一、特征选择

> 子集搜索+子集评价
> 决策树可用于特征选择，随机森林不需要
> 特征选择独立于任何机器学习算法

为什么特征选择?

- 去除不重要的，避免维数灾难
- 去除不相关的，降低任务难度

无关特征：与当前任务无关
冗余特征：可以从其他特征推演出来

1、特征选择环节

原因

遍历所有可能子集代价太大，所以维持评价候选子集，直到best

1.1子集搜索（3种贪心）

> 速度快，非最优

- 前向搜索：先选一个最优特征作为初始特征子集，逐渐加一个最优特征，最优子集越来越好

> 缺点：可能同时加多个特征会更好，但是停了

- 后向搜索：从完整特征集合开始，递减
- 双向搜索：每一轮增加相关特征，同时减少无关特征

1.2子集评价（信息熵等）

> 特征子集确定了一个划分，可以评价

偏序关系，下一层一定比上一层小，一个分支全部划掉

2、特征选择方法

> 搜索+评价相结合

2.1、过滤式filter--Relief

先对数据集进行特征选择，然后再训练学习器，不管学习器效果先把特征筛了

> 经常作为预处理
> 但关联度与最终结果好坏未必有关

> **Relief(Relevant Features)**
> 用相关统计量给特征打分

- 给每个特征一个“相关统计量”，是向量
- 特征子集重要性由子集中每个特征对应的相关统计量分量之和决定，即特征的“评分”
- 设计阈值，选大于阈值（或最大的k）个特征，作为特征选择结果

相关统计量设计--2分类

相关统计量对应于属性j的分量为

$$\delta^j = \sum_i -\text{diff}(x_i^j, x_{i,nh}^j)^2 + \text{diff}(x_i^j, x_{i,nm}^j)^2$$

- 对每个示例xi，找同类样本中最近邻（猜中近邻）和猜错近邻
- diff是一种算距离方法，可更换
- 相关统计量大：在j属性上异类最近邻远，同类最近邻近，说明j有用
- Relief 只需在数据集的采样上，而不必在整个数据集上估计相关统计量
- 开销随采样次数&原始特征数线性增加，效率很高

可拓展至k分类

一个猜中近邻和k-1个猜错近邻

pl为第l类样本在数据集D中所占的比例

$$\delta^j = \sum_i -\text{diff}(x_i^j, x_{i,\text{nh}}^j)^2 + \sum_{l \neq k} \left(p_l \times \text{diff}(x_i^j, x_{i,l,\text{nm}}^j)^2 \right) ,$$

2.2、包裹式Wrapper--LVW

直接用学习器结果评价属性子集，量身定做特征子集

> 效果好，计算量非常大，因为需要多次训练学习器

> **LVW拉斯维加斯方法（随机算法）**
> 著名随机算法，时间长不能停，停了一定是最优。时间有限则可能给不出解
> 但蒙特卡洛随机算法随时可以停，越晚停越好

- 循环每一轮随机测试一个特征子集，交叉验证得误差，选误差最小的

2.3、嵌入式选择embedding--L1

不再分开特征选择过程和学习器训练过程，而是学习器训练时自动特征选择

> **L1正则**
>
> 特征多而样本少时，为防止过拟合引入正则化项L1，称为LASSO （**最小绝对收缩选择算子**）

$$\min_{\boldsymbol{w}} \sum_{i=1}^{m} (y_i - \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_i)^2 + \lambda \|\boldsymbol{w}\|_1$$

> 若引入L2，则称为岭回归

$$\min_{\boldsymbol{w}} \sum_{i=1}^{m} (y_i - \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_i)^2 + \lambda \|\boldsymbol{w}\|_2^2$$

为什么L1可以嵌入式（优点）？

L1比L2好在易于获得稀疏解（求得的w有更多0分量，因此相当于做了特征选择，才叫嵌入式）

- 因为L1的解更容易出现在坐标轴上，L2在象限内。直观理解如图：

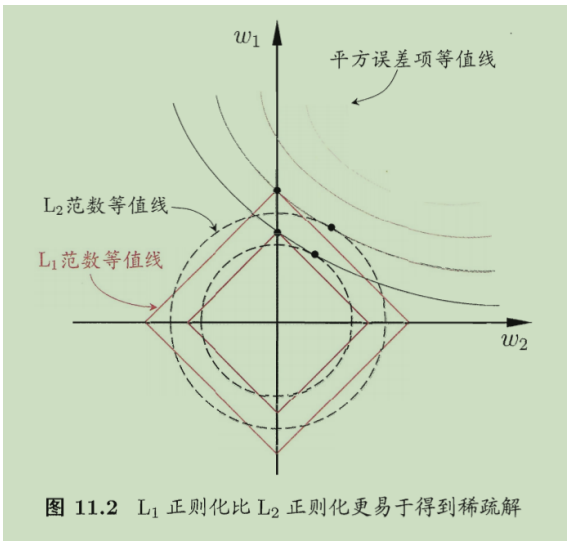

图 11.2 L_1 正则化比 L_2 正则化更易于得到稀疏解

- 因此L1得到了仅采用一部分特征的模型，不知不觉完成了特征选择

三、字典学习+稀疏表示

稀疏表示优势

> 所以要把稠密数据适当稀疏

- 使大多数问题线性可分
- 高效存储
- 文档分类任务经常是稀疏矩阵数据

字典学习：

> 为稠密样本找到合适字典，样本转化为合适稀疏表示形式

- 学习目标是字典矩阵B和稀疏表示α
- 可以设置词汇量大小k来控制字典规模，从而影响稀疏程度

压缩感知：