贝叶斯：

P(h|D) = $P(D|h)P(h)/P(D)$

P(h) 假设 h 的先验概率

P(D) 训练数据 D 的先验概率

P(D|h)给定假设 h，D 的概率(似然)

P(h|D)给定 D，h 的概率，后验概率

我们希望在给定假设训练数据 D 上得到最可能的假设

最大后验假设 hmap

$$\text{hmap} = \arg\max P(h|D) = \frac{argmaxP(D|h)P(h)}{P(D)} = argmaxP(D|h)p(h)$$

最大似然假设： hml = argmaxP(D|h)

例子：

P(cancer) = 0.008；P(¬cancer) = 0.992

P(+|cancer) = 0.98; P(−|cancer) = 0.02

P(−|¬cancer) = 0.97; P(+|¬cancer) = 0.03

给定一个病人检验结果为“+”，该病人得癌症的概率

hmap = argmax(P(cancer|+), P(¬cancer|+))

P(+|cancer)P(cancer) = 0.00784

P(+|¬cancer)P(¬cancer) = 0.02776

$$P(\text{cancer}|+) = \frac{0.00784}{0.00784 + 0.02976} = 0.21$$

$$P(\neg\text{cancer}|+) = \frac{0.02976}{0.00784 + 0.02976} = 0.79$$

$$v_{NB} \approx v_{MAP} = \arg\max_{v_j \in \{yes, no\}} P(v_j \mid sunny, cool, high, strong)$$

概率 = 观察到事件发生的次数/总的观察次数

贝叶斯分类器：

$$v_{NB} = \arg\max_{v_j \in V} \underbrace{P(v_j)}_{\text{先验概率}} \prod_i \underbrace{P(a_i \mid v_j)}_{\text{条件概率}}$$

朴素贝叶斯算法：

Naïve_Bayes_Learn (*training examples* **t**)
  For each target value v_j
    estimate $P(v_j)$ using **t** //How?
    For each attribute value a_i of each attribute *a*
      estimate $P(a_i|v_j)$ using **t** //How?

Classify_New_Instance (*x*) //*x*=(x_1,x_2,...)

$$v_{NB}(x) = \arg\max_{v_j \in V} P(v_j) \prod_{x_i \in x} P(x_i \mid v_j)$$

后验概率不一定要正确，只要决策正确就行

评价估计概率：如果有一个条件的概率为 0，会导致低估整体概率，因为最后的概率是条件概率的乘积(独立性假设问题)

解决方法：m-概率估计

$$(nc + mp)/(n + m)$$

p 和 m 是用户选择的参数，p 是估计的先验概率，m 是等价的样本数量

用贝叶斯进行文本分类：

Learn_naive_Bayes_Text (*Examples*, *V*)

vocabulary = all distinct words
For each target value v_j in V do //calculate the required probabilities
  docs_j = subset of *Examples* for which the target value is v_j
  $P(v_j)$ = |docs_j|/|Examples|
  Text_j = a document created by concatenating all members of docs_j
  n = total number of words in Text_j, i.e. |Text_j|
  for each word w_k in Vocabulary
    n_k = number of times word w_k occurs in Text_j
    $P(w_k|v_j) = (n_k+1)/(n+|\text{Vocabulary}|)$ //Why?

Classify_naive_Bayes_text (Doc)
  positions = all word positions in Doc that contain tokens found in Vocabulary
  Return v_{NB} where

$$v_{NB}(x) = \arg\max_{v_j \in V} P(v_j) \prod_{i \in positions} P(word_i \mid v_j)$$

离散 KNN

Training Algorithm

For each training example $\langle x, f(x)\rangle$
  add the example to training list.

Given a new instance x_q, and parameter k,
Search through the training list to find k nearest neighbors $x_1,\ldots,x_k$
Return

$$\hat{f}(x_q) \leftarrow \arg\max_{v \in F} \sum_{i=1}^{k} \delta(v, f(x_i)) \qquad \delta(a,b) = \begin{cases} 1 & a = b \\ 0 & a \neq b \end{cases}$$

Classification Algorithm

连续值 KNN

Given a new instance x_q, and parameter k,
Search through the training list to find k nearest neighbors $x_1,\ldots,x_k$
Return

$$\hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^{k} f(x_i)}{k}$$

离散距离权重 DWNN

$\text{f(xq)} = \text{argmax}\sum_{i=1}^{k} wi\delta(v, f(xi)); w(i) = 1/d(xq, xi)^2$

如果 xq=xi,选择 xi 类标数量最多的分类

实数距离权重 DWNN

$\text{f(xq)} = \sum_{i=1}^{k} wif(xi) / \sum_{i=1}^{k} wi$

关于 KNN

1. 偏置：新实例类别跟在欧式空间下最接近的实例相似
2. 通过权重消除无关属性
3. 快速 KNN：kd Tree、R-tree、Grid File

Locally weighted regression

基于查询点的邻居建立线性回归进行预测

Locally：查询点的预测值基于其邻居

Weighted：每个邻居的贡献值根据其距离查询点的距离决定

Regression：逼近实数值预测

局部回归：$\text{E(xq)} = 1/2\sum_{x \in KNN(xq)}(f(x) - \text{"}f(x)\text{"})^2$

加权回归：$\text{E(xq)} = 1/2\sum_{x \in D}(f(x) - \text{f(x)})^2 K(d(xq,x))$

局部加权回归：$\text{E(xq)} = 1/2\sum_{x \in KNN(xq)}(f(x) - \text{f(x)})^2 K(d(xq,x))$

权值更新

$$\Delta\text{wi} = \text{n}\sum_{x \in D}(f(x) - \text{f(x)})\, K(d(xq,x))ai(x)$$

ai(x)是样例 x 的第 i 个属性，K()一个递减的核函数

根据核函数选择的不同有很多局部线性回归的变种

kNN 总结：

优点：可以使用简单局部近似进行复杂函数建模

样例所含信息没有进行压缩或丢失

缺点：无关特征和维度灾难，计算复杂度高、很难选择好的距离度量标准

遗传算法

四种运算：

种群：If A1 AND Not A2 ->C2 : 100; If Not A1 AND Not A2 –>C1 : 001

选择：{1000(better),1100}->{1000}

交叉：Crossover(交叉) 1010+0011->1011, 0010

变异：Mutate(变异) 1000->1100

遗传算法：

GA(Fitness, Fitness_threshold, p, r, m)

Fitness: *A function that assigns an evaluation score, given a hypothesis.*
Fitness_threshold: *A threshold specifying the termination criterion.*
p: *The number of hypotheses to be included in the population.*
r: *The fraction of the population to be replaced by Crossover at each step*
m: *The mutation rate.*

- *Initialize population: P* ← Generate *p* hypotheses at random
- *Evaluate:* For each *h* in *P*, compute *Fitness(h)*
- While [$\max_h$ *Fitness(h)*] < *Fitness_threshold* do

Create a new generation P_S:

1. *Select:* Probabilistically select (1-r)*p* members of *P* to add to P_S. The probability $\Pr(h_i)$ of selecting hypothesis h_i from *P* is given by
$$\Pr(h_i) = \frac{Fitness(h_i)}{\sum_{j=1}^{p} Fitness(h_j)}$$
2. *Crossover:* Probabilistically select *rp*/2 pairs of hypotheses from *P*, according to $\Pr(h_i)$ given above. For each pair, $\langle h_1, h_2\rangle$, produce two offspring by applying the Crossover operator. Add all offspring to P_S.
3. *Mutate:* Choose *m* percent of the member of P_S with uniform probability. For each, invert one randomly selected bit in its representations.
4. *Update:* $P \leftarrow P_S$.
5. *Evaluate:* for each *h* in *P*, compute *Fitness(h)*.

•Return the hypothesis from *P* that has the highest fitness.

SVM(支持向量机)：结构风险最小化

误差函数（经验风险）：

$$R_{emp}(\alpha) = \frac{1}{2N}|yi - f(xi, \alpha)|$$

预期风险：$R(\alpha) = \frac{1}{2}\int |y - f(x,\alpha)|P(x,y)dxdy$

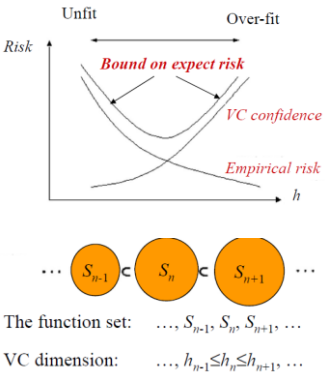
预期风险约束：$R(\alpha) \le R_{emp}(\alpha) + F(h)$

$R_{emp}(\alpha)$: *经验误差*

$F(h)$: *VC置信范围*

假设空间 H：H 能够划分的最大训练数据数量

结构风险最小化：同时减小经验风险和 VC 置信范围

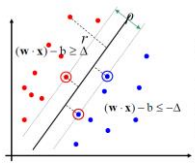


Then we can formulate the problem of Constrained Quadratic Programming (二次规划):

Find **w** and *b* such that
Minimize ½ $\mathbf{w}^T\mathbf{w}$
Subject to $y_i(w \cdot x_i + b) - 1 \ge 0,\ i = 1, 2, \ldots, N$

- ***Margin*** ρ of the separator is the width of separation between classes.
- **Maximizing the margin** is good according to intuition and theory.

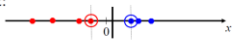
$$h \le \min\left\{\left\lceil \frac{D^2}{\rho^2} \right\rceil, m_0\right\} + 1$$



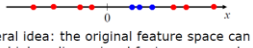
ρ is the margin,
D is the diameter of the smallest sphere that can enclose all of the training examples,
m_0 *is the dimensionality*

This implies: regardless of dimensionality m_0 we can **minimize the VC dimension by maximizing the margin ρ**.

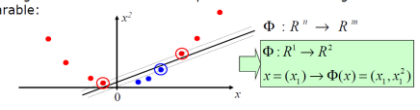
- Datasets that are linearly separable with some noise work out great:



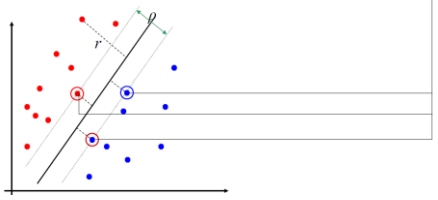
- But what are we going to do if the dataset is just too hard to separate?



- General idea: the original feature space can always be mapped to some higher-dimensional feature space where the training set is separable:



- Examples closest to the hyperplane are ***support vectors***.
- The classifier is termed ***support vector machine*** **(SVM)**.



- SVMs are based on Structural Risk Minimization, and were originally proposed by Boser, Guyon and Vapnik in 1992 and gained increasing popularity in late 1990s.
- SVMs are currently among the best performers for a number of classification tasks, e.g., text, genomic data (基因组数据).
- Minimize Structural Risk→ Minimize VC confidence → Maximize Margin → Quadratic Programming approach
- Mathematically, Learning is a problem of Quadratic Programming.
  Most popular algorithms are SMO and SVM^{light}.
- Tuning SVMs remains a black art: selecting a specific kernel and parameters is usually done in a try-and-see manner.

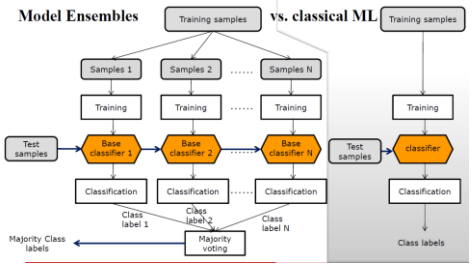
- SVMs are based on Structural Risk Minimization
- Key ideas
  - 1) Minimize **Structural Risk**
  - 2) Minimize **VC confidence**
  - 3) Maximize **Margin**
    Quadratic Programming approach
  - **Map to higher-dimensional space** where the training set is separable

算法独立的 ML 方法

模型集合：

Bootstrap、Re-sampling for classifier design

Method: Bagging、Boosting



Bagging (Bootstrap aggregation)

1. 创建数据集的多个版本(以替换的方式随机从训练集 D 中选取 d<|D|个样本)
2. 每个数据集用来训练不同的基础分类器
3. 最终分类结果：多数表决的方式

Algorithm Bagging($Instance\_set$, L, N, d)
  For $k \leftarrow 1$ to N
    $S_k \leftarrow$ random sample of size d drawn from $Instance\_set$
    $M_k \leftarrow$ the model induced by L from S_k
  For each new query instance q
    $Class(q) = argmax_{v \in V} \Sigma_{i=1}^{k} \delta(v, M_i(q))$

where V is the finite set of target class values, and $\delta(a,b) = 1$ if $a = b$ and $\delta(a,b) = 0$ otherwise.

- L: the chosen learning algorithm, e.g., decision tree or ANN
- N: the number of bags
- d: the size of each bag, i.e., number of samples/bag

Remark: Bagging 提高不稳定的分类器

1. 减少单个不稳定模型分错类的可能
2. 降低实例顺序对学习算法的影响

Boosting

Boosting 提高弱分类器的准确度

学习一系列的分类器，每个分类器更关注被上一个分类器分错的样例

训练过程

Initialize with uniform weights

Loop

  Apply learner to weighted examples (How?)

  Increase weights of misclassified examples

Combine models by majority voting

Boosting 和 Bagging 的区别：

Boosting 是迭代式的，Bagging 每个分类器是独立的，Boosting 后续的分类器依赖于前一个分类器，权重更关注难以区分的样例

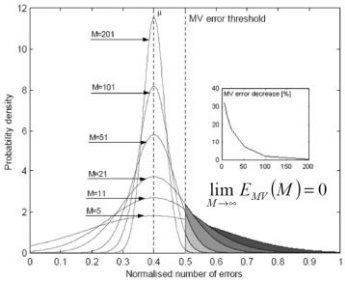
Remarks on Majority Voting

- Base classifier (Mutual-independent):
  - Number: M
  - Probability of error: e
- Probability that i base classifiers are wrong:

$$C_M^i e^i (1-e)^{M-i}$$

- Probability that the result of MV is wrong:

$$E_{MV}(M) = \sum_{i=\lceil M/2 \rceil}^{M} C_M^i e^i (1-e)^{M-i}$$



Ensemble Learning:

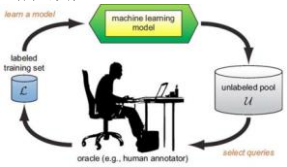
basic Idea： 与其学习一个模型，不如选择多个然后结合使用

Active Learning(主动学习)

动机：好的数据相比简单拥有更多的数据更有效

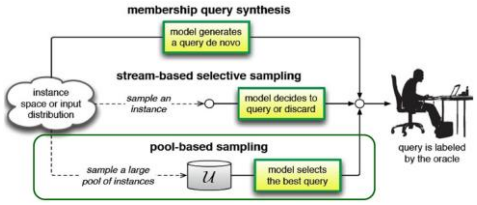
主动学习：根据已有的知识，选择收集更好的训练数据来减少整体的代价

主动学习框架：



1. 从一个小的训练样本开始
2. 从未标注的数据集中选择一些实例进行查询
3. 专业人员对查询实例进行人工标注
4. 使用新的数据集训练分类器

主动学习替代框架：



主动学习策略

启发式方法降低风险：

1. 给定模型和参数选择最不确定的实例
2. 选择信息量最多的实例优化期望增益

不确定性采样：

查询当前的分类器最不确定的实例

方法：

预测类标最没有信心的

$$x^* = \arg\min_x P(\hat{y}|x,\theta) = \arg\min_x \max_y P(y|x,\theta)$$

熵：

$$x^* = \arg\max -\sum_y P(y|x,\theta)\log P(y|x,\theta)$$

欧式距离：

在 SVM 中最靠近 margin 的

不确定采样算法

1: $\mathcal{U}$ = a pool of unlabeled instances $\{x^{(u)}\}_{u=1}^{U}$
2: $\mathcal{L}$ = set of initial labeled instances $\{\langle x, y\rangle^{(l)}\}_{l=1}^{L}$
3: **for** $t = 1, 2, \ldots$ **do**
4: $\theta = \textbf{train}(\mathcal{L})$
5: select $x^* \in \mathcal{U}$, the most uncertain instance according to model θ
6: query the oracle to obtain label y^*
7: add $\langle x^*, y^*\rangle$ to $\mathcal{L}$
8: remove x^* from $\mathcal{U}$
9: **end for**

Query by Committee

1. Build a set of classifiers C
2. Measures of disagreement
   - Entropy of predicted responses

$$x^*_{VE} = \arg\max_x -\sum_y \frac{V_C(y,x)}{|C|}\log\frac{V_C(y,x)}{|C|}$$

$V_C(y,x)$: the number of votes of label y

3. Query label of x^*_{VE}

Transfer learning

使用一些已标注的数据，抽取从相关领域中学习到的知识来帮助在特定领域中进行学习

无监督学习，聚类

无监督学习：从数据中捕获一些固有的组织形式

聚类：相近的对象在同一个簇中，不相似的在不同的类中

典型应用：

1. 作为一个独立的工具来观察数据的分布
2. 作为很多算法的预处理步骤

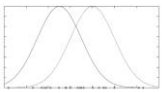
贪心聚类方法：

- Producing clusters with at most a given diameter d

```
For each point p
    Finds the cluster center q such that D(p, q) is minimum
  - If D(p, q) < d then
        p is added to the cluster whose center is q
    Else
        A new cluster with center p is formed
```

- Once the algorithm has examined all the points cluster process completes.

EM(Expectation MaxImization,期望最大化算法)



Problem: Construct a partition of n instances into a set of k clusters

- Each instance x generated by
  1) Choosing one of the k normal distributions
  2) Generating an instance according to this distribution
- Problem → estimating means of a mixture of normal distributions

Given
- Instances generated by mixture of k distributions (same variance)
  - $\langle x_i$(observable), $z_{i1}, z_{i2}, \ldots, z_{ik}\rangle$
  - z_{ij}: 1 if x_i generated by j-th distribution.
- **Don't know**
  - **means of the k distributions** $(\mu_1,\ldots,\mu_k)$ → Hypothesis
  - **which instance x_i was generated by which distribution** (z_{ij})→ hidden variable

Problem → To determine
- $h=\langle\mu_1,\ldots,\mu_k\rangle$ (and z_{ij})
- by maximizing $\mathbf{P(D|h)}$?

EM pick random initial $\langle\mu_1,\ldots,\mu_k\rangle$, then iterate
- E step:
  - Calculate the expected value $E(z_{ij})$ of each hidden variable z_{ij}
  - Assuming the current hypothesis $h=\langle\mu_1,\ldots,\mu_k\rangle$ holds
- M step:
  - Calculate a new maximum likelihood hypothesis $h'=\langle\mu_1',\ldots,\mu_k'\rangle$
  - Assuming the value taken on by each hidden variable z_{ij} is its expected value $E(z_{ij})$ calculated above
  - Replace h by h'

K-Means

算法：

1. Partition objects into k nonempty subsets randomly
2. Compute cluster centers (mean point) as the centroids of the clusters of the current partition.
3. (Re)Assign each object to the cluster with the nearest cluster center.
4. Go back to Step 2, stop when no more new assignment, i.e. converged (收敛).

K-Means 评价

- K-means:
  - Relatively efficient: $O(tkn)$, where n is # objects, k is # clusters, and t is # iterations.
  - Many improvements available
- Might terminate at a *local optimum*.
- Applicable only when *mean* is defined, then what about categorical data?
  - Handling categorical data: *k-modes*
  - A mixture of categorical and numerical data: *k-prototype* method
- Need to specify k, the *number* of clusters
  - *Recursive k means*
- Unable to handle *noisy data* and *outliers*
- Not suitable to discover clusters with *non-convex shapes*

半监督学习

- **Unsupervised Learning**
  - Problem
    - Clustering
    - PCA, ICA, MDS, Topic model...
  - Cluster analysis
    - Given no "right answers" (unsupervised)
    - To groups objects based on their similarity
    - For capturing inherent organization in the data
    - Along with supervised learning, cluster is one of the most practical learning methods and has wide applications
  - EM algorithm
  - Clustering algorithms can be categorized into
    - Partitioning methods, e.g. K means
    - Hierarchical methods, e.g. HAC, HDC
    - Density-based methods, e.g. DBSCAN

概率图模型

统计学+图理论+计算机科学

使用图的方式展现随机变量之间的概率关系

为什么选用图模型：

概率论：条理化、高效地学习接口

图论：对人来说直观清晰，高效地整合展现人类的知识

模块化：组合简单的模块构建复杂的系统

PGM 类型：

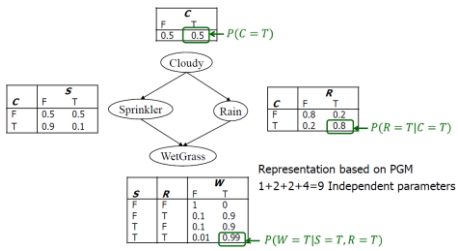
有向无环图(DAG)

贝叶斯网络，有向边表示因果依赖

模型：用图来表示随机变量间的概率关系

节点：表示随机变量(二进制事件，离散变量、连续变量)

边：表示条件依赖或者关系



无向图(UGM)s

马尔科夫随机场、因子图、玻尔兹曼机、对称的概率依赖关系

链式图(Chain graph)

推断：

Inference: Computation of the conditional probability distribution of one set of nodes, given a model and another set of nodes.

Bottom-up: "diagnosis" from effects to reasons
- Observation (e.g. wet grass)
- The probabilities of the reasons (rain, sprinkler) can be calculated

Top-down: Predict the effects
- Knowledge (e.g. "it is cloudy") influences the probability for "wet grass"

Observe: wet grass ($W=T$)
- Two possible causes: rain or sprinkler
  Which one is more likely?
- To compute the posterior probabilities of the reasons (rain $R=T$, sprinkler $S=T$):
  $P(S=T|W=T)$ $P(R=T|W=T)$
- Using Bayes' rule to calculate:

$$P(S=T|W=T)=\frac{P(S=T,W=T)}{P(W=T)}=\frac{\sum_{c,r}P(C=c,R=r,S=T,W=T)}{\sum_{c,r,s}P(C=c,R=r,S=s,W=T)}$$
$$P(R=T|W=T)=\frac{P(R=T,W=T)}{P(W=T)}=\frac{\sum_{c,s}P(C=c,R=T,S=s,W=T)}{\sum_{c,r,s}P(C=c,R=r,S=s,W=T)}$$

参数学习：find maximum likelihood estimates of parameters of each conditional probability distribution

结构学习：

find correct connectivity between existing nodes

Select a 'good' model from all possible models and use it as if it were the correct model

Structure	Observation	Method
Known	Full	Maximum Likelihood (ML) estimation
Known	Partial	Expectation Maximization algorithm (EM)
Unknown	Full	Model selection
Unknown	Partial	EM + model selection

课后习题：

1 某人进行两次 cancer 的检验均为阳性，其是否得癌症的后验概率为：

$$P(\oplus\oplus\,|\text{cancer})P(\text{cancer})$$
$$= P(\oplus\,|\text{cancer})P(\oplus\,|\text{cancer})P(\text{cancer})$$
$$=0.076832$$
$$P(\oplus\oplus\,|\neg\text{cancer})P(\neg\text{cancer})$$
$$= P(\oplus\,|\neg\text{cancer})P(\oplus\,|\neg\text{cancer})P(\neg\text{cancer})$$
$$= 0.000828$$

$$P(\text{cancer}|\oplus\oplus)=\frac{0.076832}{0.076832+0.000828}=0.98$$

$$P(\neg\text{cancer}|\oplus\oplus)=\frac{0.000828}{0.076832+0.000828}$$
$$=0.0106$$

Bootstrap:不增加任何额外的资源就能提升

组合学习：不增加额外的数据，用多个分类器提高精度，重采样的方式训练多个分类器

Bagging 特点：提升不稳定分类器的精度，对数据输入顺序不敏感

熵：Entropy(S) = -p₊ log2p₊ - p₋ log2p₋=$\sum -p_i log p_i$

$\text{Gain(S, A)} = \text{Entropy(S)} - \sum_{v\in values(A)} (|\text{Sv}|/|\text{S}|)\,\text{Entropy(Sv)}$

ID3 算法：假设空间完备、输出单个假设空间、在选定属性上不能回溯(可能进入局部最小值)、基于统计进行搜索选择(对噪声具有鲁棒性)

ID3 偏置：倾向于更短的树、高信息增益的属性接近根

神经网络：

感知器(线性阈值单元)：$\sum_{i=0}^{n} w_i x_i$>0 输出 1，否则输出-1

线性单元：o=：$\sum_{i=0}^{n} w_i x_i$

Sigmoid 单元：$o=\sigma(net)=1/(1+e^{-z})$

Sigmoid 单元偏导:

线性单元梯度下降： $E(w)=1/2\sum_{d\in D}(t_d-o_d)^2$

$$\Delta w_i=-\frac{\eta\partial E}{\partial w_i}=\eta\sum_d[(t_d-o_d)x_i] \qquad w_i=w_i+\Delta w_i$$

反向传播算法评论：

1. 很容易扩展到任意有向图
2. 算法会找到一个局部最小值
3. 可能会遇到平坦区域

实践经验：

1. 用不同的初始权重训练多个网络
2. 增加冲量$\Delta w_{i,j}(n)=\eta\delta_j x_{i,j}+\alpha\Delta w_{i,j}(n-1)$
3. 增加梯度下降速度

ANN 的能力：

1. 一个隐藏层可以表示所有布尔函数
2. 一个隐藏层可以以任意小的误差估计有限连续函数(Sigmoid + Linear)
3. 两个隐藏层可以估计任意准确度的函数(Sigmoid+Sigmoid+Linear)

SVM

支持向量机方法是建立在统计学习理论的 VC 维理论和结构风险最小原理基础上

结构化风险 = 经验风险 + 置信风险

经验风险 = 分类器在给定样本上的误差

置信风险 = 分类器在未知文本上分类的结果的误差

置信风险因素：

样本数量，给定的样本数量越大，学习结果越有可能正确，此时置信风险越小；

分类函数的 VC 维，显然 VC 维越大，推广能力越差，置信风险会变大。

提高样本数量，降低 VC 维，降低置信风险。

VC 维：模式识别中 VC 维的直观定义是：对一个指示函数集，如果存在 N 个样本能够被函数集中的函数按所有可能的 2^N 种形式分开，则称函数集能够把 N 个样本打散，函数集的 VC 维就是它能打散的最大样本数目 N ，若对任意数目的样本都有函数能将它们打散. 则函数集的 VC 维是无穷大。而 VC 维为该学习机能学习的可以由其分类函数正确给出的所有可能二值标识的最大训练样本数。

组合学习最关键的问题是基分类器的独立性问题：增加基分类器的独立性、提高基分类器性能

候选消除算法 输出和训练样例一致的所有假设
find-s 和 候选消除算法在训练数据包含噪声时性能较差

候选消除算法可以找出所有和训练样例一致的假设，假设空间中的这些假设称为变型空间

列表后消除算法：直接