Lasso是Least Absolute Shrinkage and Selection Operator的简称，是一种采用了L1正则化（L1-regularization)的线性回归方法，采用了L1正则会使得部分学习到的特征权值为0，从而达到稀疏化和特征选择的目的。

本文从最基本的Lasso开始介绍，包括数学形式以及几何意义等，然后介绍其经典的几种解法，以及一些相关的问题，最后介绍Lasso的不足和改进之处，以及Lasso的应用等等。主要会包括以下几个方面：

Lasso & Ridge Regression(岭回归)
正则化几何意义和贝叶斯解释
Forward Selection & Forward Stagewise & Least Squares Boosting
Least Angle Regression(LARS，最小角回归)
Lasso三种求解方法：闭式解、LARS、CD
ElasticNet解决Lasso存在的问题 & LARS-EN

1 Lasso & Ridge Regression(岭回归)

在考虑一般的线性回归问题，给定n个数据样本点 $\{(x_1,y_1),(x_2,y_2),\cdots,(x_n,y_n)\}$ ，其中每个 $x_i \in R^d$ 是一个d维的向量，即每个观测到的数据点是由d个变量的值组成的，每个 $y_i \in R$ 是一个实值。现在要做的是根据观察到的数据点，寻找到一个映射 $f:R^d \rightarrow R$ ，使得误差平方和最小，优化目标为：

$$\beta^*,\beta_0^* = argmin_{\beta,\beta_0}\frac{1}{n}\sum_{i=0}^{n}(y_i-\beta^T x_i-\beta_0)^2$$

其中， $\beta \in R^d, \beta_0 \in R$ 是需要优化的系数。一般来说 β_0 可以看作是一个偏置(bias)，现在我们先来看看偏置项如何处理，假设现在固定住 β 的值，那么利用一阶导数求最优 β_0 。接下来对上面的损失关于 β_0 求导得到：

$$\frac{1}{n}\sum_{i=0}^{n}\left(y_i-x_i^T\beta-\beta_0\right)=0 \;\;\Rightarrow\;\; \beta_0=\frac{1}{n}\sum_{i=0}^{n}y_i-\frac{1}{n}\sum_{i=0}^{n}x_i^T\beta=\overline{y}-\overline{x}^T\beta$$

将得到的结果代入原优化目标得到：

$$\beta^* = argmin_{\beta}\frac{1}{n}\sum_{i=0}^{n}\left((y_i-\overline{y})-\beta^T(x_i-\overline{x})\right)^2$$

从上面式子可以看出，假如我们事先对数据进行标准化（中心化），即每个样本数据减去均值，从而得到零均值的数据样本，此时做线性回归就可以不使用偏置。下面为了方便介绍，我们假定给定的n个数据样本点 $\{(x_1,y_1),(x_2,y_2),\cdots,(x_n,y_n)\}$ 是零均值的，即 $\sum_{i=1}^{n}x_i=0$ ，那么线性回归的优化目标就可以记为：

$$\beta^* = argmin_\beta \frac{1}{n}\sum_{i=0}^{n}\left(y_i - \beta^T x_i\right)^2$$

上面也可以表示为矩阵形式，记 $X = [x_1; x_2; \cdots; x_n]^T$ ，这里把每个数据点 x_i 当作列向量，那么 $X \in R^{n\times d}$ ，记 $y = (y_1, y_2, \cdots, y_n)^T$ ，那么矩阵形式的优化目标为：

$$\beta^* = argmin_\beta \frac{1}{n}\|y - X\beta\|_2^2$$

上面介绍了基本的线性回归问题，由于有d个变量，所以称之为Multiple Linear Regression，这里不要和Multivariate Linear Regression混淆了，后者指的是同时拟合多个输出值，而不是单个输出，即 y_i 不再是一个实数值，而是一个向量。

一般来说，回归问题是一个函数拟合的过程，那么我们希望模型不要太复杂，否则很容易发生过拟合现象，所以我们要加入正则化项，而不同的正则化项就产生了不同的回归方法，其中以Ridge Regression (岭回归）和Lasso最为经典，前者是加入了L2正则化项，后者加入的是L1正则化项。下面分别给出其优化目标 。

Ridge Regression的优化目标为：

$$\beta^* = argmin_\beta \frac{1}{n}\|y - X\beta\|_2^2 + \lambda\|\beta\|_2^2$$

Lasso的优化目标为：

$$\beta^* = argmin_\beta \frac{1}{n}\|y - X\beta\|_2^2 + \lambda\|\beta\|_1$$

2 正则化几何意义和贝叶斯解释

既然上文谈到了正则化项，那么下面来解释一下L2与L1两种正则化的区别。首先，L2正则化是在优化目标中加入了参数的L2-norm(范数)项，即 $\|\beta\|_2^2 = \sum_{j=1}^{d}\beta_j^2$ ；而L1正则化项则是加入了L1-norm项，即 $\|\beta\|_1 = \sum_{j=1}^{d}|\beta_j|$ 。这两者有什么区别呢?

首先，我们来说明一下Ridge Regression的等价优化问题，这里是将L2正则化项从目标函数中移除，转而变为约束条件，将 β 限制在一个半径为t的超球里面：

$$\beta^* = argmin_\beta \frac{1}{n}\|y - X\beta\|_2^2$$
$$s.t. \quad \|\beta\|_2^2 \le t$$

为了说明Constrained form和Penalty form的等价性，我们用Lagrangian乘子，将上面带有约束的问题转化为无约束问题，即为：

$$L(\beta,\lambda)=\frac{1}{n}\|y-X\beta\|_2^2+\lambda\|\beta\|_2^2-\lambda t$$

因为我们要求解的问题是线性回归，是凸优化问题，所以给定 λ,t 之后，可以看出 $L(\beta,\lambda)$ 就是带有L2正则化项的Ridge Regression，至于后面的常数项 $-\lambda t$ 和 β 没有关系，所以有无皆可。由于是凸优化问题，由强对偶性可以得到两者等价。

同样地，对于Lasso的带约束形式的优化问题为:

$$\beta^*=argmin_\beta\frac{1}{n}\|y-X\beta\|_2^2$$
$$s.t.\quad \|\beta\|_1\le t$$

所以，下面就可以放出一张非常经典的图了，在d = 2 的情况下解释了L2正则化与L1正则化的不同之处，见Figure 1(图片来自网络)，图中左边解释的是L1正则化的几何意义，右边是L2正则化。图中椭圆形的彩色线是优化目标关于参数 $\beta=(w^1,w^2)^T$ 的等高线，假设没有约束条件，那么最小值是椭圆的中心点，但是由于加入了正则化项，相当于是对参数 β 施加了约束，其中左边L1正则化将参数限制在一个菱形中，而L2正则化则将参数限制在一个圆形区域中。那么从图中可以看出，L1正则化施加的约束会使得最优值在菱形顶点处取得，即 $w^1=0$ ；而右边L2正则化项则没有这种倾向。

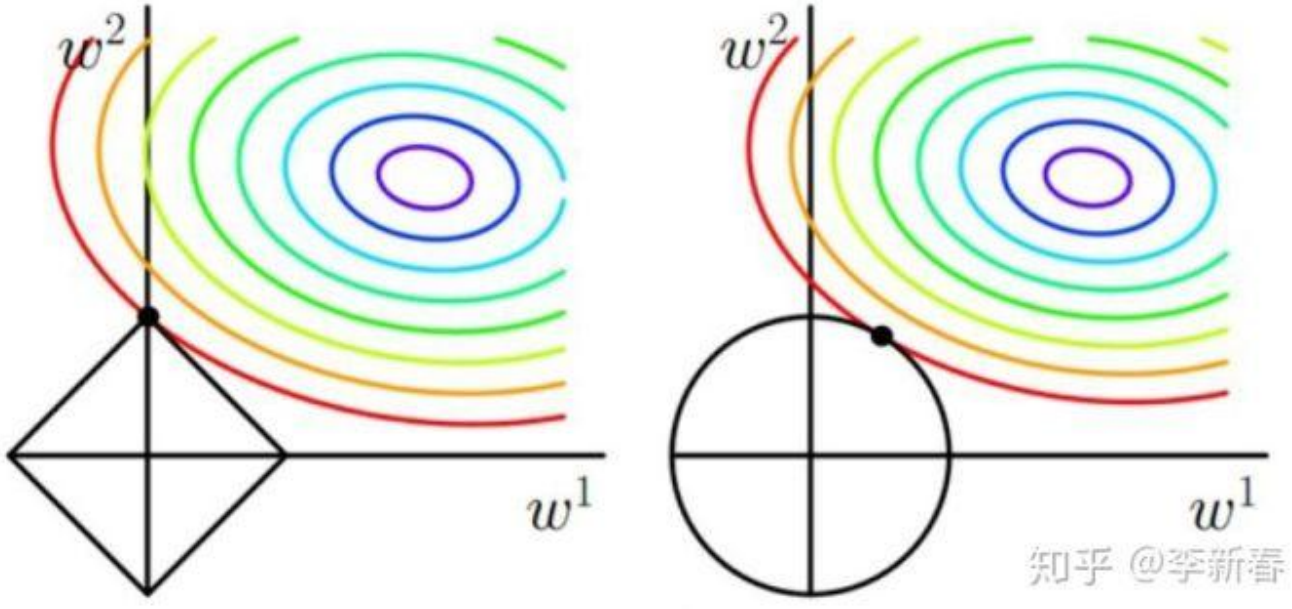

Figure 1 : L1与L2正则化的几何意义

上面的图从几何意义上解释了L1与L2正则化的区别，同时这也解释了L1与L2最大的不同：L1可以带来稀疏的结果，即L1会使得部分参数为零。这样的好处是什么呢？一方面，可以用来选择特征，一方面可以用来降维压缩数据等等。

那么，介绍到这里，L1正则化总是和稀疏挂钩，那么L2正则化呢，L2正则化做了什么事情？其实，和L2正则化挂钩的则是Weight Decay(权值衰减)。下面来简单说一下，考虑一般的优化问题：

$$J(X,y;\beta)=L(X,y;\beta)+\frac{1}{2}\lambda\|\beta\|_2^2$$

利用梯度下降来求解问题，得到：

$$\beta_{t+1}=\beta_t-\eta\nabla J(X,y;\beta)=\beta_t-\eta(\lambda\beta_t+\nabla L(X,y;\beta))=(1-\eta\lambda)\beta_t-\eta\nabla L(X,y;\beta))$$

所以可以看到，第t+1步的参数在第t步的参数前乘以了 $1-\eta\lambda$ ，所以会使得权重趋向于零，即Weight Decay的过程。

上面对比了L1正则化和L2正则化的区别，同样地，这也是Ridge Regression和Lasso的区别。正则化可以降低模型复杂度，减少模型过拟合的风险，在回归问题中的解释是：当特征之间存在高度相关关系的时候，假设有两个特征高度负相关，那么不带正则化的回归问题可能会赋予二者近似相等的很大权重，这样加权起来的结果仍然较小，但是由于权重很大，就导致了过拟合问题。Ridge Regression会倾向于在相关特征之间均匀分布权重，Lasso则倾向于从相关特征中选取出一个，其余特征权值衰减为零。另外，正则化的目的也可以理解为使模型权重之间的方差较小，从Figure 1中可以看出。

当然，正则化也可以通过贝叶斯角度来进行解释，下面还是以回归问题为例，采用贝叶斯角度进行解释。在贝叶斯学派中，任何事物都是有概率的，任何随机变量包括参数都要假设服从一定的分布，比如对于回归问题，我们假设给定参数 β 以及自变量 x_i 后，观察到的值 y_i 服从高斯分布 $p(y_i|x_i,\beta)=\mathcal{N}(\beta^Tx_i,\delta)$ ，并且我们假定参数 β 也会服从一个分布 $p(\beta)$ ，即先验prior，那么我们要求的回归问题就变成了最大化后验概率(MAP)，利用贝叶斯公式有：

$$\beta^*=argmax_\beta\ln\left(\prod_{i=1}^n p(y_i|x_i,\beta)p(\beta)\right)$$
$$=argmax_\beta\sum_{i=1}^n\left(\ln p(y_i|x_i,\beta)+\ln p(\beta)\right)$$

根据 y_i 服从高斯分布 $p(y_i|x_i,\beta)=\mathcal{N}(\beta^Tx_i,\delta)$ ，所以 $\ln p(y_i|x_i,\beta)\propto(y_i-\beta^Tx_i)^2$ ，那么可以看出现在已经得到了线性回归问题的目标损失部分，这是利用高斯分布来进行解决回归问题得到的，当然反过来也可以说线性回归的本质是高斯模型。

那下面就是先验部分 $\ln p(\beta)$ ，假设参数 β 也服从高斯分布，那么就可以得到Ridge Regression，假设服从拉普拉斯分布，那么将得到Lasso。这个可以很容易从高斯分布和拉普拉斯分布的表达式中看出，下面将L2和L1正则化对应的参数 β 的分布表示出来，在L2中参数 β_j 服从均值为0的高斯分布，在L1中服从均值为0的拉普拉斯分布。

$$p_{l2}(\beta_j)=\frac{1}{\sqrt{2\pi\alpha}}\exp\left(-\frac{\beta_j^T\beta_j}{2\alpha}\right)$$
$$p_{l1}(\beta_j)=\frac{1}{2\alpha}\exp\left(-\frac{|\beta_j|}{\alpha}\right)$$

需要注意的一点是，上面给出的是单个参数的分布，而不是向量参数 β 的分布，这可以通过独立性来简单说明一下： $p(\beta)=\prod_{j=1}^d p(\beta_j)$ ，从而有 $\ln p(\beta)=\sum_{j=1}^d\ln p(\beta_j)$ 。

所以，可以看出L1正则化和L2正则化与贝叶斯先验prior概率分布的关系，从而也就明白了怎么用贝叶斯概率角度来解释Ridge Regression和Lasso了。

3 Forward Selection & Forward Stagewise & Least Squares Boosting

下面就要来说一说更为有趣的事情了。前面两小节简单介绍了一下和Lasso相关的基本数学公式和几种解释，除此之外，在看论文或相关资料时，也会看到经常和Lasso共同出现的一些名词，很有意思，下面两节将分别介绍这几个有趣的概念，这一节会介绍三个名词，分别是：Forward Selection，Forward Stagewise和Least Squares Boosting。

3.1 Forward Selection

Forward Selection(前向选择）是一种变量选择的方法，即特征选择。在线性回归问题中，为了方便表述，我们记数据矩阵 $\boldsymbol{X}=[\boldsymbol{x}_1;\boldsymbol{x}_2;\cdots;\boldsymbol{x}_n]^T$ 的另外一种形式为 $\boldsymbol{X}=[\boldsymbol{f}_1;\boldsymbol{f}_2;\cdots;\boldsymbol{f}_d]$ ，不难看出，前者是n个样本点的表示，后者是d个维度的表示，每个维度 $\boldsymbol{f}_j$ 代表了矩阵的第j列，目标值还是用 $\boldsymbol{y}=(\boldsymbol{y}_1,\boldsymbol{y}_2,\cdots,\boldsymbol{y}_n)^T$ 来表示。Forward Selection的步骤大致如下：

Step 1 : 从d个特征中选择和目标值最为相关的一维，采用内积计算相关性，假设是第k个特征，即 $\boldsymbol{k}=\boldsymbol{argmax}_j|\boldsymbol{f}_j^T\boldsymbol{y}|$ ，然后利用这个特征进行求解回归问题，这里注意只使用了第k个特征，即单变量线性回归问题，优化问题为 $\beta_k^*=\boldsymbol{argmin}_{\beta_k}\frac{1}{n}\|\boldsymbol{y}-\beta_k\boldsymbol{f}_k\|_2^2$ ，很容易可以求解得到 $\beta_k^*=\frac{\boldsymbol{y}^T\boldsymbol{f}_k}{\boldsymbol{f}_k^T\boldsymbol{f}_k}$ 。

Step 2 : 得到第k维特征的权值 β_k^* 后，计算残差 $\hat{\boldsymbol{y}}=\boldsymbol{y}-\beta_k^*\boldsymbol{f}_k$ ，那么问题就变为了在 $\hat{\boldsymbol{X}}=[\boldsymbol{f}_1;\cdots;\boldsymbol{f}_{k-1};\boldsymbol{f}_{k+1};\cdots;\boldsymbol{f}_d]$ 上对 $\hat{\boldsymbol{y}}$ 进行线性回归。

从上面过程可以看出Forward Selection是一个非常greedy的算法，找到一个最合适的特征，然后就试图让这个特征尽可能地拟合出目标值来，拟合不出来的部分当作残差交给后续的特征继续拟合。

我写了个简单的Forward Selection来做线性回归问题，发现往往来说第一个选择的特征的权值会特别高，后面的特征的权值基本就会非常小，会趋向于零。虽然也起到了类似“稀疏化”的作用(并没有严格约束后面的权值为0)，但不是我们想要的，因为这个算法太贪婪了。之所以会产生这样的结果，是因为Forward Selection本质上就是用来做特征选择的，一般特征选择的指标可以有很多，比如相关性、相似度、信息增益(分类问题)等等。

3.2 Forward Stagewise

Forward Stagewise则是一种小心翼翼的做法，stage是“级，阶”的意思，正说明了它是一种一小步一小步往前进行试探的过程，其算法流程如下：

Step 1 : 初始 $\boldsymbol{r}=\boldsymbol{y},\beta_j=0(1\leq j\leq d)$ 。

Step 2 : 选择和 $\boldsymbol{r}$ 最为相关的一个特征k，即 $\boldsymbol{k}=\boldsymbol{argmax}_j|\boldsymbol{f}_j^T\boldsymbol{y}|$ 。

Step 3 : 更新 $\beta_k = \beta_k + \delta * sign(f_k^T r)$ 。

Step 4 : 更新残差 $r = r - \delta * sign(f_k^T r) * f_k$ 。

Step 5 : 重复上述过程一定轮数。

可以看出，本质上Forward Stagewise和Forward Selection很像，二者都是贪心地选择最相关的feature，然后更新权重和残差。唯一的不同是，Forward Selection在更新第k个特征的权重时一步到位，非常贪婪，并且后面不会再用到第k个特征；Forward Stagewise则是每次更新一小步，通过 δ 来控制学习的速率，并且特征可以反复使用。

一般来说，对于很小的步长，使用Forward Stagewise也可以产生类似Lasso的稀疏的效果，见Figure 2，所以有时可以利用Forward Stagewise的结果来当作Lasso的结果，但是有两个问题：第一，步长很难确定；第二，需要迭代的次数很难确定。

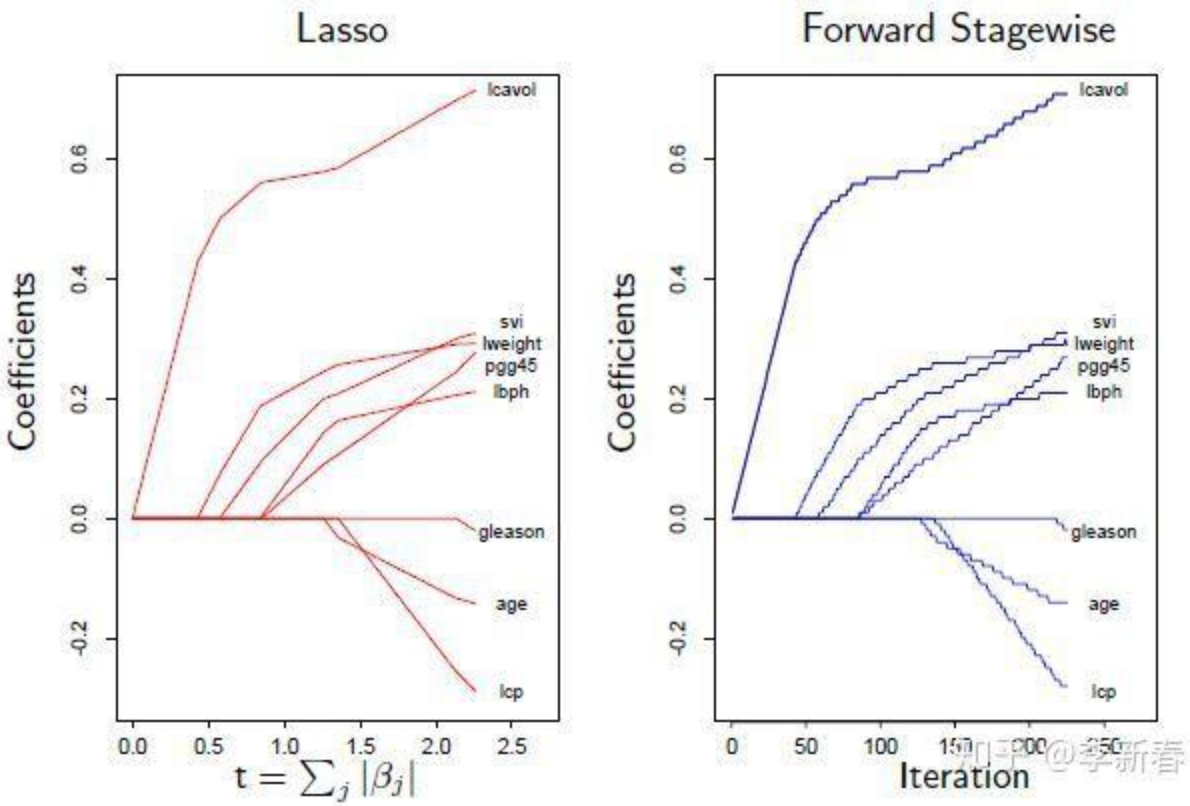

Figure 2 : Lasso和Forward Stagewise的系数

3.3 Least Squares Boosting

看到Boosting，应该就知道这大概指的是什么了。实际上，Forward Selection和Forward Stagewise都可以看作是Boosting的过程：先学习一个弱的模型，然后把没有学习到的部分，即残差，交给后面的模型去学习，最后根据弱模型的权重加权得到最终的模型。

由于Boosting经常借助树来实现，在回归问题中则借用CART树来实现，那么Least Squares Boosting的算法为：

Step 1 : 设置 $r = y, F(x) = 0$ 。

Step 2 : 在 r 上，根据数据样本 X 和算法CART树得到一个弱模型，记作 $f(x)$ 。

Step 3 : 更新 $F(x) = F(x) + \delta f(x)$ ， $r = r - \delta f(x)$ 。

Step 4 : 重复上述步骤固定轮数。

上面就是Least Squares Boosting的算法流程。

可以总结一下，上面的三个方法都是求解回归问题的算法，非常简单也很容易理解，但是对于求解Lasso，还有一个重磅型算法要介绍，那就是和Forward Selection, Forward Stagewise息息相关的LARS，下面一节将介绍何为LARS。

4 Least Angle Regression(LARS，最小角回归)

LARS，全称为Least Angle Regression，中文翻译为最小角回归，值得注意的是这里为什么多了一个“S”呢？论文中是说这个“S”暗示了Least Angle Regression与Lasso，Stagewise的关系，因为后面两个都有“S”。

下面，先直接给出LARS的步骤：

Step 1 : 初始 $r = y$ ， $\beta_j = 0(1 \leq j \leq d)$ ，ActiveSet = {}。
Step 2 : 选择和 r 最为相关的一个特征k，即 $k = argmax_j |f_j^T y|$ ，将k加入ActiveSet。
Step 3 : 沿着 $sign(f_k^T r)$ 方向更新 β_k ，即 $\beta_k = \beta_k + \delta_k * sign(f_k^T r)$ ，这里选取 δ_k 的标准是满足开始有另外一个特征s与残差 $r = r - \delta_k * sign(f_k^T r) * f_k$ 的相关性等于了特征k和残差的相关性，将s加入ActiveSet。
Step 4 : 对于当前特征k和s，沿着 f_k 与 f_s 的角平分线 $\bar{f}_{ks}$ 进行移动，即 $\beta_k = \beta_k + \delta_{ks} * sign(\bar{f}_{ks}^T r)$ ， $\beta_s = \beta_s + \delta_{ks} * sign(\bar{f}_{ks}^T r)$ ， δ_{ks} 的选择标准是满足开始有另外一个特征t与残差 $r = r - \delta_{ks} * sign(\bar{f}_{ks}^T r) * (f_k + f_s)$ 的相关性等于特征k，s与残差的相关性，将t加入ActiveSet。
Step 5 : 每次选择ActiveSet里面特征的角平分线进行更新，更新的步长的标准是满足开始有下一个特征与残差的相关性等于ActiveSet里面特征与残差的相关性。
Step 6 : 直到选择了所有特征后，或者当任意一个特征与残差的相关性都为零时结束。

上述过程是自己根据参考资料总结的，下面附上LARS算法过程的截图，希望读者能更容易理解这个过程：

1. Start with $r = y, \hat{\beta}_1, \hat{\beta}_2, \ldots \hat{\beta}_p = 0$. Assume x_j standardized.
2. Find predictor x_j most correlated with r.
3. Increase β_j in the direction of $\text{sign}(\text{corr}(r, x_j))$ until some other competitor x_k has as much correlation with current residual as does x_j.
4. Move $(\hat{\beta}_j, \hat{\beta}_k)$ in the joint least squares direction for (x_j, x_k) until some other competitor x_ℓ has as much correlation with the current residual
5. Continue in this way until all predictors have been entered. Stop when $\text{corr}(r, x_j) = 0 \; \forall \; j$, i.e. OLS solution.

Figure 3 : LARS算法描述

可以看出同Forward Selection与Forward Stagewise一样，LARS的每一步也是选择最相关的特征，然后更新参数和残差，迭代求解。三者的区别是什么呢，可以通过Figure 4看出，图片来源于经典的LARS几何意义解释（由于原图中的变量名称和本文采用的不同，所以笔者重新画了一个过程，并且将三者分别来画进行对比）。

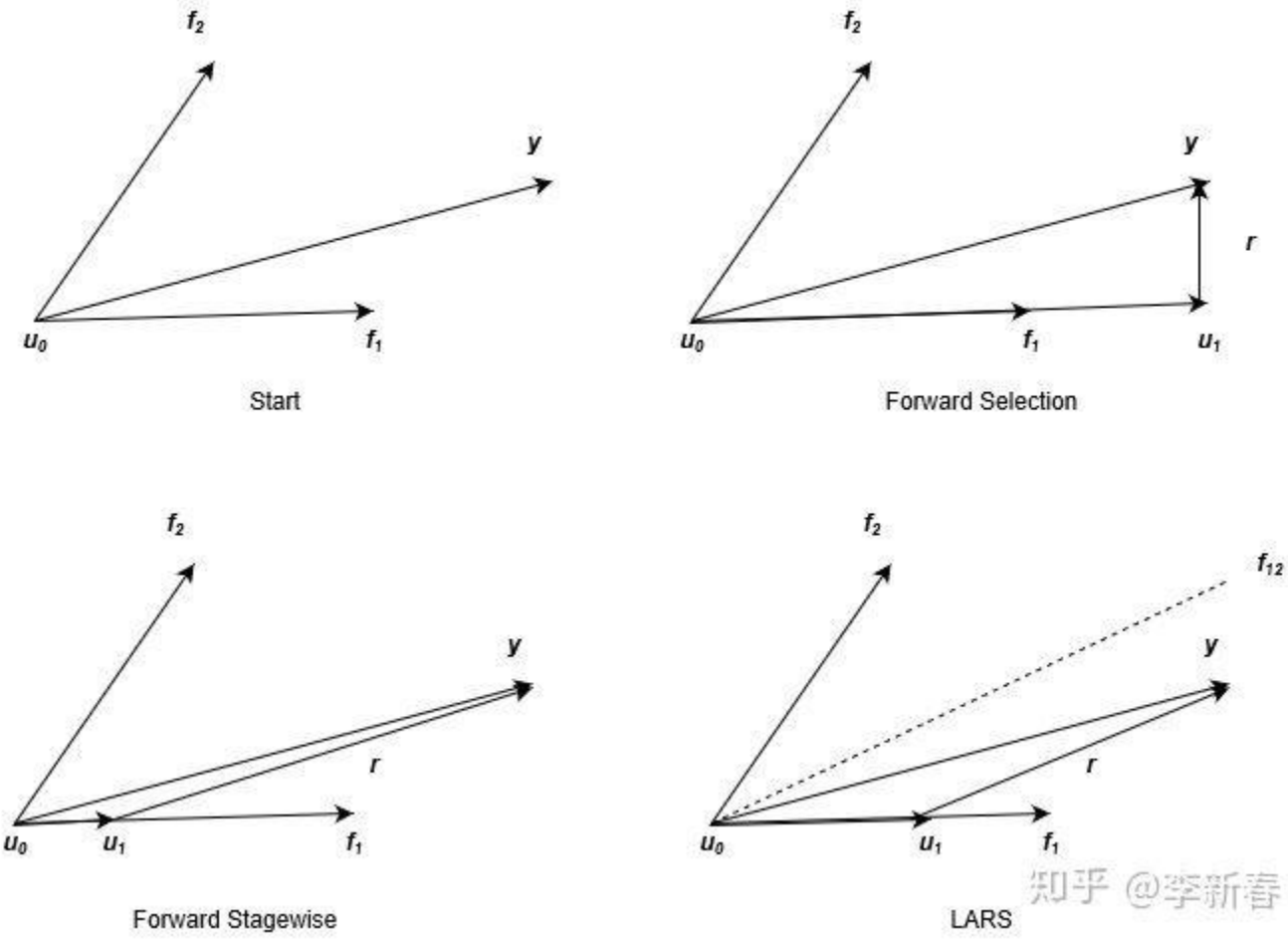

Figure 4 : Forward Selection, Forward Stagewise, LARS

在上面的图示中，我们引入了新的符号 u 表示累计更新得到的预测目标的和， $u_0 = 0$ ，下面会针对性地说明。首先，Figure 3描述了d=2时的情形，此时我们的目标就是根据 f_1, f_2 来拟合 y ，如Figure 3左上角所示。

对于Forward Selection来说，由于 f_1 与 y 更相关(表现在图中就是夹角比较小），那么根据小节3.1所述，更新过程为下面两个式子。对应于Figure 3右上角的图示，更新完后的残差 r 与特征 f_1 垂直，即不再相关，后续更新也就不会再用到 f_1 。

$$u_1 = u_0 + \frac{y^T f_1}{f_1^T f1} f_1$$
$$r = y - u_1$$

对于Forward Stagewise来说，也是会选择 f_1 的方向来更新 u ，根据小节3.2介绍的更新过程如下公式，其中 δ 是预先设定的步长，一般来说是很小的值，在Figure 3左下角所示，由于步长比较小，下一步更新时残差可能仍与 f_1 最为相关，所以可以多次利用同一个特征。

$$u_1 = u_0 + \delta f_1$$
$$r = y - u_1$$

对于LARS来说，也是会选择 f_1 的方向来更新 u ，更新方式如下公式，其中 δ_1 的选择要满足 $corr(r, f_1) = corr(r, f_2)$ ，反映在Figure 3中则是右下角，更新到残差 r 与特征 f_1, f_2 的角平分线 $\overline{f}_{12}$ 平行。

$$u_1 = u_0 + \delta_1 f_1$$
$$r = y - u_1$$

通过上面可以看出LARS和Forward Selection以及Forward Stagewise的关系，下面就仔细来推导一下LARS的数学表达，即角平分线如何求，步长 δ_k 如何求，以及下一步选择哪个特征等等(数学公式比较繁杂，不感兴趣的可以跳过)。

首先，回顾一下 $X = [f_1; f_2; \cdots; f_d]$ ，在LARS求解过程中将矩阵预处理为零均值、长度为1的向量，然后我们记ActiveSet为 $A \subseteq \{1, 2, \cdots, d\}$ ，相应的特征矩阵为 X_A ，那么这些特征的角平分线 $\overline{f}_A$ 怎么求呢?

可以这么思考，角平分线可以表示为特征的加权和，即 $\overline{f}_A = X_A w_A$ ，那么由于是角平分线，那么有:

$$X_A^T \overline{f}_A = X_A^T X_A w_A = z * 1_A$$

这里，直接使用内积而不是余弦相似度，是因为假设数据的每个特征在开始已经做了标准化(均值为0，模长为1），上面公式表示角平分线和每一个特征夹角相等，即内积相等，都等于 z ，这里 z 是一个常数， 1_A 表示全1向量 。通过上面式子又有:

$$w_A = z * (X_A^T X_A)^{-1} 1_A$$

假设 $\overline{f}_A = X_A w_A$ 是单位向量，那么有 $\|\overline{f}_A\|_2^2 = 1$ ，即：

$$z^2 1_A^T (X_A^T X_A)^{-1} X_A^T X_A (X_A^T X_A)^{-1} 1_A = 1$$
$$\Rightarrow z = \left(1_A^T (X_A^T X_A)^{-1} 1_A\right)^{-1/2}$$

所以记 $G_A = X_A^T X_A$ ， $z_A = \left(1_A^T G_A^{-1} 1_A\right)^{-1/2}$ ，那么 $w_A = z_A G_A^{-1} 1_A$ ，从而得到角平分线 $\overline{f}_A = X_A w_A$ 。

下面要在角平分线上进行更新，所以假设当前预测目标的和为 u_A ，那么更新的方法为：

$$u_{A+} = u_A + \delta_A \overline{f}_A$$

下面我们要求每个特征与残差 $r = y - u_{A+}$ 的相关系数，有：

$$C_{j+} = f_j^T (y - u_{A+}) = f_j^T (y - u_A - \delta_A \overline{f}_A) = C_j - \delta_A f_j^T \overline{f}_A$$

其中 C_j 为上一步更新结束后残差和特征的相关系数。对于 $j \in A$ ，由于ActiveSet里面的 $f_j^T \overline{f}_A$ 大小一样，即AcitiveSet里面的特征与残差的相关系数会等值衰减。另外，对于 $j \in A$ ， $f_j^T \overline{f}_A = z = z_A$ ，并且记 $C_j = C$ ，那么 $C_{j+} = C - \delta_A z_A$ 。

下面从 $A^c = \{1, 2, \cdots, d\} - A$ 中选取下一个特征 j^* 。首先对于 $j \in A^c$ ， $C_{j+} = C_j - \delta_A f_j^T \overline{f}_A = C_j - \delta_A a_j$ ，即用 $a_j = f_j^T \overline{f}_A$ 来替代一下。那么，根据LARS第4步过程， δ_A 的选择必须满足有一个特征 j 与残差的相关系数 C_{j+} 等于ActiveSet里面特征与残差的相关系数，因此有 $\exists j \in A^c,\ |C - \delta_A z_A| = |C_j - \delta_A a_j|$ ，所以得到了 δ_A 为：

$$\delta_A = min_{j \in A^c}^{+} \left\{ \frac{C - C_j}{z_A - a_j}, \frac{C + C_j}{z_A + a_j} \right\}$$

注意到，相关系数大小相等，是绝对值意义上的，既可以是负相关也可以是正相关，上面在所有 $j \in A^c$ 中及其正相关或负相关中选择最小的，并且必须是正数(步长为正数)。所以，上面最小值对应的 j 即为 j^* ，最后 $A = A \cup \{j^*\}$ ，加入ActiveSet。

总结一下，上面介绍了LARS算法数学推导的三个核心问题：更新方向ActiveSet的角平分线怎么求、步长如何选取、下一步选择哪个特征。

实际上，通过对LARS进行稍加修改即可得到Lasso和Forward Stagewise，一方面这说明了Lasso和Forward Stagewise的结果为什么会很像，另一方面这意味着可以利用修改的LARS来求解Lasso，那么后面就介绍Lasso的几种解法。

5 Lasso三种求解方法：闭式解、LARS、CD

目前为止，已经介绍了Lasso的问题形式，以及与其相关的几个概念，但是都没有涉及到如何求解Lasso。Lasso的求解有很多方法，但是较为经典的则是利用LARS来求，当然另外一种经常使用的则是坐标下降法(Coordinate Descent, CD)。

5.1 闭式解

在介绍LARS、CD求解Lasso之前，先看看Lasso的闭式解。首先，对于一般线性回归问题，即不带有正则化的线性回归问题，见第1节，有Ordinary Least Squares求解方法，记为OLS。通过简单求导计算即可得到：

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

对于Ridge Regression来说也不难，得到：

$$\beta^{Ridge} = (X^T X + n\lambda I)^{-1} X^T y$$

通过Ridge Regression的闭式解，可以发现“岭回归”中的"岭"是什么意思了，它指的就是在矩阵的对角处加上了一个大于零的常数。

下面我们来看Lasso的情况，首先求导得到：

$$\frac{2}{n}(X^T X\beta - X^T y) + \lambda \frac{\partial \|\beta\|_1}{\partial \beta} = 0$$

上面对一范数进行求导，要引入次梯度(sub gradient)，并且还要假设一个条件 $X^T X = I$ ，即任意两个特征之间正交，那么有 $\beta^{OLS} = X^T y$ ， $\beta = \beta^{OLS} - \frac{n\lambda}{2}\frac{\partial \|\beta\|_1}{\partial \beta}$ 。那么此时分情况考虑：

1）对于 $\beta_j^{OLS} > \frac{n\lambda}{2}$ ， $\beta_j = \beta_j^{OLS} - \frac{n\lambda}{2}\frac{\partial \|\beta\|_1}{\partial \beta_j}$ ，由于 $\frac{\partial \|\beta\|_1}{\partial \beta_j} \leq 1$ ，所以 $\beta_j > 0$ ，此时有 $\beta_j = \beta_j^{OLS} - \frac{n\lambda}{2}$ 。

2）对于 $\beta_j^{OLS} < -\frac{n\lambda}{2}$ ， $\beta_j = \beta_j^{OLS} - \frac{n\lambda}{2}\frac{\partial \|\beta\|_1}{\partial \beta_j}$ ，由于 $\frac{\partial \|\beta\|_1}{\partial \beta_j} \geq -1$ ，所以 $\beta_j < 0$ ，此时有 $\beta_j = \beta_j^{OLS} + \frac{n\lambda}{2}$ 。

3）对于 $-\frac{n\lambda}{2} \leq \beta_j^{OLS} \leq \frac{n\lambda}{2}$ ，用反证法，假设 $\beta_j > 0$ ，那么 $\frac{\partial \|\beta\|_1}{\partial \beta_j} = 1$ ，所以 $\beta_j = \beta_j^{OLS} - \frac{n\lambda}{2} \leq 0$ ，矛盾；同理可证 $\beta_j < 0$ 也不成立。那么， $\beta_j = 0$ 。

将各种情况考虑进来，得到 $\beta_j^{Lasso} = sign(\beta_j^{OLS})(|\beta_j^{OLS}| - \frac{n\lambda}{2})_+$ ，这就是Lasso的闭式解，可以看出，Lasso将OLS得到的系数的绝对值进行衰减 $\frac{n\lambda}{2}$ ，如果小于零了，就将其变为0，即稀疏化了系数。

下面给出一张图，表示OLS，Ridge Regression, Lasso之间系数的关系，分别在Figure 5中分别用灰色点虚线、灰色杠虚线、黑色杠虚线来表示。另外一个黑色实线是后面要介绍的ElasticNet的系数。

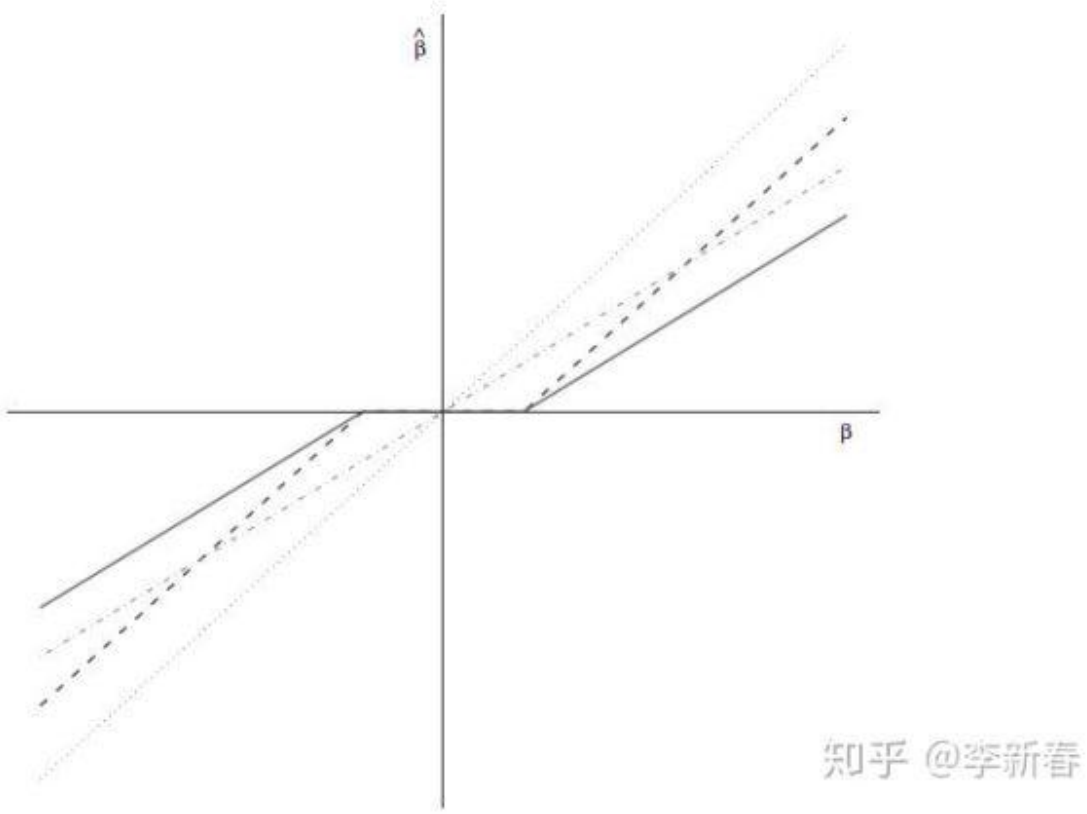

Figure 5 : beta of OLS, Ridge Regression, Lasso, ElasticNet

5.2 修改LARS求解Lasso

第四节介绍的LARS是用来求解线性回归问题的，但并不能直接拿来求解Lasso，而是要进行修改一部分才可以。下面先给出修改的方法：

在ActiveSet里面，假设有 $j \in A$ ，在更新系数的过程中 $\beta_{j+} = \beta_j + \delta_A sign(\bar{f}_A y) < 0$ ，(更新过程参考第4小节对LARS的介绍)，即有一个系数的值小于了零或大于了零，即改变了符号，将其从ActiveSet里移除即可。

这里根据论文里面给出的为什么要将其移除才能满足Lasso的约束条件，来简单说明一下。

首先，考虑Lasso的式子：

$$\frac{\partial\left(\|y-X\beta\|_2^2+\lambda\|\beta\|_1\right)}{\partial\beta}=0$$

$$\Leftrightarrow$$

$$X^T\left(y-X\beta\right)=\frac{n\lambda}{2}\frac{\partial\|\beta\|_1}{\partial\beta}$$

$$\Leftrightarrow$$

$$<r,f_j>=\frac{n\lambda}{2}sign(\beta_j)$$

即权重的符号和相关性的符号一致。而在LARS中，更新过程中相关系数的更新为 $\forall j\in A,\ C_{j+}=C-\delta_A z_A$ ，在前面介绍LARS中， C,C_{j+} 实际上表示的是相关系数的绝对值，因而在LARS中，当权重相关系数改变符号时，相关系数却不会改变符号，这不满足上面的约束。所以标准LARS得到的解不会是Lasso的解，所以要修改LARS，至于按照上述过程修改后的LARS为什么就是Lasso的解，这个问题有点复杂，看论文并没有看懂(貌似论文也就只给了一个定理)。这里省去。

论文中提到LARS的复杂度，假设LARS更新了m步，即选择了m个特征，那么时间复杂度将是 $O(m^3+nm^2)$ ，首先m会小于等于d，因为LARS的每一步都会选择一个特征。其中 m^3 是因为要求ActiveSet的角平分线，利用Cholesky分解求逆矩阵。

另外，利用LARS求解Lasso时，最多只能选择出min(n, d)个特征，这实际上也是由Lasso的性质决定的，事实上当n << d时，Lasso的结果里面最多有n个是non-zero的。下面通过Lasso的KKT条件来说明一下。首先KKT条件为：

$$X^T(y-X\beta)=\frac{n\lambda}{2}sign(\beta)$$

其中 $sign(\beta)$ 是次梯度，我们考虑系数不为零的 $\beta_J, J\subseteq\{1,2,\cdots,n\}$ ，$|sign(\beta_j)|=1, j\in J$ ，那么这部分的KKT条件为：

$$|X_J^T(y-X_J\beta_J)|=\frac{n\lambda}{2}$$

由于 $X_J^TX_J\in R^{|J|\times|J|}$ ，假若 $|J|>n$ ，那么因为：

$$rank(X_J^TX_J)\le rank(X_J)\le min\{n,d\}=n<|J|$$

考虑非齐次线性方程组 $Ax=b, A\in R^{n\times n}$ ，假设 $rank(A)=k<n, rank(A|b)\neq rank(A)$ ，此时方程组无解。应用到Lasso的KKT约束条件上，即 $|J|>n$ 是不成立的，所以 $|J|\le n$ ，即Lasso最多只能选择出min(n, d)个特征。

利用LARS的求解过程也可以说明，还是在n << d的情况下，假设现在已经选择了n个特征加入了ActiveSet里面，那么是否还可以继续选择特征呢？再选一个特征的话，这个特征就可以用前面的特征进行线性表示了，那么角平分线就没办法定义了。为什么是这样，举个简单的例子：假设二维平面上选择了三个向量，那么角平分线如何定义呢？所以可以通过LARS求解过程直观地理解为什么最后只能选择出min(n, d)个特征。

而Ridge Regression却没有这个限制，是因为Ridge Regression的KKT条件为：

$$X^T(y - X\beta) = n\lambda\beta$$
$$\Rightarrow$$
$$(X^TX + n\lambda I)\beta = X^Ty$$

可以看出由于加入了一个常数倍数的对角单位阵，则消除了Lasso里面矩阵 $rank(X^TX) < n$ 的限制。

5.3 Coordinate Descent

利用坐标下降法求解Lasso，也是求解Lasso的一种经典算法，并且具有很大的优势，因为坐标下降法特别适合那些在单个维度上有闭式解，但是在整体维度上却没有闭式解的问题。这里给出其算法流程：

随机初始化参数 $\beta_1, \beta_2, \cdots, \beta_d$
For k = 1, 2, ... , d

求解目标：

$$L(\beta_j|\beta_{-j}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \sum_{s\neq k} X_{is}\beta_s - X_{ik}\beta_k)^2 + \lambda|\beta_k| + const$$

得到 $\beta_k^{(t+1)*}$ ，求解过程参考前面介绍的Lasso闭式解求解过程，这一步会得到稀疏的系数，即 $\beta_k^{(t+1)*}$ 可能会被Shrink为0

迭代上面的步骤直到收敛

可以看出利用Coordinate Descent来求解Lasso就是将其看作d个一维变量的Lasso逐步求解，迭代直到收敛即可。由于并不涉及到矩阵分解操作，所以单步求解只需要 $O(nd)$ 的时间复杂度，唯一要确定的就是迭代的轮数。

下面稍微总结一下三种解法。闭式解需要满足正交性 $X^TX = I$ ，并且需要利用矩阵求逆 $(X^TX)^{-1}$ ，一般来说比较难满足，并且求逆太复杂；利用LARS则很简单，最多只用d步即可以完成求解；而Coordinate Descent则非常快，因为其每一步求解过程都是在一个坐标轴上进行求解单变量Lasso问题。总的来说，LARS和Coordinate Descent是最为常用的两个Lasso求解方法。

由于Lasso的目标函数是Convex的，所以只要算法可以收敛，即可得到全局最小值，但是由于Lasso不是严格凸的，所以可以有很多全局最小值，所以得到的解有时会不一样。

6 ElasticNet解决Lasso存在的问题 & LARS-EN

前文说到，当d >> n的时候，Lasso最多只能选择出n个特征，其余特征的权值全为0，那么这不是我们想要的，因为很多情况下数据样本很少，但特征维度很多。

并且即便当d < n时，假设有几个特征高度相关，这种情况下，Lasso倾向于选择其中一个作为代表，而Ridge Regression会倾向于将权重均匀分配到这些相关特征上面。事实上，这种情况下，实验表明Ridge Regression的表现会比Lasso好很多。

所以，Lasso的使用场合还是比较适合于d < n，并且特征之间相关性很小的情况下。

那么为了综合Ridge Regression和Lasso的性质，又出现了ElasticNet，其优化目标为：

$$\beta^* = argmin_\beta \frac{1}{n}\|y - X\beta\|_2^2 + \lambda_1\|\beta\|_2^2 + \lambda_2\|\beta\|_1$$
$$\lambda_1 + \lambda_2 = 1, \lambda_1, \lambda_2 \geq 0$$

即加入了L2和L1形式的正则化，由于L2正则化项的存在，即便在n << d的情况下，也可以选择多于n个特征。其求解过程也可以采用对LARS进行稍微修改得到，为什么呢？其实是因为，ElasticNet可以写成Lasso的形式。

令 $\hat{y} = [y^T, 0^T]^T \in R^{n+d}$ ， $\hat{X} = \left[X^T; \sqrt{n\lambda_1}I\right]^T \in R^{(n+d)\times d}$ ，那么有：

$$\|\hat{y} - \hat{X}\beta\|_2^2 = \|\left[(y - X\beta)^T; \sqrt{n\lambda_1}\beta^T\right]^T\|_2^2$$
$$\Rightarrow$$
$$\beta^* = argmin_\beta \frac{1}{n}\|\hat{y} - \hat{X}\beta\|_2^2 + \lambda_2\|\beta\|_1$$
$$\lambda_1 + \lambda_2 = 1, \lambda_1, \lambda_2 \geq 0$$

所以，ElasticNet相当于是在新的数据矩阵上求解Lasso问题，由于新的数据矩阵 $\hat{X} \in R^{(d+n)\times d}$ 不会出现n + d << d的情况，所以就可以选择超过n个特征，并且解法可以对LARS稍加改变即可得到，算法为LARS-EN，这里不再介绍了。

最后附上一张图Figure 6，ElasticNet的几何意义：

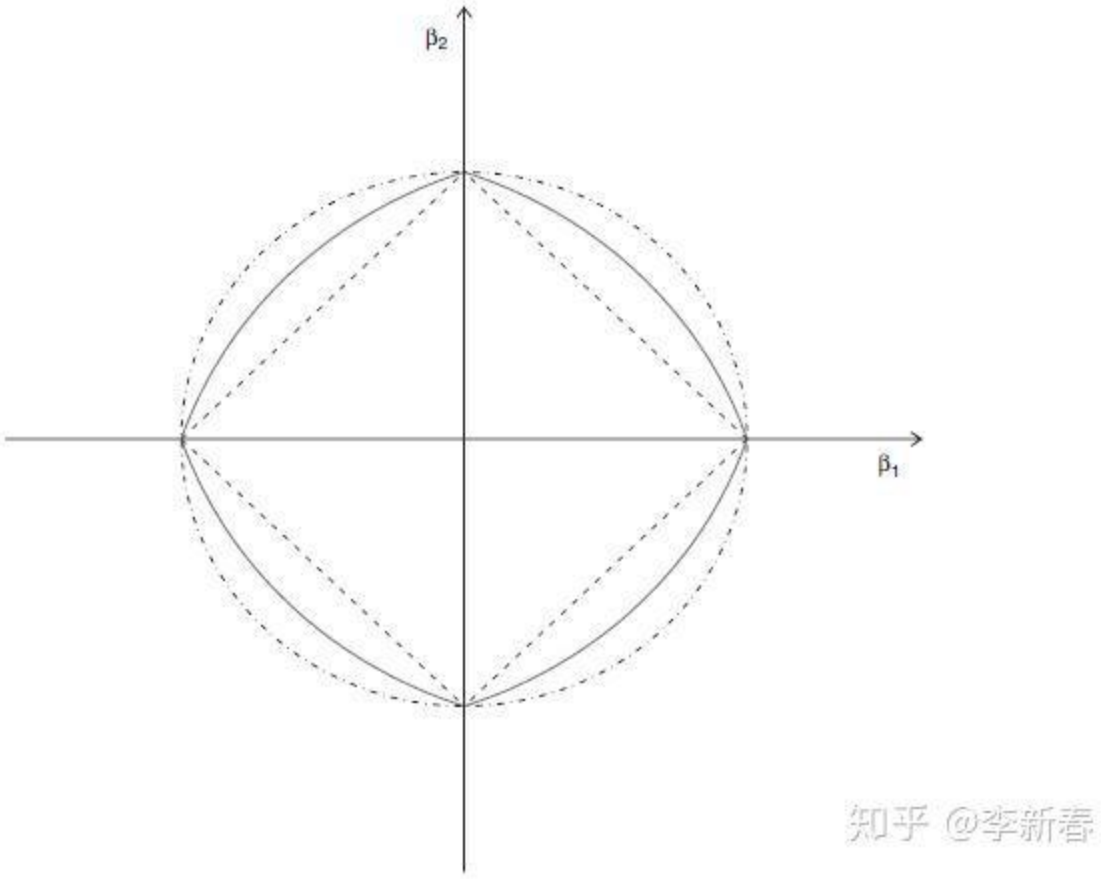

Figure 6： ElasticNet

参考文献

1. Tikhonov regularization

2. Lasso (statistics)

3. Least-angle regression

4. 101.110.118.19/statweb.

5. web.stanford.edu/~hasti

6. cs.cmu.edu/~ggordon/107

7. courses.cs.washington.edu

8. Efron, Bradley; Hastie, Trevor; Johnstone, Iain; Tibshirani, Robert. Least Angle Regression. Annals of Statistics, 2004.