第8章

特征选择与稀疏学习

Feature Selection and Sparse Learning

孟 高 峰

gfmeng@nlpr.ia.ac.cn

http://people.ucas.ac.cn/~gfmeng

时空数据分析与学习课题组(STDAL)

中科院自动化研究所　模式识别国家重点实验室

内容提要

- 子集搜索与评价
- 特征选择方法
  - 过滤式选择
  - 包裹式选择
  - 嵌入式选择
- 稀疏表示与字典学习
  - KSVD算法
- 压缩感知

特征选择

- 处理高维数据的两大主流技术之一
- 特征选择：给定一个学习任务，对于给定的数据属性（特征）集，从中选出与任务相关（对学习任务有利）的特征子集的过程。
  - 减少数据维度，缓解“维数灾难”，减少计算量
  - 通过去除与任务不相关特征、冗余特征、或者关联性较小的特征，降低学习任务的难度
  - 通过选择与任务相关的特征，提高分类器性能

子集搜索与评价

- 特征选择涉及两个关键环节：
  - 子集搜索: 从特征集合中搜索最优的特征子集
  - 子集评价: 对特征子集的分类性能进行评价
- 子集搜索(subset search)
  - 给定特征集合 $\{a_1, a_2, \cdots, a_d\}$，搜索最优的特征子集
- 子集评价(subset evaluation)
  - 对给定的特征子集，依据某种评价准则，对该特征子集进行优劣评价
  - 通常基于类别可分性来进行特征子集评价
  - 常用的判定准则包括：信息增益、信息熵等

子集评价

- 特征子集评价判据：评价一组特征性能好坏的客观标准。
  - 直接判据：分类器的分类错误率（通常很难计算）
  - 间接判据：与分类器的分类性能存在一定关系的判据，例如：不同类别数据的可分程度、不同类别的概率分布的差异性、特征对于分类的不确定性程度。
- 评价准则
  - 基于距离的准则（Distance-based criterion）
  - 基于分布的准则（Distribution-based criterion）
  - 基于熵的准则（Entropy-based criterion）

评价准则

- 什么是“理想的”评价准则？
- 评价准则J_{ij}反映了在一组特征下，第i和第j类的可分程度。理想的评价准则应满足：
  - 应该与分类错误率具有正相关性，以反映特征的分类性能
  - 对于独立特征，评价标准应具有可加性

$$J_{ij}(x_1,\ldots,x_d)=\sum_{k=1}^{d}J_{ij}(x_k)$$

  - 应该是一个度量（metric）

$$\begin{aligned} J_{ij} &> 0, \quad \text{for } i \neq j \\ J_{ij} &= 0, \quad \text{for } i = j \\ J_{ij} &= J_{ji} \end{aligned}$$

  - 是特征数目的单调函数，也即新加入特征不会减少可分度$J_{ij}(x_1,\ldots,x_d) \leq J_{ij}(x_1,\ldots,x_d,x_{d+1})$

基于距离的评价准则

- 记$\mathbf{x}_k^{(i)} \in \mathbb{R}^d$和$\mathbf{x}_l^{(j)} \in \mathbb{R}^d$为类别$\omega_i$和$\omega_j$对应的特征
- 记$\mathbf{x}_k^{(i)} \in \mathbb{R}^d$和$\mathbf{x}_l^{(j)} \in \mathbb{R}^d$之间的距离为$\delta(\mathbf{x}_k^{(i)}, \mathbf{x}_l^{(j)})$
- 可定义所有类别上的总距离为：

$$J_d(\mathbf{x}) = \frac{1}{2}\sum_{i=1}^{c} P_i \sum_{j=1}^{c} P_j \frac{1}{n_i n_j} \sum_{k=1}^{n_i}\sum_{l=1}^{n_j} \delta\left(\mathbf{x}_k^{(i)}, \mathbf{x}_l^{(j)}\right)$$

- 如果基于欧式距离 $\delta\left(\mathbf{x}_k^{(i)}, \mathbf{x}_l^{(j)}\right) = \left(\mathbf{x}_k^{(i)} - \mathbf{x}_l^{(j)}\right)^T \left(\mathbf{x}_k^{(i)} - \mathbf{x}_l^{(j)}\right)$
- 则有

$$J_d(\mathbf{x}) = \sum_{i=1}^{c} P_i \left[\frac{1}{n_i} \sum_{k=1}^{n_i} \left(\mathbf{x}_k^{(i)} - \mathbf{m}_i\right)^T \left(\mathbf{x}_k^{(i)} - \mathbf{m}_i\right) + (\mathbf{m}_i - \mathbf{m})^T (\mathbf{m}_i - \mathbf{m}) \right]$$

类别i的重心　　所有数据点的重心

基于距离的评价准则

- 利用散度矩阵，可将上式整理成更简单的形式
- 定义类间散度如下：

$$\mathbf{S}_b = \sum_{i=1}^{c} P_i (\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^T$$

- 定义类内散度如下：

$$\mathbf{S}_w = \sum_{i=1}^{c} P_i \frac{1}{n_i} \sum_{k=1}^{n_i} \left(\mathbf{x}_k^{(i)} - \mathbf{m}_i \right) \left(\mathbf{x}_k^{(i)} - \mathbf{m}_i \right)^T$$

- 则有：

$$J_d(\mathbf{x}) = \mathrm{tr}(\mathbf{S}_w + \mathbf{S}_b)$$

基于距离的评价准则

- 类似的，根据Fisher线性判别准则，可定义如下的评价准则，使得类内散度尽可能小，类间散度尽可能大

$$J_2 = \mathrm{tr}\left(S_w^{-1} S_b\right)$$

$$J_3 = \frac{tr S_b}{\mathrm{tr}\, S_w}$$

$$J_4 = \frac{|S_b|}{|S_w|}$$

基于分布的评价准则

- 假如我们定义了分布 $p(\mathbf{x}|\omega_i)$和 $p(\mathbf{x}|\omega_j)$之间的一个“距离”函数，该距离反映了这两个条件分布之间的重合程度；
- 这个“距离”函数应该是非负的；
- 当这两个条件分布不重叠时，该距离函数取得最大值，而当这两个条件分布一样时，该距离函数应该取零值；
- 该距离函数可用两个分布之间的KL散度来表示；

Kullback-Leibler散度

- 对数似然比

$$L_{ij}(\mathbf{x}) = \ln \frac{p(\mathbf{x}|\omega_i)}{p(\mathbf{x}|\omega_j)}$$

- KL散度又称为相对熵(relative entropy)，定义为对数似然比的数学期望

$$KL_{ij}(\mathbf{x}) \triangleq E[L_{ij}(\mathbf{x})] = \int p(\mathbf{x}|\omega_i) \ln \frac{p(\mathbf{x}|\omega_i)}{p(\mathbf{x}|\omega_j)} d\mathbf{x}$$

  - 注意：KL散度不是一个度量 $KL_{ij}(\mathbf{x}) \neq KL_{ji}(\mathbf{x})$
  - 可将其变成一个度量

$$J_D(\mathbf{x}) = KL_{ij}(\mathbf{x}) + KL_{ji}(\mathbf{x}) = \int [p(\mathbf{x}|\omega_i) - p(\mathbf{x}|\omega_j)] \ln \frac{p(\mathbf{x}|\omega_i)}{p(\mathbf{x}|\omega_j)} d\mathbf{x}$$

基于熵的可分性准则

- 后验概率$P(\omega_i|\mathbf{x})$反映了特征x刻画类别i的有效性
- 两个极端例子
  - 如果后验概率对于所有的类别都一样，如$P(\omega_i|\mathbf{x})=\frac{1}{c}$，则说明该特征对类别没有任何鉴别性；
  - 如果后验概率对于类别i为1，而对其他类别均为0，也即$P(\omega_i|\mathbf{x})=1$，则说明特征x非常有效(informative)；
    - 对于某个给定特征，样本属于各类的后验概率越平均，越不利于分类；如果越集中于某一类，则越有利于分类。
- 因此，我们可利用后验概率的信息熵来度量特征对类别的可分性

基于熵的可分性准则

- 信息熵可用来衡量一个随机事件发生的不确定性，不确定越大，信息熵越大
- 香农熵：

$$H(x) = -\sum_{i=1}^{c} P(w_i|x) \log_2 P(w_i|x)$$

- 平方熵：

$$H(x) = 2\left[1 - \sum_{i=1}^{c} P^2(w_i|x)\right]$$

子集搜索

- 从给定的含有特征数目D的特征集合中选择最优的特征子集
  - 候选特征数目可能很多
  - 特征的维数可能很高（特征变换、特征降维等）
  - 子集搜索是典型的组合问题 (组合爆炸)

$$\text{特征的组合数目} \frac{D!}{(D-d)!d!}$$

子集搜索

- 根据子集搜索策略不同：
  - 穷举法：搜索所有的特征组合。
  - 前向搜索策略：在特征选择的迭代过程中，每次只加入一个新特征，并对得到的特征子集进行评价，直到增加特征不会优于增加特征之前的子集为止。
  - 后向搜索策略：从完整特征集合开始，每次迭代去掉一个无关特征，直到去掉特征后会导致剩余特征子集的性能显著减少。
  - 双向搜索策略：将前向特征选择和后向特征选择相结合。
  - 随机搜索策略：使用随机策略进行子集搜索，然后对得到的特征子集进行评价。

特征选择方法

- 对于给定的特征集合 $\{a_1, a_2, \cdots, a_d\}$ 确定其中的最优特征子集
- 经典的特征选择方法可划分为：
  - 暴力搜索（穷举法）
  - 包裹式特征选择方法
  - 过滤式特征选择方法
  - 分支定界法(最优方法)
  - 嵌入式特征选择方法

包裹式选择

- 包裹式特征选择直接把最终将要使用的学习器的性能作为特征子集的评价准则；
- 目的是为给定学习器选择最有利于其性能、“量身定做”的特征子集；
- 通常采用交叉验证来评价选取的特征子集的好坏
- 交叉验证：
  - 留出法（hold-out cross validation）
  - K折交叉验证(k-fold cross validation)
  - 留一法(Leave-one-out)

包裹式选择

- 基于前向搜索的包裹式特征选择算法

1. Initialize $\mathcal{F} = \emptyset$.
2. Repeat {
   (a) For $i = 1, \ldots, n$ if $i \notin \mathcal{F}$, let $\mathcal{F}_i = \mathcal{F} \cup \{i\}$, and use some version of cross validation to evaluate features $\mathcal{F}_i$. (I.e., train your learning algorithm using only the features in $\mathcal{F}_i$, and estimate its generalization error.)
   (b) Set $\mathcal{F}$ to be the best feature subset found on step (a).
   }
3. Select and output the best feature subset that was evaluated during the entire search procedure.

包裹式选择

Algorithm 5: AdaBoost for Feature Selection

Given: $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, where $\mathbf{x}_i \in \mathbb{X}$, $y_i \in \{1, -1\}$

Init: $W_1(i) = 1/n$

for $t = 1, \ldots, T$

- Normalize the weights W_t
- $\forall$ feature j, train h_j and obtain its error rate ϵ_j
- Choose the classifier h_t with the lowest ϵ_t
- Set $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$ and $\alpha_t = -\log \beta_t$
- Updating the data distribution

$$W_{t+1}(i) = \begin{cases} W_t(i)\beta_t & \text{if } h_t(x_i) = y_i \\ W_t(i) & \text{otherwise} \end{cases}$$

end

Return: a strong classifier $H(\mathbf{x}) = sgn\left(\sum_{t=1}^{T} \alpha_t h_t(\mathbf{x})\right)$

一个具体应用

- 实时人脸检测 (Viola&Jones, CVPR'01)

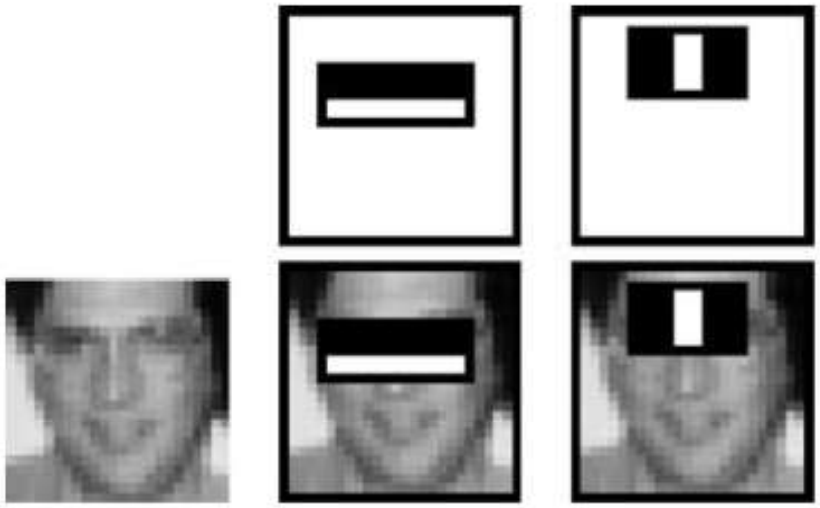
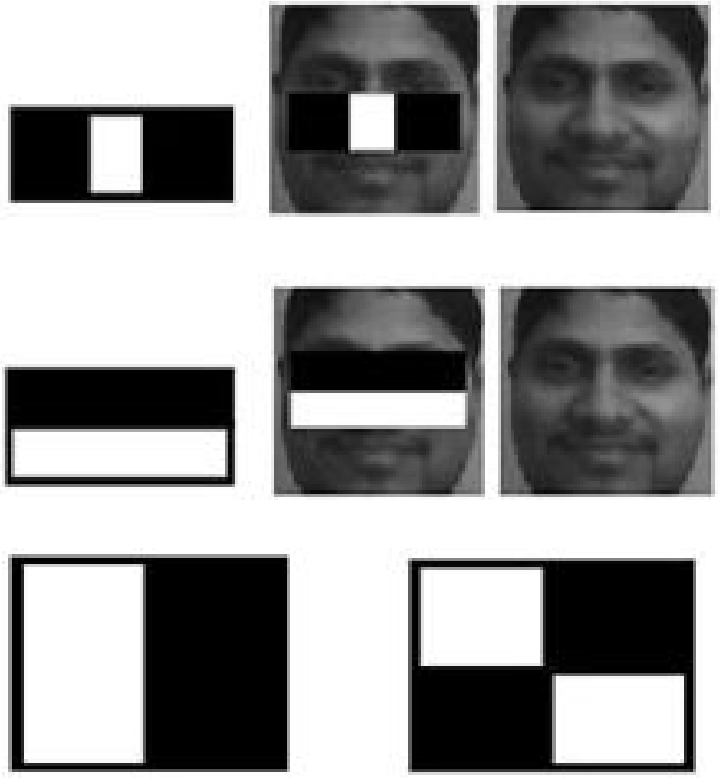


Figure 5. The first and second features selected by AdaBoost. The two features are shown in the top row and then overlayed on a typical training face in the bottom row. The first feature measures the difference in intensity between the region of the eyes and a region across the upper cheeks. The feature capitalizes on the observation that the eye region is often darker than the cheeks. The second feature compares the intensities in the eye regions to the intensity across the bridge of the nose.



包裹式选择分析

- 包裹式特征选择方法对分类器的基本要求：
  - 分类器能够处理高维特征向量；
  - 在特征维度很高、样本个数较少时，分类器依然可以取得较好的效果。
- 启发式方法，无法保证得到最优子集
- 需频繁调用学习算法进行候选特征子集的评价
- 通常特征选择效果很好，但计算量很大

过滤式选择

- 过滤式选择先对数据集进行特征选择，然后再训练学习器。特征选择过程与后续学习器无关；
- 基本思想：首先定义一个评价函数，来度量某个给定特征与类别标签之间的相关度；最后选取具有最大相关度的k个特征作为选择结果；
- 核心是如何定义特征的评价函数；
- 启发式特征选择方法，无法获得最优子集；
- 与包裹式选择方法相比，计算量降低了很多；

单独特征选择法

单独特征选择法

输入：数据集、特征集合、待选择特征个数k

输出：已选择特征集合Φ

1、计算每个特征的性能评价判据；

2、根据特征的性能评价判据，对所有特征进行排序

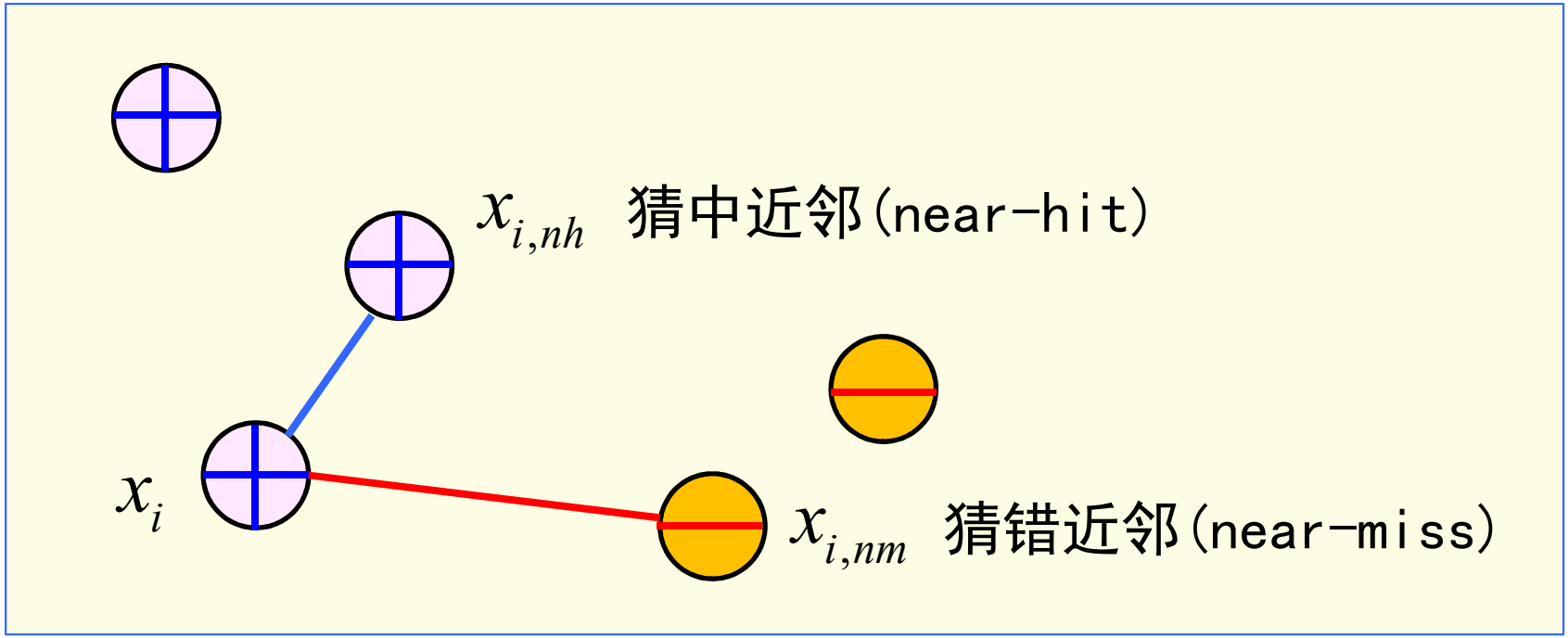
3、取前k个特征加入特征选择集合Φ

基本假设：单独作用时性能最优的特征，它们组合起来性能也是最优的。

- 没有考虑特征的组合特性
- 假设太强，和很多实际情况不符
- 简单、计算快速

Relief (Relevant Features)算法

- 一种著名的过滤式特征选择方法
- 设计了一个“相关统计量”来度量特征的重要性
- 考虑二类分类问题



Relief (Relevant Features)算法

- 定义如下的对应于属性*j*的相关统计量

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

其中：x_i^j 表示第*i*个样本点的第*j*个属性

$diff(\cdot)$用于度量样本点属性值的差异

$$diff(x_a^j, x_b^j) = \left|x_a^j - x_b^j\right|$$

可见，若样本点与其猜中近邻在属性*j*上的距离小于其猜错近邻的距离，则说明属性j对区分同类和异类样本有益，于是增大属性j所对应的统计量分量。反之亦然

Relief (Relevant Features)算法

- 为估计相关统计量，Relief只需在数据集的采样上而不是整个数据集上进行，因此Relief的时间开销随采样次数以及原始特征数线性增长，运行效率很高；
- Relief是为二分类问题设计的，其扩展变体Relief-F能处理多分类问题。

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

其中 p_l 为第 l 类样本在数据集 D 中所占的比例.

过滤式选择

- “装袋”法
  - 顺序前进特征选择法
  - 顺序后退特征选择法
  - 前向-后向特征选择法
- 启发式选择方法

顺序前进特征选择法

顺序前进特征选择法

输入：数据集、特征集合、待选择特征个数k

输出：已选择特征集合Φ

1、计算每个特征的性能评价判据，选择最优的特征加入特征选择集合Φ

2、对于每个剩余特征，分别计算它**与已选择特征组合在一起的性能评价判据**

3、根据评价判据，**选择最优的特征**加入特征选择集合Φ

4、重复2-3步，直到已选择特征数量达到预定数量

顺序前进特征选择法

- 优点：
  - 相比单独特征选择法更鲁棒一些，考虑了一定的特征组合因素；
  - 计算速度依然很快；
- 缺点：
  - 特征一旦入选，就无法被剔除。

顺序后退特征选择法

顺序后退特征选择法

输入：数据集、特征集合、待选择特征个数k

输出：已选择特征集合Φ

1、将所有特征加入特征选择集合Φ

2、对于已选择特征集合Φ中的每一个特征，计算去掉该特征后剩余特征的性能评价判据

3、根据评价判据，选择使得判据最优所对应的特征，将其从特征选择集合Φ中去除

4、重复2-3步，直到已选择特征数量达到预定数量

顺序后退特征选择法

- 顺序后退特征选择缺点：
  - 自顶向下的方法，相对顺序前进法（自底向上），**计算量更大**，因为大部分计算都在高维空间（特征个数从最大逐渐较少）；
  - 特征一旦剔除，就无法再加入。

前向-后向特征选择法

- **增l减r特征选择法（$l > r$）**
- **基本思想**：为了使得已选择或者剔除的特征**有机会重新被考虑**，将顺序前进特征选择法和顺序后退特征选择法相结合。
  - 采用l次顺序前进特征选择，选择l个特征；
  - 对已选择特征集合，采用r次顺序后退特征选择；
  - 重复上述特征选择和剔除过程，直到选择到所需数目的特征。
- 顺序前进步骤和顺序后退步骤**可以对调**，此时，$r > l$，**对应减r增l法**。

包裹式选择 vs 过滤式选择

- **过滤式**特征选择方法：
  - 先对数据集进行特征选择，然后再训练分类器；**特征选择过程与分类单独进行**，特征选择评价判据**间接**反应分类性能。
- **包裹式**特征选择方法：
  - **特征选择过程与分类性能相结合**，特征评价判据为分类器性能。为**给定分类**方法，选择最有利于其性能的特征子集。
  - 直接以分类性能为准则的特征选择方法

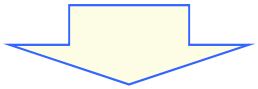
分支定界法（Branch and Bound）

- 最优方法之一：穷举法
  - 从给定的n个特征中，挑选出最优特征子集，若采用穷举法，需要遍历2^n个子集。当n很大时，该方法计算量巨大$O(2^n)$。
  - 能否有更聪明的搜索方法？
- 最优方法之二：分支定界法（Branch and Bound）
  - 基本思想：将所有可能的特征选择组合以树的形式进行表示，采用分枝定界方法对树进行搜索，使得搜索过程尽早达到最优解，而不必搜索整个树。
  - 特征子集的评价函数需满足一定的前提条件

分支定界法（Branch and Bound）

- 前提条件：
  - 特征子集的评价准则判据对特征集合具有单调性，特征数目增多时，判据值不会减少

$$\overline{\mathcal{X}}_1 \supset \overline{\mathcal{X}}_2 \supset \ldots \supset \overline{\mathcal{X}}_k$$

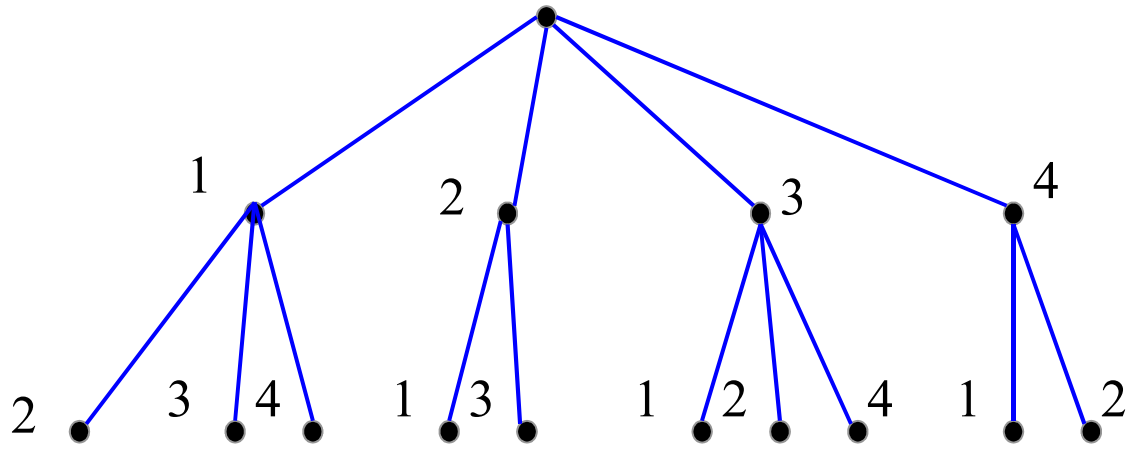


$$J\left(\overline{\mathcal{X}}_1\right) \geq J\left(\overline{\mathcal{X}}_2\right) \geq \ldots \geq J\left(\overline{\mathcal{X}}_k\right)$$

  - 基于KL散度和基于信息熵的评价准则满足上述要求

分支定界法（Branch and Bound）

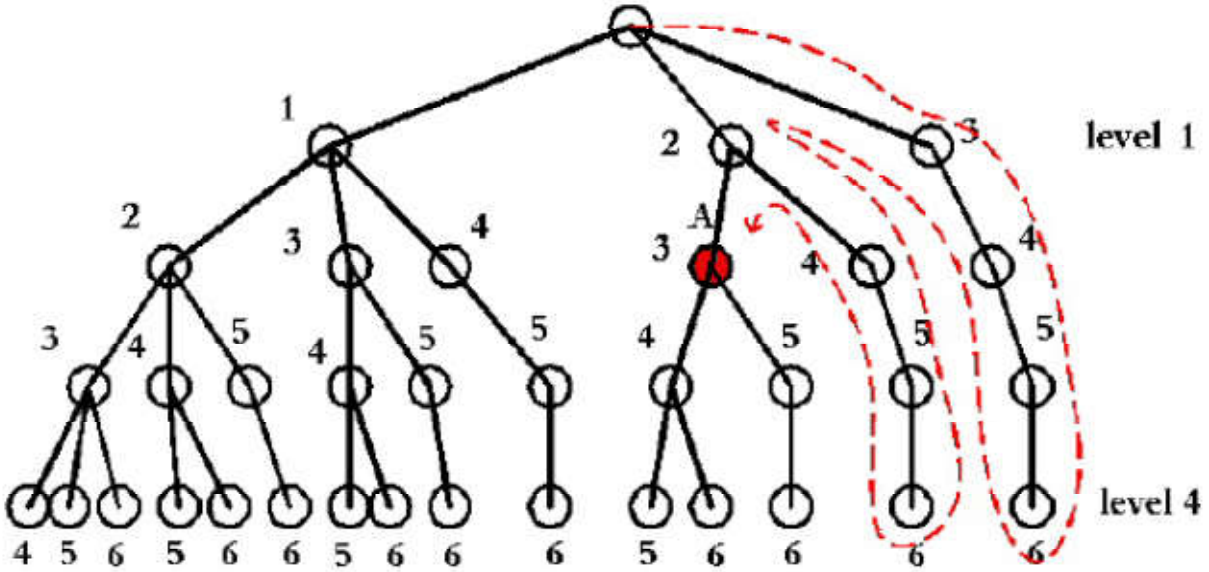
- 特征子集的树表示
  - 根节点包含全部特征；
  - 每一个节点，在父节点基础上去掉一个特征；
  - d维特征选择k个特征，需要d-k层达到需要的特征数量，即树的深度为d-k



分支定界法（Branch and Bound）

- 树的生长过程（树的构造）
  - 1. 将**所有特征组成根节点**，根节点记为第0层
  - 2. 对于第1层，分别**计算去掉上一层节点单个特征后的评价判据值的减少量**，根据**减少量从大到小**对上一层节点中的特征排序，并把相应的特征去掉、按照**从左到右**的顺序生成第1层的节点
  - 3. 对于第2层，针对**上一层最右侧**的节点，重复第2步
  - 4. 依次类推，直到第d-k层，到达叶子结点，记录对应的准则判据值B，同时记录对应的特征选择集合
- 从第一个叶子结点开始，对树进行**回溯**

分支定界法（Branch and Bound）



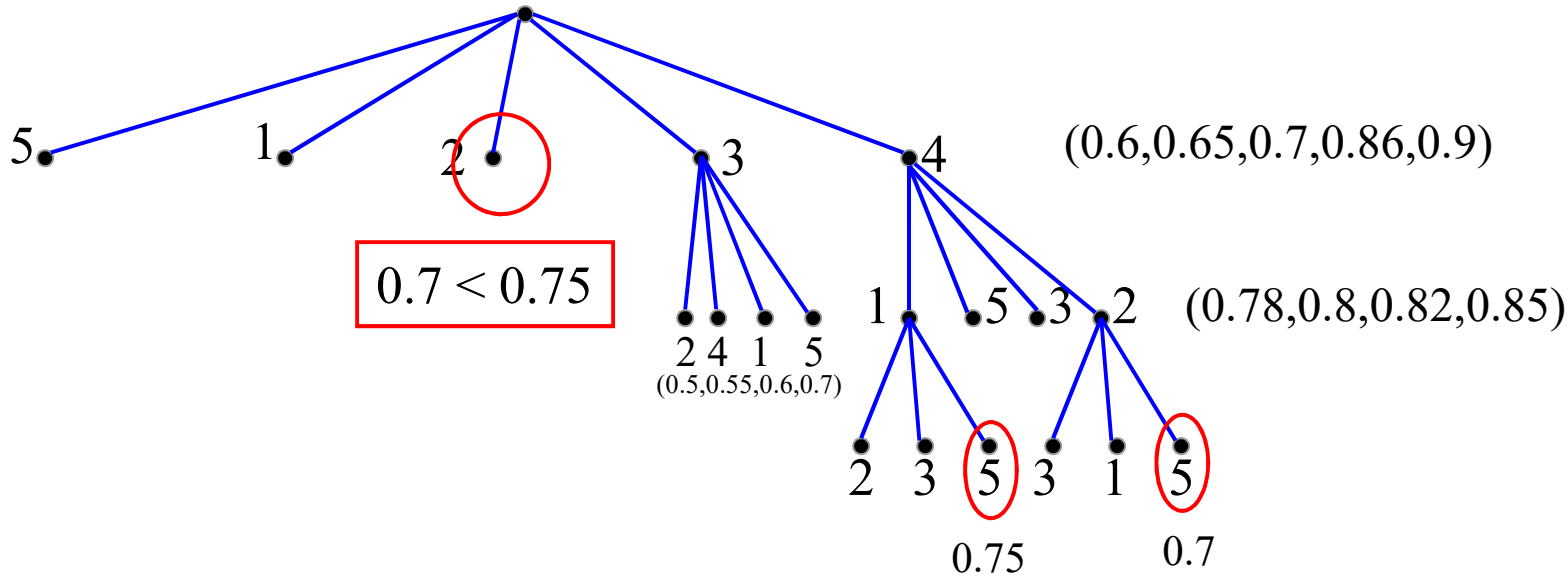
- **树的生长过程**，使得**同一层左侧节点**对应的特征集合**判据值小于右侧节点**；
- **评价判据的单调性**，使得**下层节点**对应的特征集合**判据值小于上层节点**
- 回溯算法根据这两个特性，在对树搜索的时候进行分枝限界（从右到左、从上到下）

分支定界法（Branch and Bound）

- **树的回溯**过程
  - 1. 从某个节点开始往上面对树进行回溯，**直到遇见分枝节点**，搜索分枝节点**最右侧未处理**的一个分枝
  - 2. 对于该分枝下每一层节点，计算对应特征集合的判据值V
  - 3. 如果V<B，根据判据的单调性，该节点以下的判据值**都小于V，无需往下搜索**，往上回溯，转到第1步；否则继续往下搜索，转第2步；若**遇见叶节点**，转第4步
  - 4. 计算叶节点对应特征集合的判据值B′，如果B′ > B，**更新B和相应的特征选择集合**。转第1步；否则，算法终止

举个例子

示例：共有5个特征，选择2个



{1，3},0.7

{2，3},0.75

嵌入式选择

- 将分类器学习与特征选择融为一体，分类器训练过程自动完成了特征选择。
- 考虑线性回归问题

$$f(x) = w^T x$$

  - w_i等于0，第i个特征对分类没有影响
  - w_i不等于0，第i个特征属于有用特征
- **基本思路**：在学习w的时候，对w进行限制，使得w不仅能满足训练样本的误差要求，同时使得w**中非零元素尽可能少**（只使用少数特征）

基于L1范数的特征选择

- 如何衡量一个向量中非零元素的个数？

该向量的L0范数！

L1范数：$\|w\|_1 = |w_1| + |w_2| + \cdots + |w_d|$

$$\min_w \sum_i (w^T x_i - y_i)^2 + \lambda \|w\|_1$$

均方损失

稀疏约束

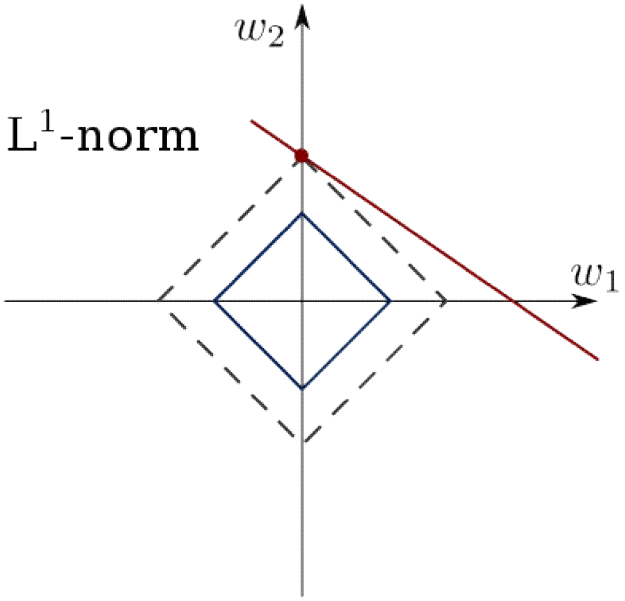
LASSO (Least Absolute Shrinkage and Selection Operation)

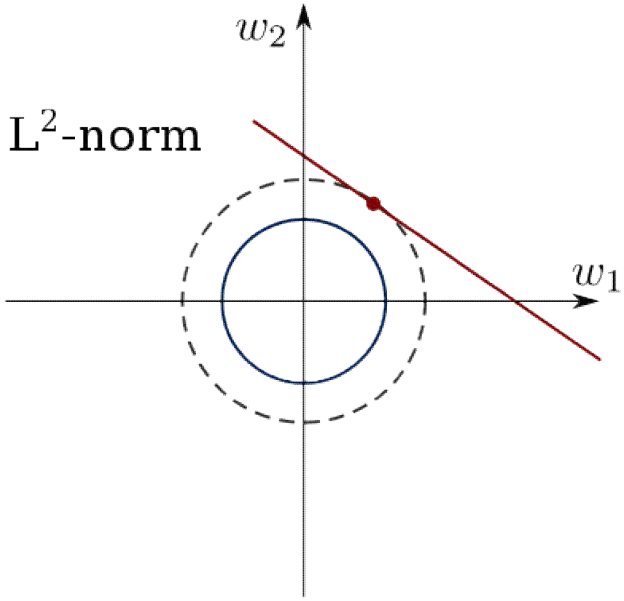
基于L1范数的特征选择

- 不能直接设置最终选择特征的个数k；
- 通过设置正则化系数λ来隐式控制k；
- λ值越大，模型越关注稀疏性，得到的非零系数个数越少；反之，非零稀疏个数越多；
- 可以设置一个选择特征个数的上限，通过设置不同λ值，得到满足要求的特征。

L1范数 vs L2范数

- 最小化L1范数可得到稀疏解
- 什么是稀疏性？
  - 解向量的大部分位置值为零，只有少数部分位置值不为零





LASSO

- 基本形式：

$$\min_{\beta_0,\beta}\left\{\frac{1}{N}\sum_{i=1}^{N}(y_i-\beta_0-x_i^T\beta)^2\right\}\ \text{subject to}\ \sum_{j=1}^{p}|\beta_j|\le t$$

  - 样本点为p维向量 $x_i:=(x_1,x_2,\ldots,x_p)^T$
  - N: 样本点的总数目
  - t为指定的自由参数，用于控制正则化的程度

向量形式

- 将上式可进一步写成对应的向量形式

$$\min_{\beta_0,\beta}\left\{\frac{1}{N}\|y-\beta_0-X\beta\|_2^2\right\} \text{ subject to } \|\beta\|_1 \le t$$

其中 $X_{ij}=(x_i)_j$

- 注意到给定 β 后，β_0 的优化有闭式解

$$\beta_0=\overline{y}-\overline{x}^T\beta$$

我们可将其带入原目标函数，进一步简化目标函数的形式。

向量形式

- 记 $\hat{\beta}_0 = \bar{y} - \bar{x}^T\beta$，于是

$$y_i - \hat{\beta}_0 - x_i^T\beta = y_i - (\bar{y} - \bar{x}^T\beta) - x_i^T\beta$$
$$= (y_i - \bar{y}) - (x_i - \bar{x})^T\beta$$

因此只需要把数据进行规范化，LASSO可写为如下更为简洁的向量形式：

$$\min_{\beta\in\mathbb{R}^p}\left\{\frac{1}{N}\|y - X\beta\|_2^2\right\} \text{ subject to } \|\beta\|_1 \le t$$

$$\min_{\beta\in\mathbb{R}^p}\left\{\frac{1}{N}\|y - X\beta\|_2^2 + \lambda\|\beta\|_1\right\}$$

Lagrangian形式

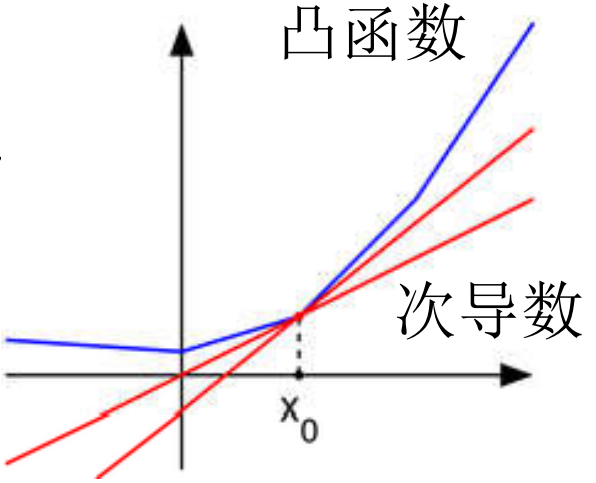
LASSO的求解

- 求解LASSO的几种常用方法：
  - 次梯度下降法(subgradient descent methods)
    - 梯度下降法的推广
  - 最小角度回归(least-angle regression, LARS)
    - 与LASSO模型密切相关
  - 近端梯度下降法(proximal gradient descent methods)
    - 目前非常流行，效果也是最好的
  - 半二次切分(half-quadratic splitting)

次梯度下降法

- 次导数(subderivative)的定义
  对于实值凸函数$f:\mathbb{R}^n \to R$，定义该函数在x_0点处的次导数为满足下式的所有实数c：

$$f(x)-f(x_0)>c(x-x_0)$$



- 一维实值函数的次导数对应一个闭区间$[a,b]$

$$a=\lim_{x\to x_0^-}\frac{f(x)-f(x_0)}{x-x_0} \qquad b=\lim_{x\to x_0^+}\frac{f(x)-f(x_0)}{x-x_0}$$

- 对于不可导的点，函数的次导数不唯一；而对于可导的点，函数的次导数唯一。

次梯度下降法

- 同理，对多元函数可相应的定义函数的次梯度
- 次梯度(subgradient)
  - 一个凸函数在给定点x_0处的次梯度为所有满足下式的向量的集合：$f(x) - f(x_0) \geq v \cdot (x - x_0)$
- 次梯度下降法
  - 对于凸函数$f(x)$，次梯度下降迭代计算下式：

$$x^{(k+1)} = x^{(k)} - \alpha_k g^{(k)}$$

其中，$g^{(k)}$ 为函数 $f(x)$在 $x^{(k)}$ 处的次梯度

  - 由于并非所有的次梯度方向都为下降方向，因此下降过程需要保存当前最优的函数值

$$f_{\text{best}}^{(k)} = \min\{f_{\text{best}}^{(k-1)}, f(x^{(k)})\}$$

近端梯度下降法

- 考虑如下一般的形式

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) + \lambda \| \boldsymbol{x}\|_1$$

若$f(x)$可导且其导数满足L-Lipschitz条件：

$$\left\|\nabla f(\boldsymbol{x}') - \nabla f(\boldsymbol{x})\right\|_2^2 \leqslant L\left\|\boldsymbol{x}' - \boldsymbol{x}\right\|_2^2 \quad (\forall \boldsymbol{x}, \boldsymbol{x}')$$

则可将$f(x)$ 在x_k 附件利用泰勒展开式近似为：

$$\begin{aligned}\hat{f}(\boldsymbol{x}) &\simeq f(\boldsymbol{x}_k) + \langle \nabla f(\boldsymbol{x}_k), \boldsymbol{x} - \boldsymbol{x}_k\rangle + \frac{L}{2}\|\boldsymbol{x} - \boldsymbol{x}_k\|^2 \\ &= \frac{L}{2}\left\|\boldsymbol{x} - \left(\boldsymbol{x}_k - \frac{1}{L}\nabla f(\boldsymbol{x}_k)\right)\right\|_2^2 + \text{const}\end{aligned}$$

近端梯度下降法

- 上式的最小值在如下 x_{k+1} 出获得：

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \frac{1}{L}\nabla f(\boldsymbol{x}_k)$$

- 类似的，考虑L1范数正则化下，每一步的迭代应为：

$$\boldsymbol{x}_{k+1} = \arg\min_{\boldsymbol{x}} \frac{L}{2}\left\|\boldsymbol{x} - \left(\boldsymbol{x}_k - \frac{1}{L}\nabla f(\boldsymbol{x}_k)\right)\right\|_2^2 + \lambda \|\boldsymbol{x}\|_1$$

- 上式的优化可通过引入一个辅助变量来实现

$$\boldsymbol{z} = \boldsymbol{x}_k - \frac{1}{L}\nabla f(\boldsymbol{x}_k)$$

近端梯度下降法

- 于是，优化问题转变为：

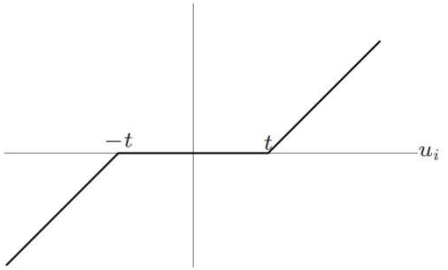
$$\boldsymbol{x}_{k+1} = \arg\min_{\boldsymbol{x}} \frac{L}{2} \| \boldsymbol{x} - \boldsymbol{z} \|_2^2 + \lambda \| \boldsymbol{x} \|_1$$

- 该优化问题可转化成$\boldsymbol{n}$个独立的1D优化问题

$$x_{k+1}^i = \arg\min_x \tfrac{L}{2}\left(x - z^i\right)^2 + \lambda |x|$$

- 上述优化问题有闭式解：

$$x_{k+1}^i = \begin{cases} z^i - \lambda/L, & \lambda/L < z^i \\ 0, & \left|z^i\right| \leqslant \lambda/L \\ z^i + \lambda/L, & z^i < -\lambda/L \end{cases}$$



软阈值算子(soft-thresholding operator)

半二次切分法

$$\min_{\beta\in\mathbb{R}^p}\left\{\frac{1}{N}\|y-X\beta\|_2^2+\lambda\|\beta\|_1\right\}$$

$\alpha=\beta$

$$\min_{\alpha,\beta}\frac{1}{N}\|Y-X\alpha\|_2^2+\gamma\|\alpha-\beta\|_2^2+\lambda\|\beta\|_1$$

坐标下降法

固定α $\min_{\beta}\gamma\|\alpha-\beta\|_2^2+\lambda\|\beta\|_1$

$\min_{\alpha}\frac{1}{N}\|Y-X\alpha\|_2^2+\gamma\|\alpha-\beta\|_2^2$ 固定β

LASSO的解释

- 几种回归方法的对比
  - 线性回归 $\min_{\beta} \frac{1}{N} \|Y - X\beta\|_2^2$
  - 岭回归(ridge regression) $\min_{\beta} \frac{1}{N} \|Y - X\beta\|_2^2 + \lambda \|\beta\|_2^2$
  - LASSO $\min_{\beta} \frac{1}{N} \|Y - X\beta\|_2^2 + \lambda \|\beta\|_1$
  - Best Subset Selection $\min_{\beta \in \mathbb{R}^p} \left\{ \frac{1}{N} \| y - X\beta \|_2^2 + \lambda \| \beta \|_0 \right\}$

LASSO的解释

- 仅考虑变量正交情形，即$X^T X = I$
- LASSO的解为：

$$\hat{\beta}_j = S_{N\lambda}(\hat{\beta}_j^{\text{OLS}}) = \hat{\beta}_j^{\text{OLS}} \max\left(0, 1 - \frac{N\lambda}{|\hat{\beta}_j^{\text{OLS}}|}\right)$$

$$\text{where } \hat{\beta}^{\text{OLS}} = (X^T X)^{-1} X^T y$$

- 岭回归的解为：$\hat{\beta}_j = (1 + N\lambda)^{-1} \hat{\beta}_j^{\text{OLS}}$
- 最优子集选择的解为：

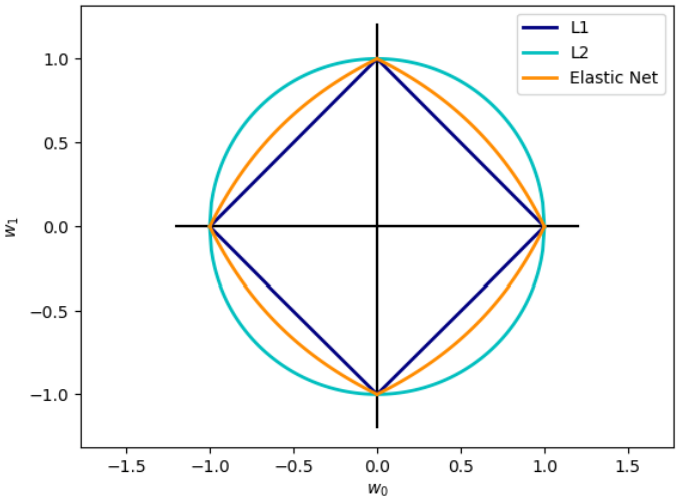
$$\hat{\beta}_j = H_{\sqrt{N\lambda}}\left(\hat{\beta}_j^{\text{OLS}}\right) = \hat{\beta}_j^{\text{OLS}} \mathrm{I}\left(\left|\hat{\beta}_j^{\text{OLS}}\right| \geq \sqrt{N\lambda}\right)$$

LASSO的解释

- LASSO可看作最优子集选择回归问题的凸松弛

$$\min_{\beta \in \mathbb{R}^p} \left\{ \frac{1}{N} \|y - X\beta\|_2^2 + \lambda \|\beta\|_0 \right\}$$

$$\min_{\beta \in \mathbb{R}^p} \left\{ \frac{1}{N} \|y - X\beta\|_2^2 + \lambda \|\beta\|_1 \right\}$$



- Elastic Net

$$\min_{\beta \in \mathbb{R}^p} \left\{ \|y - X\beta\|_2^2 + \lambda_1 \|\beta\|_1 + \lambda_2 \|\beta\|_2^2 \right\}$$

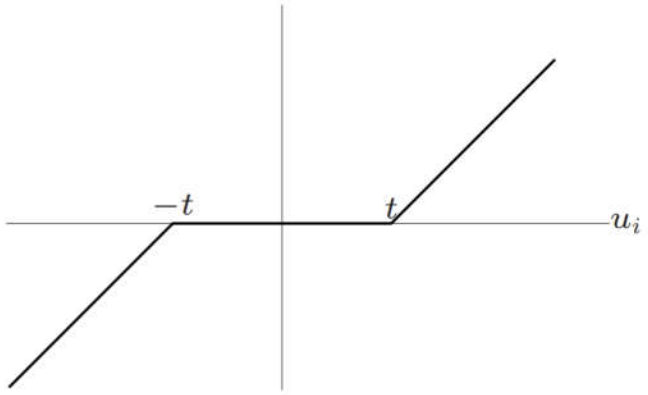
LASSO的解释

$$\hat{\beta}_j = S_{N\lambda}(\hat{\beta}_j^{\text{OLS}}) = \hat{\beta}_j^{\text{OLS}} \max\left(0, 1 - \frac{N\lambda}{|\hat{\beta}_j^{\text{OLS}}|}\right)$$

$$\text{where } \hat{\beta}^{\text{OLS}} = (X^T X)^{-1} X^T y$$

其中，$S_{N\lambda}(\cdot)$ 为soft-thresholding operator

线性算子



联想：神经网络中的激活函数

稀疏表示与字典学习

- 把数据集考虑成一个矩阵，其每行对应一个样本，每列对应于一个特征
- 稀疏性能带来好处
  - 特征选择：矩阵中的许多列与当前学习任务无关，因此通过特征选择剔除这些列，则学习器训练过程仅需在较小的矩阵上进行，学习任务将变得简单；
  - 稀疏矩阵：矩阵中存在很多零元素，这些零元素并不是以整行、整列的形式存在。数据的这种稀疏性，使得对应的问题常变得线性可分，且存储负担大大降低，学得的模型的可解释性也会提高；

字典学习

- 如果给定的数据集 D 是稠密的，那么能否将其转化为“稀疏表示”(Sparse Representation)形式？
- 字典学习(Dictionary Learning)
  - 为普通稠密表达的样本找到合适的字典，将样本转化为合适的稀疏表示形式，从而使得学习任务得以简化、模型复杂度得以降低；
  - 字典学习与稀疏编码(sparse coding)
    - 前者更侧重于学得字典的过程，后者更侧重于对样本进行稀疏表达的过程。这两者通常在同一个优化求解过程中完成。

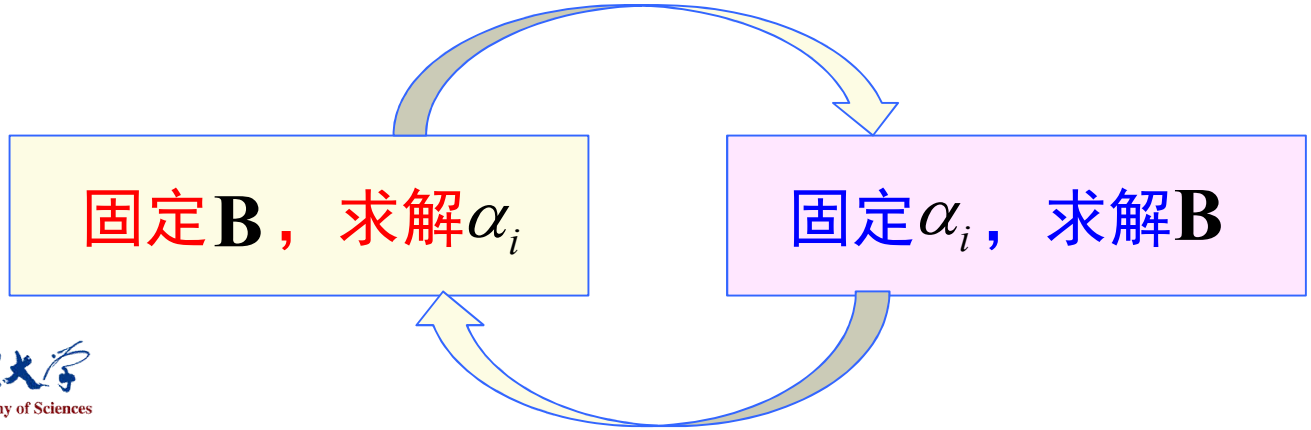
字典学习

- 给定数据集$\{x_1, x_2, \cdots, x_m\}$，字典学习最简单的形式为

$$\min_{\mathbf{B},\alpha_i} \sum_{i=1}^{m} \left\| x_i - \mathbf{B}\alpha_i \right\|_2^2 + \lambda \sum_{i=1}^{m} \left\| \alpha_i \right\|_1$$

其中，$\mathbf{B} \in \mathbb{R}^{d \times k}$ 为字典矩阵，k称为字典的词汇量，通常由用户指定，$\alpha_i \in \mathbb{R}^k$ 则是样本$x_i \in \mathbb{R}^d$的稀疏表示

- 与LASSO相比，字典学习显然更复杂
- 求解时采用变量交替优化策略



K-SVD算法

- K-SVD算法是求解字典学习的常用算法
- K-SVD算法是K-means聚类算法的推广形式
- K-means聚类算法回顾

Given an initial set of k means $m_1^{(1)},\ldots,m_k^{(1)}$

Assignment step:

$$S_i^{(t)} = \left\{x_p : \left\|x_p - m_i^{(t)}\right\|^2 \le \left\|x_p - m_j^{(t)}\right\|^2 \ \forall j, 1 \le j \le k\right\},$$

where each x_p is assigned to exactly one $S^{(t)}$, even if it could be assigned to two or more of them.

Update step: Calculate the new means to be the centroids of the observations in the new clusters.

$$m_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{x_j \in S_i^{(t)}} x_j$$

K-SVD算法 vs K-means算法

- 从字典学习的角度来看K-means聚类算法
  - 通过最近邻来寻找数据点的最优表示字典

$$\min_{D,X}\{\|Y-DX\|_F^2\} \qquad \text{subject to } \forall i, x_i = e_k \text{ for some } k.$$

数据点的稀疏编码仅有一个非零值

  - 一般的字典学习

$$\min_{D,X}\{\|Y-DX\|_F^2\} \qquad \text{subject to} \quad \forall i\ , \|x_i\|_0 \le T_0$$

数据点的稀疏编码有多个非零值

K-SVD算法

$$\min_{D,X}\{\|Y-DX\|_F^2\} \qquad \text{subject to} \quad \forall i\ , \|x_i\|_0 \le T_0$$

- 算法基本步骤：
  - 1）稀疏编码：固定字典 D，计算最优的系数矩阵 X
  - 2）字典更新：固定系数矩阵 X，更新字典 D 。更新时，每次仅更新字典一列，而保持其他列固定。

稀疏编码步骤

- 固定字典，求样本点的稀疏编码，也即

$$\min_{X} \|Y - DX\|_F^2 \quad subject\ to \quad \forall i, \|x_i\|_0 \le T_0$$

$$\min_{\{x_i\}} \sum_{i=1}^{m} \|y_i - Dx_i\|_2^2 \quad subject\ to \quad \forall i, \|x_i\|_0 \le T_0$$

$$\min_{x_i} \|y_i - Dx_i\|_2^2 \quad subject\ to \quad \forall i, \|x_i\|_0 \le T_0$$

可用orthogonal matching pursuit (OMP)算法求解

L1范数松弛

$$\min_{x_i} \|y_i - Dx_i\|_2^2 + \lambda \|x_i\|_1$$ LASSO

字典更新步骤

- 固定样本点的编码系数矩阵，更新字典
- 为减少字典更新的计算复杂度，每次仅更新字典的一列

$$\|Y - DX\|_F^2 = \left\| Y - \sum_{j=1}^{K} d_j x_T^j \right\|_F^2$$

$$= \left\| \left(Y - \sum_{j \neq k} d_j x_T^j \right) - d_k x_T^k \right\|_F^2$$

$$= \|E_k - d_k x_T^k\|_F^2$$

where x_T^k denotes the k-th row of X.

调整 d_k 和 x_T^k，使得该误差最小

字典更新步骤

- 调整d_k和x_T^k，使得该误差最小：

$$\left\| E_k - d_k x_T^k \right\|_F^2$$

- 可采用矩阵的SVD分解

$$E_k = \mathbf{U} \sum \mathbf{V}^T = \sum_{i=1}^{n} \lambda_i u_i v_i^T$$

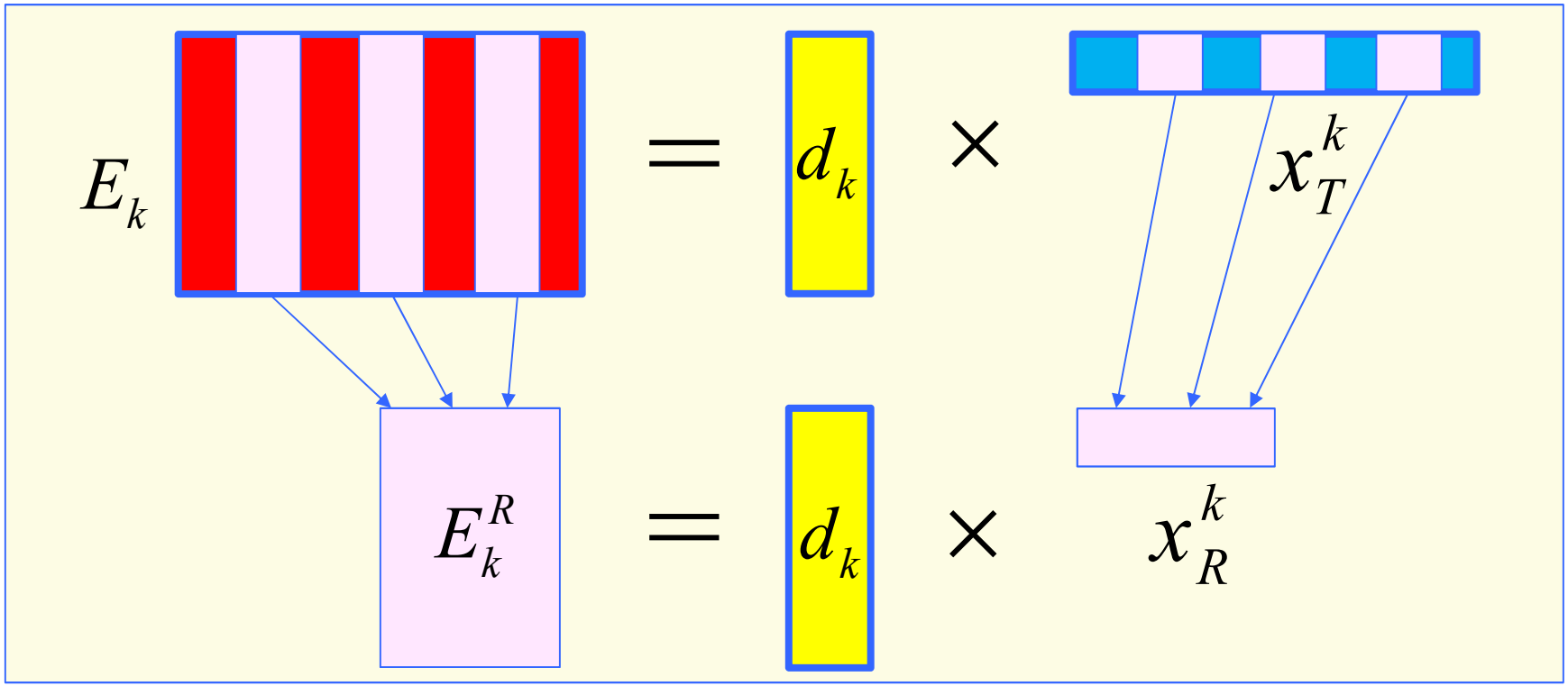
- 问题：对上式直接进行SVD分解，未考虑稀疏性约束，得到的x_T^k可能不是稀疏的。

字典更新步骤

- 解决办法：
  - x_T^k仅保留非零元素
  - E_k则仅保留对应于x_T^k 非零元素的列
  - 然后再进行SVD分解
- 定义 $\omega_k = \{i \mid 1 \le i \le N, x_T^k(i) \ne 0\}$
- 引入一个列选择矩阵Ω_k, 大小为 $N \times |\omega_k|$，满足 $\Omega_k\left(i, \omega_k(i)\right) = 1$, 而其他元素为零。于是

$$\left\| E_k \Omega_k - d_k x_T^k \Omega_k \right\|_F^2 = \left\| E_k^R - d_k x_R^k \right\|_F^2$$

字典更新步骤



$$\left\| E_k\Omega_k - d_k x_T^k \Omega_k \right\|_F^2 = \left\| E_k^R - d_k x_R^k \right\|_F^2$$

K-SVD算法

Task: Find the best dictionary to represent the data samples $\{\mathbf{y}_i\}_{i=1}^N$ as sparse compositions, by solving

$$\min_{\mathbf{D},\mathbf{X}}\left\{\| \mathbf{Y}-\mathbf{D}\mathbf{X} \|_F^2\right\} \quad \text{subject to} \quad \forall i, \|\mathbf{x}_i\|_0 \le T_0$$

Initialization : Set the dictionary matrix $\mathbf{D}^{(0)} \in \mathbb{R}^{n\times K}$ with ℓ^2 normalized columns. Set $J=1$.

Repeat until convergence (stopping rule):

- *Sparse Coding Stage*: Use any pursuit algorithm to compute the representation vectors $\mathbf{x}_i$ for each example $\mathbf{y}_i$, by approximating the solution of

$$i=1,2,\ldots,N, \quad \min_{\mathbf{x}_i}\left\{\|\mathbf{y}_i-\mathbf{D}\mathbf{x}_i\|_2^2\right\} \quad \text{subject to } \|\mathbf{x}_i\|_0 \le T_0$$

Michal Aharon, et al., *K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation*, IEEE Trans. On Signal Processing, 2006.

K-SVD算法

- *Codebook Update Stage*: For each column $k = 1, 2, \ldots, K$ in $\mathbf{D}^{(J-1)}$, update it by
  - Define the group of examples that use this atom, $\omega_k = \{i \mid 1 \leq i \leq N,\ \mathbf{x}_T^k(i) \neq 0\}$.
  - Compute the overall representation error matrix, $\mathbf{E}_k$, by

$$\mathbf{E}_k = \mathbf{Y} - \sum_{j \neq k} \mathbf{d}_j \mathbf{x}_T^j$$

  - Restrict $\mathbf{E}_k$ by choosing only the columns corresponding to ω_k, and obtain $\mathbf{E}_k^R$.
  - Apply SVD decomposition $\mathbf{E}_k^R = \mathbf{U}\mathbf{\Delta}\mathbf{V}^T$. Choose the updated dictionary column $\tilde{\mathbf{d}}_k$ to be the first column of $\mathbf{U}$. Update the coefficient vector $\mathbf{x}_R^k$ to be the first column of $\mathbf{V}$ multiplied by $\mathbf{\Delta}(1,1)$.
- Set $J = J + 1$.

Michal Aharon, et al., *K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation*, IEEE Trans. On Signal Processing, 2006.

字典学习与压缩感知

- 信号压缩（采样）与传输

$$y = \Phi x$$

一般有 $n \ll m$

长度为n的采样后信号　　测量矩阵　　长度为m的离散信号

- 信号重建
  - 给定测量值y 和测量矩阵，重建原始信号x 通常很难，因为对应的方程是一个欠定方程
- 假设存在某个线性变换Ψ，使得

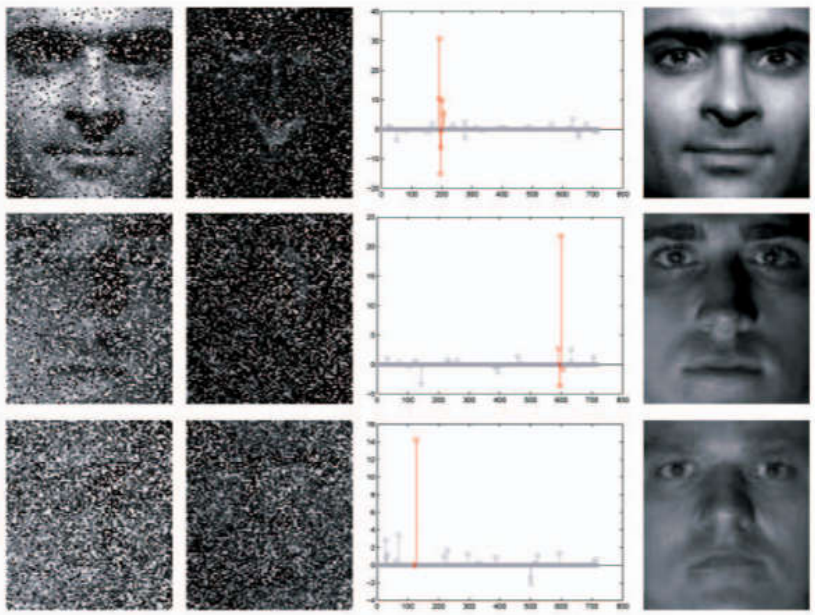
$$y = \Phi x = \Phi \Psi s = \mathbf{A}s$$

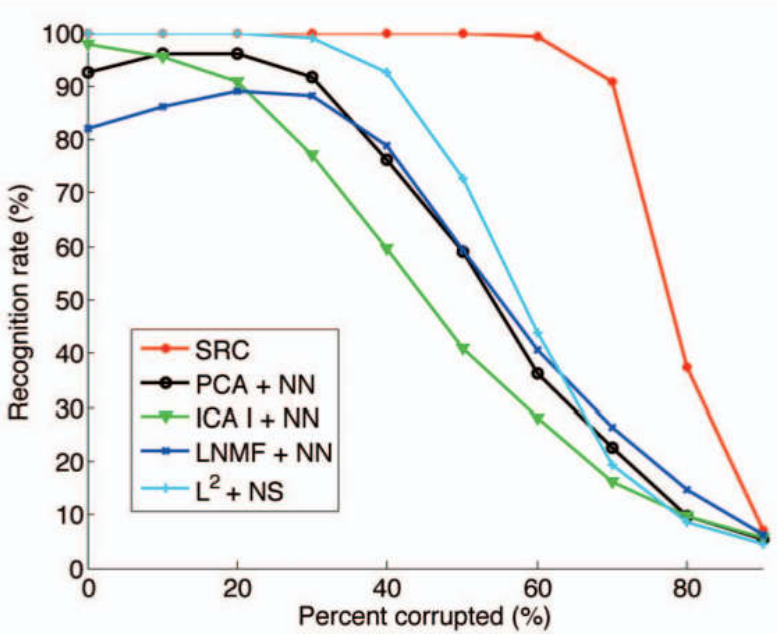
字典学习与压缩感知

$$y = \mathbf{\Phi} x = \mathbf{\Phi}\mathbf{\Psi} s = \mathbf{A} s$$

- 恢复x的问题转化为恢复s的问题
- 如果s具有稀疏性，则这个问题能很好的得到解决！这里的$\mathbf{A}$作用类似于字典，能将信号转换为稀疏表示
- 与特征选择、稀疏表示不同，压缩感知关注的是如何利用信号本身所具有的稀疏性，从部分观测样本中恢复原信号。通常包括“感知测量”和“重构恢复”两个部分。

举个例子





(a) (b) (c) (d) (e)

Percent corrupted	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%
Recognition rate	100%	100%	100%	100%	100%	100%	99.3%	90.7%	37.5%	7.1%

Fig. 11. **Recognition under random corruption.** (a) Test images y from Extended Yale B, with random corruption. Top row: 30 percent of pixels are corrupted. Middle row: 50 percent corrupted. Bottom row: 70 percent corrupted. (b) Estimated errors $\hat{e}_1$. (c) Estimated sparse coefficients $\hat{x}_1$. (d) Reconstructed images y_r. SRC correctly identifies all three corrupted face images. (e) The recognition rate across the entire range of corruption for various algorithms. SRC (red curve) significantly outperforms others, performing almost perfectly upto 60 percent random corruption (see table below).

J. Wright, *et al.*, *Robust Face Recognition via Sparse Representation*, TPAMI 2009.

本节内容回顾

- 特征子集评价准则
- 特征选择方法
  - 过滤式选择
  - 包裹式选择
  - 嵌入式选择
- LASSO及其求解方法
- 稀疏表示与字典学习
  - K-SVD算法

参考文献

- 周志华，《机器学习》
- Andrew Ng, CS229 Lecture notes: Part VII Regularization and model selection
- https://en.wikipedia.org/wiki/Lasso_(statistics)

Thank All of You!

(Questions?)

孟高峰

gfmeng@nlpr.ia.ac.cn

时空数据分析与学习课题组(STDAL)

中科院自动化研究所· 模式识别国家重点实验室