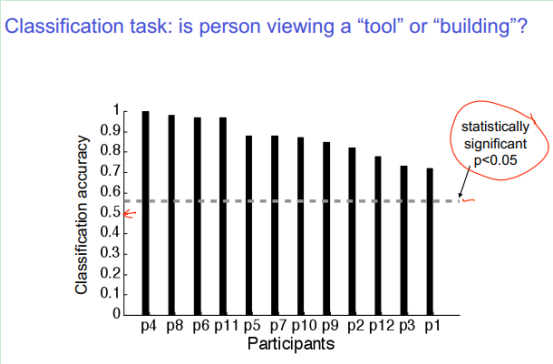
用 mle 估计均值和方差

MLE 就是当 y 固定时，求特征 Xi 的均值和方差!!



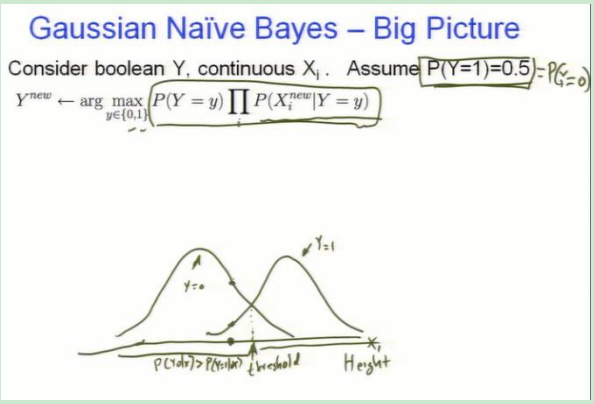


mental state 对于不同的 word
X 是大脑皮层活动　红色代表反映比较激烈的部位

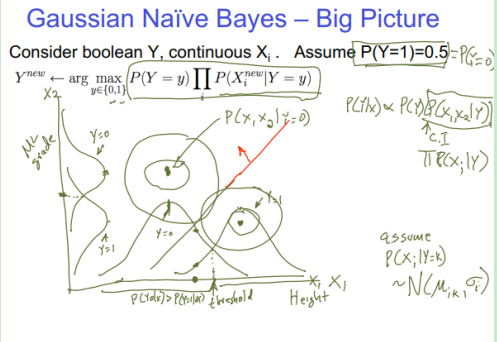


每个人一个 classifier

Gaussian Naive Bayes Big pic



X1 身高 single continuous feature　Y 是否是 basketball player
train Naïve Bayes 其实就是 use data estimate 两个高斯 distribution
这里假设 P(Y=1)=0.5
最右边是 y=1 的 distribution　左边 y=0
横坐标的任意一点可以预测 Y，谁的 Y 值大**预测谁！**
交叉点是 threshold(decision point)，来判断是 Y=1 or 0



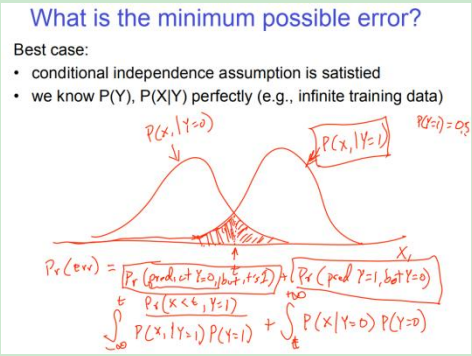
假如 X 是 2 维度：**X1 身高　X2 ML 成绩**
y=0 时需要对 X1 X2 分别画高斯分布
右边：因为条件独立，可以写成乘积的形式，也就是**两个高斯分布的乘积**
左边圆圈是高斯等高线图，也就是**两个高斯的最高点的乘积还是最高点，边缘的乘积还是边缘！实际变成了 2 dimension 高斯分布！**p(X1,X2|Y=0)
p(X1,X2|Y=0)也有一个分布
此时对于任意一个 X,还是可以带入求概率值,我们会发现,threshold 是个 plane!
红色

上面图像的假设是每个 P(Xi|Y)都有共同大小的方差，**因此二维的高斯是圆形！**

question 以上是概率密度，为什么概率密度可以直接可以直接应用于概率？
跟概率密度的定义有关？

Naïve Bayes 最小的错误

最好的情况，最小的错误！
1 假设满足条件独立假设
2 假设参数都知道，data 充足



其实就是 Y=0 的高斯分布也可能很大(长尾)，会进入 Y=1 分布，但我们会强制判断为 Y=1，因此会有 error，unavoidable error，也是 **best error(假设条件都满足,曲线正确)**
因为对于概率分布，面积是概率，p(error)其实就是**上面阴影的面积，积分和**

解释，如左边的一半面积：身高低会被预测为 Y=0 不是篮球队员，但实际存在身高低的篮球队员，因此会**预测错误**。这是
即使 data 都有了，假设也满足了，还是会有错误。
因为这就是 the case in the world，人类对于身高也只能猜测，一定有错误！

如何特征筛选？

1 top feature
statistic test: X 和 Y 的 mutual information，选出最高的几个
2 single best feature
top feature 可能 interdepent，需要选一个最好的
选择一个 X 跟剩余的 X mutu al information question?
question

Logistic Regression

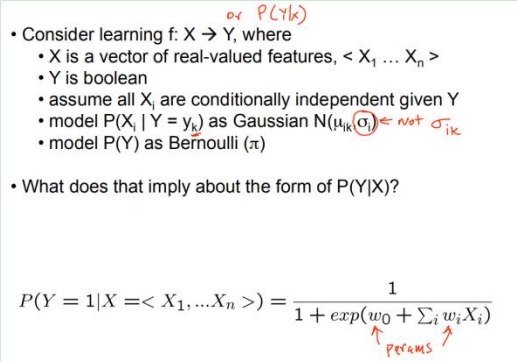
Logistic Regression

Idea:

- Naïve Bayes allows computing P(Y|X) by learning P(Y) and P(X|Y)
- Why not learn P(Y|X) directly?

前面 NB 和 GNB 都是利用 bayes 间接的 learn，learn P(Y|X)
P(Y|X)和 P(Y)P(X|Y)成比例，只需计算和比较 P(Y)P(X|Y)的值

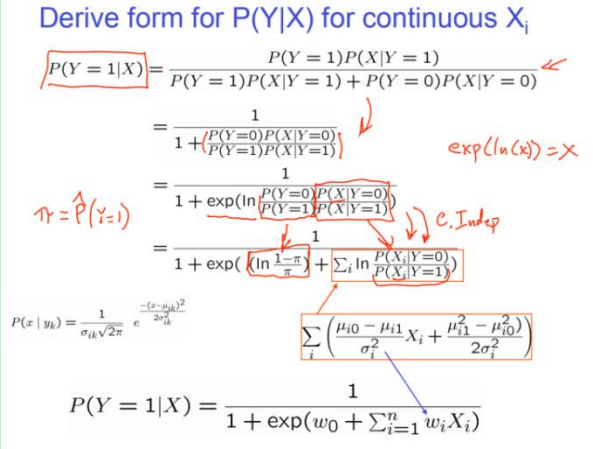
为什么不直接 learn P(Y|X),之前是因为参数太多
我们想看下，P(Y|X)真正的形式，而不是上面的比例形式，如果同样做了条件独立 assumption,p(Y)伯努利分布,p(X|Y)高斯分布,看看 P(Y|X)的形式,Gaussain NB 就是 logistic regression



还要多一个假设：p(X|Y)的方差只跟 feature Xi 有关,跟 Y 无关！

$$P(Y|X)=\frac{P(X|Y)P(Y)}{P(X)}=\frac{P(X|Y)P(Y)}{\sum P(X|Yi)P(Yi)}$$

结果是如上形式，w 是我们需要 estimate 的参数



推导：P(Y|X)

先考虑 Y=1. 运用 bayes rule

1 除以分子　2 exp ln 变换　3 ln 转换为和 右边又是条件独立，进一步变成和！

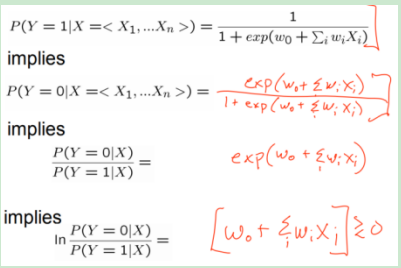
然后带入我们的假设的分布

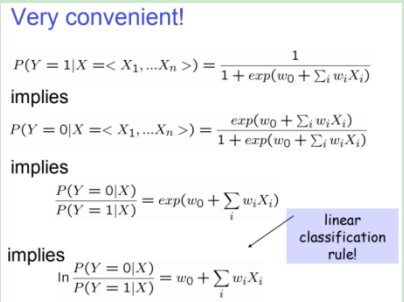
π 是伯努利的参数　　右边带入高斯分布的公式，就得到和的形式

然后就成了 logistics regression，伯努利参数和高斯参数都变成了 w 和 w0 了

也就是有上面的假设才能推导出 LR 模型!!! LR 模型的假设!!!

因此，我们可以直接学习 w 参数(不需要先 train 高斯参数)





因为是线性函数，会比较方便

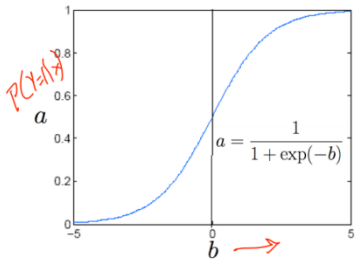
P(Y=0|X)+P(Y=1|X)=1

P(Y=0|X)和 P(Y=1|X)相比，得出如上结果

之前我们知道了，P(Y=0|X)和 P(Y=1|X)的比值大于小于 1 可以作为 decision rule，因此加上 log 就是 lr 的线性边界了，大于或小于 0 decision rule

decision rule ： linear term 是否大于 0
得到了一个 equivalent decision rule
此时不需要再从 bayes 开始计算了，变得更容易计算了，只是一个线性计算！
复杂的 bayse rule 等同于简单的线性计算!!

Logistic function



$$P(Y=1|X) = \frac{1}{1+\exp(w_0 + \sum_{i=1}^{n} w_i X_i)}$$

即使条件独立不满足，LR 比 naive bayes 效果更好!!

Logistic regression more generally

- Logistic regression when Y not boolean (but still discrete-valued).
- Now $y \in \{y_1 \dots y_R\}$: learn $R-1$ sets of weights

for $k<R$ $$P(Y=y_k|X) = \frac{\exp(w_{k0} + \sum_{i=1}^{n} w_{ki} X_i)}{1 + \sum_{j=1}^{R-1} \exp(w_{j0} + \sum_{i=1}^{n} w_{ji} X_i)}$$

for $k=R$ $$P(Y=y_R|X) = \frac{1}{1 + \sum_{j=1}^{R-1} \exp(w_{j0} + \sum_{i=1}^{n} w_{ji} X_i)}$$

如果 Y 是多 category R 个值
因为所有 Y 和为 1 ，分母是 R 项的和，最后一项的分子就是 1
因此只需要 R-1 套参数，对于每个 y 一套参数，最后一项是 1-前 R-1 项，不需要参数
因此对于同一个 X，可以计算 R 个 p(Y|X)概率值
Question 一般 softmax 没有 1，是全部的 exp?

train LR directly MCLE

直接 train 出 w,不需要再经过高斯和 bayes 的过程

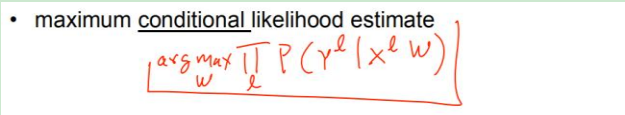
可以先用 MLE: 给定 W 下使得 X 的 likelihood P(<XY>|W)最大,选择 W

- we have L training examples: $\{\langle X^1, Y^1\rangle, \dots \langle X^L, Y^L\rangle\}$
- maximum likelihood estimate for parameters W

$$W_{MLE} = \arg\max_W P(< X^1, Y^1 > \dots < X^L, Y^L > |W)$$
$$= \arg\max_W \prod_l P(< X^l, Y^l > |W)$$ obs data

需要假设：data 每个 train example 在给定 W 条件下独立
需要<XY>的联合分布

有更好的方法：我们作分类，是已知了 X!!
Naive bayes rule learn P(Y)和 P(X|Y),其实就是 learn 了 P(XY)JD
这是 Naive bayes 的 learning 目标
因此上面对 P(Y)和 P(X|Y)我们其实不怎么 care,因为 X Y 都已知了 train

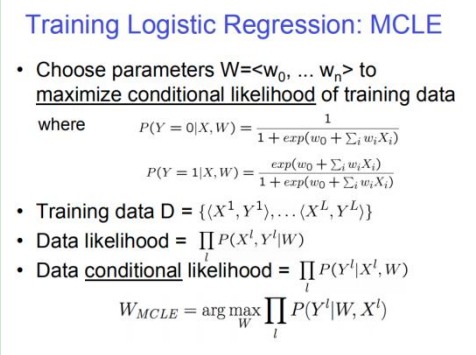


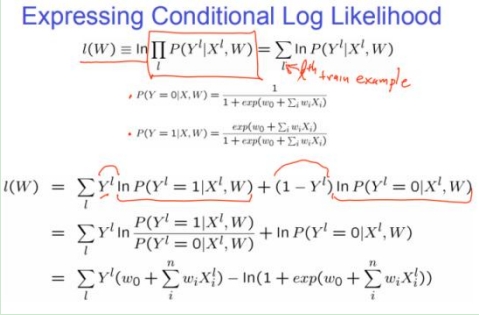
上面才是我们最关心的!!!
也就是 X 已知下，每个 observer data 的概率是 P(Yi|Xi,W)，每个 data 也是假设独立，乘积形式,因此 likelihood 需要相乘，如上
跟 MLE 相比，只是把 P(<XY>|W)换成了条件概率 P(Y|X|W)

Data conditional likelihood: πP(Yi|Xi,W)
Data likelihood: πP(<Xi,Yi>|W)

参数 W 作为条件，其实就是该目标函数(该式子)的参数形式，也就是假设参数以及存在，但是以变量形式存在

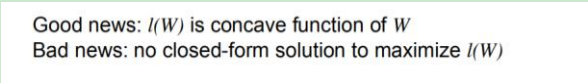
MCLE: Conditional



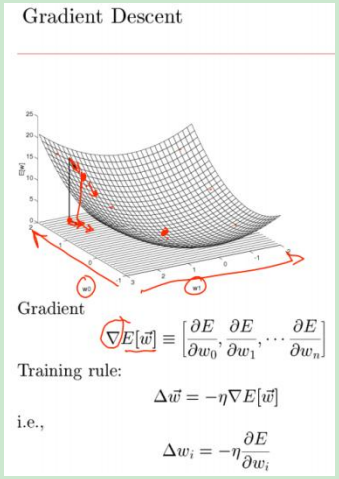


带入我们上面得到的 P(Y|X)条件概率
因为每个 X 的 Y 不确定，因此用 Y 作为 indicate 函数，两种情况一起写！
P(Y|X)我们刚刚推导出了，带入，得到目标函数的最终形式，lr 的目标函数直接训练出 W

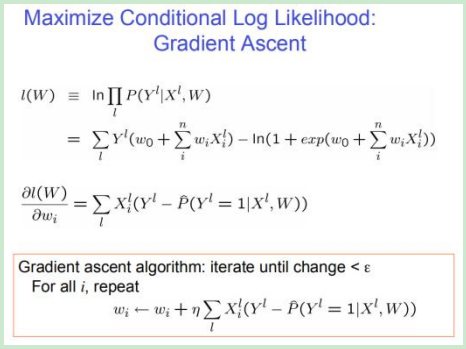
Train 参数：先要写出目标函数,目标函数由 MLE MAP MLCE 写出



max 目标函数 choose w 又成了优化问题：直接求 w 的倒数的话
该函数是 convex function，但 MCLE 没有 closed form solution(得不到公式 question)，但有唯一的 global optimal，只能通过 GD 来达到最优解
需要 gradient descent

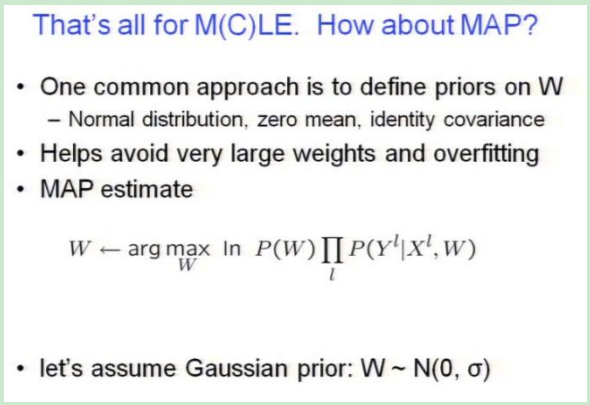


E 是 error function
上三角形是 w 的改变量，对 w 的每个分量独立更新



只需对每个 w 单独的求导数，然后所有 w 同时更新
假设 Y=0 我们想让 P(Y=1|XW)预测 0，此时 step=0，此时的 w 就最好
也就是通过对比 label 和我们得到的概率，最终让这两个值一致

MAP 正则化



也可以 MAP,对 W 有个高斯分布的 prior p(W),weight center with 0
对 W 的 p(W)，就是 W 的正则化
在 MLCE 基础上乘以 p(W)

MLE vs MAP

- Maximum conditional likelihood estimate

$$W \leftarrow \arg\max_W \ln \prod_l P(Y^l|X^l,W)$$

$$w_i \leftarrow w_i + \eta \sum_l X_i^l (Y^l - \hat{P}(Y^l = 1|X^l,W))$$

- Maximum a posteriori estimate with prior W~N(0,σI)

$$W \leftarrow \arg\max_W \ln[P(W) \prod_l P(Y^l|X^l,W)]$$

$$w_i \leftarrow w_i - \eta\lambda w_i + \eta \sum_l X_i^l (Y^l - \hat{P}(Y^l = 1|X^l,W))$$

Machine Learning 10-701

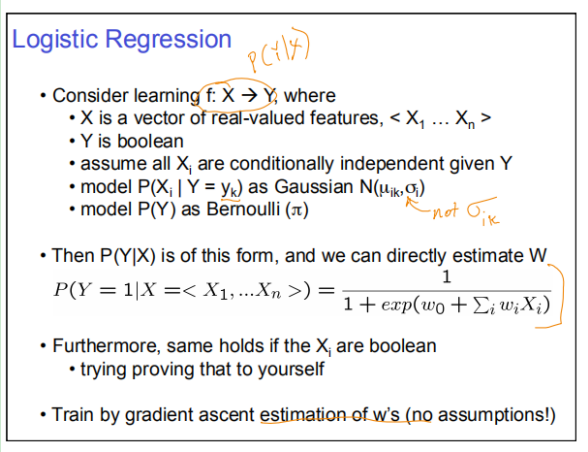
Tom M. Mitchell
Machine Learning Department
Carnegie Mellon University

February 1, 2011

Today:
- Generative – discriminative classifiers
- Linear regression
- Decomposition of error into bias, variance, unavoidable

Readings:
- Mitchell: "Naïve Bayes and Logistic Regression" (see class website)
- Ng and Jordan paper (class website)
- Bishop, Ch 9.1, 9.2

Assumption of LR



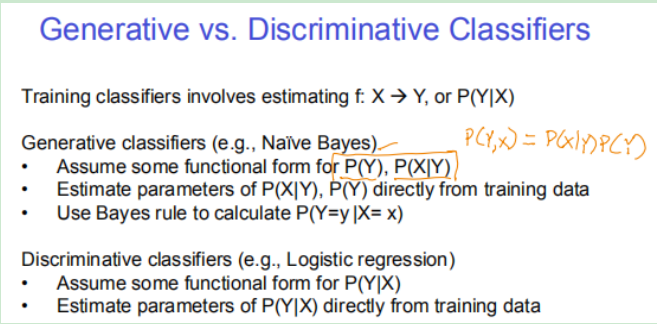
LR 的假设

1 得到 derive 上面的 LR form，需要上面的假设(跟 GNB 一个区别)

2 一旦我们得到这个 form，在我们 train lr 时，没有任何假设！

因此 LR 更简单！更容易 train

Generative and Discriminative Classifier



NB : generative

estimate p(Y) p(X|Y),其实是学了 JD

可以从来 generate X Y pairs

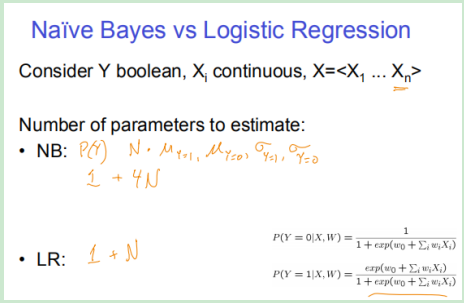
先根据 p(Y) generate y，然后根据 P(X|Y) generate X

LR: Discriminative 只关注 Y

直接从 data 学习条件概率 p(Y|X)

NB GNB LR 算法对比

参数个数



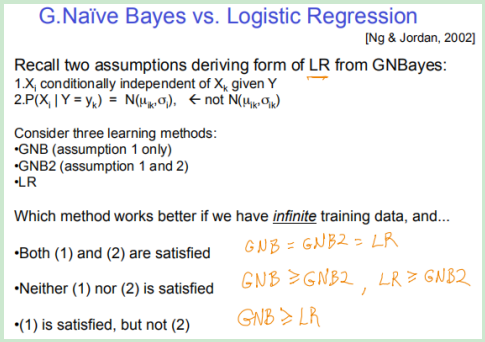
假设 N 个特征，多少参数？
N+1：wi+w0

GNB2 和 LR 的最优解
如果有 infinite data 并且三个假设都满足(高斯，条件独立，variance 只 depend feature)
train 高斯 Naive bayes 的**均值和方差参数**，此时模型形式写成 LR 得到 w 参数，应该是就是 LR 的最优解 w,才会和 LR 的解一样好，一样的结果

有了高斯 Naive bayes 参数可以写成 LR 的形式，但有 LR，写不出高斯 Naive bayes，**因为 LR 只 learn 条件概率，而高斯 Naive bayes 需要 JD**

用哪一个算法：**如果有 infinite data 并且三个假设都满足，用哪一个都一样，如果不满足，就用 LR.**

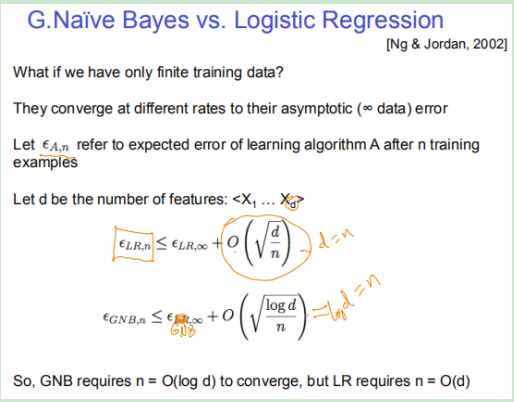
假设有 infinite data：



question 需要再听一遍
GNB 只需要第一个假设
GNB2 是满足两个条件的 GNB，可以推出 LR
因此 LR 需要前两个 assumption
如果假设都满足并且 infinite data，结果一致

如果都不满足：GNB2 比 GNB 参数更少，GNB2 假设更严格 learn 的 h space 是 GNB 的 h space 的 subspace，GNB2 只能 learn linear decision boundary，GNB 可以 learn 非线性决策边界；LR 没有假设，直接 train w learn 的 model 限制更少
第一个满足：GNB 可以非线性边界，LR 不行!!
question 这里没听懂！挺重要的

假设只有 finite data:

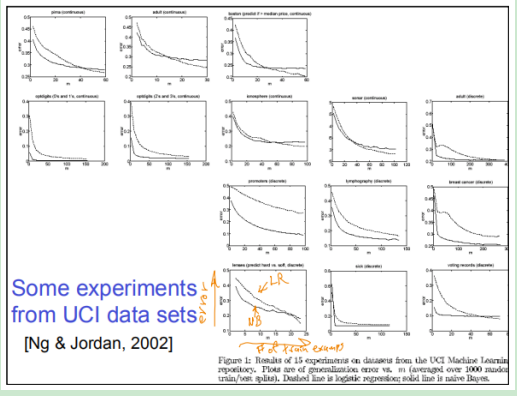


如果数据有限：
draw n 个 example sample，train A algorithm，重复多次，得到的平均 error
d 是 feature 个数　n 是 examples 个数
LR 就算有 **infinite data 也会有错误**(big pic，overlap)这个是最理想的 error
E(LR,n)有上界，converge to optimal error
feature 数量越少(**需要 estimate 的参数越少**)，或者增加 n，**E(LR,n)越小**

对于 GNB2 有另一个公式，
因为 GNB 每个参数都是独立的 MLE 估计出来，参数之间独立，计算一个 estimate 错误，也不影响其他的参数
对于 LR 之间的参数是 coupled 连接的！参数之间是互相影响，一个 estimate 不准会影响下一个（数据的 variance 会影响 estimate）

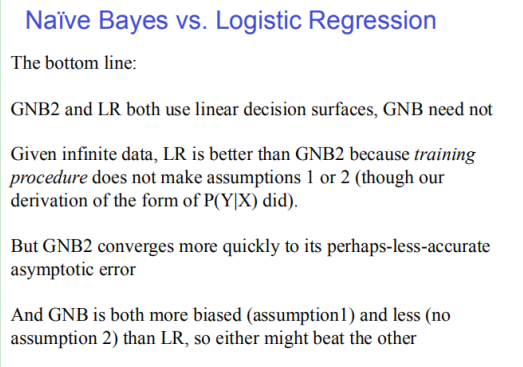
结论：GNB2 需要更少的 data：n=O(log g) to converge to optimal error

其实就是 overfitting，数据越多，overfitting 越少，但下降速度不同



上面线是 LR，下面 NB2，x 是 data 数量，y 是 error

总结：



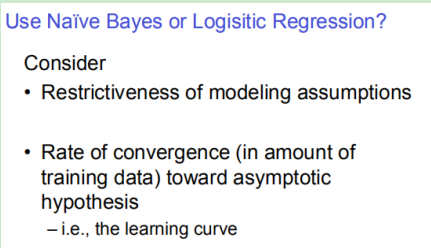
GNB have richer model space

LR GNB2 只能线性分类

一般情况下**数据充足，lr 会好一点**，但实际是万一非线性分类，**GNB 更好!!**

Discriminative work better when enough data

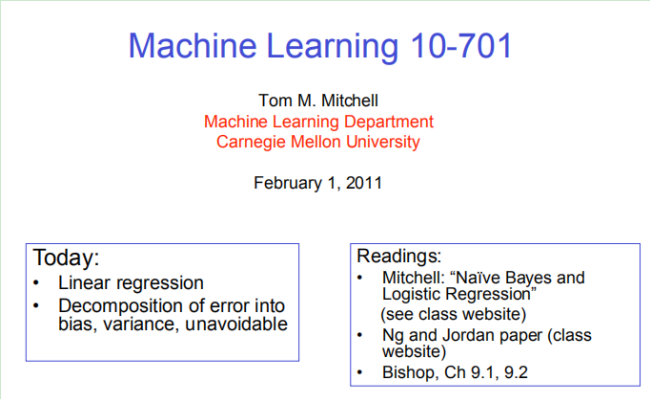
generative model：HMM 成对存在 Discriminative model：直接预测隐状态



算法选择的标准：

1 model 的假设是否严格

2 收敛不同



Regression

Regression

So far, we've been interested in learning P(Y|X) where Y has discrete values (called 'classification')

What if Y is continuous? (called 'regression')

- predict weight from gender, height, age, …
- predict Google stock price today from Google, Yahoo, MSFT prices yesterday
- predict each pixel intensity in robot's current camera image, from previous image and previous action

Y continue value

Regression

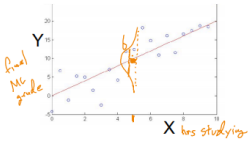
Wish to learn f:X→Y, where Y is real, given {<x^1,y^1>…<x^n,y^n>}

Approach:

1. choose some parameterized form for P(Y|X; θ)
   (θ is the vector of parameters)
2. derive learning algorithm as MLE or MAP estimate for θ

和前面方法一样，选择合适的参数化的模型！

1. Choose parameterized form for P(Y|X; θ)



Assume Y is some deterministic f(X), plus random noise

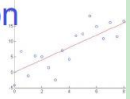
$y = f(x) + \epsilon$ where $\epsilon \sim N(0, \sigma)$

x 是 studying 时间 y 是成绩
我们看到数据点有 random:同样的 x，y 可能不同，需要 model 这种 noise
假设 true Y 是由两部分组成：1 决定性部分 f(x)线性部分(baseline) 2 随机 noise 部分（不确定）0 mean 高斯
random noise 如图所示，true Y 是个高斯分布，noise 是在 f(X)上正负波动

我们目标是得到决定性部分 f(X)的形式，只关心参数 W，不关心 sigma
f(X)就是线性函数!!!

Y 值服从高斯分布，均值 E[Y]是 f(x)
此时 Y 是随机变量,P(Y)是正态分布 注意不是 Y 的值，而是 Y 值取值的概率!!
我们已经知道 E[Y],并不知道 Y 的具体值，是随机的

Consider Linear Regression



$$p(y|x) = N(f(x), \sigma)$$

E.g., assume f(x) is linear function of x

$$p(y|x) = N(w_0 + w_1 x, \sigma)$$

$$E[y|x] = w_0 + w_1 x$$

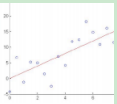
Notation: to make our parameters explicit, let's write

$$W = < w_0, w_1 >$$

$$p(y|x; W) = N(w_0 + w_1 x, \sigma)$$

P(Y|X; W)是高斯分布　**该式子的参数是 W，也急速写出高斯分布公式，参数也是 W**
得到了 P(Y|X; W) 我们想要的形式
用 MLCE 来估计！

Training Linear Regression



$$p(y|x; W) = N(w_0 + w_1 x, \sigma)$$

How can we learn W from the training data?

Learn Maximum Conditional Likelihood Estimate!

$$W_{MCLE} = \arg\max_W \prod_l p(y^l|x^l, W)$$

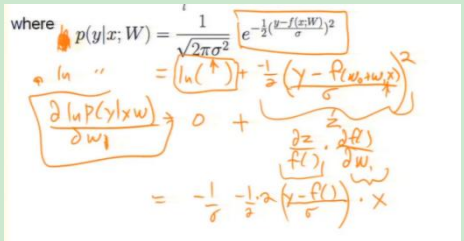
$$W_{MCLE} = \arg\max_W \sum_l \ln p(y^l|x^l, W)$$

where

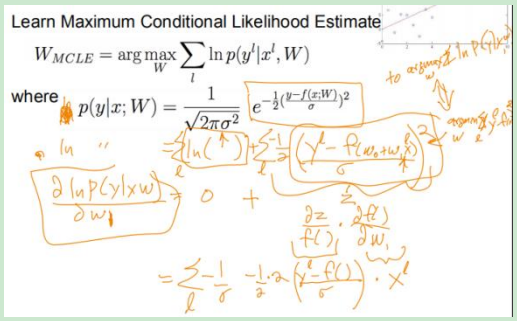
$$p(y|x; W) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{y - f(x;W)}{\sigma}\right)^2}$$

不同的 data example 的 p(y|x)，有不同的高斯公式，因为 w0+w1x 均值不同，但同方差

MCLE，目标函数是 sum
可以求出 w 的 derivative，然后跑 gradient descent
对于 sum 求导，窍诀就是对**每个单独求导然后相加**
先 ln 变换，然后求导：



右边 **剥葱法!! 剥掉外面两层函数　f(X)就是线性函数**



每一个 train exmaple 的 derivative 相加，就得到 W 最终的导数

可以看出：当 predict 正确 y=f(x),就是 0，参数不需要更新，如果 predict wrong ，参数更新

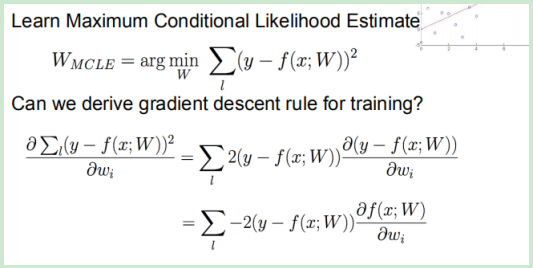
另一个视角：等价的目标函数

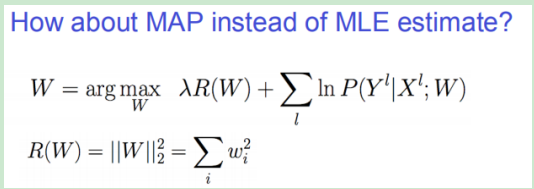
目标函数的 w 只 depend on term2,

max 目标函数<=> min squared error (true y-f(x))^2

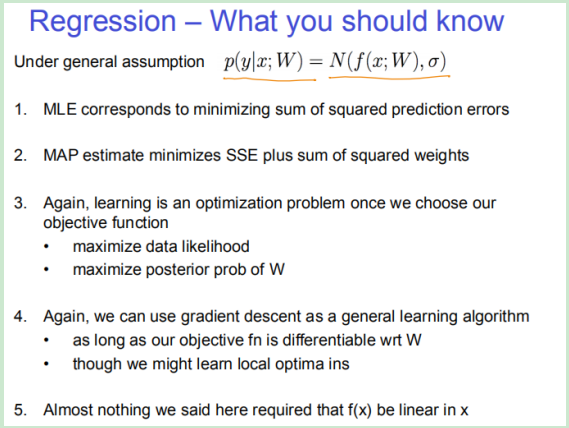
$$\underset{W}{\arg\max} \sum_l \ln P(Y \mid X_l, W) \Leftrightarrow \underset{W}{\arg\min} \sum_l (y_{\text{true}} - f(X_l))^2$$

minmize the squared error 更简单





MAP 就是加个对 w 的先验分布，其实就是正则化



有了上面的假设，1 MLE 就是 最小二乘，2 MAP 结果是正则

3 learning 是从 training data 里 estimate 参数

learning and optimization

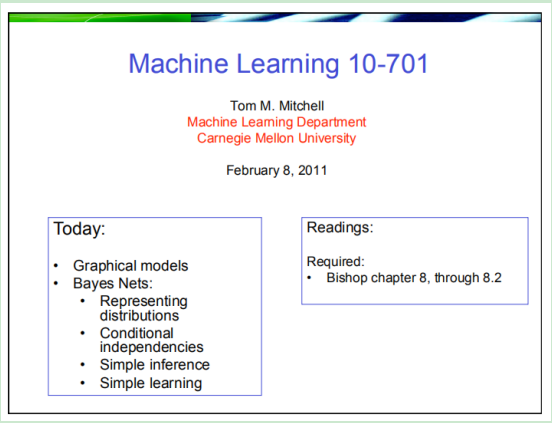
learning 是从 training data 里 estimate 参数

Again, learning is an optimization problem once we choose our objective function

- maximize data likelihood
- maximize posterior prob of W

一旦确定了目标函数，learning 就成了**优化**!
training 过程就是优化过程

所以，ML 更多是 optimization


Machine Learning 10-701

Tom M. Mitchell
Machine Learning Department
Carnegie Mellon University

February 8, 2011

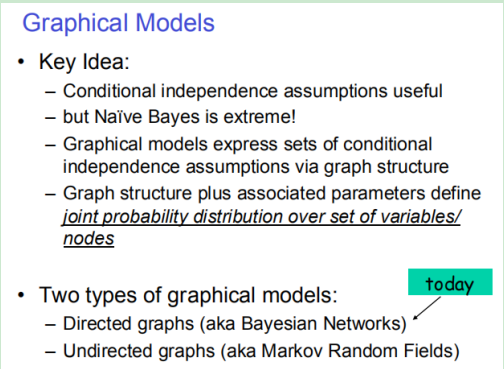
Today:
- Graphical models
- Bayes Nets:
  - Representing distributions
  - Conditional independencies
  - Simple inference
  - Simple learning

Readings:

Required:
- Bishop chapter 8, through 8.2

Graphical Model

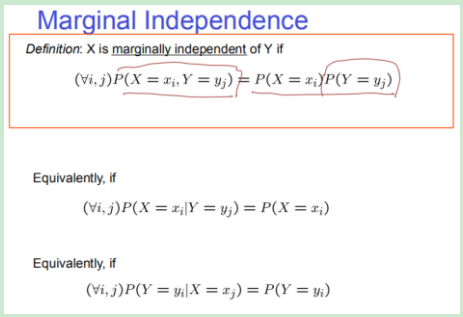
是 ML 非常重要的 field!! **bitshop 讲的最好!!!!!!!!**


Graphical Models

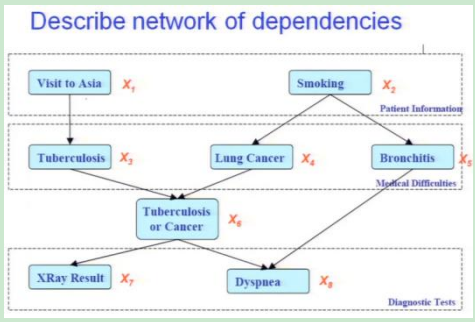
- Key Idea:
  - Conditional independence assumptions useful
  - but Naïve Bayes is extreme!
  - Graphical models express sets of conditional independence assumptions via graph structure
  - Graph structure plus associated parameters define *joint probability distribution over set of variables/ nodes*
- Two types of graphical models: today
  - Directed graphs (aka Bayesian Networks)
  - Undirected graphs (aka Markov Random Fields)

为了不去学习 full JD,用了 conditional independent，但 Naive Bayes 的假设太 naive，simple 了，**ridiculous，有些假设根本不成立**

Graphic model 是 JD 和 conditional independent 的折中
保存了某些 conditional independent，并增加了**变量之间的 dependency**



Bayes network

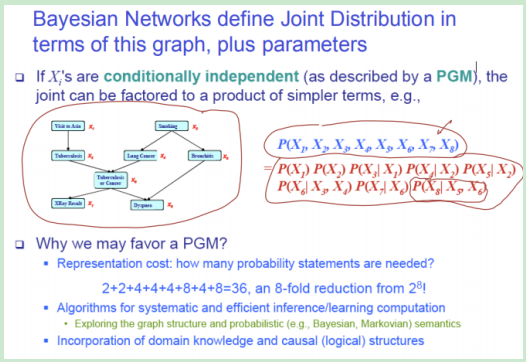


A-->B　A 影响 B　或 B dependent on A

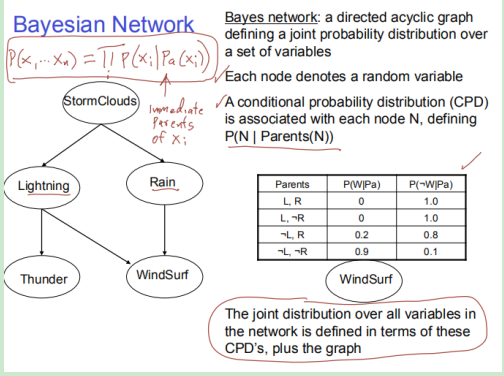
formalize:其实就是将**所有变量的联合分布**以一种特殊方式分解(依据假设分解)

分解方式对应不同的 structure

分解也是为了通过假设降低参数数量！



example:

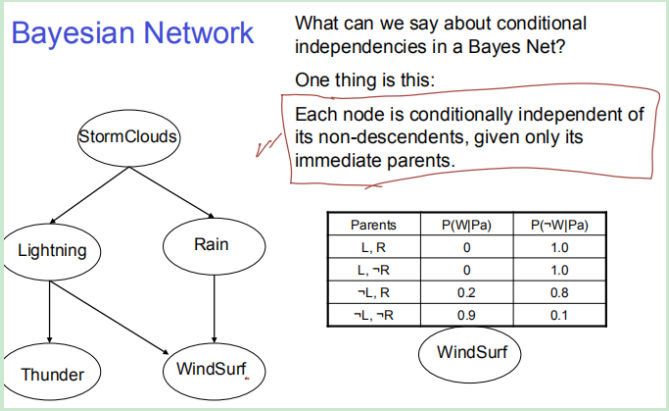


Bayes network: directed acyclic graph 有向无环图

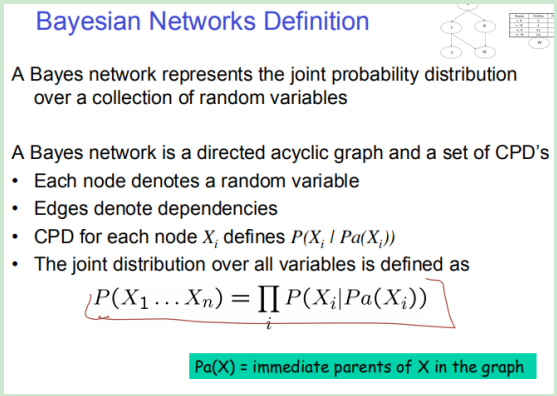
node 代表 variable　　edge 代表 dependency，箭头代表**影响**

每个 node 有个 **CPD** p(N|Parent(N)) conditional probability distribution
例如 WindSurf 有 2 个 parent，**两个条件**
P(WindSurf|Lighting,Rain)，每一行代表不同的条件

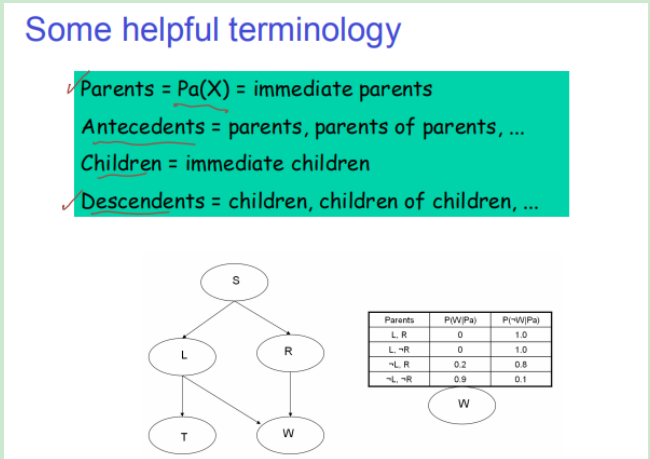
给定所有变量的 CPD,和 graph structure，就可以求出**所有变量的 JD**
就等于所有节点的条件概率的乘积
从上到下父节点的值确定了，子节点的的条件就确定了，条件概率也确定了



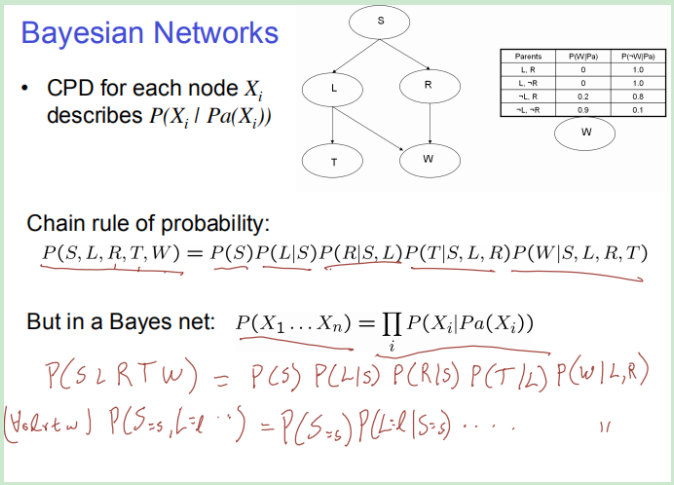
每个 node 和**他得非后代的 node，在 given 直接 parent** 的条件下独立！
given rain 和 lighting ，windsurf independent with thunder
giving lighting ，thunder independent with rain 和 storm



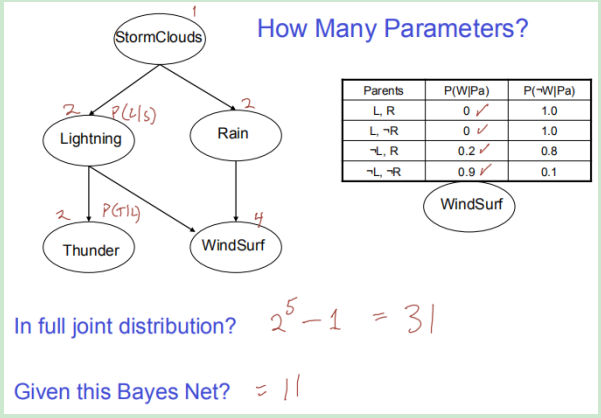
Bayse net 代表 JD of 多个变量



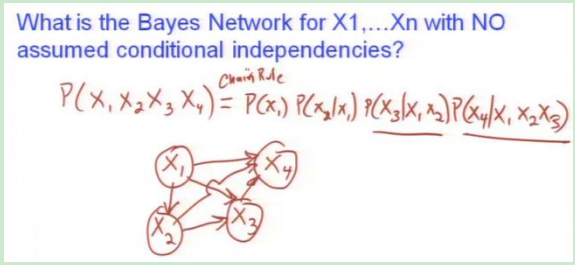
parent children 都是 immediate



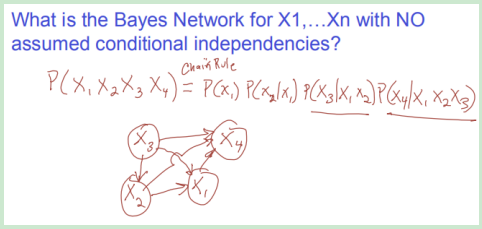
从上到下的顺序写
上面是普通的 chain rule 分解
下面是 bayes net 分解
诀窍:先 chain rule 分解,然后再改写成 bayes net 的 conditional probability
条件里只写 parent!! 因为有假设 conditional independent on parent



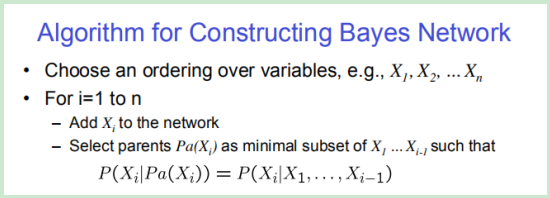
假设都是 boolean
JD parameter 需要 31 个
BN 需要 11 个，对于 windsurf 2 个条件 8 种组合如上表：**只需要考虑条件的组合就好**，因为条件下发不发生和为 1！ **2 个条件，组合为 4，4 个参数**



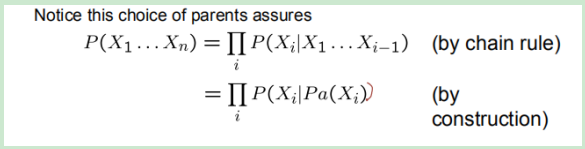
假设没有 conditional independent
先用 chain rule，画出 chain rule 的 graphic model 如上!!
每个 node 都相连(如果不考虑方向)



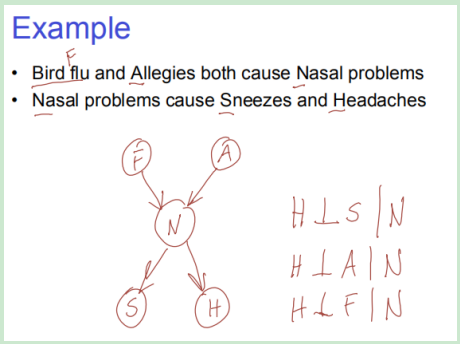
但 chain rule 可以有很多分解方式!! 很多 graphic model!!



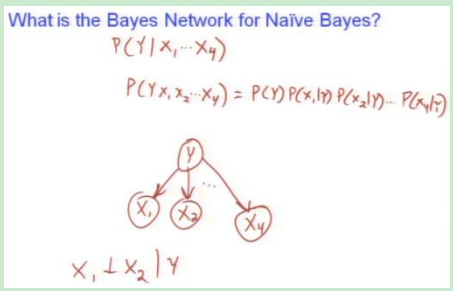
需要 choose 一个 order(也就是 chain rule 的**分解顺序**)
遍历每个 node，对于每个 node，剩余的其他节点作为他的 parent，
以上过程就是 chain rule 的分解:
p(X1|X2,X3,X4)p(X2|X3,X4)p(X3|X4)p(X4)=p(X1,X2,X3,X4)



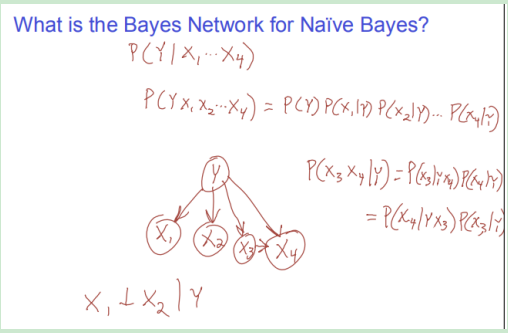
然后再加入我们的假设，修改 chain rule



Naive bayes



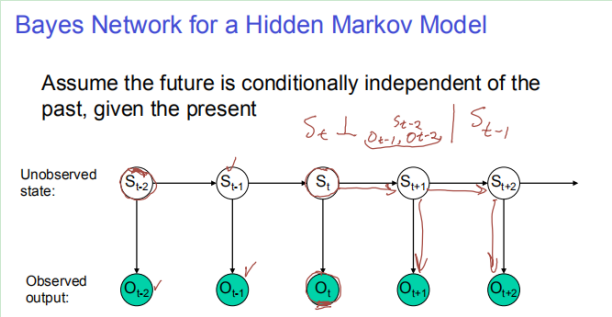
先 chain rule 分解
p(Y,x1,x2)=p(Y)p(x1|Y)p(x2|x1,Y)
naive bayes 假设 p(Y)p(x1|Y)p(x2|Y)



假如我们想让其中 x3 x4 不条件独立，也就是 P(X3 X4|Y) 不再分解成乘积了，**只能按 chain rule 分解**，两种方式，**箭头方向两种都可以，之前的式子就变了，**画如上的图

HMM

HMM 也是一致 GM，这种结构可以写出联合概率分布，也隐含了假设



HMM 本身就是 **graphic model**

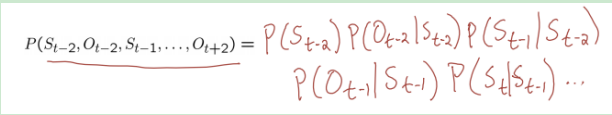
1 Ot 在给定 St 下，和所有所有其他 node independent！

tail-tail 和 head tail 两种情况 block

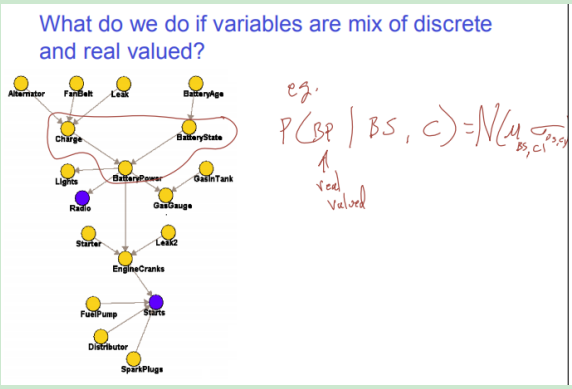
2 St **在给定 St-1 下，和所有的 non-decendent(不包括 Ot 和 St future node) 也就是和 past 无关。**

tail-tail 和 head tail 两种情况 block

跟未来的都有 path



上面的有固定的顺序;先 hidden state 然后 obs，然后 hidden state..



既有离散又有连续，分解方式不变

p(x|j)

x 是离散变量，假如用 category distribution 来 model

x 是连续变量，用其他合适的连续分布，如高斯分布 参数 condition on parents

这就是 Graphic model 的魅力

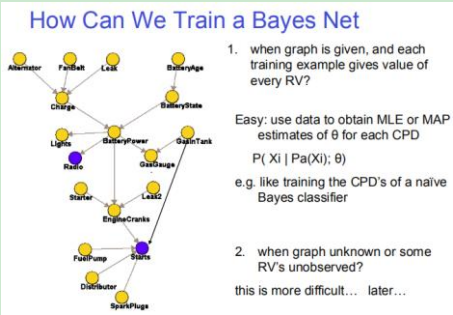
learn Graphic model

Graphic model 可以包含两部分：

1 the graph(代表关系结构，分解) 2 具体的 distribution 公式(高斯，伯努利)

将简单的分布集成为复杂的分布！

Naive bayes 就是个特例，用 graph 将连续和离散变量联系在一起！



CPD conditional probability distribution

1 graph 结构有了

2 所有变量可观测(examples)有 data

如何 train BN?

其实就是 learn 每个 conditional probability

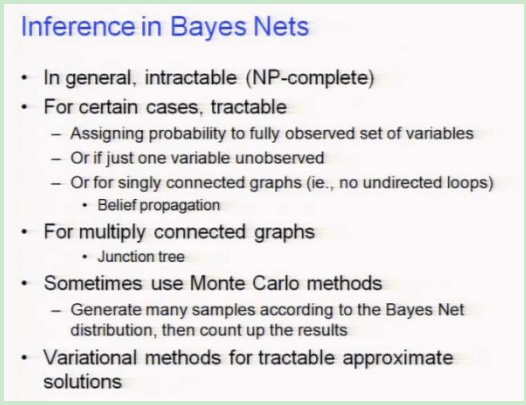
Naïve bayse 是个特例，$p(Y|X)=P(X|Y)P(Y)=P(Y)\pi p(Xi|Y)$

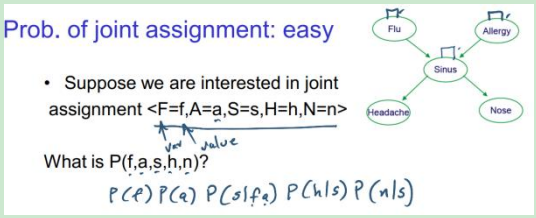
也是在 learn 条件概率 CPD，因此方法一样，也是 count 比例

GM 可能更复杂一点:1 是条件更多 2 各种 CPD 分布

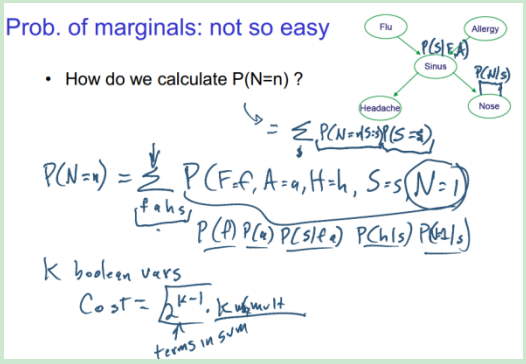
inference in GM

inference 最难，比 learn
inference **也就是计算某个给定观测后的概率，或者是条件概率，边缘概率的值**





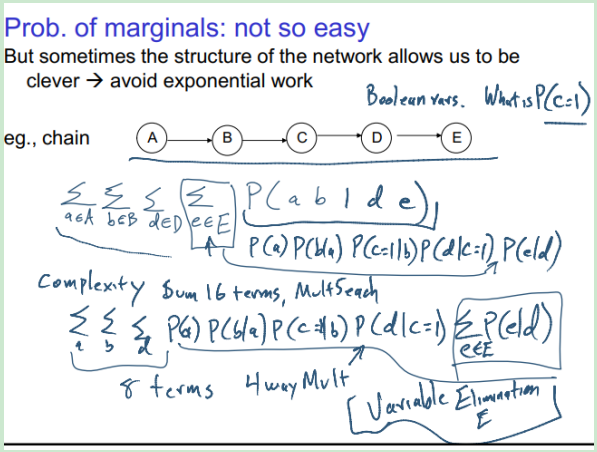
1 计算给定值的**联合分布**概率值　好求
因为分解后，**每个因子的概率分布已知**，只需从头到尾带入，从左到右求得概率值就好!! 如 p(f)　p(a) 然后带入 f a s 值，求条件概率 p(s|f,a)



2 计算 marginal probability
其实就是将 JD 的其他变量求 sum
也可以求**联合分布**的其他变量的 sum，**写成乘积的形式**
假如都是 boolean 变量
计算量分析:
sum 4 个 boolean 变量是加 2 的 k-1 次
每个都是 k multiple(k 个相乘)
计算量太大了!!

3 对于 conditional probability:
P(F,A,S,H|N=n)= P(F,A,S,H,N)/ P(N=n)

在边缘分布计算的基础上完成



求 P(C=1)的概率，利用 JD 求边缘分布
再将 JD 分解，再移项，将和 E 无关的移到左边，先 sum E
计算量：
ABCD 每个 2 个取值 共 16 个值，每个值都是 5 个乘积

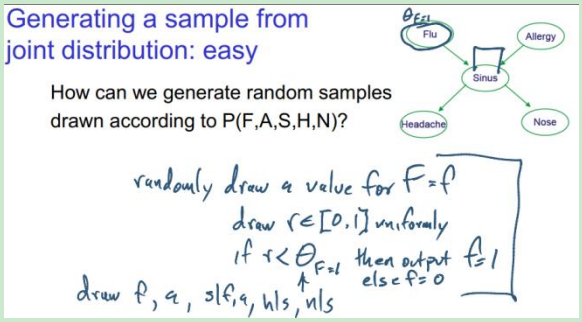
SUM 动态规划 variable elimination question

可以把不 depend on E 的 term 提前，相当于提公因式!!
这样可以减少运算!! 变成 8 个值，每个值 4 个乘积
消去了 variable E
variable elimination (dymamic programing!!!)
pre-compute sub-expression once, store it, 减少 repetition

可以再消 D, B … question 不太明白 这里的 sum 求和公式不是很明白!

monty carlo sample

用 monty carlo sample 来模拟，也就是用 BN generate a random sample from JD，然后来 estimate 我们想要概率(只需要统计，看 fraction 就行)
how to random sample?



1 如何 sample 一个伯努利？ 参数是 theta
draw r from [0,1]uniformly
if r <theta output 1 else 0

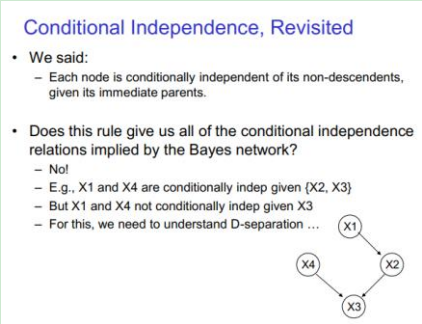
就可以 draw F 和 A

S 是 condition p（S|F）
F 取 0,1 时，**S 只是 theta 不同(不同的伯努利分布)**，其实也是上面的过程
同理 p（S|FA） 不同的 FA 组合 S 不同的 theta

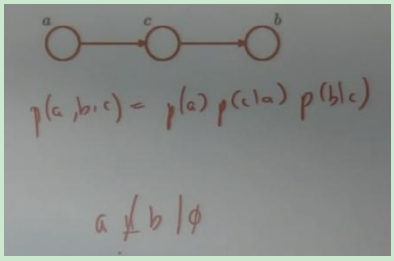
但如果有一些概率如 A=1 很小，sample 可能不会出现，就没办法估计

conditional independent D-separated

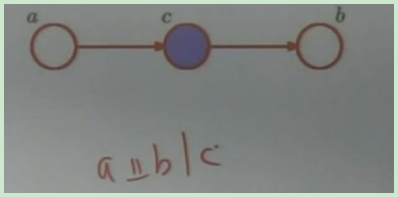
(prml bishop 讲得好)



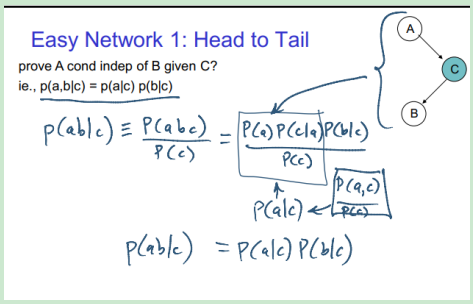
之前的 condition independent 不能反映全部的情况
X1 X4 在 X2X3 条件下独立，在单独的 X2 X3 不独立



A 可以跟 B 通讯，因此不独立



知道 c 以后就会 block a b，就独立：证明：



这个是满足我们之前的 rule 条件独立

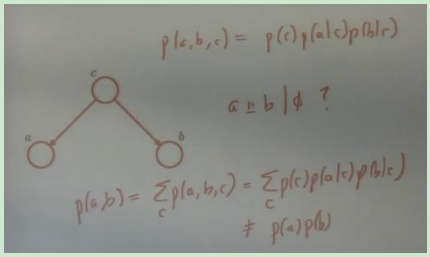
prove：

用 JD 的 chain rule，此时**分解有顺序!!**

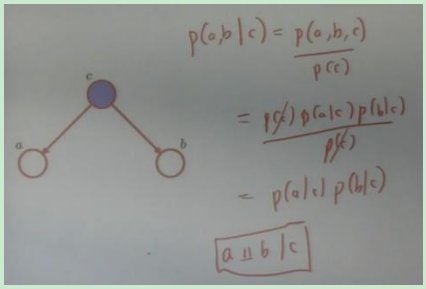
再利用 bayes rule ，得证!!

记住：不同的 graph 决定了不同的 JD 分解方式!!!

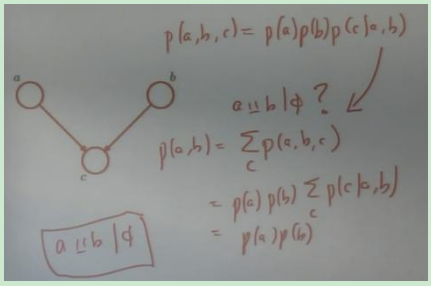
同样满足 rule 条件独立



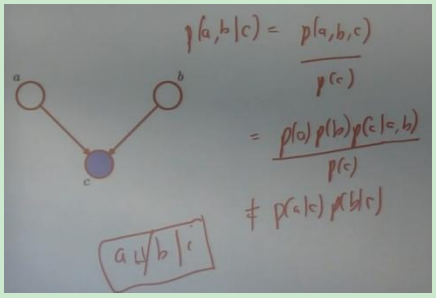
如果 C 不知道，**则 a b 不独立** 以上证明



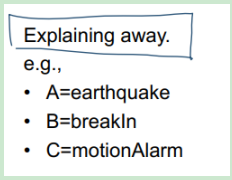
3 collider



c 不知道，ab 独立！



知道 c 后或 c 的 desendent， ab 不独立



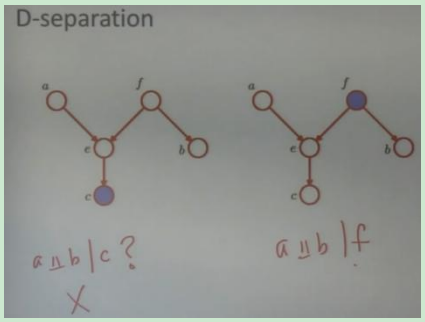
A 和 B 独立　地震和偷窃　都会使得门铃响

如果在**条件 C 发生的情况下，就不再独立!! 假设 C 知道，报警了**

A 是否发生会影响 B 的概率!! AB 不再独立

大总结：D-separation

A 和 B 的 all path blocked，AB 不能通讯了就独立!!，有一个 path 通了就不独立！（这里的 path 不考虑方向!!）



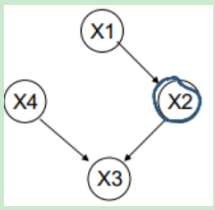
condition on 什么，就说明什么已经 observed 发生了

1 ab 只有一条 path 左边是第二种情况，并且 unblocked，右边是第一站情况 unblocked，因此 ab 可以通讯，不独立！

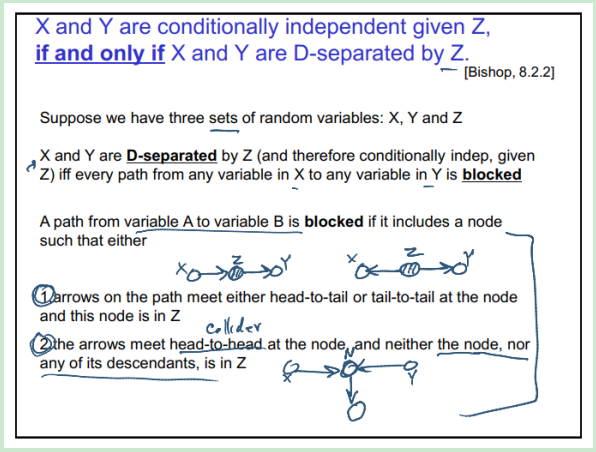
2 a －f 已经 block，整条路就 block 了

a path blocked:也就是说一个 path 中,只要有一段(任意两个节点出现 block，这段就垮掉！)没办法通讯！就独立！

如果有多个 path，需要 block every path!



X1 和 X4 独立，因为右边 unblock，但左边 block
X1 和 X4 given X2 也独立，都 block 了

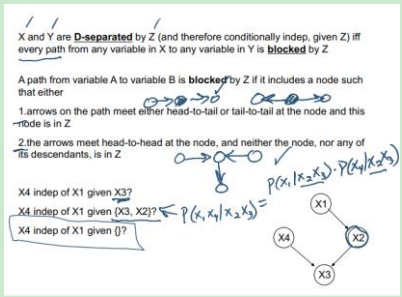


此时，X Y Z 不仅是一个 variable 也可以是 set of variables

X Y 条件独立于 Z IF ONLY X Y D-separated by Z
X Y D-separated by Z IF ONLY X 的所有变量和 Y 所有变量的 path 被 block

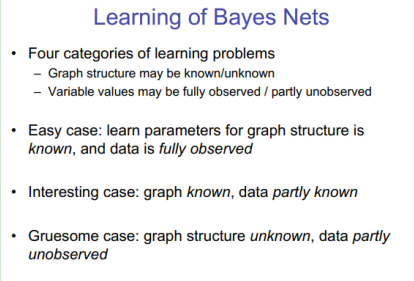
A 和 B 两个变量 block 的两种情况：
1 前两种，已经证明。
  存在一个 head tail node，并且 observed
  存在一个 tail tail node，并且 observed
就 block AB

2 存在一个 head head node，并且该 node 和其 desendent 都 unobserved
就 block AB



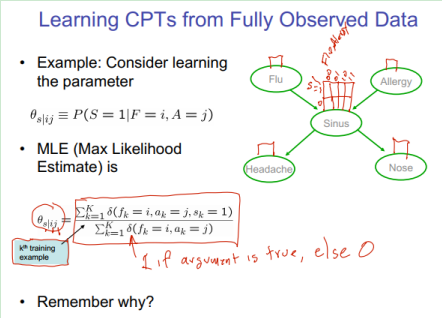
1 不独立，左边 block 右边还 block
2 独立，左边 unblock，右边 block 可以如上分解
3 独立 左边 unblock

learn Bayes Net



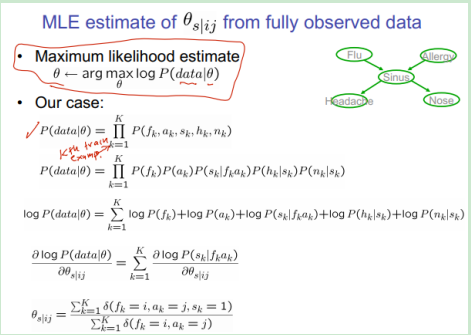
1 是否需要 learn **graphic structure**
2 变量的值 **可能没有观测全**，之前的 train 都是 fully observed
但有些 example 的**变量值**没有，很多实际情况！

fully observed data



已知 structure and all observed data，来估计参数
MLE 是我们标准的方法，esitimate 参数，来 max 概率 of observed train data

上面每个节点代表一个分布表，sinus 是 p(S|F A)
此时概率表的**每个概率就是我们要学的参数**
共有 K 个 train example
MLE 就是频率统计结果 分母是 ij 的个数，分子是 ij 并且 s 的个数
delta：是 indicate function if argument= trure 函数是 1
具体证明：



choose the theta 使得 data likelihoode 最大，theta 必须是条件(theta 存在，只是变量而已) theta 是联合概率分布的参数!!!
假设每个 example 都是 iid，都是从 JD 里 sample

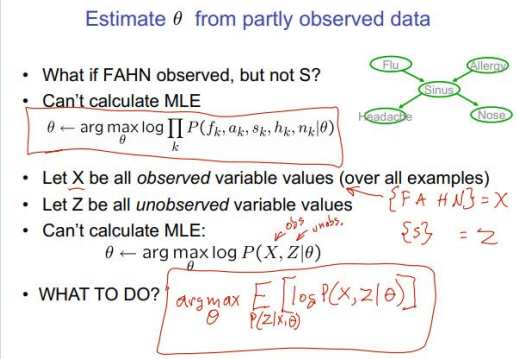
每个 example 的概率就是**所有变量的联合分布概率，可以根据 GM 结构分解**
所有 example 联合概率的乘积就是 data likelihood：p(D|theta)
然后取 log，然后求导，假设对..求导

$\theta_{s|ij} \equiv P(S = 1|F = i, A = j)$

其他 term 不 depend on theta，都是常数
令导数为 0
p(s|f,a)要么是 theta，要么是(1-theta)

partly observed data and EM

PRML 讲的更好!!!

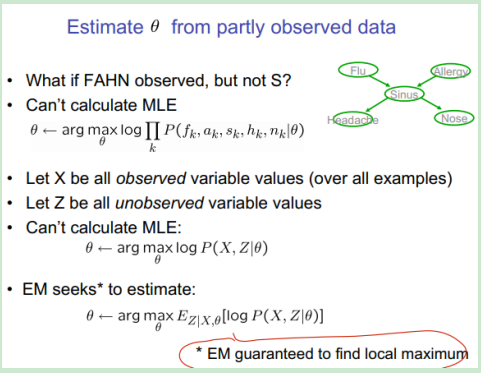


当某一个变量观测不到，或者部分观测
没办法再 MLE 原因：
1 没办法写出联合概率 P(X,Z)(Z 值不知道) 没办法写出 likelihood
2 没法统计 count 某个分布的参数了
但**联合分布的公式形式我们知道**(参数作为变量)：就是分解的形式，就是每个分布公式相乘得到的公式。
直接计算 log likehood 不行，我们计算它的**期望**

我们得到了 log likelihood
但含有 Z 的因子，我们因为不知道 s，也就不知道其具体值，没办法求 log likehood 怎么办？
因为每个可能的 Z 值,会对应有一个 log p(X,Z)
我们将 **Z 的所有可能取值**进行平均，weighted by the probability of Z
得到 **expected log likelihood!!**

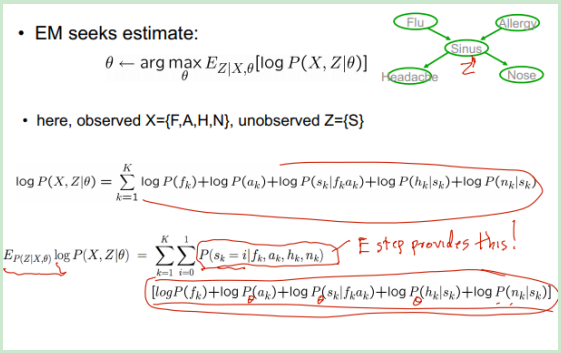
maximize expected log likelihood!
E :expect of A **with respect of 概率 distribution**

EM 算法 ： expectation maxmization!!
maximize expected log likelihood!



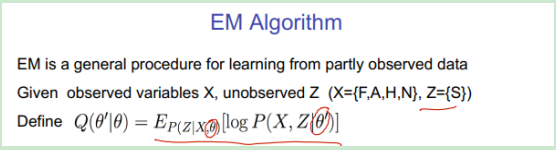
expected log likelihood 的解释：

假设 X 是 Sinus S 如下：



上面是先对 k 个 example 遍历，当第 i 个 example 时，我们需要计算他的 log likehood 均值！

但我们不知道 theta，上面的概率分布公式不能确定？如何计算上面的概率：EM 算法



theta’ 和 theta 不同

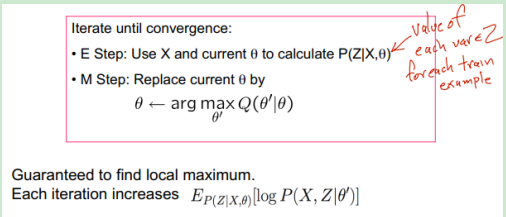
theta 是我们 guess 的初始值，theta’ 是 likelihood 里的参数(需要 MLE 估计)

1 先 guess theta，求的 P(X,Z|theta)的概率

2 然后就可以 expected log likelihood

E[P(X,Z|theta) logP(X,Z)],此时是 theta(未知数)

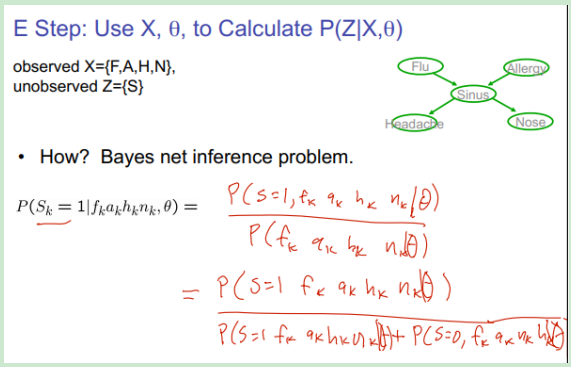
3 然后 max E 再求 theta’,作为新的 guess，重复上面



1 E step 本质是 calculate 所有 train examples 的 unobserved feature 的 expected value(相当于补全了 data)，得到了 data 的 log likelihood。

2 然后再做 MLE,再重新计算 expected value

EM 保证每一步都会 non-decrease(max 时)，保证有 local maximum



如何求 P(Z|X,theta) conditional distribution? 条件概率的定义

因此 P(Z|X,theta) 也是由 P(X,Z|theta) 计算出，因此 theta 一样!!!

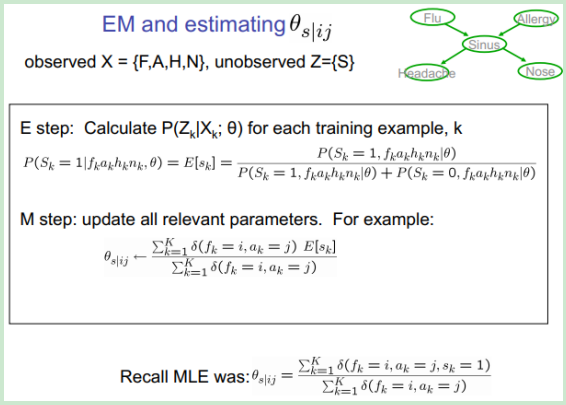
分母再 sum rule(任意两个互斥事件 s=1 s=0)

$$P(S_k = 1|f_k a_k h_k n_k, \theta) = \frac{P(S_k = 1, f_k a_k h_k n_k|\theta)}{P(S_k = 1, f_k a_k h_k n_k|\theta) + P(S_k = 0, f_k a_k h_k n_k|\theta)}$$

完全可以用 JD 来计算，但 JD 分解后的每个 factor 分布的 theta 我们不知道

我们 guess theta，计算出 p(S) ，此时 theta 已经不存在了

然后将 p(S) 带入计算 Expection of log likelihood，此时的参数是 theta'（未知数）



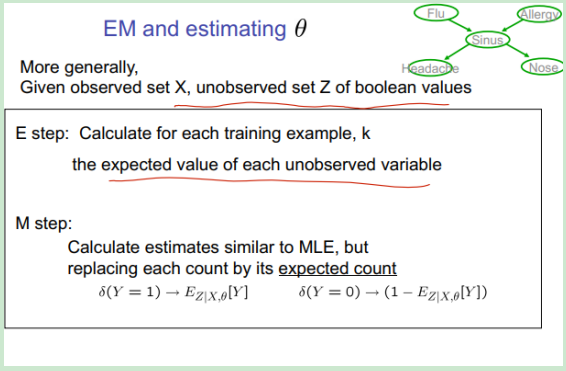
max E 求出 theta'，结果如上，类似于 MLE

得出 JD 分解后每个 factor 分布的参数

结果可以看出本质:

P(s|a,f) 是伯努利分布，参数是 p(s=1|a,f)，所以 MLE 是统计 S=1 的比例

EM 跟 MLE 不同的是，E[s] 是 s=1 的期望，统计每个 example s=1 的期望值

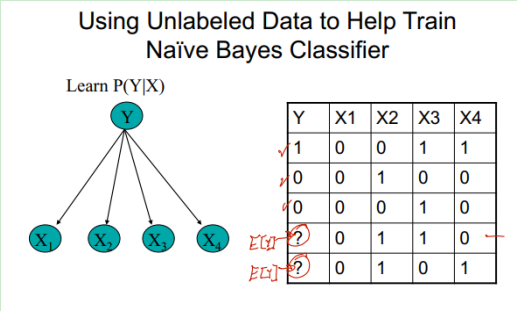


EM: 任何 structure+ observed and unobserved data

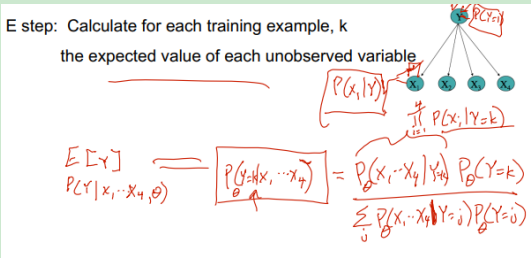
EM 本质

其实上面过程的也就是 EM 本质：
其实就是对缺失值进行分类!!!!
用 observed 的数据 train NB classify unlabeled data!! 得到 Y 概率
然后 assign to 缺失值，然后 retrain!! 然后得到再 assign，再 train…

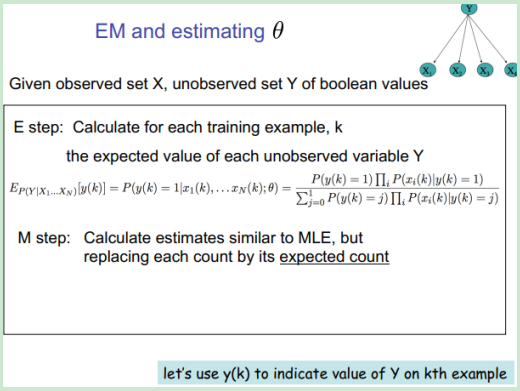
以上都是部分观测数据
NB:



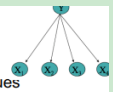
缺失的原数据部分用 E[Y]是 Y 的期望代替，E[Y]=sum：p(Y|其他)*Y 的取值
每一行缺失值的 p(Y)不同，所以 E[Y]也不同，p(Y)受其他变量观测值的影响
具体如何计算？首先伯努利的 E[Y]=P(Y),只需计算 p(Y)



NB 的参数是 p(Y) P(Xi|Y)
P(Y|X1..X4)用 bayes rule 计算
分母是 P(X1..X4)=SUM P(X1..X4,Y)=SUM P(X1..X4|Y)P(Y)
通过 bayes rule 和边缘分布变换，都变成我们已知的参数(用我们部分能观测的数据计算)可以计算出 E(Y)



EM and estimating θ



Given observed set X, unobserved set Y of boolean values

E step: Calculate for each training example, k

the expected value of each unobserved variable Y

$$E_{P(Y|X_1 \ldots X_N)}[y(k)] = P(y(k)=1|x_1(k), \ldots x_N(k); \theta) = \frac{P(y(k)=1)\prod_i P(x_i(k)|y(k)=1)}{\sum_{j=0}^{1} P(y(k)=j)\prod_i P(x_i(k)|y(k)=j)}$$

M step: Calculate estimates similar to MLE, but replacing each count by its expected count

$$\theta_{i|m} = \hat{P}(X_i = j|Y = m) = \frac{\sum_k P(y(k)=m|x_1(k)\ldots x_N(k))\ \delta(x_i(k)=j)}{\sum_k P(y(k)=m|x_1(k)\ldots x_N(k))}$$

MLE would be: $\hat{P}(X_i = j|Y = m) = \frac{\sum_k \delta((y(k)=m) \wedge (x_i(k)=j))}{\sum_k \delta(y(k)=m)}$

- **Inputs:** Collections $\mathcal{D}^l$ of labeled documents and $\mathcal{D}^u$ of unlabeled documents.
- Build an initial naive Bayes classifier, $\hat{\theta}$, from the labeled documents, $\mathcal{D}^l$, only. Use maximum a posteriori parameter estimation to find $\hat{\theta} = \arg\max_\theta P(\mathcal{D}|\theta)P(\theta)$ (see Equations 5 and 6).
- Loop while classifier parameters improve, as measured by the change in $l_c(\theta|\mathcal{D}; \mathbf{z})$ (the complete log probability of the labeled and unlabeled data
  - **(E-step)** Use the current classifier, $\hat{\theta}$, to estimate component membership of each unlabeled document, *i.e.*, the probability that each mixture component (and class) generated each document, $P(c_j|d_i; \hat{\theta})$ (see Equation 7).
  - **(M-step)** Re-estimate the classifier, $\hat{\theta}$, given the estimated component membership of each document. Use maximum a posteriori parameter estimation to find $\hat{\theta} = \arg\max_\theta P(\mathcal{D}|\theta)P(\theta)$ (see Equations 5 and 6).
- **Output:** A classifier, $\hat{\theta}$, that takes an unlabeled document and predicts a class label.

From [Nigam et al., 2000]

cluster and mixture distribution

Usupervised clustering

Just extreme case for EM with zero labeled examples…

Clustering

- Given set of data points, group them
- Unsupervised learning
- Which patients are similar? (or which earthquakes, customers, faces, web pages, …)

将 cluster 问题转化成 estimate mixture distribution 参数

Mixture Distributions

Model joint $P(X_1 \ldots X_n)$ as mixture of multiple distributions.

Use discrete-valued random variable Z to indicate which distribution is being use for each random draw

So $P(X_1 \ldots X_n) = \sum_i P(Z=i)\, P(X_1 \ldots X_n|Z)$

Name of the cluster

Mixture of *Gaussians*:

- Assume each data point X=<X1, … Xn> is generated by one of several Gaussians, as follows:

1. randomly choose Gaussian i, according to P(Z=i)
2. randomly generate a data point <x1,x2 .. xn> according to $N(\mu_i, \Sigma_i)$

混合分布也是一种 JD 概率分布!!

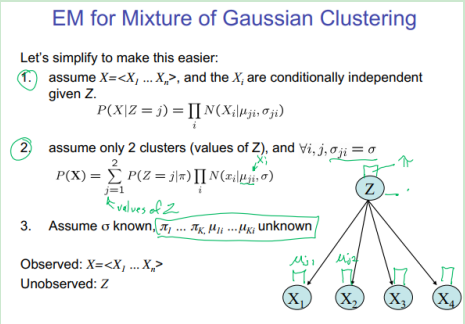
也就是假设有隐变量，不同的隐变量有不同的分布，来生成我们的 data，更具体：不同的隐变量和不同的 feature Xi，p(Xi|Z)是不同的分布
但一般情况我们会假设条件独立 p(X|Z)=πp(Xi|Z)，其实不同的 Z 就是多元高斯分布，代表 data 是从多个分布产生

每一次 sample:X1..Xn,都可能来自不同的分布，因此需要 Z 来指定
Z 是 name of cluster 也是 X 的 latent variable(unobserved)
P(Z)P(X1..Xn|Z)有先后顺序，但具体是那个 Z 未知，但都有可能，所以概率相加(或 or)，其实也是一种期望！
就是将 JD transform，利用边缘化概率

Z 的 number，可以用 test data 来 pick，选择 p(tets data)最大的 num

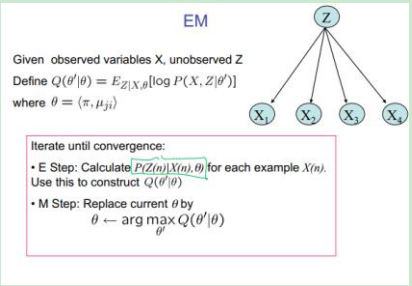
如果是多个高斯分布的混合，当 p(X|Z)为高斯分布，就是混合高斯分布
每个 data 是从多个高斯分布产生

EM：Z 是完全不可观测

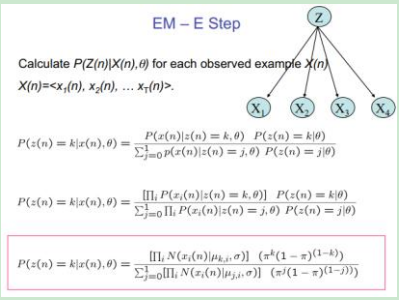


假设 NB condition independent，不同的 Z 和不同的 feature Xi 有不同的高斯分布 p(X|Z)其实就是 n 元元高斯
假设只有 2 个 cluster，且方差全部相同其已知
π是 category 分布参数，u 也是参数，需要估计

我们有了 structure，用 EM learn P(Z)P(X|Z) 参数



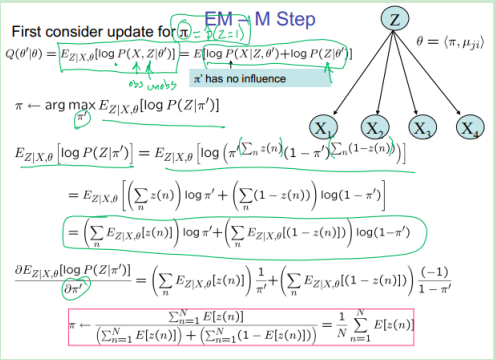
计算出 P(Z|X,theta),也就是每个 example 属于每个类的概率
然后计算 log likelihood 的期望，对于未观测变量的概率 P(Z|X,theta)



计算 P(Z|X,theta)，用 bayes rule（其实就是 NB 的分类算法得到 p(Y|X)）
此时得到 P(Z|X,theta)的具体公式
guess 参数值，并带入具体分布，计算每个 example 属于每个类的概率 P(Z|X,theta)

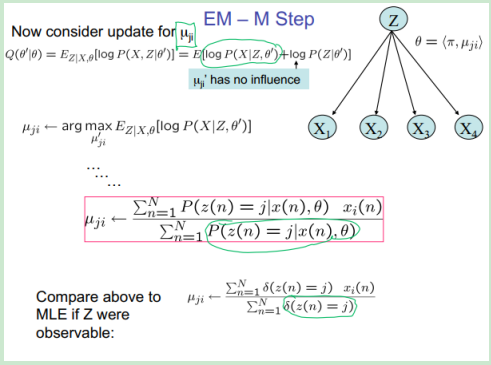
再带入 E[log P(X,Z)]就是我们的参数的公式
注意：此时 log P(X,Z) ；P(X,Z)是 likelihood
P(X,Z)=P(X|Z)P(Z) 也都是 likelihood

最后得到了含有参数的 log likelihood 公式



分别对 π 和 u 进行 estimate
先对 π 优化：用 chain rule 展开，π 不影响第一个 term
只需 max 第二个 term
z()是 label 函数，z(n)如果第 n 个 z 是 1，否则返回 0
n 是 exmaple 的数量
第二个 term p(Z)是 likelihood！因此是 multinomial,写成多项式形式
第三行 展开期望，得到每个 exmaple 平均是 1 期望，然后再求和
每个 example 的 z(n)是随机变量要么是 0，要么是 1，因此 E 可以刺穿 SUM 变成第三行
相当于 E[sum(Xi)]=sum E[Xi]
每个 example 都有对应的 P(Z|X,theta),E[z(n)]也就是计算该 example z 是 1 的均值
E[1-z(n)] 此时 z=0 时 1-z 是 1，也就是出现 z=0 的均值
最后所有 example SUM,所有 example Z 值的期望

对 π 求导： π 就是 所有 example 是 Z=1 的期望的和

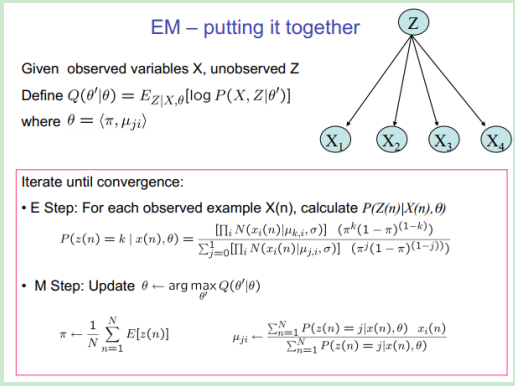


x(n)

MLE uji 所有 label 是 j 的数据里，每个数据有多个特征，每个特征 xi 值相加除以所有 label 是 j 数据个数 sample mean

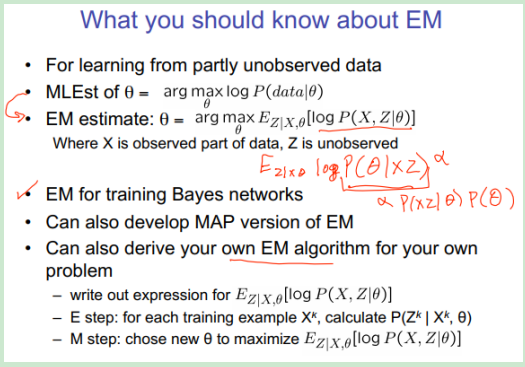
EM 此时由 z=1 的**个数**变成了 z 的均值

硬编码 z=1 是 1 z 是 0 变成 soft code z 是 0-1 的概率值



E: 计算每个 example 属于每个 cluster 的概率

M: 拿到概率就可以直接计算参数，然后再重复训练!!!



MAP 版本的 EM 如上 max E[log(P(theta|XZ))]=E[log P(X,Z|theta)P(theta)]

可以自己实现 EM

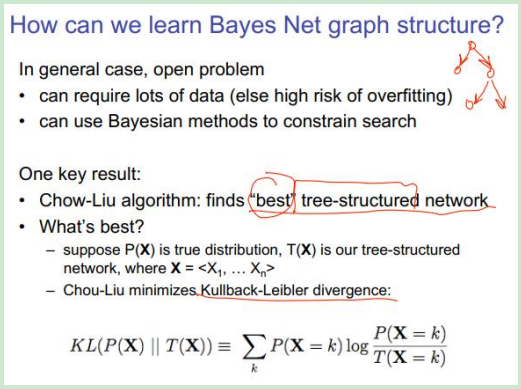
只要按照上面的**三步走!!**

Learning Bayes Net Structure

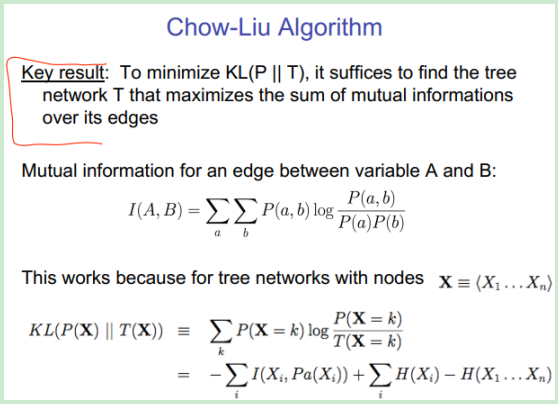
given data and 多个 structure

用什么来判断，**哪个 structure 更好？**

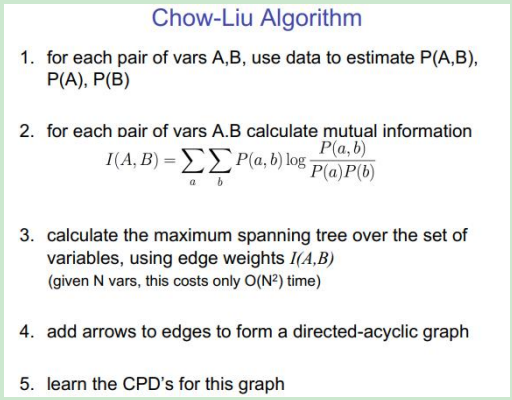
标准做法：用 **validation set 来挑选**，看哪个 strut 对于 validation 有更高的 **likelihood!!**
但有很多 graphic，**很难 search**



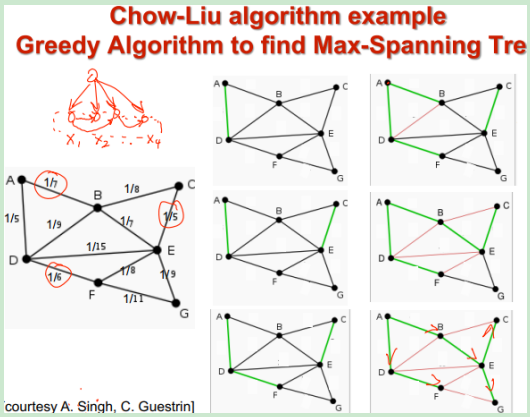
假设 restrict the structure a tree!! chow-liu alg
也就是每个 node 只有一个 immediate parent
best: minimize KL
P(X) 未知，是真正的 distribution



min KL 可以转化为 **max MI** **根据 I 的定义和 KL 的定义**
choose edge max 第一个 term，跟第二个 term 无关
question KL 和 mutual information 和 entropy!



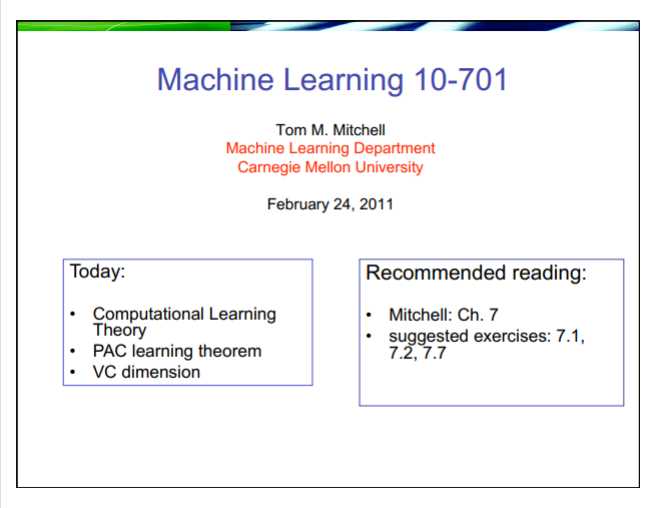
2 得到一个全连接 graph
3 用 maximum spanning tree 算法！

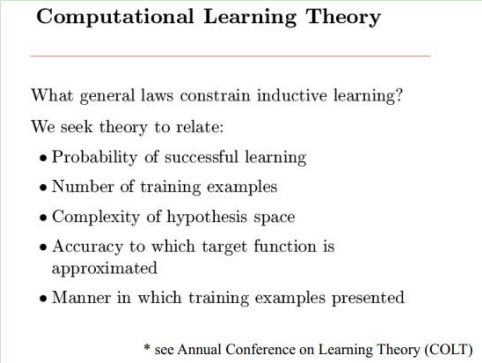


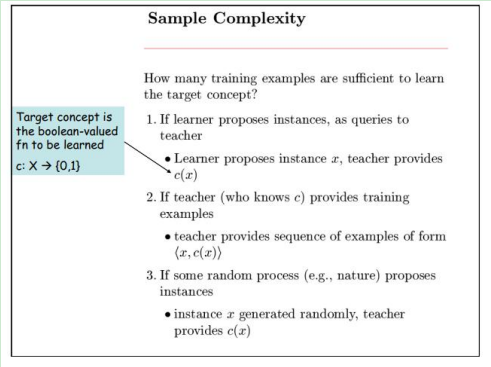
左边是 mutual information of graph
每次都选择**最大的** MI edge
等到 BD 时，不满足 tree

Bayes Nets – What You Should Know

- Representation
  - Bayes nets represent joint distribution as a DAG + Conditional Distributions
  - D-separation lets us decode conditional independence assumptions
- Inference
  - NP-hard in general
  - For some graphs, closed form inference is feasible
  - Approximate methods too, e.g., Monte Carlo methods, …
- Learning
  - Easy for known graph, fully observed data (MLE's, MAP est.)
  - EM for partly observed data
  - Learning graph structure: Chow-Liu for tree-structured networks

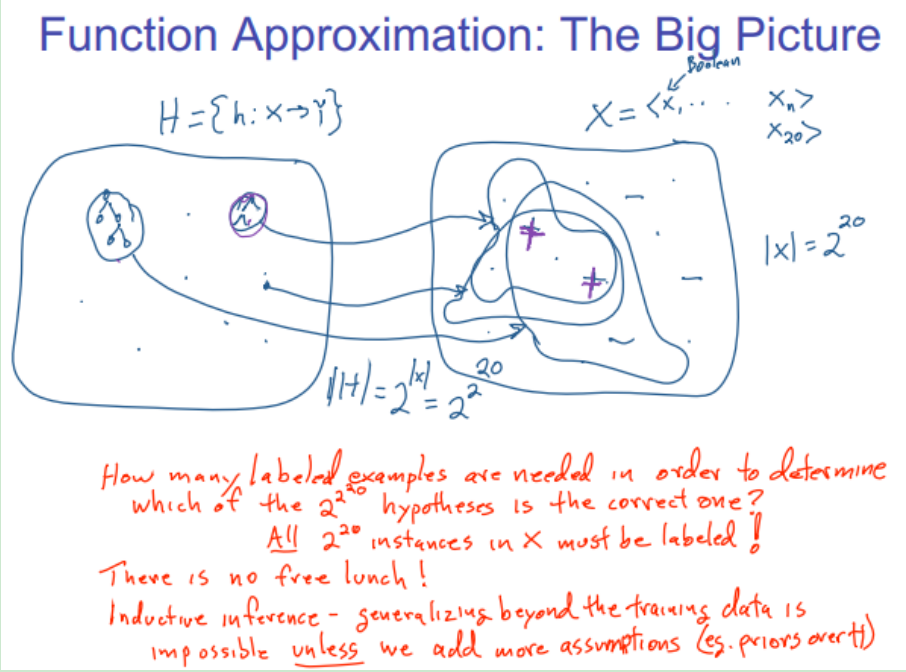






1 learner random choose arbitrary X,teacher label
需要多少 example to learn C(x) **(active learn)**
2 teacher choose particular example(efficient),给 learning **training**
3 都不 pick example，用随机过程来 choose

需要多少数据使得 learner 的错误有上界



右边是全部的 example，左边是 function space(可能是 DT LR)
给定一个 sample，可能存在多个 fucntion，标记对 sample
对于 DT,需要全部 example，error=0

需要 more strict，error 不能为 0
需要多少数据使得 learner 的**错误有上界**
我们假设 learner learn consisitent learner label 对**全部的** train data

但我们关系 true error

Sample Complexity: 3

Given:

- set of instances X
- set of hypotheses H
- set of possible target concepts $C = \{c: X \to \{0,1\}\}$
- training instances generated by a fixed, unknown probability distribution $\mathcal{D}$ over X ← $P(X)$

Learner observes a sequence D of training examples of form $\langle x, c(x)\rangle$, for some target concept $c \in C$

- instances x are drawn from distribution $\mathcal{D}$
- teacher provides target value $c(x)$ for each

Learner must output a hypothesis h estimating c

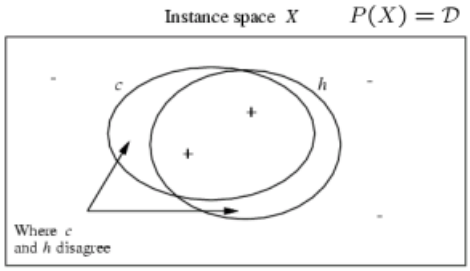
- h is evaluated by its performance on subsequent instances drawn according to $\mathcal{D}$

Note: randomly drawn instances, noise-free classifications

C 是 set of function，teacher teach 我们
sample 是按照一个分布来生成 random 产生
learner 选择一个 h 来 estimate c，然后用接下来的 sample 来验证准确率

x 是 random，label 没有 random noise

True Error of a Hypothesis



Definition: The **true error** (denoted $error_{\mathcal{D}}(h)$) of hypothesis h with respect to target concept c and distribution $\mathcal{D}$ is the probability that h will misclassify an instance drawn at random according to $\mathcal{D}$.

$$error_{\mathcal{D}}(h) \equiv \Pr_{x \in \mathcal{D}}[c(x) \neq h(x)]$$

true error of h 也就是对**接下来 sample 的**错误率

Two Notions of Error

Can we bound $error_{\mathcal{D}}(h)$ in terms of $error_{D}(h)$??

Training error of hypothesis h with respect to target concept c

- How often $h(x) \neq c(x)$ over training instances D

$$error_{D}(h) \equiv \Pr_{x \in D}[c(x) \neq h(x)] \equiv \frac{\sum_{x \in D} \delta(c(x) \neq h(x))}{|D|}$$

training examples

True error of hypothesis h with respect to c

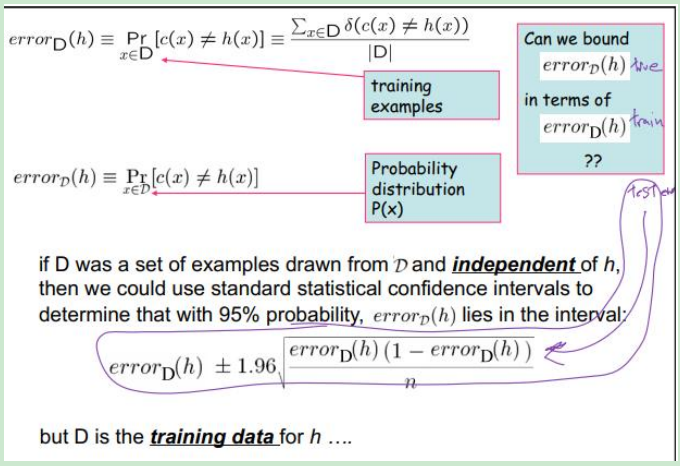
- How often $h(x) \neq c(x)$ over future instances drawn at random from $\mathcal{D}$

$$error_{\mathcal{D}}(h) \equiv \Pr_{x \in \mathcal{D}}[c(x) \neq h(x)]$$

Probability distribution P(x)

两种 error 的关系

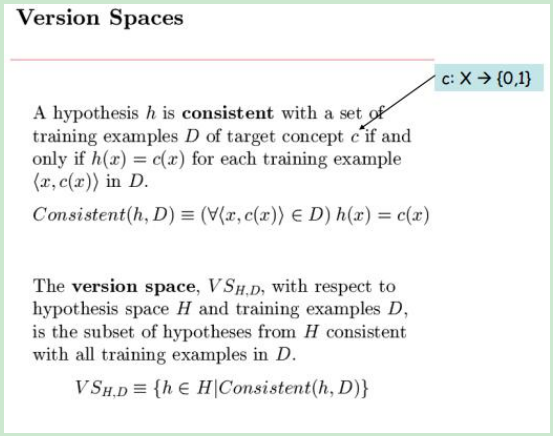
我们想 bound **ture eorror** in terms of **train error**



大 D 是 trian 小 D 是 true dtat sample from p(x)
test error:
假设 D 是从 true data 的 sample(test data)，计算 h，得到 error
问 how sure(confident) of the true error
和 test 大小有关，越大 test，confidence 越小
需要用到置信区间：
true error lies in 置信区间等于
test error+- 一个值，n 越大区间越小，越不 confident
n 是 true data 的数量

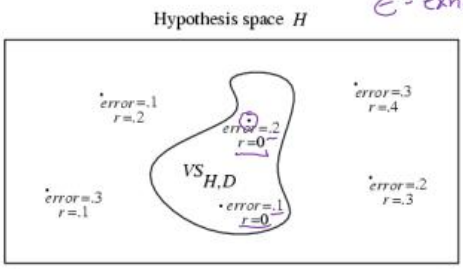
上面是 test error 和 true error 的关系

我们想知道 train error 和 true error 的关系!!



一个 D 对应一个 H space
D 包含 X 和对应的 Y，而 H 不是 true fucntion 不一定非要准确预测,Y 可以任意预测，其实就是 X 和 Y 的各种可能的组合|H|
VS 是 H 的 sunset of consistence h（多个可以将同一份 train label 全对的 h 集合）

Exhausting the Version Space



(r = training error, $error$ = true error)

Definition: The version space $VS_{H,D}$ is said to be ϵ-**exhausted** with respect to c and $\mathcal{D}$, if every hypothesis h in $VS_{H,D}$ has true error less than ϵ with respect to c and $\mathcal{D}$.

$$(\forall h \in VS_{H,D})\ error_{\mathcal{D}}(h) < \epsilon$$

VS epsilon-exhausted

epsilon 是小于 1 的很小的数

全部的 h，的 true error 都小于 epsilon

这样可以保证，VS 里所有 consitence h 可以 reach a upper bound

How many examples will ϵ-exhaust the VS?

Theorem: [Haussler, 1988].

If the hypothesis space H is finite, and D is a sequence of $m \geq 1$ independent random examples of some target concept c, then for any $0 \leq \epsilon \leq 1$, the probability that the version space with respect to H and D is not ϵ-exhausted (with respect to c) is less than

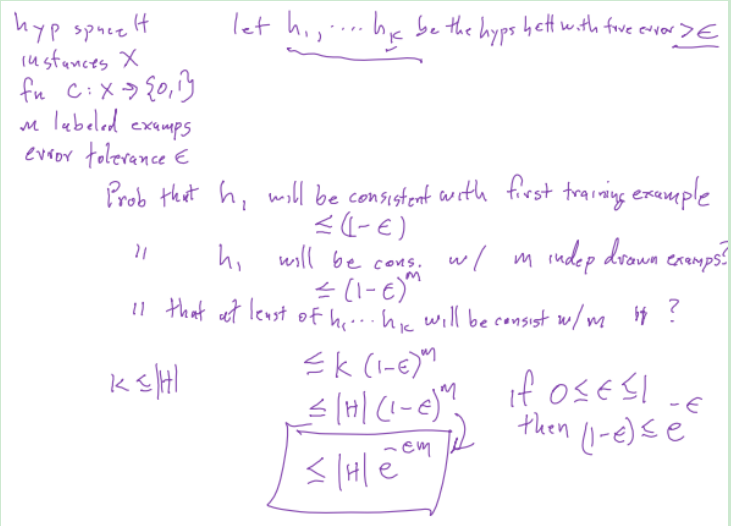
$$|H|e^{-\epsilon m}$$

D 是 sample from p(x)

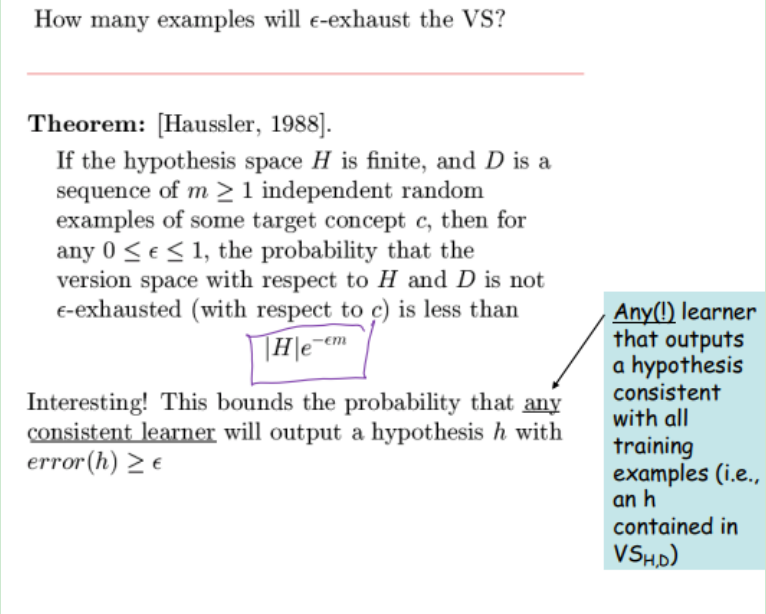
VC 不是 e-exhausted 的概率小于 有上界

随着 m 增加下降 随着 space 大小上升升高

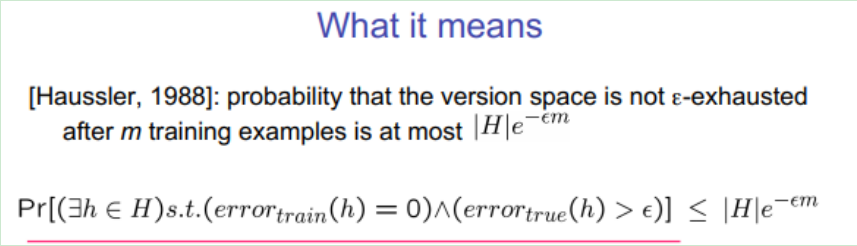
m 越大越好，space 大小越小越好



question 证明略过



对于所有 consistent learner error(h)>=epcilon 的概率有上界，被 bounded
(任意类型的 learner，只要 consitent)



VC 不是 e-exhausted 的概率小于　有上界
也就是存在一个 h 是 consistent learner，true error 小于 epsilon 的概率小于 有上界

Suppose we want this probability to be at most δ

1. How many training examples suffice?

$$m \geq \frac{1}{\epsilon}(\ln|H| + \ln(1/\delta))$$

2. If $error_{train}(h) = 0$ then with probability at least (1-δ):

$$error_{true}(h) \leq \frac{1}{m}(\ln|H| + \ln(1/\delta))$$

1 可以用来 choose train example 数量，可以 guantee 上界
2 又因为 true error 小于 epcilon，可以解出 epicilon

$$\Pr[(\exists h \in H) s.t. (error_{train}(h) = 0) \wedge (error_{true}(h) > \epsilon)] \leq |H| e^{-\epsilon m}$$

概率对应的**实验**是：从 P(X)多次 draw **不同的 train data**，不同的 set consistent h，记录他们的 true error 小于 epsilon 的次数

Example: H is Conjunction of Boolean Literals

$$m \geq \frac{1}{\epsilon}(\ln|H| + \ln(1/\delta))$$

Consider classification problem f:X→Y:

- instances: $X = \langle X_1\ X_2\ X_3\ X_4 \rangle$ where each X_i is boolean
- learned hypotheses are rules of the form:
  - IF $\langle X_1\ X_2\ X_3\ X_4 \rangle = \langle 0, ?, 1, ? \rangle$, THEN Y=1, ELSE Y=0
  - i.e., rules constrain any subset of the X_i

How many training examples *m* suffice to assure that with probability at least 0.99, *any* consistent learner will output a hypothesis with true error at most 0.05?

$$m \geq \frac{1}{.05}\left(\ln|H| + \ln\left(\frac{1}{.01}\right)\right)$$

$|H| = 3^n$

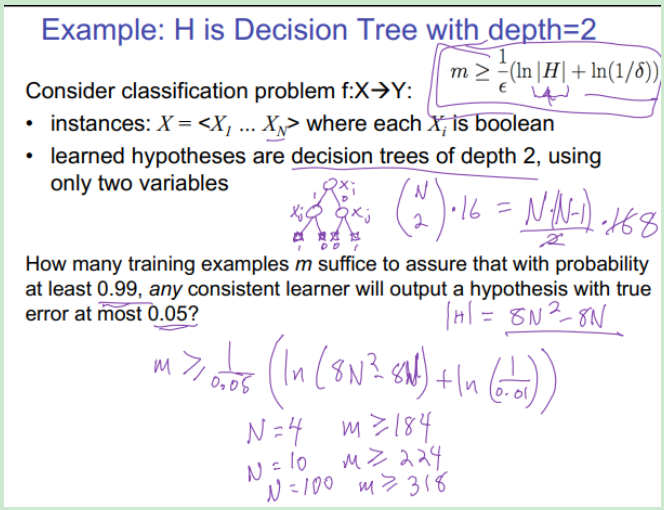
$n=4 \rightarrow m \geq 180$ ← 3^4, $n=4$

$\ln|H| = n \ln 3$

$n=10 \quad \geq 312$

$n=100 \quad \geq 2290$

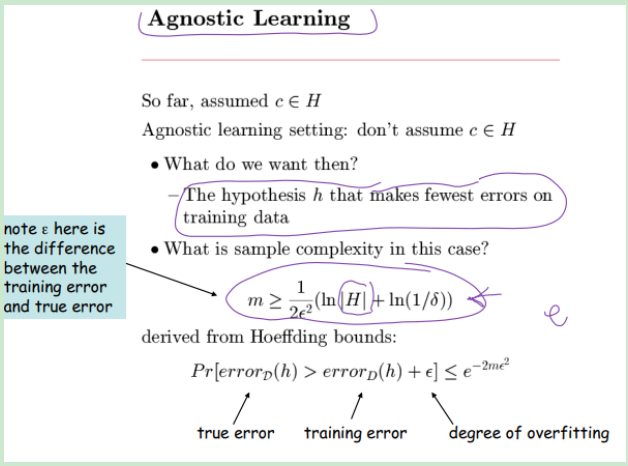
h: 如果 X，fit the pattern(正则只关心 X1 X3)，Y=1 或 0
假设 我们有 consistent learner(任意类型的 learner)
带入上面公式
|H| 只需考虑预测 1 的情况，X 有 3 的 4 次方的组合 question 越大，需要越多的 m



DT

N 个变量，但 DT 只用 2 个 先选两个 C N2 个 DT,然后最终的 leave 有 16 种结果，相乘 就是|H|(两个节点谁在上在下都是一样的 h，所以不考虑)

H 越复杂，需要越多的 data!!!



如果**不一定是 consistent learner**

Hoeffding bound

epcilon 是 true error 和 train error 的差

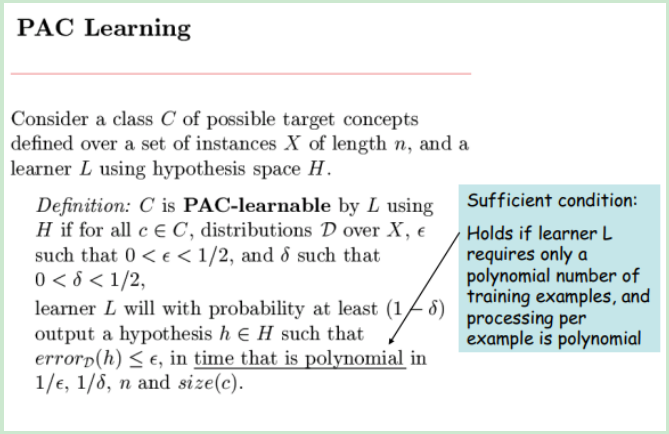
true error 会大一点

PAC Learning

Consider a class C of possible target concepts defined over a set of instances X of length n, and a learner L using hypothesis space H.

Definition: C is **PAC-learnable** by L using H if for all $c \in C$, distributions $\mathcal{D}$ over X, ϵ such that $0 < \epsilon < 1/2$, and δ such that $0 < \delta < 1/2$, learner L will with probability at least $(1 - \delta)$ output a hypothesis $h \in H$ such that $error_{\mathcal{D}}(h) \leq \epsilon$, in time that is polynomial in $1/\epsilon$, $1/\delta$, n and $size(c)$.

D 任意 distribution of X p(X)
Learner can do this in polynomial time of...
n coding 长度 of X,会受 example 数量影响



因此 learner 运行时间和 example 数量也有关
learner 只需要 polynomial number of training example
sample complexity

$$m \geq \frac{1}{\epsilon}(\ln |H| + \ln(1/\delta))$$

Question: If H = {h | h: X → Y} is infinite, what measure of complexity should we use in place of |H| ?

Answer: The largest subset of X for which H can guarantee zero training error (regardless of the target function c)

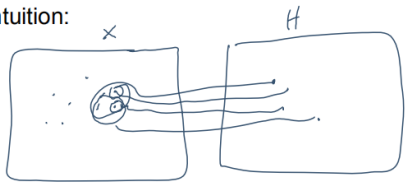
VC dimension of H is the size of this subset

只关心多少 h 可以 represent X. 这个 size 是 VC dimension

Question: If H = {h | h: X → Y} is infinite, what measure of complexity should we use in place of |H| ?

Answer: The largest subset of X for which H can guarantee zero training error (regardless of the target function c)

Informal intuition:



如果 H 可以表示，所有的 X 上的函数，我们需要**所有的 X**，才能找到这个 true hypothesis

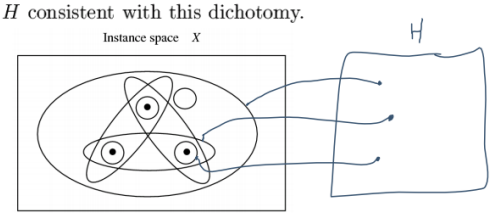
VC dimension=3，也就是 X 里有 3 个 example，h 可以保证 0 train error

Shattering a Set of Instances

Definition: a **dichotomy** of a set S is a partition of S into two disjoint subsets.

a labeling of each member of S as positive or negative

Definition: a set of instances S is **shattered** by hypothesis space H if and only if for every dichotomy of S there exists some hypothesis in H consistent with this dichotomy.



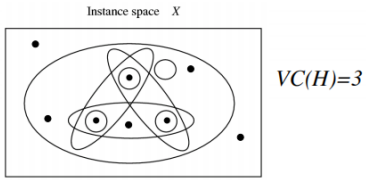
1 分割 set 其实就是每个 example 的 **labeling**

2 存在 h consistent with every dichotomy(各种可能的 labeling)

上面的图有 8 种 label 方法，如果存在 h 可以全部分对

The Vapnik-Chervonenkis Dimension

Definition: The **Vapnik-Chervonenkis dimension**, $VC(H)$, of hypothesis space H defined over instance space X is the size of the largest finite subset of X shattered by H. If arbitrarily large finite sets of X can be shattered by H, then $VC(H) \equiv \infty$.



VC(H)=3

也就是 size of largest subset of X, H 能保证存在 h 可以 0 train error(无论 X 如何 label)

Sample Complexity based on VC dimension

How many randomly drawn examples suffice to ε-exhaust $VS_{H,D}$ with probability at least (1-δ)?

ie., to guarantee that any hypothesis that perfectly fits the training data is probably (1-δ) approximately (ε) correct

$$m \geq \frac{1}{\epsilon}(4\log_2(2/\delta) + 8VC(H)\log_2(13/\epsilon))$$

Compare to our earlier results based on $|H|$:

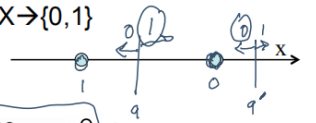
$$m \geq \frac{1}{\epsilon}(\ln(1/\delta) + \ln|H|)$$

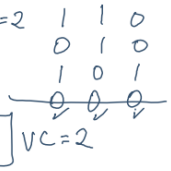
之前用的是 count of H
现在用 VC d 代替

VC dimension: examples

Consider X = ℝ, want to learn c:X→{0,1}
What is VC dimension of

- Open intervals:
  - H1: if $x > a$ then $y = 1$ else $y = 0$ VC=1
  - H2: if $x > a$ then $y = 1$ else $y = 0$ VC=2
    or, if $x > a$ then $y = 0$ else $y = 1$





- Closed intervals:
  - H3: if $a < x < b$ then $y = 1$ else $y = 0$ VC=2
  - H4: if $a < x < b$ then $y = 1$ else $y = 0$
    or, if $a < x < b$ then $y = 0$ else $y = 1$

X 是线上的点，我们看一下我们的 H 的 VC d
H1 ： 只对于一个 example，可以任意 label a
但对于两个 example，没法标记 0,1(11,00,10 都可以)
因此 VC d 是 1
H2 ： VC d 是 2 可以标记标记 0,1

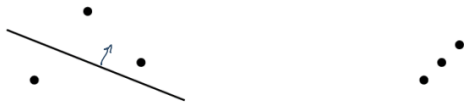
H3 ： 如果三个 example 101 就就不能被分开 VC d=2
H4 ： VC=4 因为 0101 不行

VC dimension: examples

$X = \mathbb{R}^2$

What is VC dimension of lines in a plane?

- $H_2 = \{ ((w_0 + w_1x_1 + w_2x_2)>0 \rightarrow y=1) \}$ $VC \geq 3$



X 是平面上的点 H2 的 VC d=3，三个点任意 label，都有 h 可以 0 train error

VC dimension: examples

What is VC dimension of

- $H_2 = \{ ((w_0 + w_1x_1 + w_2x_2)>0 \rightarrow y=1) \}$
  - $VC(H_2)=3$
- For H_n = linear separating hyperplanes in n dimensions, $VC(H_n)=n+1$

线性分类器，在 n dimension 的 X 上，VC=n+1
可以带入公式，得到我们的 example 数量

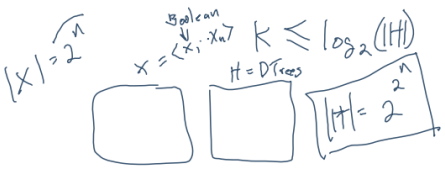
$$m \geq \frac{1}{\epsilon}(4\log_2(2/\delta) + 8VC(H)\log_2(13/\epsilon))$$

X dimension 增加 VC 也增加 question 需要的 m 更多!!!
feature 越多，越容易被分开？

For any finite hypothesis space H, can you give an upper bound on VC(H) in terms of |H| ? (hint: yes)

$VC(H) = K$

H can express $\geq 2^K$ fns.

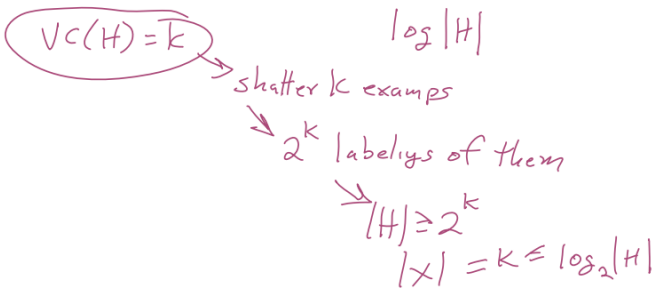


对于有限的 H space，的 VC d 有没有上限？
假设 VC=K K 个 example，有几种 label 结果？ 2 的 k 次方，此时只是考虑 label 的分类结

果，还没有考虑 X，但 H 至少是 2 的 k 次方，**因此 K 有上界**

对于 DT　　X n 个 feature
X example 有 2 的 n 次方，每一个 example 有不同的分类结果 2 的 2 的 n 次方，**也就是 H 的数量，　log2 以后 K 还是很大，保证 0 train error**

Can you give an upper bound on VC(H) in terms of |H|, for any hypothesis space H? (hint: yes)



Tightness of Bounds on Sample Complexity

How many examples m suffice to assure that any hypothesis that fits the training data perfectly is probably (1-δ) approximately (ε) correct?

$$m \geq \frac{1}{\epsilon}(4\log_2(2/\delta) + 8VC(H)\log_2(13/\epsilon))$$

How tight is this bound?

Lower bound on sample complexity (Ehrenfeucht et al., 1989):

Consider any class C of concepts such that VC(C) > 1, any learner L, any 0 < ε < 1/8, and any 0 < δ < 0.01. Then there exists a distribution $\mathcal{D}$ and a target concept in C, such that if L observes fewer examples than

$$\max\left[\frac{1}{\epsilon}\log(1/\delta), \frac{VC(C)-1}{32\epsilon}\right]$$

Then with probability at least δ, L outputs a hypothesis with $error_{\mathcal{D}}(h) > \epsilon$

上面的 bound 是最坏的情况，是 teacher 随机给的 X example
另一种 bound：
teacher 可以 choose X，需要更少的 example

PAC Learning: What You Should Know

- PAC learning: Probably (1-δ) Approximately (error ε) Correct
- Problem setting

- Finite H, perfectly consistent learner result
- If target function is not in H, *agnostic learning*
- If |H| = ∞ , use VC dimension to characterize H

- Most important:
  - Sample complexity grows with complexity of H
  - Quantitative characterization of overfitting

- Much more: see Prof. Blum's course on Computational Learning Theory

most important：

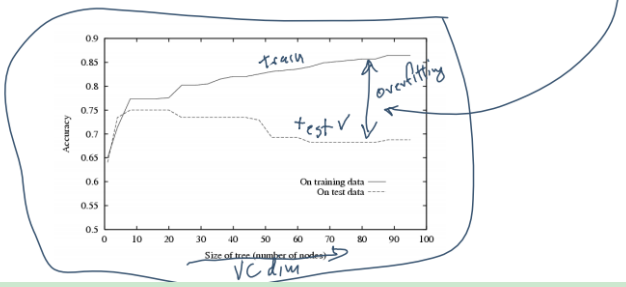
question　这节课后面的 mid term 总结没有听

Agnostic Learning: VC Bounds

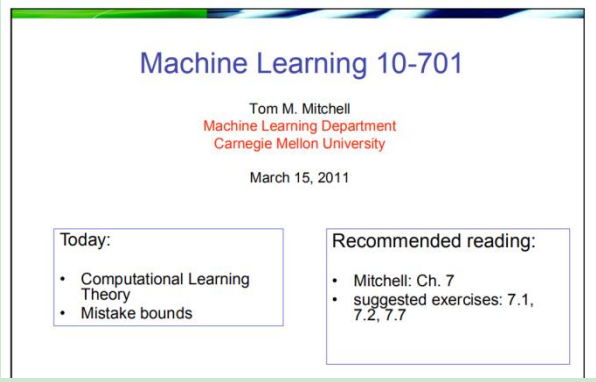
[Schölkopf and Smola, 2002]

With probability at least (1-δ) every $h \in H$ satisfies

$$error_{true}(h) < error_{train}(h) + \sqrt{\frac{VC(H)(\ln \frac{2m}{VC(H)} + 1) + \ln \frac{4}{\delta}}{m}}$$



VC d 增加 **overfitting 增加！！！**



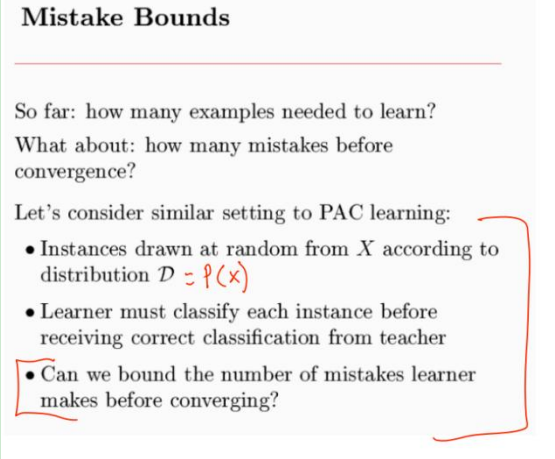
Machine Learning 10-701

Tom M. Mitchell
Machine Learning Department
Carnegie Mellon University

March 15, 2011

Today:
- Computational Learning Theory
- Mistake bounds

Recommended reading:
- Mitchell: Ch. 7
- suggested exercises: 7.1, 7.2, 7.7



Mistake Bounds

So far: how many examples needed to learn?

What about: how many mistakes before convergence?

Let's consider similar setting to PAC learning:

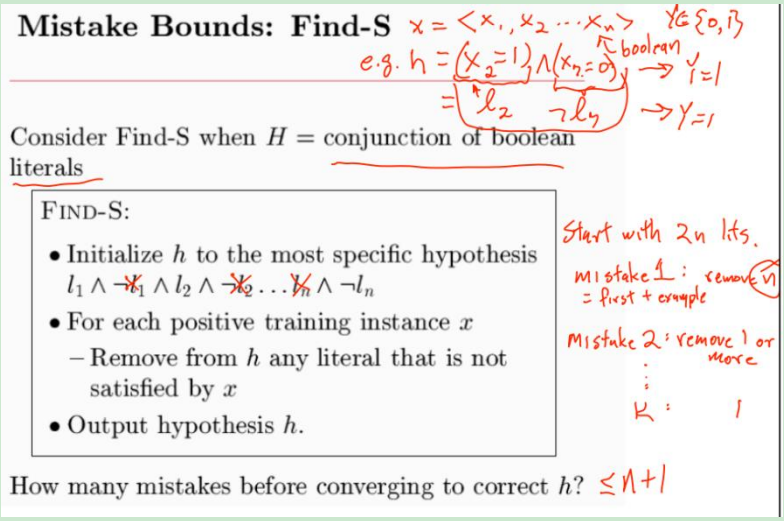
- Instances drawn at random from X according to distribution $\mathcal{D}$ = P(X)
- Learner must classify each instance before receiving correct classification from teacher
- Can we bound the number of mistakes learner makes before converging?

consistent learner

达到最优 error 之前，会犯多少错误？

online learning：example 一直进来，要犯多少错误，model 才能稳定

learner 必须先分类，然后才能知道**正确答案，学习！**



Mistake Bounds: Find-S $x = \langle x_1, x_2 \cdots x_n \rangle$ $y \in \{0,1\}$ boolean

e.g. $h = (x_2 = 1) \wedge (x_4 = 0) \rightarrow y = 1$

$= l_2 \; \neg l_4 \rightarrow y = 1$

Consider Find-S when H = conjunction of boolean literals

FIND-S:
- Initialize h to the most specific hypothesis $l_1 \wedge \neg l_1 \wedge l_2 \wedge \neg l_2 \ldots l_n \wedge \neg l_n$
- For each positive training instance x
  - Remove from h any literal that is not satisfied by x
- Output hypothesis h.

Start with 2n lits.

Mistake 1: remove n (= first + example)

Mistake 2: remove 1 or more

⋮

K: 1

How many mistakes before converging to correct h? $\leq n+1$

X 是 boolean feature

所有的 **hypothesis 都是 conjunction of the X value**，如果满足 Y=1

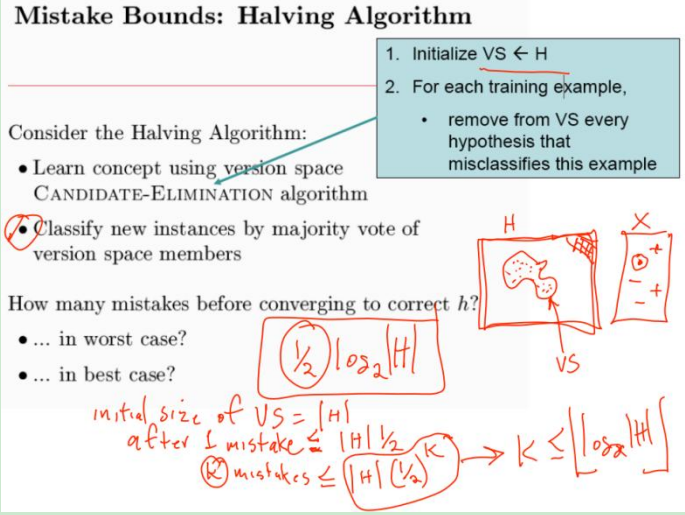
上面也可以用 1 表示，1 代表是 1，非 1 是 0

1 initial h 对于所有 X， 一直输出 Y=0，一旦 mistake，就 modify!!
2 train 负样本忽略(因为没有分错)，对于正样本(分类错误)，删掉所有不满足的 literal 上面红叉
假设 x1=1 x2=1 xn=0，删掉 not x1，not x2 和 xn，这些都是为 0 的
最后 output h

最开始有 2n literals
第一个正样本：需要 remove n 个 literal
第二个正样本：最理想的情况下，只需 remove 1 个 literal
因为第一次已经去掉一半了，不保证一直输出 Y=0 了
也就是需要 remove 1 or more than1

最多 n+1,不然没得删了

不同的算法有不同的 mistake bound 公式



得到 VS 的算法：initialize VS(consistent learner)为全部的 H
对于所有 train example 去除分错的 h
右边图：
左边是我们剩下的 VS，右边的是 new example
VS 的所有 h 会对 new example 投票
如果投对了，没投对的 h 被 removed， VS 范围缩小了
mistake:大多数 h 都投错

假设;每个 h 有 0.5 的概率 vote 正确，每次都淘汰一半，最后只剩一个
initial VS=|H|
after one mistake :most h wrong, VS<=0.5|H|
after k mistake VS<=0.5 的 K 次方*|H|
when VS=1? 让 0.5 的 K 次方*|H|=1 如上面公式 ，得到 K

Optimal Mistake Bounds

Let $M_A(C)$ be the max number of mistakes made by algorithm A to learn concepts in C. (maximum over all possible $c \in C$, and all possible training sequences)

$$M_A(C) \equiv \max_{c \in C} M_A(c)$$

Definition: Let C be an arbitrary non-empty concept class. The **optimal mistake bound** for C, denoted $Opt(C)$, is the minimum over all possible learning algorithms A of $M_A(C)$.

$$Opt(C) \equiv \min_{A \in learning\ algorithms} M_A(C)$$

$$VC(C) \leq Opt(C) \leq M_{Halving}(C) \leq log_2(|C|).$$

MAC 是最坏的情况，其中里面最好的 A 算法，optimal mistake bound question？？

因为 VC 是我们的 example 数量，除非全 mistake

Weighted Majority Algorithm

a_i denotes the i^{th} prediction algorithm in the pool A of algorithms. w_i denotes the weight associated with a_i.

- For all i initialize $w_i \leftarrow 1$
- For each training example $\langle x, c(x) \rangle$
  - Initialize q_0 and q_1 to 0
  - For each prediction algorithm a_i
    - If $a_i(x) = 0$ then $q_0 \leftarrow q_0 + w_i$
    - If $a_i(x) = 1$ then $q_1 \leftarrow q_1 + w_i$
  - If $q_1 > q_0$ then predict $c(x) = 1$
    If $q_0 > q_1$ then predict $c(x) = 0$
    If $q_1 = q_0$ then predict 0 or 1 at random for $c(x)$
  - For each prediction algorithm a_i in A do
    If $a_i(x) \neq c(x)$ then $w_i \leftarrow \beta w_i$

when β=0, equivalent to the Halving algorithm...

$\beta = 0.5$

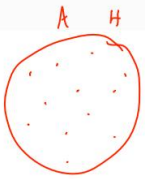
question 待看

Weighted Majority

Even algorithms that learn or change over time...

[Relative mistake bound for WEIGHTED-MAJORITY] Let D be any sequence of training examples, let A be any set of n prediction algorithms, and let k be the minimum number of mistakes made by any algorithm in A for the training sequence D. Then the number of mistakes over D made by the WEIGHTED-MAJORITY algorithm using $\beta = \frac{1}{2}$ is at most

$2.4(k + \log_2 n)$ $\geq$ # mistakes by Wtd Maj



let M be # of mistakes made by Wtd Maj. Alg using n algs.

K # " " by best $a_i \in A$

$W = \sum_i w_i$

What is final wt of alg a_i? $\left(\frac{1}{2}\right)^k$

What is final $\sum_{j=1}^{n} w_j$

What is initial $W = n$

after mistake #1, $W \leq \frac{3}{4} n$

after mistake M, $W \leq \left(\frac{3}{4}\right)^M n$

$w_i \leq W$

$\left(\frac{1}{2}\right)^k \leq W \leq \left(\frac{3}{4}\right)^M n$

$\left(\frac{1}{2}\right)^k \leq \left(\frac{3}{4}\right)^M n$

Machine Learning 10-701

Tom M. Mitchell
Machine Learning Department
Carnegie Mellon University

March 17, 2011

Today:

- Semi-supervised learning
- Co-Training
- Never ending learning

Recommended reading:
(see class website)

- Carlson et al., 2010
- Blum & Mitchell 1998

When can Unlabeled Data Help Learn f: X→Y ?

Consider problem setting:

- Set *X* of instances drawn from unknown distribution *P(X)*
- Wish to learn target function *f*: *X*→ *Y* (or, P(Y|X))
- Given a set *H* of possible hypotheses for *f*

Given:

- i.i.d. labeled examples $L = \{\langle x_1, y_1\rangle \ldots \langle x_m, y_m\rangle\}$
- i.i.d. unlabeled examples $U = \{x_{m+1}, \ldots x_{m+n}\}$

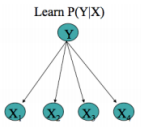
Wish to find hypothesis with lowest true error:

$$\hat{f} \leftarrow \arg\min_{h \in H} \Pr_{x \in P(X)} [h(x) \neq f(x)]$$

有 label 也有 unlabel
最终使得 H(X)和 true function f(X)不一致的概率最小

When can Unlabeled Data Help Learn f: X→Y ?

- EM



Y	X1	X2	X3	X4
1	0	0	1	1
0	0	1	0	0
0	0	0	1	0
?	0	1	1	0
?	0	1	0	1

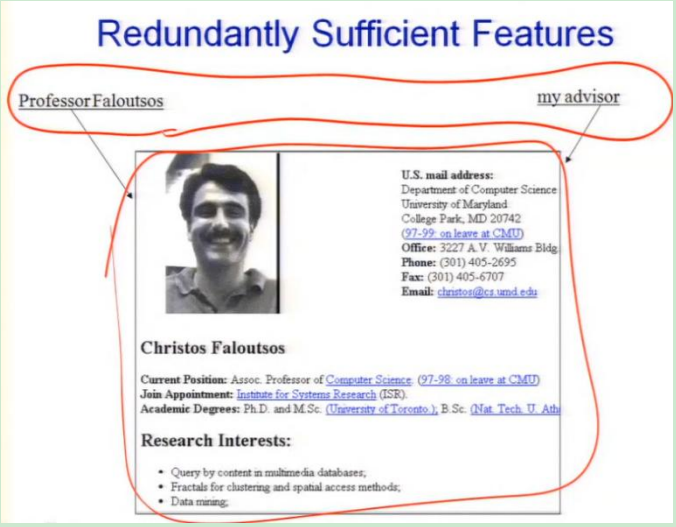
- Metric regularization
  - [Schuurmans & Southey, MLJ 2002]
  - use unlabeled data to detect (and avoid) overfitting
- CoTraining, Multiview learning, CoRegularization

可以用 EM
也有其他 4 个方法

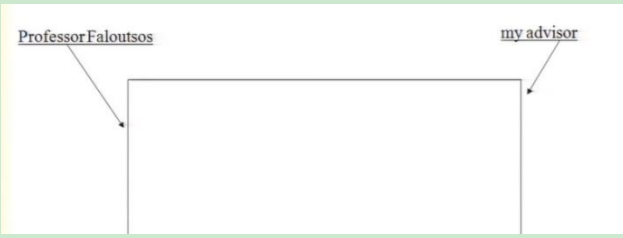
CoTraining

- In some settings, available data features are redundant and we can train two classifiers based on disjoint features
- In this case, the two classifiers should agree on the classification for each unlabeled example
- Therefore, we can use the unlabeled data to constrain joint training of both classifiers

feature 冗余
可以 train 2 classifier using 2 subset X
这两个算发应该在 unlabeled 的 data 上得到的结果一致!!
可以用这个来约束，来 cotrian 2 classifier



classifier web page; whether faculty page
feature 是 1 正文，2 也包括 link，可以 link 进来的连接，冗余了



两个 subset feature 都可以足够做分类

CoTraining Algorithm #1
[Blum&Mitchell, 1998]

Given: labeled data L,

unlabeled data U

Loop:

Train g1 (hyperlink classifier) using L

Train g2 (page classifier) using L

Allow g1 to label *p* positive, *n* negative examps from U

Allow g2 to label *p* positive, *n* negative examps from U

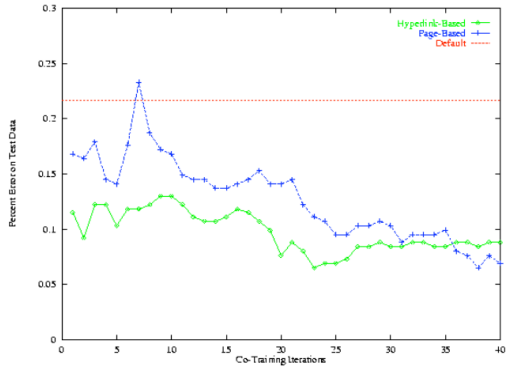
Add these self-labeled examples to L

1 先用 label data L train 两个 classifier
2 让后来 label unlabel data
3 然后将被标记的 labeled data(不是全部，agree 而且是最 confident 的) 到 L,retrain
每次 pick up more unlabeled data
这样做的目的是我们想从 unlabeled data 里获取信息

CoTraining: Experimental Results

- begin with 12 labeled web pages (academic course)
- provide 1,000 additional unlabeled web pages
- average error: learning from labeled data 11.1%;
- average error: cotraining 5.0%

Typical run:



随着 iteration，error 都会下降，并且接近

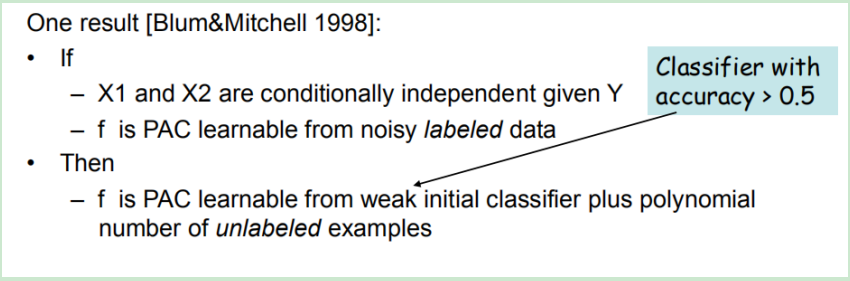
CoTraining setting:

- wish to learn f: X → Y, given L and U drawn from P(X)
- features describing X can be partitioned (X = X1 x X2)

such that f can be computed from either X1 or X2

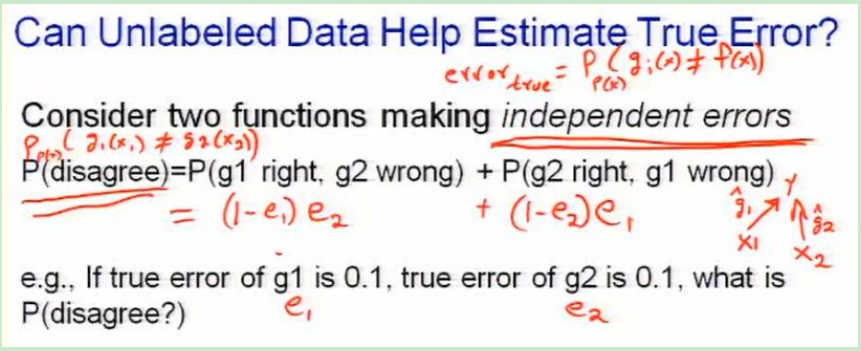
$$(\exists g_1, g_2)(\forall x \in X) \quad g_1(x_1) = f(x) = g_2(x_2)$$

feature 可以被 partitioned X1 X2
true function 可以从 X1 X2 计算到
最后两个 fuction 会相同



f 是 PAC

weak initial classifier 就是我们的 labeled example train classifier



f 的 true error 可以 bounded by **g1 g2 disagree on unlabeled data**

p(g1,g1 disagree)

$$\text{error}_{\text{true}} = P_{P(x)}(g_i(x) \neq f(x))$$

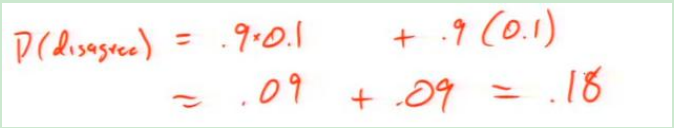
p(x)是 distribution we draw

$$P(disagree) = P_{P(x)}(g_1(x) \neq g_1(x))$$

假设 g1 g2 的 true error **independent**(link 和正文都是由不同的人独立判断)

P(g1 对，g2 错)=（1-e1）e2

g1 g2 disagree 的概率：假设 g1g2 的 true error 相等



可以由 0.18 反推 P1P2 true error

0.18 就是 g1g2 对于 unlabeled data 的 disagree 率

PAC Generalization Bounds on CoTraining

[Dasgupta et al., NIPS 2001]

This theorem assumes X1 and X2 are conditionally independent given Y

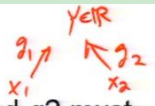
Theorem 1 *With probability at least $1-\delta$ over the choice of the sample S, we have that for all h_1 and h_2, if $\gamma_i(h_1,h_2,\delta) > 0$ for $1 \le i \le k$ then (a) f is a permutation and (b) for all $1 \le i \le k$,*

$$P(h_1 \neq i \mid f(y) = i, h_1 \neq \perp) \;\le\; \frac{\widehat{P}(h_1 \neq i \mid h_2 = i, h_1 \neq \perp) + \epsilon_i(h_1,h_2,\delta)}{\gamma_i(h_1,h_2,\delta)}.$$

The theorem states, in essence, that if the sample size is large, and h_1 and h_2 largely agree on the unlabeled data, then $\widehat{P}(h_1 \neq i \mid h_2 = i, h_1 \neq \perp)$ is a good estimate of the error rate $P(h_1 \neq i \mid f(y) = i, h_1 \neq \perp)$.

核心思想是：the assumption the reason 我们可以同时用 p1 p2 来计算 Y，是由 constrain：p1 p2 必须 agree on unlabeled data

Co Regularization



- Let's build our assumption that g1 and g2 must agree directly into the objective we're optimizing

- e.g.,

$$\langle \theta_1, \theta_2 \rangle \leftarrow \arg\min_{\langle \theta_1,\theta_2 \rangle} \sum_{x^l \in L} (y^l - g_1(x^l;\theta_1))^2 + \sum_{x^l \in L} (y^l - g_2(x^l;\theta_2))^2 + \sum_{x^u \in U} (g_1(x^u;\theta_1) - g2(x^u;\theta_2))^2$$

Max likelihood over L

假设 g1 g2 的参数分别是 theta1 theta2
不仅要 minimize g1 g2 的错误(在)
还要 minimize the disagreement of g1 g2
对于 noise data 非常 robust

CoTraining Summary

- Unlabeled data improves supervised learning when example features are redundantly sufficient
  - Family of algorithms that train multiple classifiers
- Theoretical results
  - If X1,X2 conditionally independent given Y, Then
    - PAC learnable from weak initial classifier plus unlabeled data
    - disagreement between g1(x1) and g2(x2) bounds final classifier error
- Many real-world problems of this type
  - Semantic lexicon generation [Riloff, Jones 99], [Collins, Singer 99]
  - Web page classification [Blum, Mitchell 98]
  - Word sense disambiguation [Yarowsky 95]
  - Speech recognition [de Sa, Ballard 98]
  - Visual classification of cars [Levin, Viola, Freund 03]

Further Reading

- Semi-Supervised Learning, O. Chapelle, B. Sholkopf, and A. Zien (eds.), MIT Press, 2006. (excellent book)
- EM for Naïve Bayes classifiers: K.Nigam, et al., 2000. "Text Classification from Labeled and Unlabeled Documents using EM", *Machine Learning,* 39, pp.103—134.
- CoTraining: A. Blum and T. Mitchell, 1998. "Combining Labeled and Unlabeled Data with Co-Training," *Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT-98).*
- S. Dasgupta, et al., "PAC Generalization Bounds for Co-training", *NIPS 2001*
- Model selection: D. Schuurmans and F. Southey, 2002. "Metric-Based methods for Adaptive Model Selection and Regularization," Machine Learning, 48, 51—84.

如果假设满足 cotrain 会 performance better than EM

Never Ending Learning

Tom M. Mitchell

Justin Betteridge, Jamie Callan, Andy Carlson, William Cohen, Estevam Hruschka, Bryan Kisiel, Mahaveer Jain, Jayant Krishnamurthy, Edith Law, Thahir Mohamed, Mehdi Samadi, Burr Settles, Richard Wang, Derry Wijaya

Machine Learning Department
Carnegie Mellon University

March 2011

Humans learn many things, for years, and become better learners over time

Why not machines?

学习的时间越长,而且learn很多任务,应该越来越智能,never ending learning,像人一样

Never Ending Learning

Task: acquire a growing competence without asymptote

- over years
- multiple functions
- where learning one thing improves ability to learn the next
- acquiring data from humans, environment

Many candidate domains:

- Robots
- Softbots
- Game players
- Tweeters

multiple function 同时学习多任务!!

NELL: Never-Ending Language Learner

Inputs:

- initial ontology
- handful of examples of each predicate in ontology
- the web
- occasional interaction with human trainers

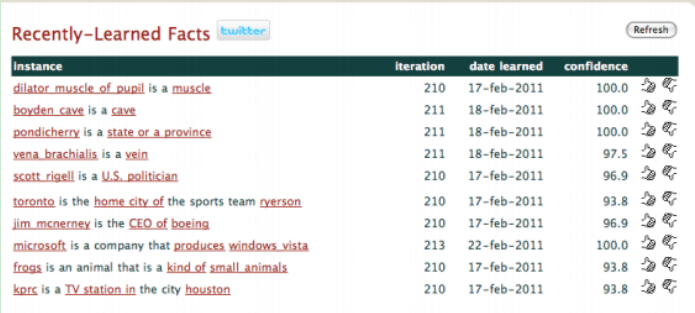
The task:

- run 24x7, forever
- each day:
  1. extract more facts from the web to populate the initial ontology
  2. learn to read (perform #1) better than yesterday

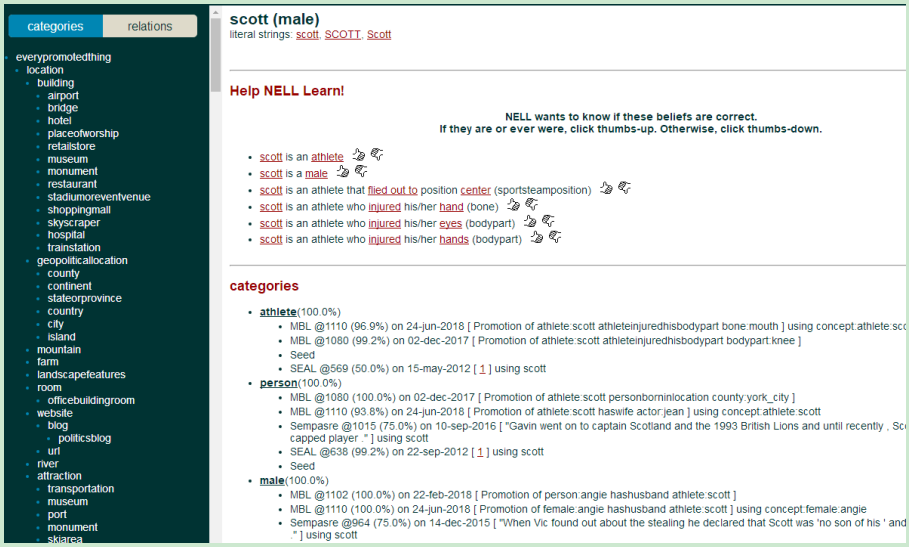
learn read the web

NELL Today

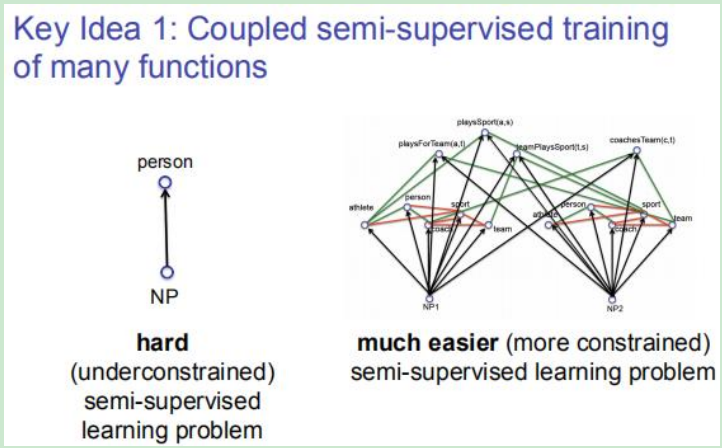
- http://rtw.ml.cmu.edu
- eg., “Disney”, “Mets”, “IBM”, “Pittsburgh” …



learn the fact，也就是 term 和 term 的关系



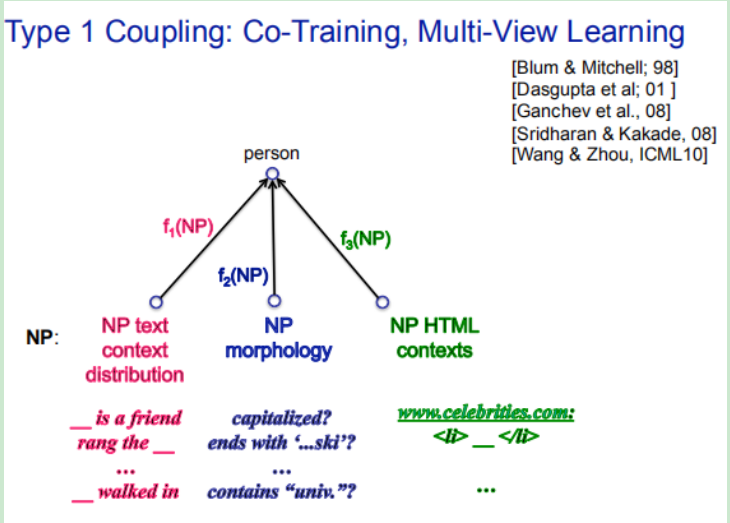
其实就是多个 classifier，对同一个 term



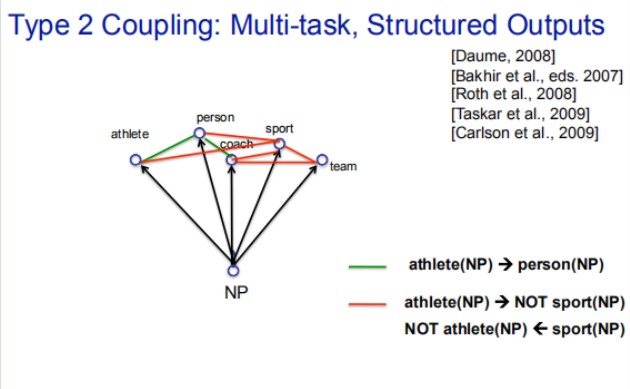
NP 是 norn phrase 名词!!!

learn a little labeled data, then learn from unlabeled data from web

全部两两配对约束



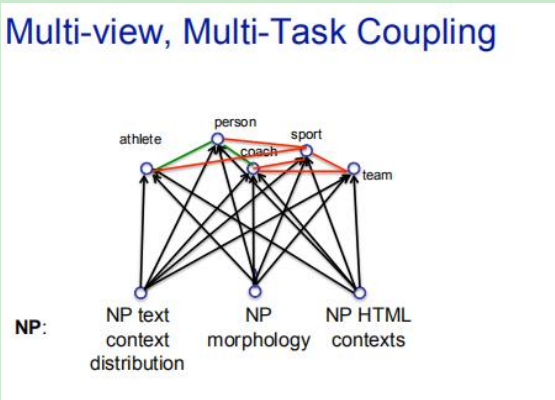
可以用不同的 view 角度对 term 分类
1 上下文　2 语法 3 链接的规律　因此是属于不同的 feature，做同一种分类
这就形成了 co-training，可以两两配对 co-train
对于 unlabeled data，同一个 term 理论上应该三个算法 **agree**，如果 disagree 就需要 change 了，modify 算法(function 肯定有错误)
对于输出同一 label 的 function 进行 1constrain



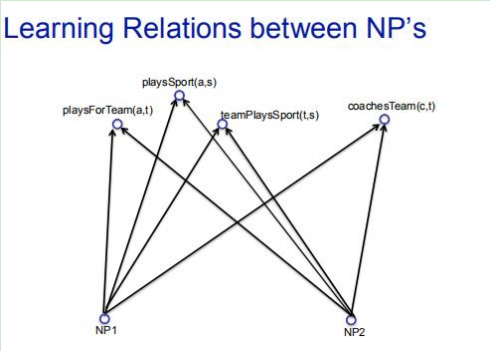
对 NPterm 进行 更多的分类进行预测，每一个箭头独立的 classifier

但类别和类别直接也有约束
person 是 1　athlete 是 0 没问题
athlete 是 1　person　是 0　有问题，需要调整算法(肯定有一个 fuction 有错误)
我们可以 2constrain on 输出不同 label 的 function!!!(独立的 classifier)

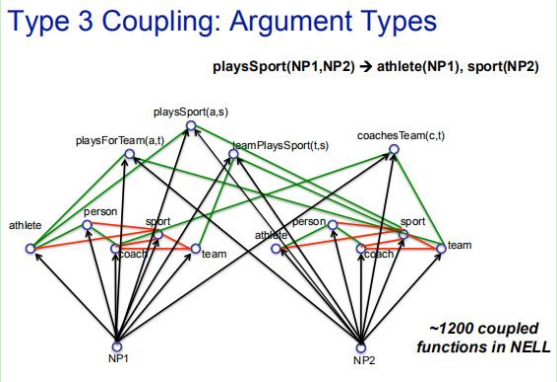
将两种 **constrain** 结合在一起



不同的 view+不同类别的 classifier



最后：
不再是 learn 分类，而是 learn NP pair 的 relation
需要 NP pair 的 classifier，来判断 NP pair 的 relation
这些 relation 在 function 直接也需要 3constrain

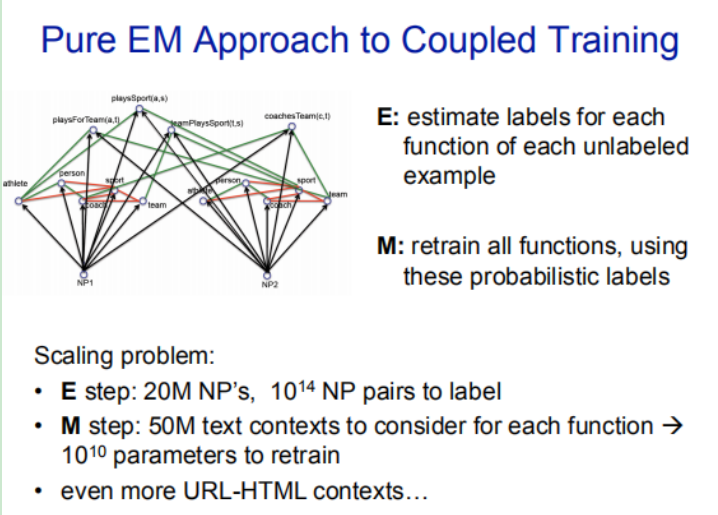


得到最终的形式

1 semi-supervise 让 system learn 许多个 classifier，这样会更好的利用大量的 unlabeled data

2 对于 classifier 增加更多的 constrains，更好的利用大量的 unlabeled data

3 给 system stage 的课程，从简单的 classifier 开始学起，然后越来越复杂一直 improve！



E:start with **labeled data**，然后 **estimate** 所有的 unlabeled data(全部 NP)
M:retain all function
但是计算量太大，没办法用 EM，太慢了!!

NELL's Approximation to EM

E' step:

- Consider only a growing subset of the latent variable assignments
  - category variables: up to 250 new NP's per category per iteration
  - relation variables: add only if confident and args of correct type
  - this set of explicit latent assignments ***IS*** the knowledge base

M' step:

- Each view-based learner retrains itself from the updated KB
- "context" methods create growing subsets of contexts

需要近似的算法

E: 只 label 我们 most confident 的 unlabeled NP(subset of unlabeled 变量)

unlabeled 变量就是 latent 变量

knowledge base 就是我们最近的 label assignment

M: retrain 也是用 subset of feature, mutual information 最大的 feature

Machine Learning 10-701

Tom M. Mitchell
Machine Learning Department
Carnegie Mellon University

March 22, 2011

Today:

- Time series data
- Markov Models
- Hidden Markov Models
- Dynamic Bayes Nets

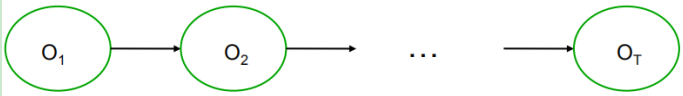
Reading:

- Bishop: Chapter 13 (very thorough)

thanks to Professors Venu Govindaraju, Carlos Guestrin, Aarti Singh, and Eric Xing for access to slides on which some of these are based

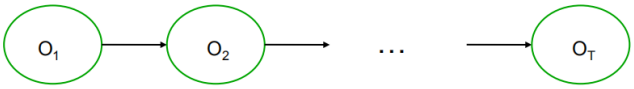
Sequential Data

- stock market prediction
- speech recognition
- gene data analysis



how shall we represent and learn $P(O_1, O_2 \dots O_T)$?

Markov Model

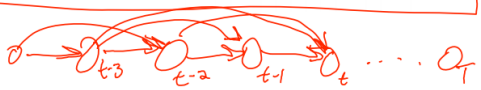


how shall we represent and learn $P(O_1, O_2 \dots O_T)$?

Use a Bayes net: $P(O_1 \dots O_T) = \prod_{t=1}^{T} P(O_t | Pa(O_t))$

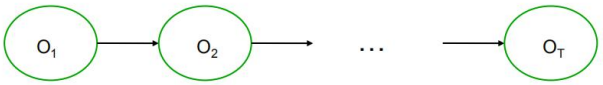
Markov model: $Pa(O_t) \equiv O_{t-1}$

nth order Markov model: $Pa(O_t) \equiv O_{t-1}, O_{t-2}, \dots O_{t-n}$



Markov Model 假设：observation 只 depend on **有限的 past 节点**
此时假设是只 depend on immediate parent，GM 是一条链
nth order Markov Model,depend on n past 节点
n 越大，计算量越大

Markov Model



how shall we represent and learn $P(O_1, O_2 \dots O_T)$?

Use a Bayes net: $P(O_1 \dots O_T) = \prod_{t=1}^{T} P(O_t | Pa(O_t))$

Markov model: $Pa(O_t) \equiv O_{t-1}$

nth order Markov model: $Pa(O_t) \equiv O_{t-1}, O_{t-2}, \dots O_{t-n}$

if O_t real valued and assume $P(O_t) \sim N(f(O_{t-1}, O_{t-2} \dots O_{t-n}), \sigma)$, where f is some linear function, called nth order autoregressive (AR) model

如果 o 是 continue value

AR model Autoregressive model

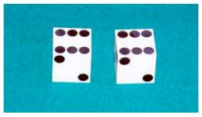
predict from past

O 服从正态分布，均值是前面 depend o 的 linear function

linear function 是 learn 出来的，就是 linear regression，就是现行模型的 determinant 部分，然后再加上 noise，就得到我们的真正的模型

Hidden Markov Models: Example

An experience in a casino



Game:

1. You bet $1
2. You roll (always with a fair die)
3. Casino player rolls (sometimes with fair die, sometimes with loaded die)
4. Highest number wins $2

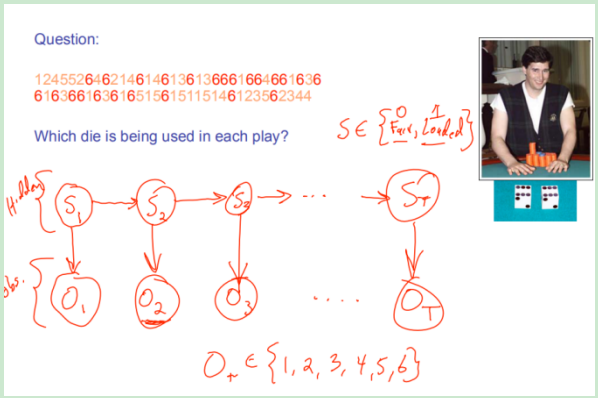
Here is **his** sequence of die rolls:

12455264621461461361366616646616366163661636165156151151461235623 44

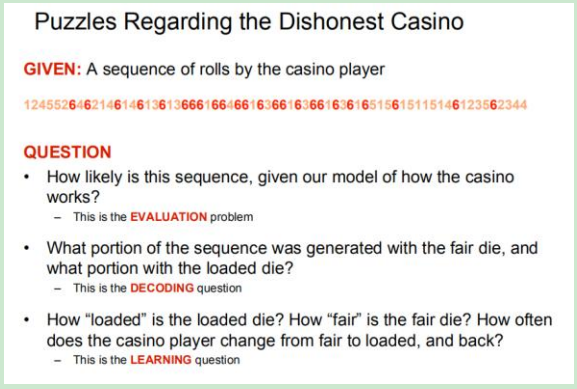
Which die is being used in each play?

有两种筛子，作为隐变量

由观测变量 推测 隐变量



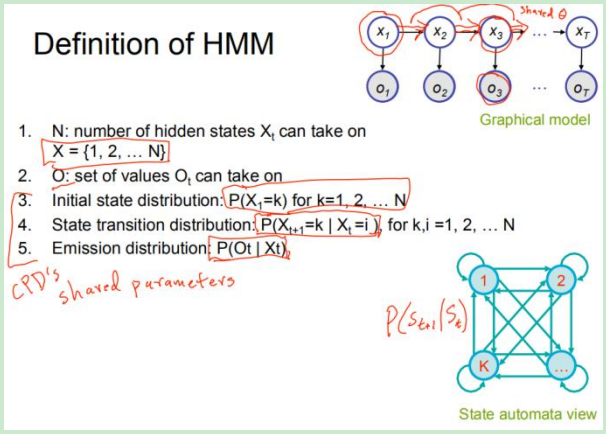
S set O set
前面用哪个筛子，会影响本次用哪个筛子的决定
d-separate
given S2 O2 和前面所有 S O 独立
given S2 S3 和前面所有 S O 独立



HMM 三个基本问题
levaluate 2decoding 3 learning transition,emission distribution

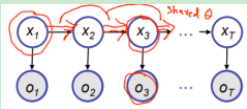
speech recognition
每种语言语音由最基本的元素 39 phony，是我们的 observed
hidden ： 不同的 word

HMM 其实就是个 bayse net!! 只是有些变量不可观测，可以用 EM 算法！

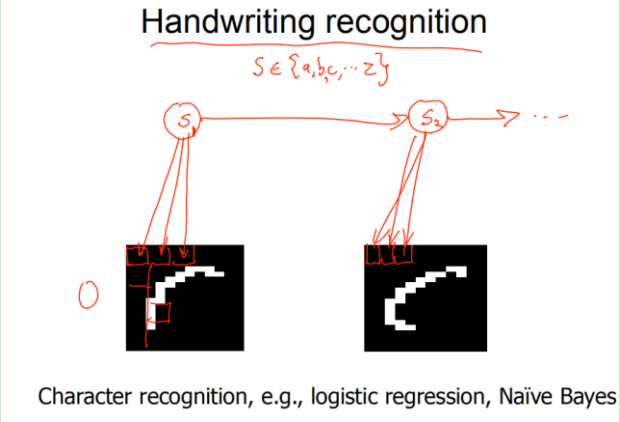


我们需要知道 hidden state 的数量
initial state prob 是第一个 state 的 category 的分布!!
3,4,5 bayse netd 的 CPD conditional prob distribution
X1 没有 parent，所以需要 P(X1) initial Prob
transition dis
emmission dis

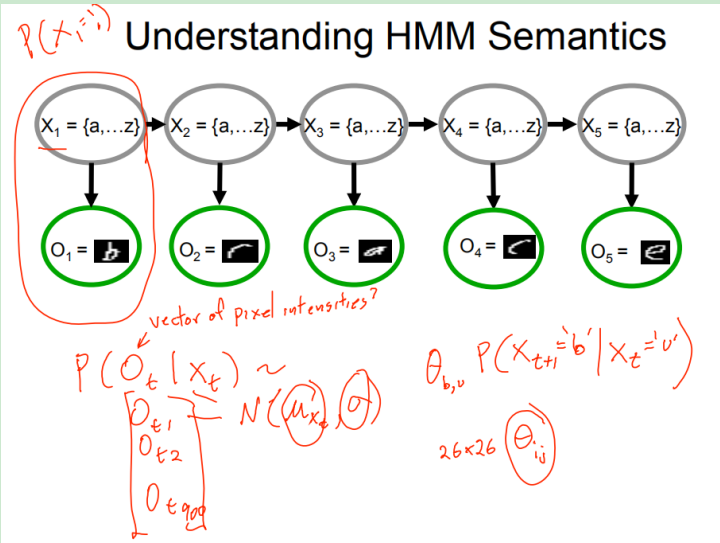
有一个隐含的假设：CPD share same 参数
transition dis share same 参数



每个时刻都 share 相同的 CPD 概率分布

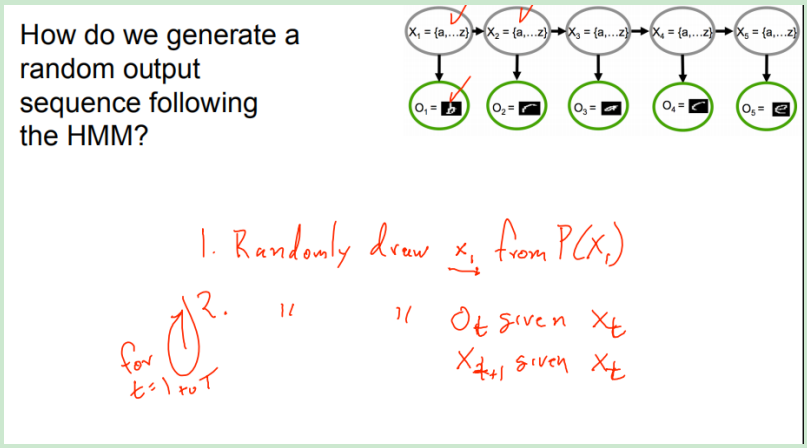


用 HMM 来识别图片
只看左边单个，是 GNB　S 是字符 set
HMM：前面的字符会某种程度上影响下一个字符的出现 sequence 字符
HMM 是 joint classifier，多个 GNB 联合在一起!!!
HMM 就是 sequence of NB classifiers，共享一套参数

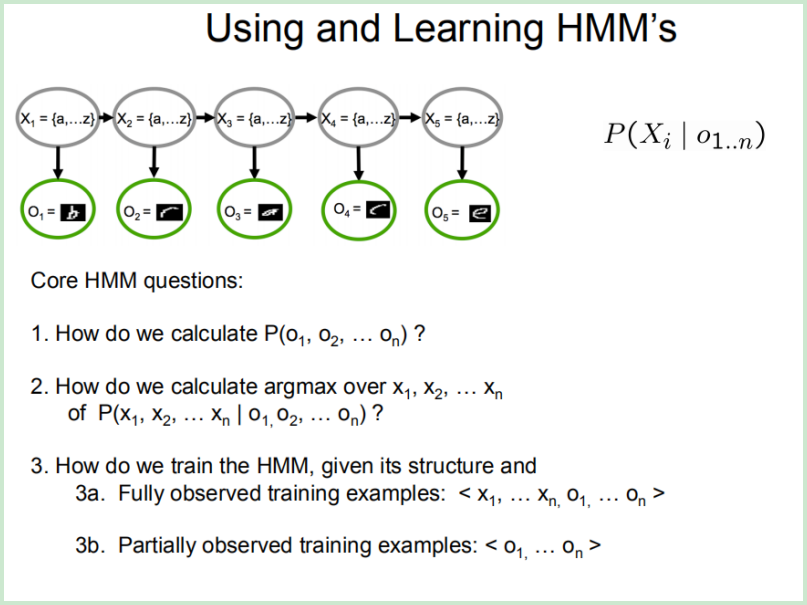


O 应该是高斯分布(高维的，同方差)，这个分布随时间不变
X 是 category 分布

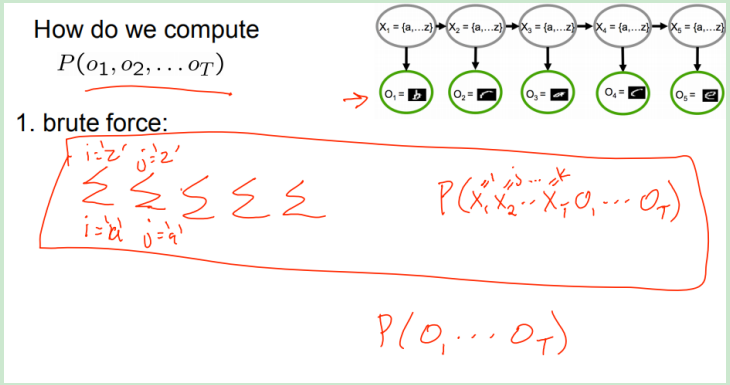
参数是 均值 方差 category 分布参数，initial state prob 是第一个 state 的 category 的分布!!



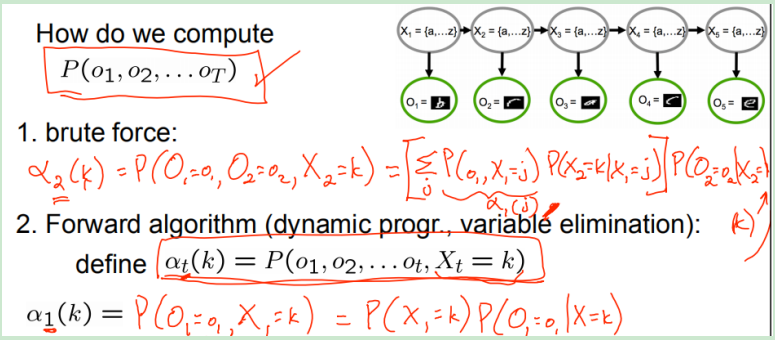
如何 sample？ X1->O1->X2->O2



inference ：



计算 P(o1o2..on)
求 P(x1,x2..xn,o1,o2..on)的**边缘分布**
计算量太大



用动态规划，递归

forward algorithm

$\alpha_t(k) = P(o_1, o_2 ..., o_t, X_t = k)$ 是观测从 1-t 的序列，并且最后一个 state 是 k(只关心最后一个 state)

上面是 alpha1 和 alpha2 两者直接是递归关系，如何定义？

$$p(O_1 = o_1, X_2 = k) = \sum_j p(O_1 = o_1, X_1 = j) p(X_2 = k \mid X_1 = j)$$

因为 $p(O_1 = o_1, X_1 = j) p(X_2 = k \mid X_1 = j) = p(O_1 = o_1, X_1 = j, X_2 = k)$

对 j 求和得到 marginal 概率 $p(O_1 = o_1, X_2 = k)$，消掉了 X1

其中 $p(O_1 = o_1, X_1 = j)$ 就是 alpha1(j)

$$p(O_1 = o_1, X_2 = k)\ p(O_2 = o_2 \mid X_2 = k) = p(O_1 = o_1, O_2 = o_2, X_2 = k)$$

$$\alpha_{t+1}(k) = \sum_{j=1}^{N} \alpha_t(j) P(X_{t+1} = k \mid X_t = j) P(O_{t+1} = o_{t+1} \mid X_{t+1} = k)$$

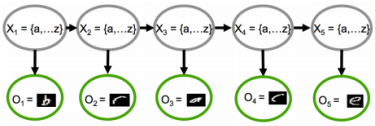
运行 T 次(线性时间)，就可以计算出 alpha t

$$\alpha_t(k) = P(o_1, o_2, \ldots o_t, X_t = k)$$

最后计算 p(o)只需 sum Xt

$$P(o_1, o_2, \ldots o_T) = \sum_{k=1}^{N} \alpha_T(k)$$

How do we compute
$P(X_t = k|o_1, o_2, \ldots o_T)$



2. Backward algorithm (dynamic progr., variable elimination):

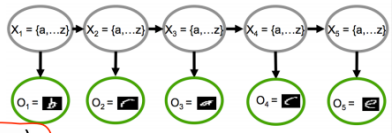
$\alpha_t(k) = P(o_1, o_2, \ldots o_t, X_t = k)$

define $\beta_t(k) = P(o_{t+1}, o_{t+2}, \ldots o_T|X_t = k)$

$$P(X_t = k|o_1, o_2, \ldots o_T) = \frac{P(X_t = k, o_1, o_2, \ldots o_T)}{P(o_1, o_2, \ldots o_T)} = \frac{\alpha_t(k)\beta_t(k)}{\sum_k \alpha_T(k)}$$

给定所有 obs，推测出**某个时刻的 latent 值**
Backward algorithm
define beta 是 future obs+当前的 state
分子，分母我们都能计算

How do we compute
$\arg \max_{x_1,\ldots x_T} P(x_1, \ldots x_T|o_1, o_2, \ldots o_T)$



Viterbi algorithm, based on recursive computation of

$\delta_t(k) = \max_{x_1,\ldots x_{t-1}} P(x_1, \ldots x_{t-1} X_t = k, o_1, o_2, \ldots o_t)$

给定观测序列，推测全部 latent 序列
define 一个 delta

learning:

Learning HMMs from fully observable data: easy

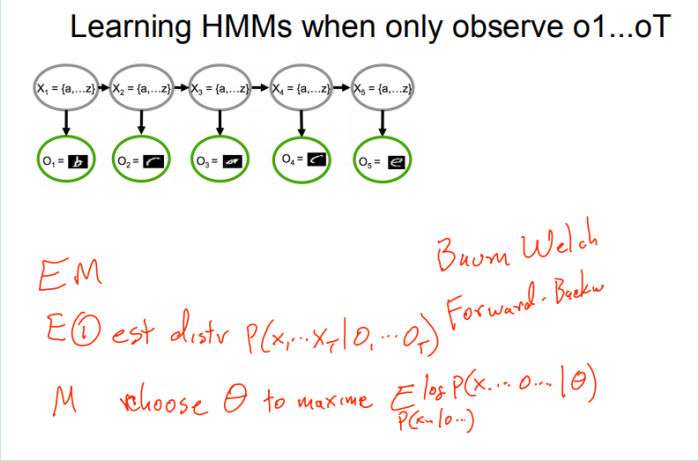
Learn 3 distributions:

$P(X_1)$

$P(O_i \mid X_i)$

$P(X_i \mid X_{i-1})$

如果知道 latent variable 就简单了



latent 不能观测，用 EM

E 计算因变量的概率分布

M 计算 full data likelihood 的期望

其他 bayse net structure

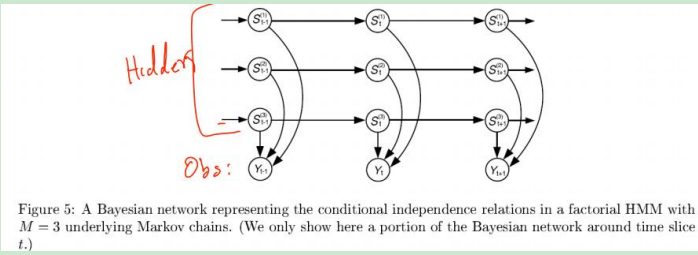


Figure 5: A Bayesian network representing the conditional independence relations in a factorial HMM with $M = 3$ underlying Markov chains. (We only show here a portion of the Bayesian network around time slice t.)

factorial HMM 多个 hidden state

股票价格受多个 state 影响

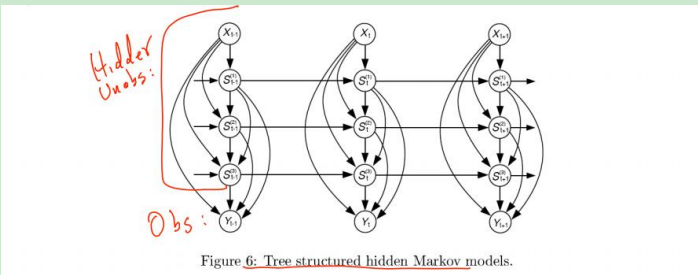


Figure 6: Tree structured hidden Markov models.

X 是连续变量的 latent

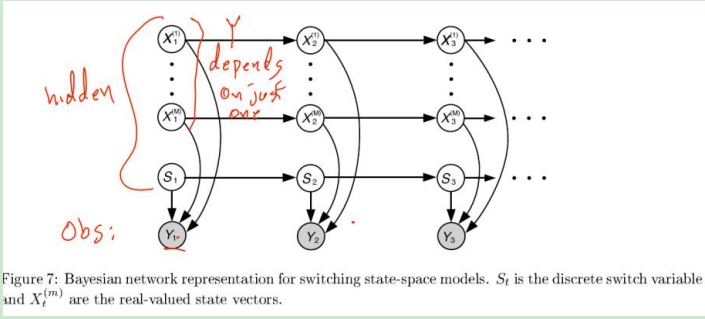


Figure 7: Bayesian network representation for switching state-space models. S_t is the discrete switch variable and $X_t^{(m)}$ are the real-valued state vectors.

Machine Learning 10-701

Tom M. Mitchell
Machine Learning Department
Carnegie Mellon University

March 24, 2011

Today:
- Non-linear regression
- Artificial neural networks
- Backpropagation
- Cognitive modeling
- Deep belief networks

Reading:
- Mitchell: Chapter 4
- Bishop: Chapter 5

NN 是 learn representation

non-linear regression

Artificial Neural Networks to learn f: X → Y

- f might be non-linear function
- X (vector of) continuous and/or discrete vars
- Y (vector of) continuous and/or discrete vars

- Represent f by *network* of logistic units
- Each unit is a logistic function

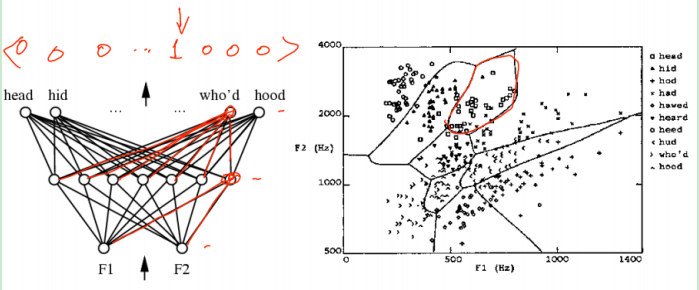
$$unit\ output = \frac{1}{1 + exp(w_0 + \sum_i w_i x_i)}$$

- MLE: train weights of all units to minimize sum of squared errors of predicted network outputs
- MAP: train to minimize sum of squared errors plus weight magnitudes

network of LR

MLE 和 MAP 的结果

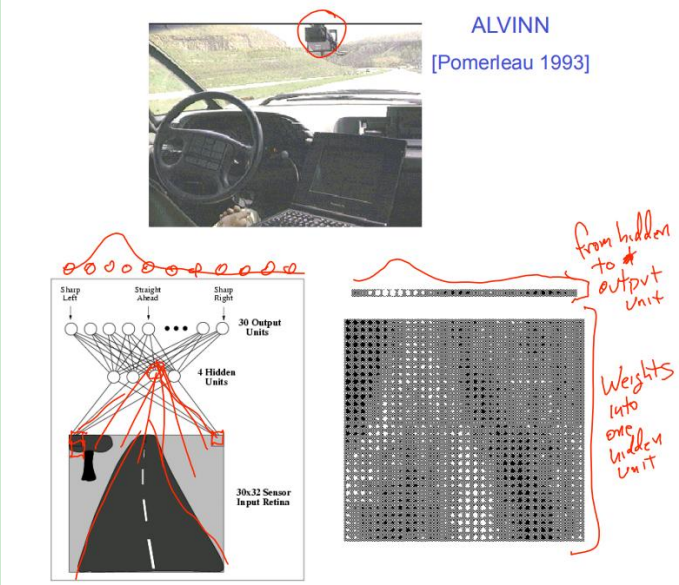
Multilayer Networks of Sigmoid Units



本质其实就是 LR unit 的 stack 在一起

最后一层其实就是每个类别的 LR 的结果，一个类别一个 LR

non-linear decision boundary



auto-drive

output ;direction to steer

我们想知道最一个 hidden layer 的节点的 weight(连的是 image)

因此看到每个像素对应的 weight，白色正　黑负 灰 0

hidden unit 相当于是一个**模板**，模板的地方如果图片白，像素值高，unit 越容易被激活，也就是**符合这样模板的 unit 的图片**才会被激活!!!

Connectionist Models

Consider humans:

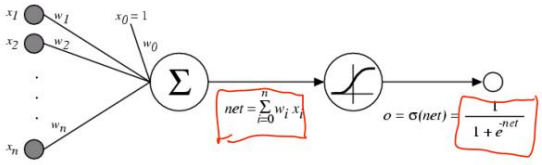
- Neuron switching time ~ .001 second
- Number of neurons ~ 10^{10}
- Connections per neuron ~ 10^{4-5}
- Scene recognition time ~ .1 second
- 100 inference steps doesn't seem like enough

→ much parallel computation

Properties of artificial neural nets (ANN's):

- Many neuron-like threshold switching units
- Many weighted interconnections among units
- Highly parallel, distributed process

Sigmoid Unit



$\sigma(x)$ is the sigmoid function

$$\frac{1}{1 + e^{-x}}$$

Nice property: $\frac{d\sigma(x)}{dx} = \sigma(x)(1 - \sigma(x))$

We can derive gradient decent rules to train

- One sigmoid unit
- *Multilayer networks* of sigmoid units → Backpropagation

M(C)LE Training for Neural Networks

- Consider regression problem f:X→Y , for scalar Y

  y = f(x) + ε ← assume noise N(0,σ_ε), iid

  deterministic

- Let's maximize the conditional data likelihood

$$W \leftarrow \arg\max_W \ln \prod_l P(Y^l | X^l, W)$$

$$W \leftarrow \arg\min_W \sum_l (y^l - \hat{f}(x^l))^2$$

Learned neural network

MAP Training for Neural Networks

- Consider regression problem f:X→Y , for scalar Y

  y = f(x) + ε ← noise N(0,σ_ε)

  deterministic

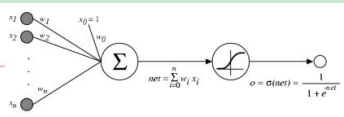
Gaussian P(W) = N(0,σI)

$$W \leftarrow \arg\max_W \ln P(W) \prod_l P(Y^l | X^l, W)$$

$$W \leftarrow \arg\min_W \left[c \sum_i w_i^2 \right] + \left[\sum_l (y^l - \hat{f}(x^l))^2 \right]$$

ln P(W) ↔ c $\sum_i w_i^2$

Error Gradient for a Sigmoid Unit



$$\begin{aligned}
\frac{\partial E}{\partial w_i} &= \frac{\partial}{\partial w_i} \frac{1}{2} \sum_{d \in D} (t_d - o_d)^2 \\
&= \frac{1}{2} \sum_d \frac{\partial}{\partial w_i} (t_d - o_d)^2 \\
&= \frac{1}{2} \sum_d 2(t_d - o_d) \frac{\partial}{\partial w_i} (t_d - o_d) \\
&= \sum_d (t_d - o_d) \left(-\frac{\partial o_d}{\partial w_i} \right) \\
&= -\sum_d (t_d - o_d) \frac{\partial o_d}{\partial net_d} \frac{\partial net_d}{\partial w_i}
\end{aligned}$$

x_d = input

t_d = target output

o_d = observed unit output

w_i = weight i

But we know:

$$\frac{\partial o_d}{\partial net_d} = \frac{\partial \sigma(net_d)}{\partial net_d} = o_d(1 - o_d)$$

$$\frac{\partial net_d}{\partial w_i} = \frac{\partial (\vec{w} \cdot \vec{x}_d)}{\partial w_i} = x_{i,d}$$

So:

$$\frac{\partial E}{\partial w_i} = -\sum_{d \in D} (t_d - o_d) o_d (1 - o_d) x_{i,d}$$