软件分析

机器学习基础

熊英飞
北京大学

2017

机器学习概述

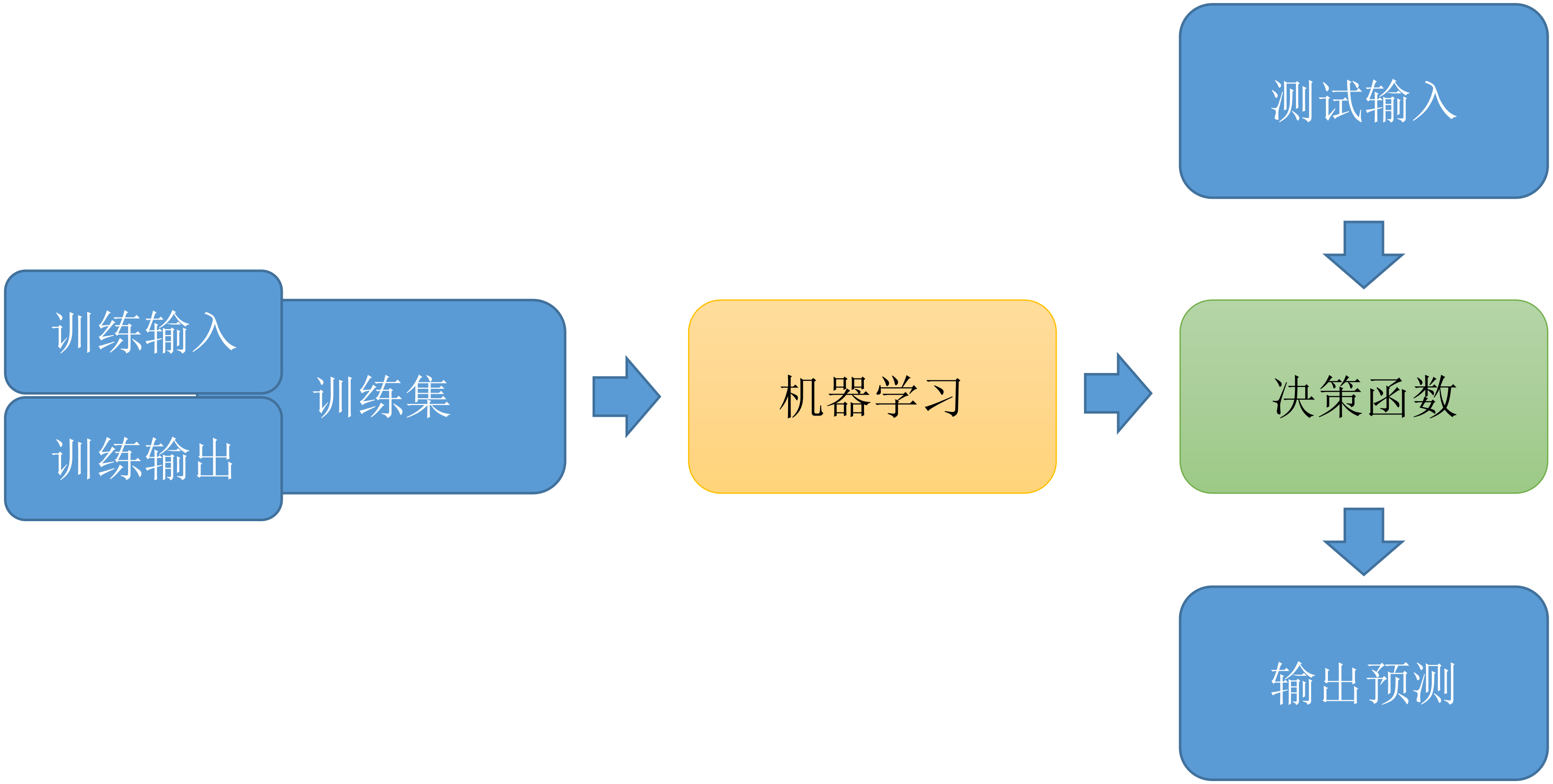
周志华西瓜问题

- 已知若干西瓜的外在属性和内部质量

色泽青绿程度(1-3)	根蒂硬挺程度(1-3)	敲声清脆程度(1-3)	好瓜
3	1	1	是
1	1	1	是
3	3	3	否
1	2	2	否

- 给定任意西瓜的外在属性，如
  - 色泽=2
  - 根蒂=3
  - 敲声=1
- 问这个西瓜是不是一个好瓜？

（有监督的）机器学习问题



从概率的角度看机器学习

- Y为代表输出取值的随机变量
- X为代表输入取值的随机变量
- 理想的决策函数是求一个条件概率分布
  - P(Y|X=输入)
  - 并且输出概率最高的Y取值
- 机器学习算法通过对训练集进行学习来逼近这个条件概率分布
  - 如果训练集足够大，直接采样即可
  - 实际中通常没有这么大的训练集

机器学习的三要素

- 模型
  - 决策函数的抽象形式，如：
    - $f_{a,b,c,d}$(色泽, 根蒂, 敲声)=a∗色泽+b∗根蒂+c∗敲声>d
  - 机器学习过程就是在决策函数的空间中选择一个具体的决策函数，常表现为选择一组参数的值
- 策略
  - 如何判断决策函数的好坏？
  - 如：在训练集上算对一个答案，优秀程度加一，否则不加
  - 机器学习问题转变成优化问题
- 算法
  - 求解该优化问题的具体算法
  - 如：随机产生a,b,c,d的若干组值，选择效果最好的一组

过拟合Overfitting

- 决策函数在训练集上表现良好，在测试集上表现不好
  - $f(3,1,1) = true$
  - $f(1,1,1) = true$
  - f(其他色泽，其他根蒂，其他敲声) $= false$
- 常见原因
  - 模型参数较多
  - 训练集数量较少
  - 模型与问题特征不符

决策函数的常见输入和输出

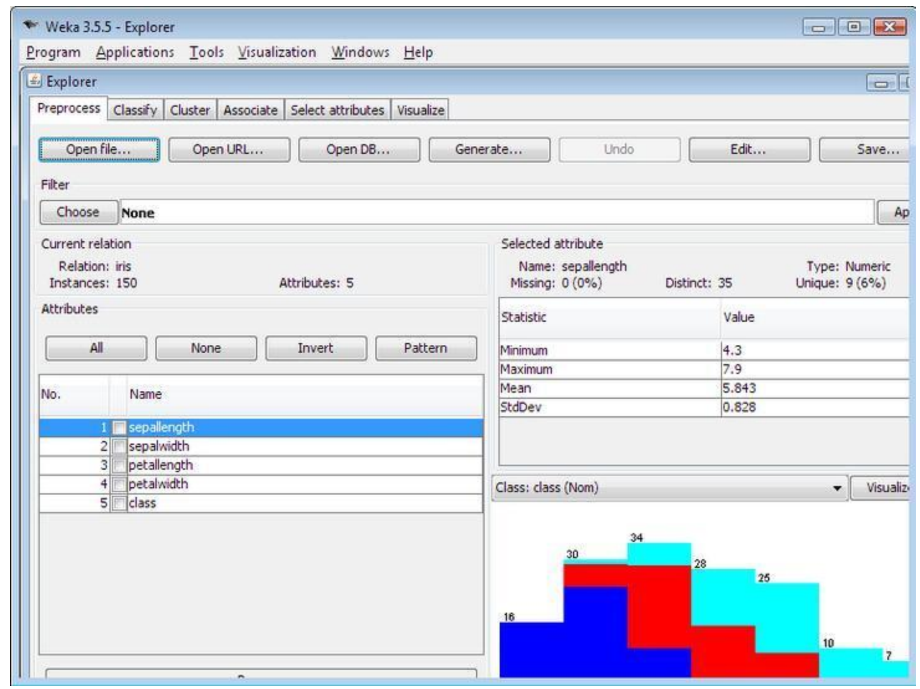
- 输入
  - 一个长度固定的向量，每维的值是数值，称为特征
- 输出
  - 分类问题
    - 从一个有限大小的集合中选择一个元素
  - 回归问题
    - 从一个连续空间中选择一个值
- 应用机器学习的关键是
  1. 问题能转换成这样的形式
  2. 能收集到该问题输入输出的大量实例

常见衡量指标

- Accuracy：输出有多少是正确的
  - $\frac{\text{正确输出数}}{\text{输出总数}}$
- Precision/Recall/F1：某一个特定类别上的准确度
  - $Precision_A = \frac{\text{输出中正确选择}A\text{的数量}}{\text{输出中}A\text{的数量}}$
  - $Recall_A = \frac{\text{输出中正确选择}A\text{的数量}}{A\text{的数量}}$
  - $F1 = \frac{2*precision*recall}{precision+recall}$

机器学习工具Weka

- Weka：包含一组实现好的机器学习算法的图形工具
- 基于Weka的软件分析
  - 将问题转换成向量输入和目标输出的形式
  - 收集该问题的大量实例
  - 将实例分成训练集和测试集
    - 常用k折交叉验证
    - 将数据分成k个集合，用一个集合做测试，剩下的做训练，重复k次
  - 在Weka中选择不同的算法和参数，直到效果好为止
  - 最后一步甚至可以用搜索算法自动进行 (AutoWeka)



K近邻法

K近邻法

- 把输入向量看做是超空间中的一个点
- 给定测试输入，找到超空间中和该输入点最临近的k个训练输入
- 输出训练输入的对应输出的平均值（连续情况）或者出现次数最多的值（离散情况）

- 缺点：
  - 实际中常常没有足够的临近点
  - 假设所有特征是等价的（通常不成立）
  - 假设特征对结果的影响和特征大小相关（有可能不成立）

朴素贝叶斯

朴素贝叶斯Naïve Bayes

- 困难：直接统计条件概率$P(Y|X=\bar{x})$样本数不够
- 方法：假设特征值相对输出都是彼此条件独立的

- 应用贝叶斯公式（y为任意输出，$\bar{x}$为任意输入）
  - $P(y|\bar{x}) = \frac{P(y)P(\bar{x}|y)}{P(\bar{x})}$
- 假设$\bar{x}$相对输出y条件独立，即$P(\bar{x}|y) = \prod_i P(x_i|y)$
  - $P(y|\bar{x}) = \frac{P(y)}{P(\bar{x})}\prod_i P(x_i|y)$
- 针对不同的y，$P(\bar{x})$不变，只需要统计$P(y)$和$P(x_i|y)$即可

朴素贝叶斯——举例

色泽青绿程度(1-3)	根蒂硬挺程度(1-3)	敲声清脆程度(1-3)	好瓜
3	1	1	是
1	1	1	是
3	3	3	否
1	2	2	否

- P(好瓜)=50%
- P(色泽=2|好瓜)=0%
- P(根蒂=1|好瓜)=100%
- P(敲声=1|好瓜)=100%
- P(好瓜|色泽=2,根蒂=1,敲声=1)=0%

拉普拉斯修正Laplacian Correction

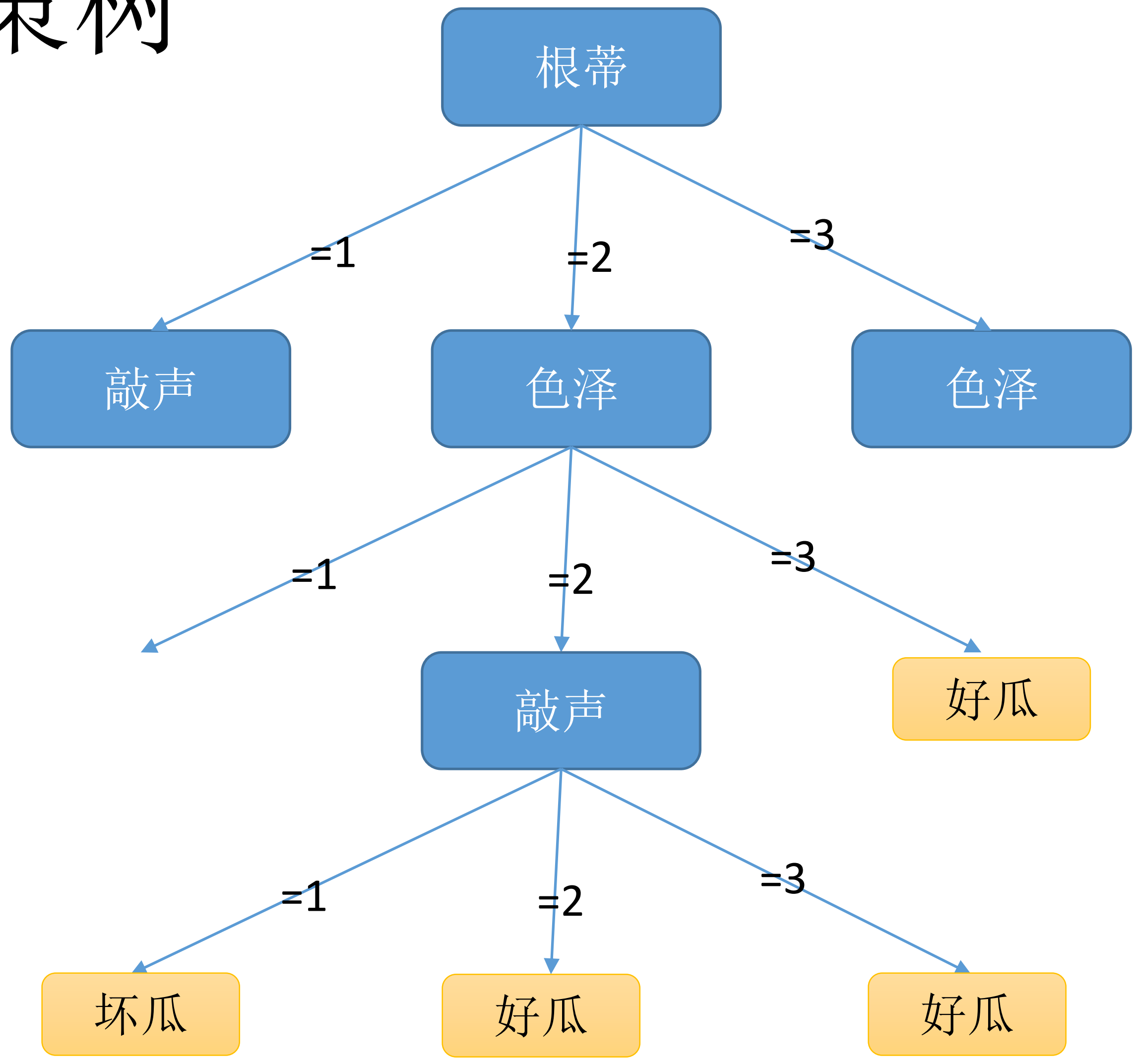
- 防止有概率为0的时候其他特征直接被忽略
- 假设存在一些虚拟样本，在虚拟样本中每种可能都均匀地出现一次
  - P(X=x)=(x的样本数+1)/(样本总数+X的取值可能数)
- 如：
  - P(好瓜)=(2+1)/(4+2)=50%
  - P(色泽=2|好瓜)=(0+1)/(2+3)=20%
  - P(根蒂=1|好瓜)=(2+1)/(2+3)=60%
  - P(敲声=1|好瓜)=(2+1)/(2+3)=60%
  - P(好瓜|色泽=2,根蒂=1,敲声=1)=3.6%/P(色泽=2,根蒂=1,敲声=1)

计算方法

- 连乘的时候，如果特征值很多，计算结果非常小，容易溢出
- 解决方案:
  - 计算概率的对数
  - $\log\big(P(y|\bar{x})\big) = logP(y) - logP(\bar{x}) + \sum_i logP(x_i|y)$
- 因为log是单调函数，所以最后直接比较对数即可

决策树

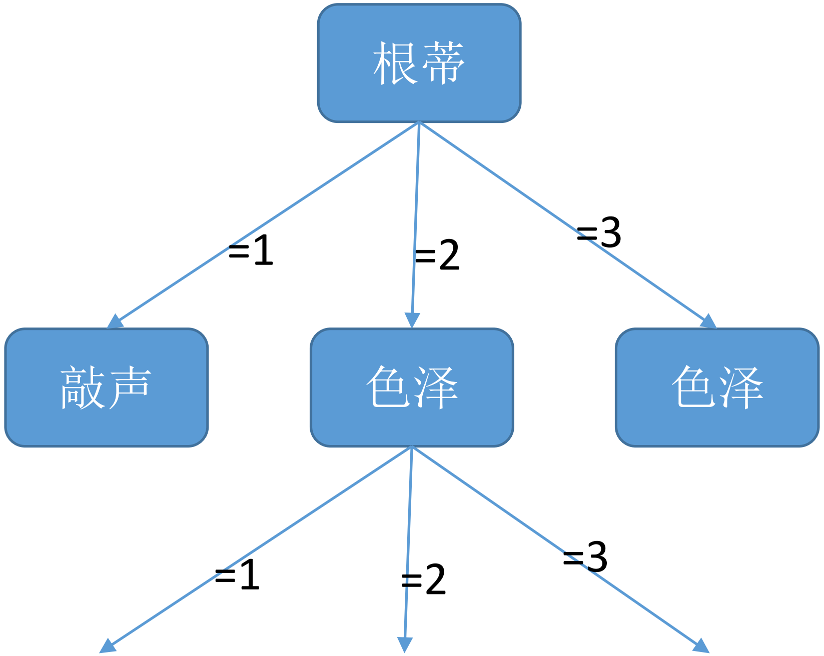
决策树



决策树生成过程

- 每次选一个特征
- 将样本按该特征值分类，重复上诉过程

- 终止条件：
  - 样本中都只包含一种输出值——叶节点为该值
  - 样本在剩余属性上取值都一样——选出现次数最多的输出值
  - 样本集合为空——选父节点的次数最多的输出值



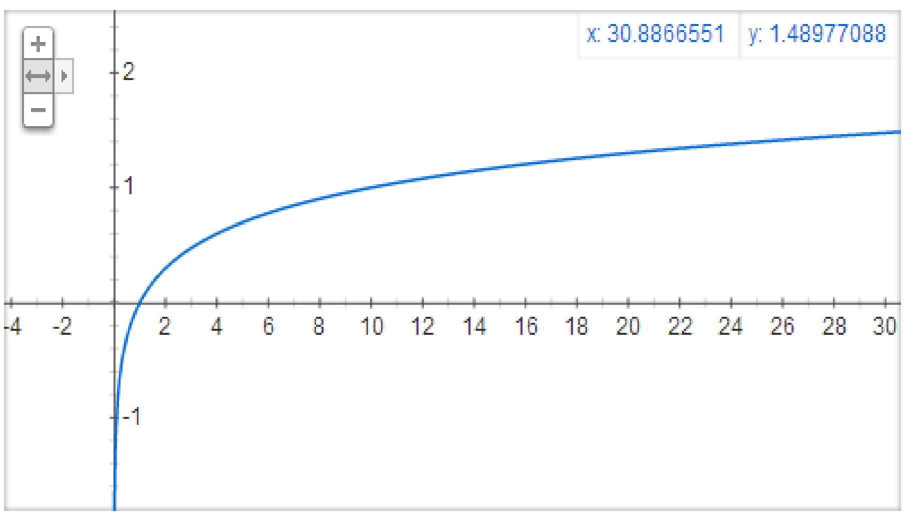
决策树特征选择

- 目标：尽可能生成最小的树
  - 即：尽量不用遍历所有特征就能得出结果
- 基本思路：贪心法
  - 每次尽量让选择的集合中的输出“纯度”最高
- 如何定义“纯度”？

信息熵Information Entropy

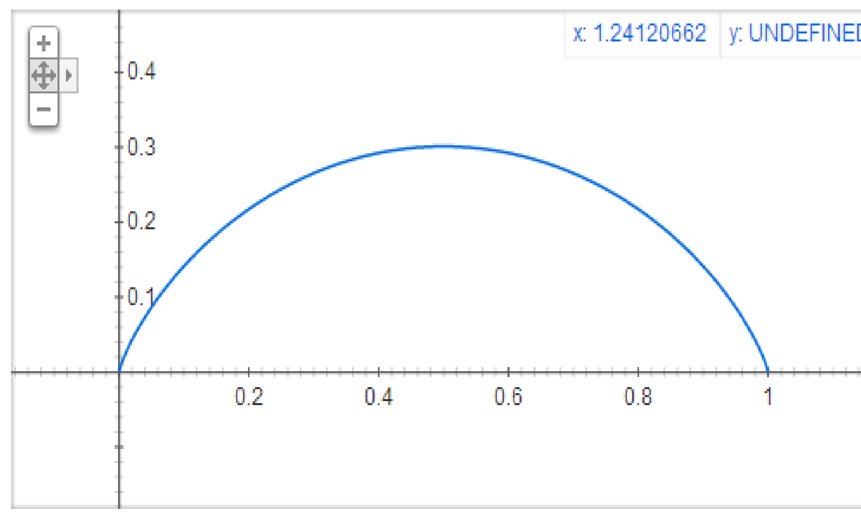
- 假设样本集合D中包含n种输出取值，每种占的比例为p_k，则整体信息熵为：
  - $Ent(D) = -\sum_{k=1}^{n} p_k \log p_k$

Graph for (-x)*1/x*log(1/x)



种类越多，信息熵越大

Graph for (-x)*log(x)-(1-x)*log(1-x)



分类越均匀，信息熵越大

决策树特征选择

- 目标：尽可能生成最小的树
  - 即：尽量不用遍历所有特征就能得出结果
- 基本思路：贪心法
  - 每次尽量让选择的集合中的输出“纯度”最高
- 如何定义“纯度”？
  - 信息熵最小
- 最小化一个信息熵度量
  - ID3算法：$\Sigma_i \frac{|D^i|}{|D|} Ent(D^i)$ D^i为第i个分类
  - C4.5算法：$\frac{\Sigma_i \frac{|D^i|}{|D|} Ent(D^i) - Ent(D)}{\Sigma_i \frac{|D^i|}{|D|} \log \frac{|D^i|}{|D|}}$ 平衡ID3容易选择分类多的属性的倾向

决策树的其他扩展

- 剪枝：为避免过拟合，去掉一部分分支
  - 留出一部分数据作为验证集，在验证集上效果不好的分支删掉
- 处理连续特征
  - 采用二分法查找连续特征（C4.5）

决策树小结

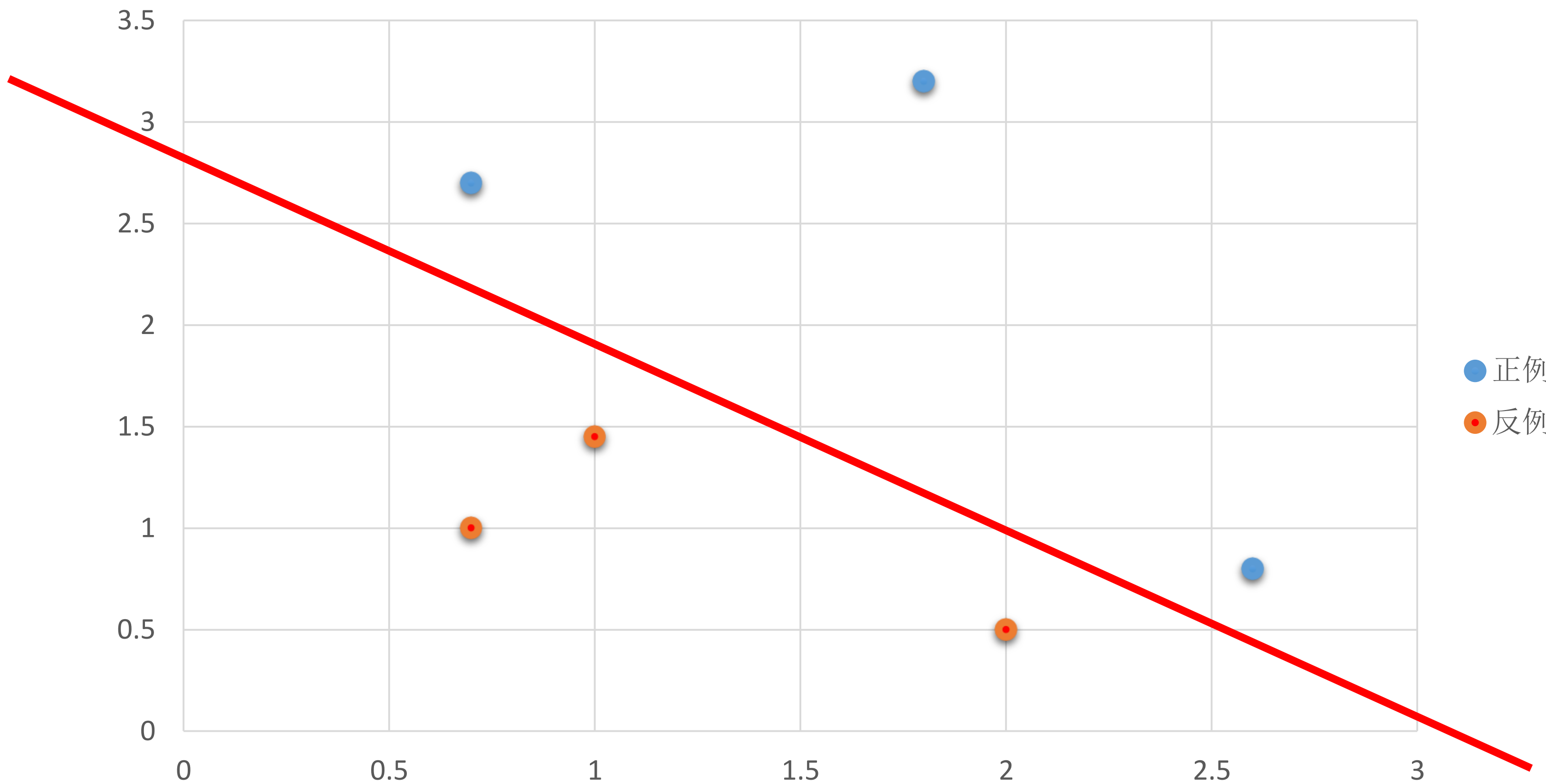
- 决策树考虑了特征之间的依赖关系
- 决策树主要依赖对输入分段，在理想决策函数为线性或多项式关系时效果不好

支持向量机

支持向量机

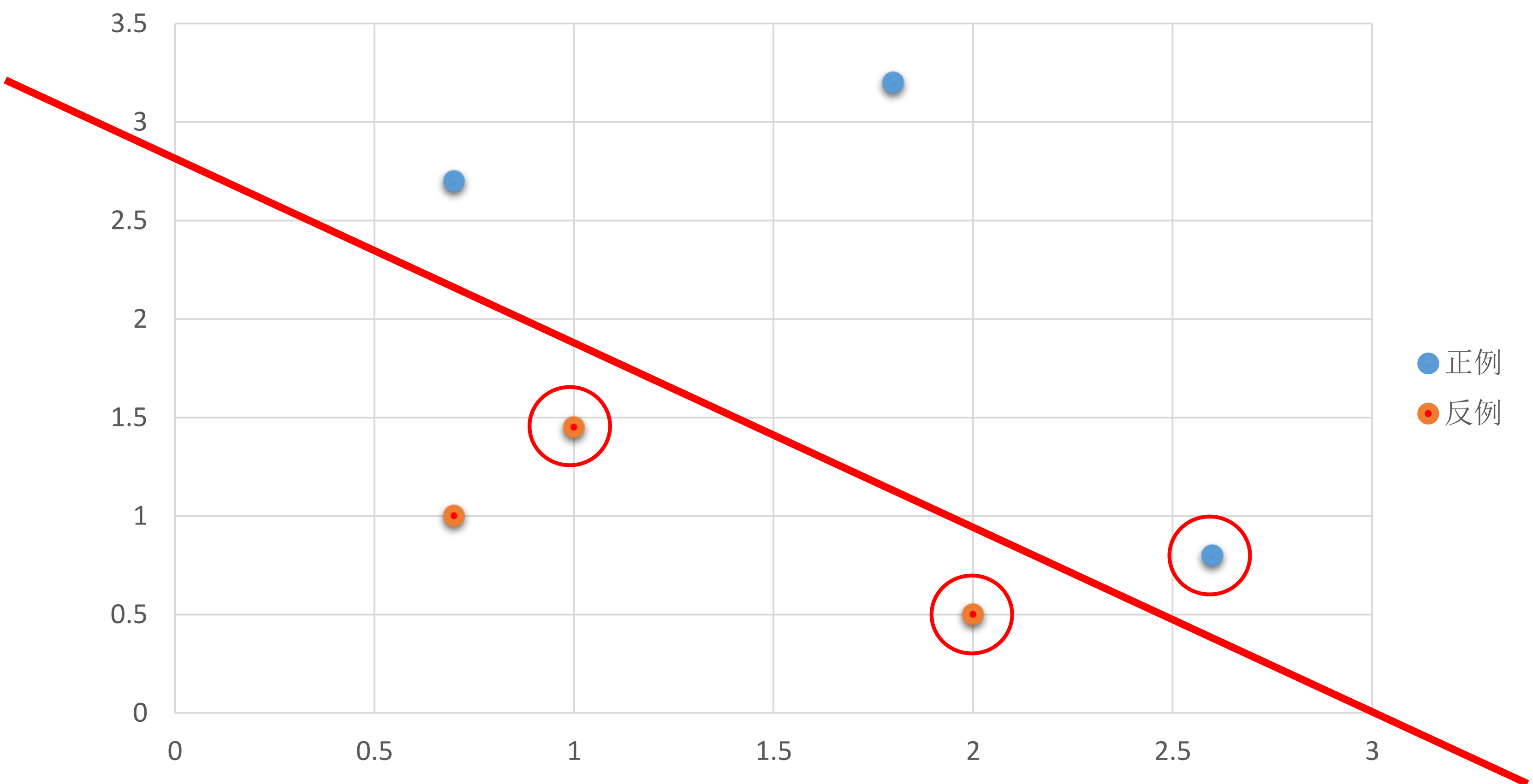
- 假设输出只有两种可能的取值
  - 多种取值可以用多个分类器多次分类（稍后介绍）
- 假设决策函数是一个线性函数
  - 非线性函数可以用核函数的方式变成线性的（稍后介绍）

支持向量机



寻找一个超平面，将所有的正例和反例点分开，并且距离所有点的最小距离最大。

支持向量



超平面完全由距离最近的几个点决定，所以叫做支持向量

复习：线性代数

- 超平面的表示
  - $\bar{\omega}\bar{x} + b = 0$
  - 其中$\bar{\omega}$为行向量，b为常数；$\bar{x}$为列向量，表示平面中的点。
- 点$\bar{x}$到超平面的距离
  - $\frac{\bar{\omega}\bar{x}+b}{||\omega||}$
  - 其中$||\omega|| = \sqrt{w_1^2 + \cdots + w_n^2}$
  - 如果结果为正，则点在超平面的一侧，如果结果为负，则点在超平面的另一侧

支持向量机转优化问题

- 如果样本$\overline{x_i}$是正例，令$y_i = 1$，否则$y_i = -1$
- 原问题变为最大化
  - $y_i \frac{\overline{\omega}\overline{x_i}+b}{||\omega||}$的最小值
  - 且对于任意样本i，$y_i \frac{\overline{\omega}\overline{x_i}+b}{||\omega||}$为正
- 注意到分子分母可以等比例增加，不妨限定支持向量上$y_i(\overline{\omega}\overline{x_i} + b)=1$，则原问题可变为如下支持向量机基本型
  - 最小化$\frac{1}{2}||\omega||^2$
  - 保证对于任意样本i，$y_i(\overline{\omega}\overline{x_i} + b) \geq 1$
- 凸包优化问题，可用拉格朗日乘子法和SMO算法求解

软间隔

- 如果样例无法用超平面分开怎么办？
- 情况一：只有少数样例无法区分
- 解决方案一：允许分类出现错误，同时最小化出错的样本个数和出错程度
- 对于每个样本i，引入变量ξ_i，同时对标准型进行如下改写
  - 最小化$\frac{1}{2}||\omega||^2 + C\sum_i \xi_i$
  - 保证对于任意样本i，$y_i(\overline{\omega}\overline{x_i} + b) \geq 1 - \xi_i$
  - 且$\xi_i \geq 0$
  - C为惩罚参数，是大于0的常数
- 仍然是凸包优化问题，可用拉格朗日乘子法和SMO算法求解

核函数

- 如果样例无法用超平面分开怎么办？
- 情况二：较多样例无法区分，即理想决策函数很可能不是线性的
- 解决方案二：引入新的特征来代表非线性的项
- 例
  - 原始特征：x_1, x_2
  - 理想决策函数：$x_1^2 + x_2^2 > 0$
  - 引入新特征：$x_3 = x_1^2, x_4 = x_2^2$
- 然后重新用支持向量机学习即可

核函数

- 假设存在映射ϕ将原来的输入向量映射到新的向量，基本型变换为
  - 最小化$\frac{1}{2}||\omega||^2$
  - 保证对于任意样本i，$\mathrm{y_i}(\overline{\omega}\phi(\overline{x_i})+b)\geq 1$
- 在求解过程中，我们实际不需要知道ϕ的定义，只需要知道$\mathrm{K}(\bar{x})=\phi^T(\bar{x})\phi(\bar{x})$的定义即可，称为核函数
  - 注意一个K可以对应多个不同的ϕ
- 所以在支持向量机中采用直接定义核函数的方式

常见核函数

名称	表达式	参数
线性核	$K(x,y)=\mathrm{x}^{\mathrm{T}}\mathrm{y}$	
多项式核	$K(x,y)=(x^Ty)^d$	$d\geq 1$为多项式的次数
高斯核	$K(x,y)=e^{-\frac{\|\|x-y\|\|^2}{2\sigma^2}}$	$\sigma>0$为高斯核的带宽
拉普拉斯核	$K(x,y)=e^{-\frac{\|\|x-y\|\|}{\sigma}}$	$\sigma>0$
Sigmoid核	$K(x,y)=\tanh(\beta x^Ty+\theta)$	$\beta>0,\theta<0$

- 线性核等价于支持向量机基本型
- 周志华老师建议：文本通常采用线性核，情况不明的时候采用高斯核

多分类问题转二分类问题

- OvO：给定N个分类，对任意两个分类都训练一个分类器，总共N(N-1)/2个。最终分类通过投票产生。
- OvR：给定N个分类，对任意分类（正类）和其他所有分类（负类）训练一个分类器，总共N个。如果有多个分类器预测为正类，则选择预测可能性最大的，即$\bar{\omega}\bar{x}+b$最大的
- ECOC：用纠错码的方式进行编码，这样当只有少量分类器分类错误的时候仍然可以得到正确结果。

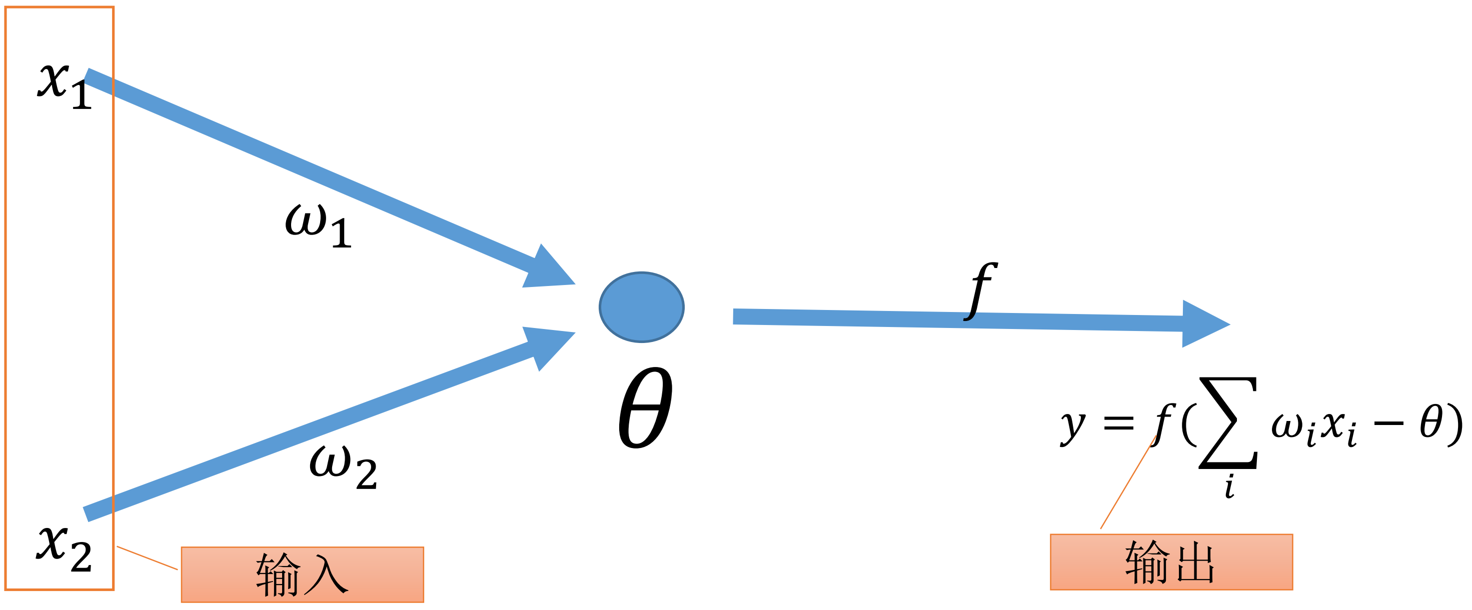
支持向量机小结

- 适用于特征值连续的情况
- 对训练集的大小要求不是特别强
- 90年代和00年代初期被最广泛使用的学习算法之一

神经网络

感知机

- 模拟人类神经元的功能，对输入信号进行处理后，输出[0, 1]之间的输出信号

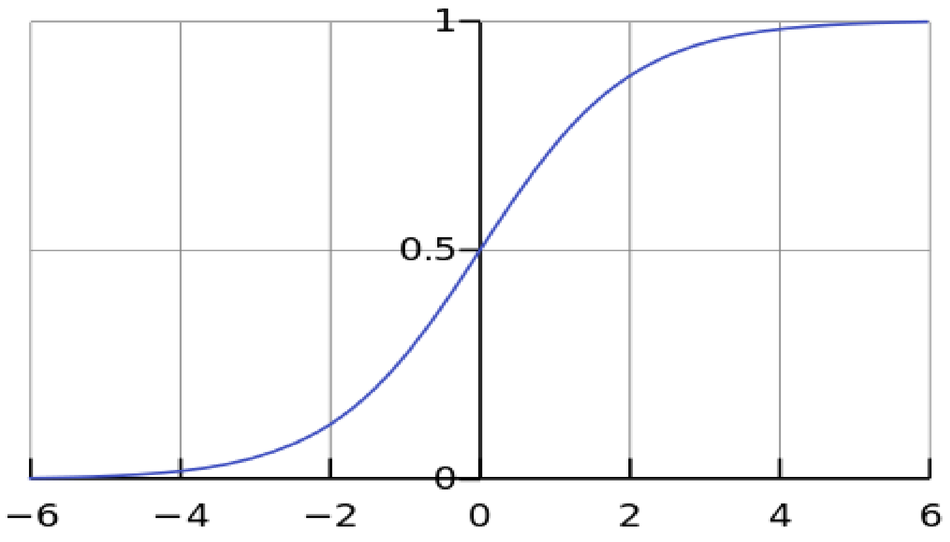


- f为激活参数，用于将一个实数域上的值尽可能映射成0和1两种信号

常见激活函数

- 阶跃函数
  - $f(x) = 1$ 如果$x \geq 0$
  - $f(x) = 0$ 如果$x < 0$
- Sigmoid函数

$$f(x) = \frac{1}{1 + \mathrm{e}^{-x}}$$



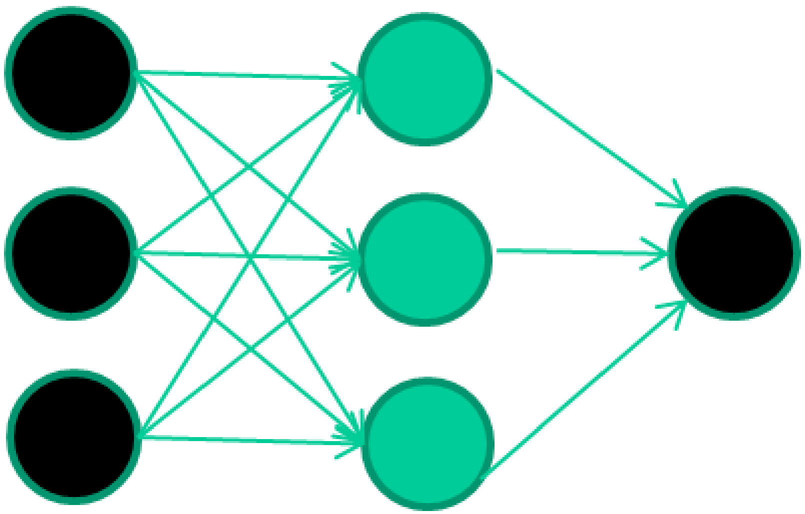
感知机模拟基本逻辑运算

- 逻辑与：$\omega_1 = 1, \omega_2 = 1, \theta = 2$
- 逻辑或：$\omega_1 = 1, \omega_2 = 1, \theta = 1$
- 逻辑非：$\omega_1 = -1, \theta = -0.5$

- 其他没有用到的输入对应的ω都是0

神经网络

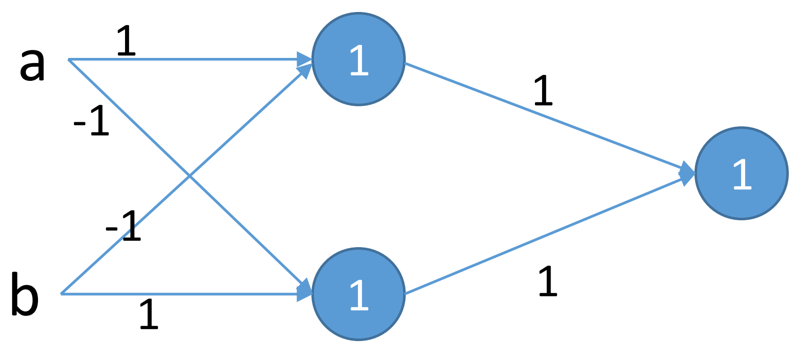
- 将多个感知机连接成网络



- 可以模拟更复杂的计算

例：神经网络模拟更多运算

- 异或运算无法用线性模型表达，也就无法用感知机表达
- $a \oplus b = (a \wedge \neg b) \vee (\neg a \wedge b)$
- 可以用两层神经网络



误差反向传播算法
Error Backward Propagation, BP

- 基本思路：
  - 首先随机生成参数，计算结果误差
  - 根据结果误差和参数对结果影响大小按比例调整参数
  - 根据训练集反复迭代多次直到收敛
- 通用公式：
  - $v \coloneqq v + \Delta v$
  - $\Delta v \coloneqq \eta(expected - output)\frac{doutput}{dv}$

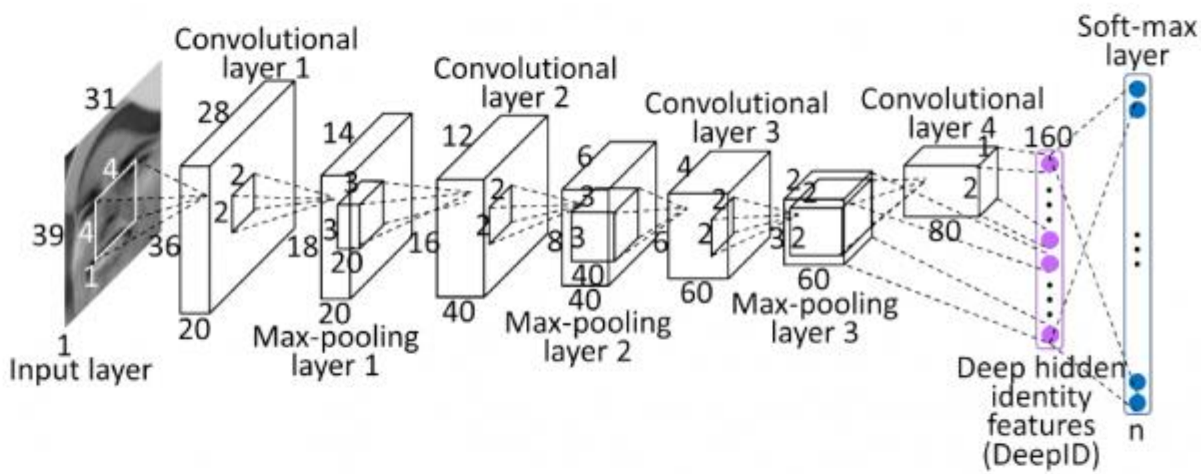
深度学习

- 层数比较多的神经网络
- 给定足够多的训练样本，能模拟较复杂的行为
- BP算法针对深度神经网络也有一定优化

- 下面介绍两种用于适合处理程序的神经网络
  - 卷积神经网络
  - 循环神经网络
- 程序特点：长度不固定，且如果直接编码成特征维度太高

卷积神经网络

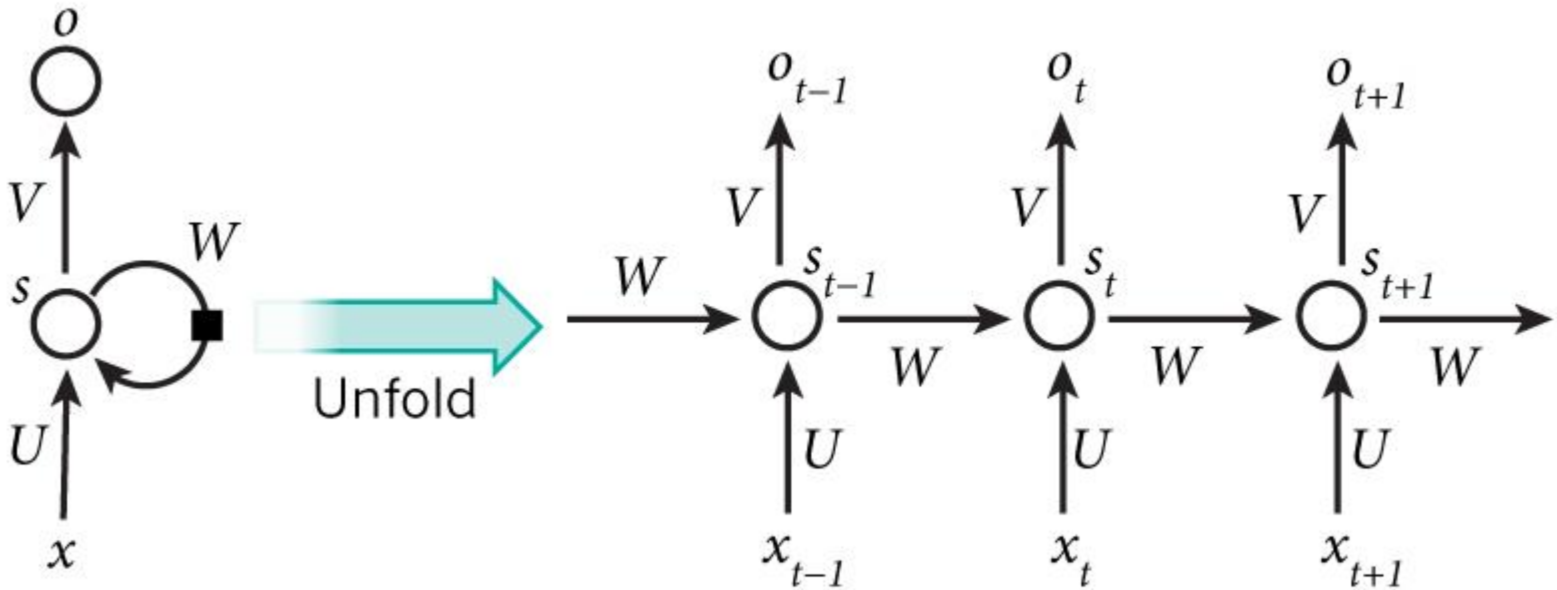
- 被广泛应用于图像处理，也可以处理程序[牟力立等,AAAI 2016]
- 两个典型过程：
  - 卷积(convolution)：对一个大小固定的子块提取特征
    - 图像：n*n的小图像
    - 程序：包含n个节点的内部子树
  - 池化(pooling)：选择卷积提取的特征中的典型值
    - 典型值：最大值、最小值、平均值等等
    - 可以对一个字块做（结果大小不固定），也可以对一个特征做（结果大小固定）
- 卷积和池化可反复进行多次



ImageNet网络结构

循环神经网络

- 适合处理连续文本的神经网络
- 上一次的输出同时作为这一次的输入



神经网络实现框架

- Tensor flow （Google开发）
- MXNet（多个大学和公司联合开发，含百度）
- ……

- 实现了神经网络的学习算法
- 用户指定神经网络的结构和训练集，算法自动训练出神经网络
- 提供分布式、异构编程支持等

N-Gram

N-Gram

- 处理连续文本的基本模型
- N-Gram模型统计在出现了n-1个词的概率下第n个词的出现概率
  - 即统计$P(\mathrm{w_n} \mid \mathrm{w_1 w_2} \dots w_{n-1})$
  - 通过直接采样进行统计$P(\mathrm{w_n} \mid \mathrm{w_1 w_2} \dots w_{n-1}) = \frac{P(w_1 w_2 \dots w_n)}{P(w_1 w_2 \dots w_{n-1})} = \frac{w_1 w_2 \dots w_n\text{的出现次数}}{w_1 w_2 \dots w_{n-1}\text{的出现次数}}$
- 由于训练集的样本有限，n通常比较小，常见n=2-5
- 通常应用拉普拉斯修正来处理没有出现的序列

统计语言模型与N-Gram

- 统计语言模型：用于输出一个句子（单词序列）出现概率的模型。
  - 即计算$P(w_1 w_2 \ldots w_m)$
- N-Gram是一种典型的统计语言模型
  - $P(w_1 w_2 \ldots w_m) = \prod_{i=0}^{m} P(w_i \mid w_1 w_2 \ldots w_{i-1}) \approx \prod_{i=0}^{m} P(w_i \mid w_{i-n+1} w_{i-n+2} \ldots w_{i-1})$
- 循环神经网络也可以认为是一种统计语言模型
- 研究人员也提出了部分用于编程语言的统计语言模型

N-Gram与其他分类器

- 借鉴N-Gram的思想，可以把连续文本映射到有限长度的向量上
- 给定n=3，特征$F_{w_1 w_2 w_3}$表示串$w_1 w_2 w_3$出现的次数
- 可以用其他机器学习算法给文本分类

线性回归分析

线性回归分析

- 以上方法针对离散的情况
- 决策函数输出是连续值的时候怎么办？
- 线性回归分析：假设决策函数是线性函数
  - 即：$f(\bar{x}) = \bar{\omega}\bar{x} + b$
- 非线性的情况可以和SVM类似用添加特征的方式来支持

线性回归分析训练算法

- 考虑一个特征的情况
  - 模型：决策函数为$f(x) = wx + b$
  - 策略：均方差衡量误差，即在训练集上的误差为
    - $E = \sum_{\mathrm{i}=1}^{m}(f(x_i) - y_i)^2 = \sum_{\mathrm{i}=1}^{m}(\mathrm{w}x_i + b - y_i)^2$
  - 算法：
    - E相对w和b均为二次项为正的二次式，曲率为0的时候最小
    - 将上式分别对w和b求导，得
      - $\frac{dE}{dw} = 2\left(w\sum_{\mathrm{i}=1}^{m} x_i^2 - \sum_{\mathrm{i}=1}^{m}(y_i - b)x_i\right)$
      - $\frac{dE}{d\mathrm{b}} = 2(mb - \sum_{\mathrm{i}=1}^{m}(y_i - wx_i))$
    - 另$\frac{dE}{dw}$=$\frac{dE}{d\mathrm{b}}$=0，得到二元一次方程，求解即可
- 多特征的情况原理类似，但计算过程更加复杂

训练数据预处理

训练集的平衡问题

- 训练集不平衡时可能无法得到想要的决策函数
  - 比如，训练集中99%都是正例，那么决策函数直接返回正例就有99%的机会正确
- 通常做法
  - 欠采样Undersampling：扔掉一些反例
    - 随机扔掉
    - EasyEnsemble：用不同的反例子集训练多个分类器，并集成结果
  - 过采样Oversampling：添加一些正例
    - 直接复制——容易导致过拟合
    - SMOTE：随机选择两个正例，然后在欧式空间生成一个他们的中点
  - 代价敏感学习：为分类错误赋不同的权重

归一化

- 很多算法都依赖超空间的距离
  - K邻近、SVM等
- 如果特征的取值范围不一，那么会偏向取值范围大的特征
- 常见归一法：

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

$$x^* = \frac{x - \mu}{\sigma}$$

μ：均值
σ：标准差

线性归一化：最小值归一成0，最大值归一成1

标准差归一化：均值归一化成0，标准差归一化成1

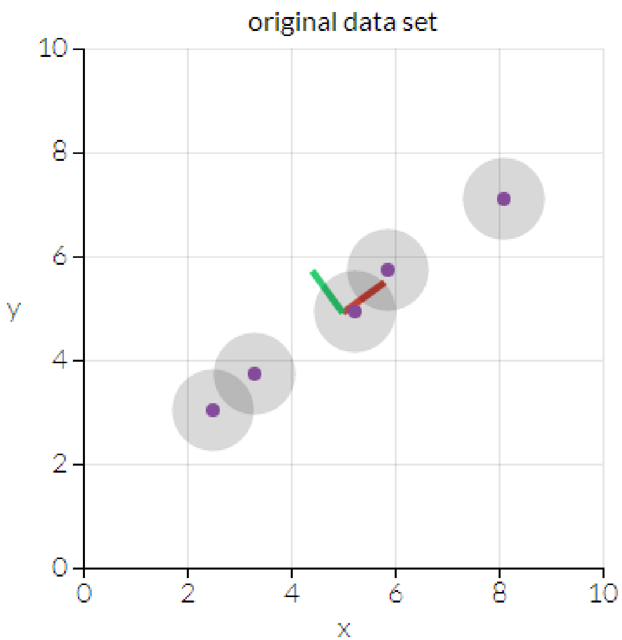
降维和特征选择

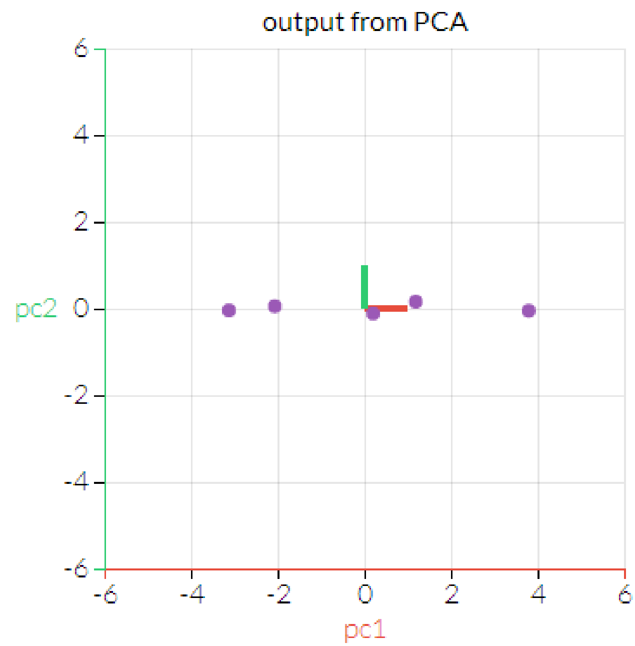
- 特征数量太多常常会影响学习的效果
  - 考虑K近邻法，会找不到周边的结点
  - 考虑SVM，特征越多，支持向量上所需要的点就越多
- 降维：将原始高维空间投影到低维空间，使得高维空间的样本距离尽量在低维保持
- 选择：扔掉一些不重要的特征

降维——主成分分析PCA

- 输入：高维样本集合X，低维空间的维度d
- 输出：一个线性变化W将样本映射到低维空间Z=WX，使得
  - 每一维的方差都尽可能大
    - 方差较小的维度可能是误差
  - 不同维之间的协方差绝对值都尽可能小
    - 协方差：$\sigma_{AB}^2 = \frac{\sum_{i=1}^{n}(a_i - \overline{a})(b_i - \overline{b})}{n-1}$
    - 协方差绝对值越小表明变量之间越独立
- 低维空间的维度d也可以通过设定一个信息保留比例t得到，比如设置t=95%

PCA示例





线性变换可以看做是从高维空间到低维空间的投影

PCA使用注意

- PCA假设特征的重要性由方差决定，所以根据需要很可能要对数据归一化
- PCA的映射是线性的，如果要寻找更复杂的映射也可以采用Kernel PCA

特征选择

- 特征选择保留下重要的特征子集
- 如何确定特征子集是重要的？
  - 和决策树类似，采用信息熵确定
  - 给定特征子集，将D分成了子集$D_1, D_2, \ldots \ldots$
  - 计算$\Sigma_i \frac{|D^i|}{|D|} Ent(D^i)$ 或者$\Sigma_i \log \frac{|D^i|}{|D|} Ent(D^i)$，越小说明分类的纯度越高，则特征子集越好
- 选择标准：
  - 给定特征子集大小，选出最好的特征子集
  - 给定阈值A，选择令$Ent(D) - \Sigma_i \frac{|D^i|}{|D|} Ent\left(D^i\right) > A$的最小子集
- 选择算法：通常采用搜索算法