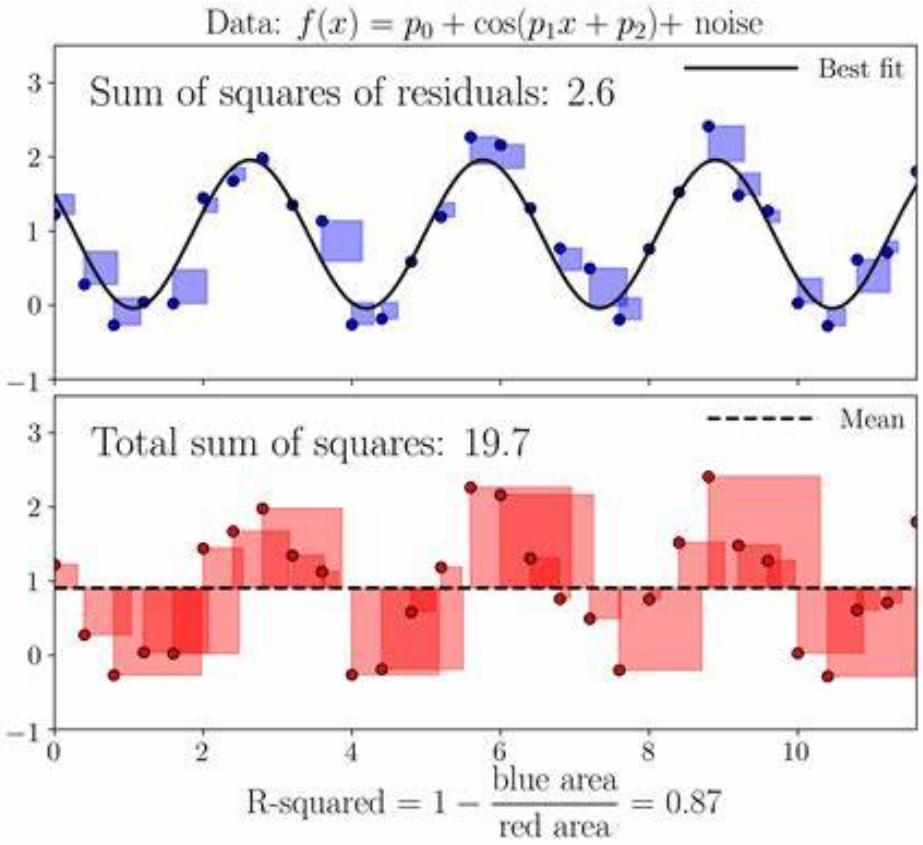
机器学习算法 - 2

Guangrui Qian

最小二乘法(Least Squares Method)

- 最小二乘法(Least Squares Method,简记为LSE)是一个比较古老的方法，源于天文学和测地学上的应用需要。在早期数理统计方法的发展中，这两门科学起了很大的作用。
- 最小二乘方法方法与高斯误差理论的结合,是数理统计史上最重大的成就之一。
- 最小二乘法的本质是最小化系数矩阵所张成的向量空间到观测向量的欧式误差距离.

$$H = \sum_{0}^{m} (y - y_i)^2$$

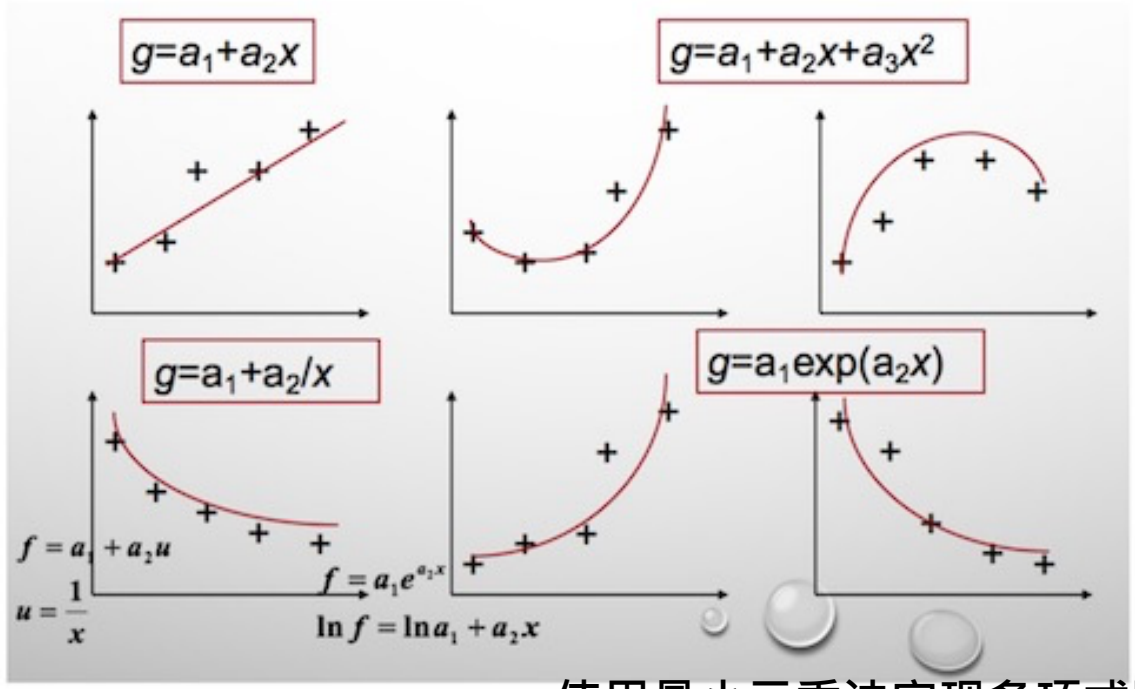


最小二乘法的应用和原理

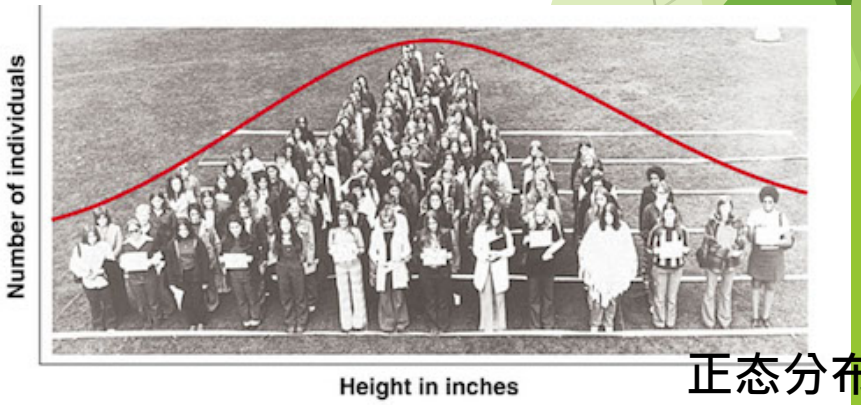
- 测量数据中让总的误差的平方最小的y 就是真值，这是基于如果误差是随机的，应该围绕真值上下波动而得出的
- 中心极限定理：在适当的条件下，大量相互独立随机变量的均值经适当标准化后依分布收敛于正态分布
  - 独立
  - 随机
  - 相加
- 求解方法：
  - 投影法
  - 矩阵法

$$\epsilon = \sum (y - y_i)^2 \text{最小} \implies \text{真值} y$$

$$\frac{d}{dy}\epsilon = \frac{d}{dy}\sum (y - y_i)^2$$



使用最小二乘法实现多项式回归



正态分布

常见回归

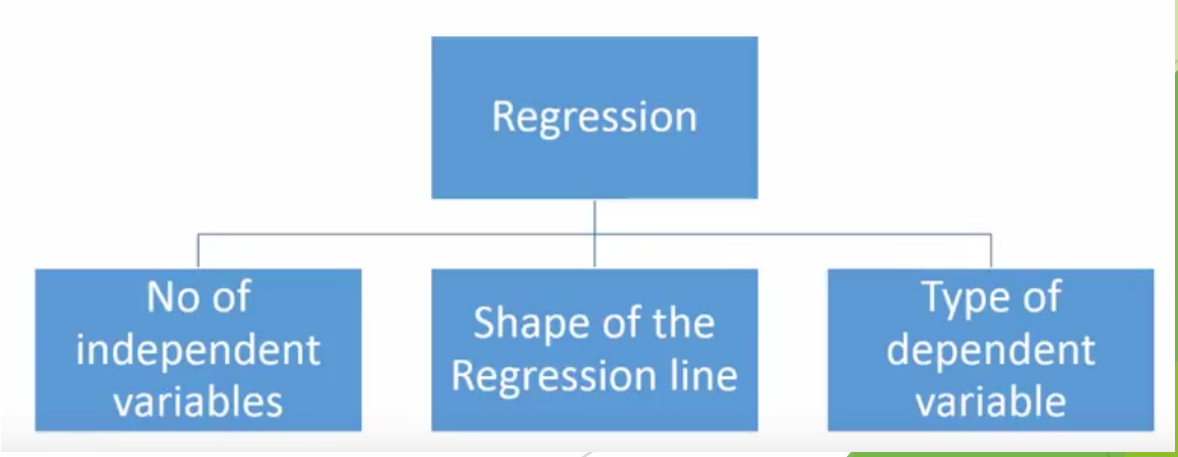
- 线性回归
- 多项式回归
- 逻辑回归
- 岭回归
- Lasso回归
- ElasticNet回归
- 逐步回归

“回归”的来历：

出自高尔顿种豆子的实验，通过大量数据统计，他发现个体小的豆子往往倾向于产生比其更大的子代，而个体大的豆子则倾向于产生比其小的子代，然后高尔顿认为这是由于新个体在向这种豆子的平均尺寸“回归”，大概的意思就是事物总是倾向于朝着某种“平均”发展，也可以说是回归于事物本来的面目

回归分析

- **什么是回归分析：**
  - 回归分析是研究自变量和因变量之间关系的一种预测模型技术。这些技术应用于预测，时间序列模型和找到变量之间关系。例如可以通过回归去研究超速与交通事故发生次数的关系。
- **为什么用回归分析：**
  - 它指示出自变量与因变量之间的显著关系
  - 它指示出多个自变量对因变量的强烈影响
- **回归分析的重要指标：**
  - 自变量的个数
  - 因变量的类型
  - 回归线的形状



线性回归重点

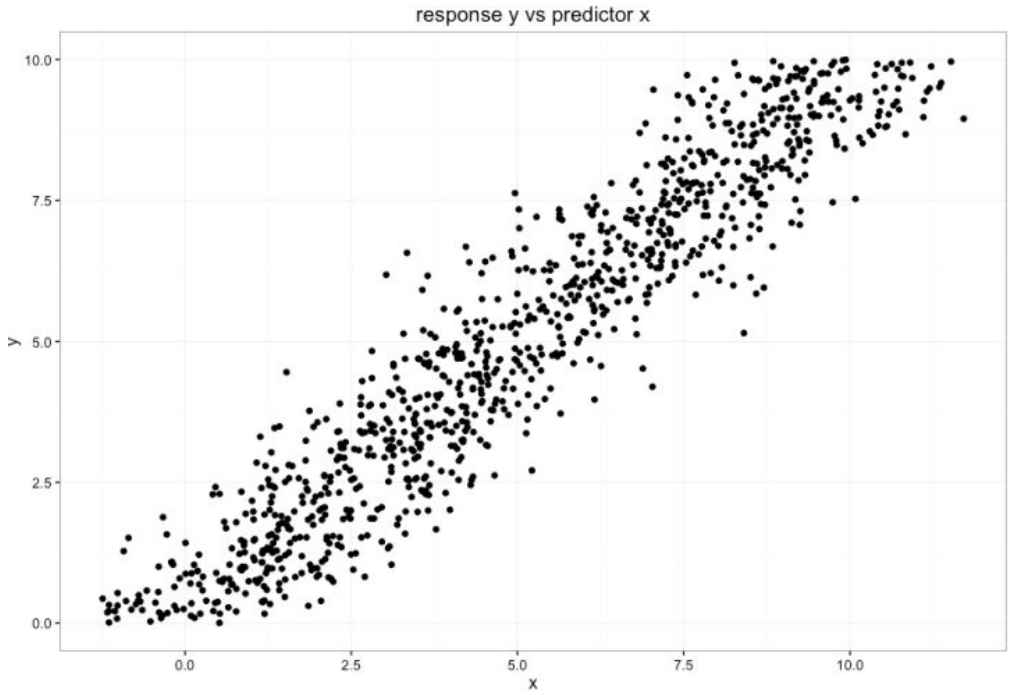
- 自变量与因变量之间必须要有线性关系。
- 多重共线性、自相关和异方差对多元线性回归的影响很大。
- 线性回归对异常值非常敏感，其能严重影响回归线，最终影响预测值。
- 多重共线性会增加参数估计的变化，使得估计对模型细微变化会异常敏感。导致参数的估计也会非常不稳定。
- 在多元的自变量中，我们可以通过前进法，后退法和逐步法去选择最显著的自变量。

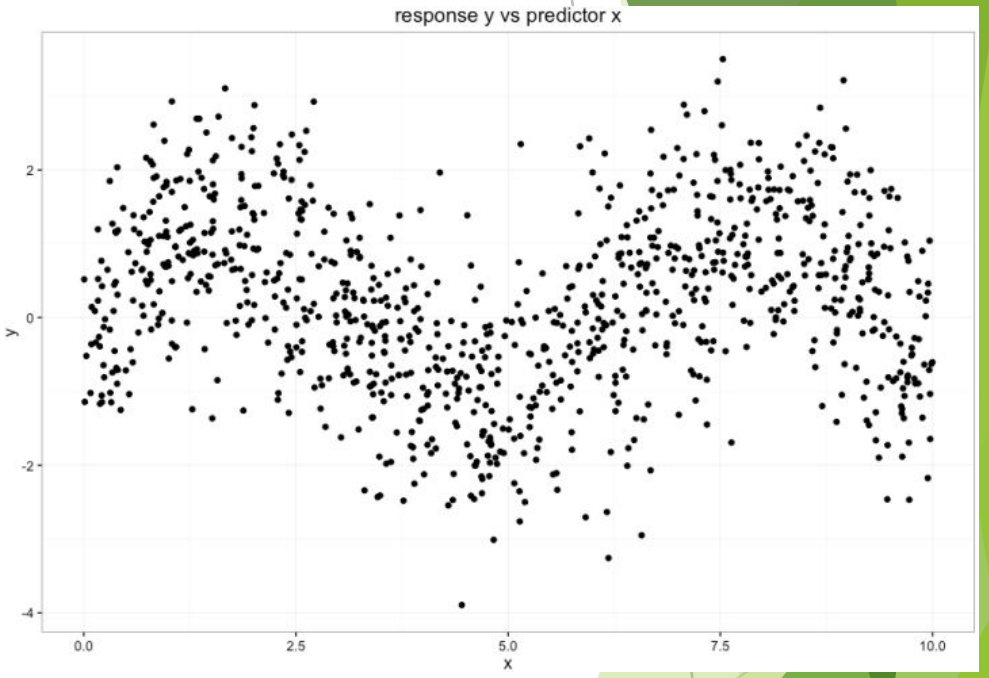
多重共线性请参考pdf文件

线性回归假设模型必须要符合以下条件

- **线性性**
- **误差项的独立性**
- **同方差性（homoscedasticity)**
- **正态性**
- **不存在多重共线性（multicollinearity）**

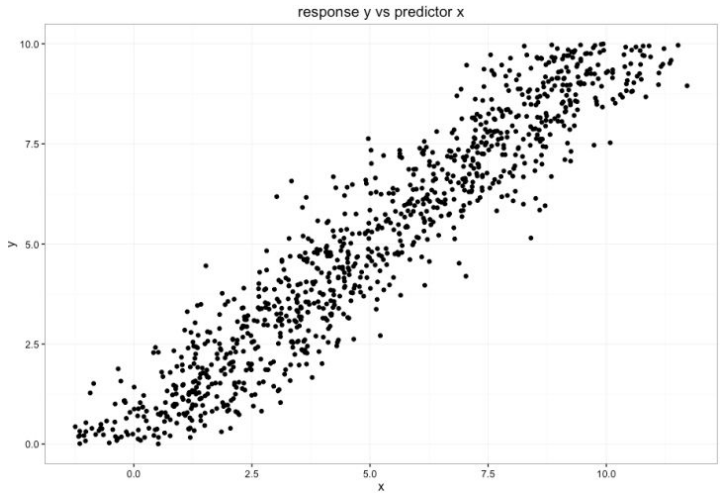
线性回归常见技巧

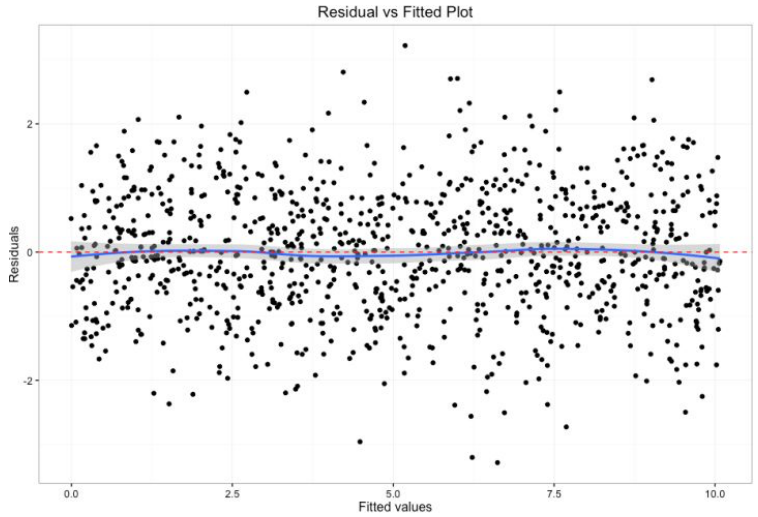


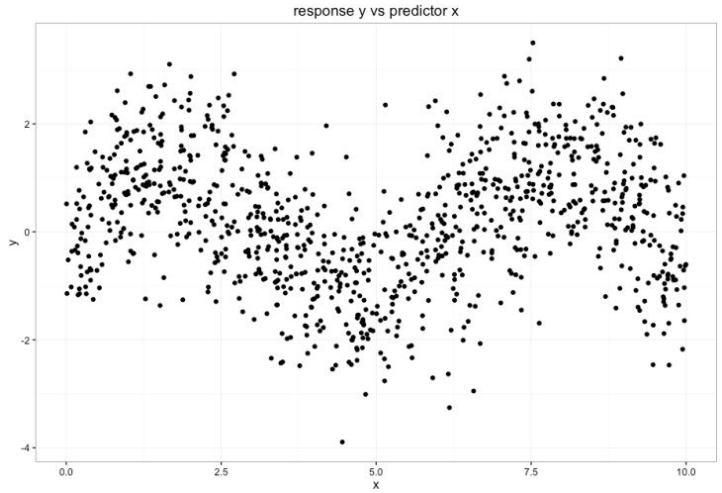


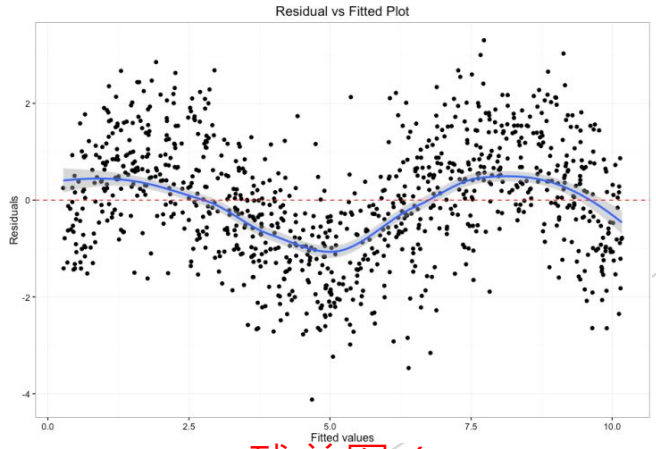
对于不满足线性性的数据强行使用线性回归模型会导致一个非常不准确的结果，在使用此模型做外部推断（**extrapolate**）或者说样本外预测（**out-of-sample prediction**）时会得到错误的数值。

线性回归常见技巧









残差图（residual plot）

数据修正 – 满足符合线性性特征

- 对响应变量或者预测变量或者对于两者同时使用合适的非线性变换（nonlinear transformation）。比如说，如果某个变量的值全部都是正的，可以考虑log变换（log transformation），如果变量大于等于0，则可以考虑log+1变换（log+1 transformation）。
- 可以考虑造一些新的预测变量加入到数据中。使得新加入的变量与已有的预测变量x有某种非线性关系。比方说，加入x^2或者x^3或者甚至于更高阶的。但这个办法常常会引入过拟合（overfitting）
- 还有种可能就是你在收集数据（data collection）的时候漏掉了很重要的预测变量。这些变量可能与响应变量有很大的关系，或者跟其他的预测变量的交互能够很好的解释响应变量。

示例：在调查价格和需求的关系（price-demand）时，经常会发现预测变量的百分比变化会导致响应变量发生相应的百分比变化而且两个变化之间竟然是成近似正比的关系，这时对解释变量和预测变量同时使用log变化再对变化后的变量搞一个线性回归就很合理了。

示例：金融机构里搞时间序列模型，经常会有这种情况，数据实际上是应该分成一段一段的，每一段上有一个明显的线性关系，但是把每一段全部合在一起看，并不存在一个简单的线性关系。那么很容易想象，如果我们能对数据进行正确的分段并且加入一个示性变量来表示观测值属于哪一段，那么最终得到的模型就是分段线性回归模型，此时的数据模拟与预测就是准确的了。当然，正如前面所说，随着新变量的加入，modeler需要时时关注过拟合的程度，找到一个好的平衡点。

同方差性

- 即误差项的方差应该是一样的
- 违反同方差性会导致如下问题：
  - 线性回归的参数估计是每一个观测值对应的预测变量和响应变量的某种方程的线性组合。如果观测值对应的变动越大，则它在此线性组合中的权重也越大。所以参数估计基本上都被少数几个高变动的观测决定了。这样建出来的模型一方面损失了原始数据中的信息，一方面不稳定，因为模型是由高变动的观测决定的，做外部预测的时候结果也会变动极大，预测值非常不准确。
  - 对方差的估计是有偏的。这个是很自然的错误，因为OLS假设方差恒定，估出来的方差是一个数，而真实方差是变动的。所以方差估计不会跟真实值一样。
  - 参数估计是无偏但是低效的。
- 怎么检测同方差性假设是否成立？
  - 画出残差与预测值的对比图（residuals vs predicted values）或者画出残差与时间的对比图（residuals vs time）或者画出残差与预测变量的对比图（residuals vs predictors）。如果残差随着预测值或者时间或者预测变量的增加而单调递增的话，那就说明同方差性被违背。

统计量

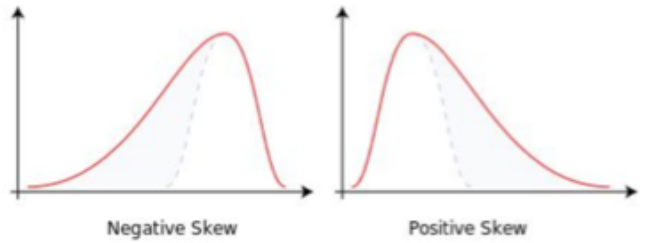
- 统计量是统计学的基本概念，它是关于样本的函数，不包含未知值，通过统计量我们可以借助样本数据去推断总体的一些性质。依据推断目的的不同，我们构造的统计量也不同。
- 统计量有严格的定义： 设 $X_1,X_2,\ldots,X_n$是从总体X 中抽取容量为n的一个样本，如果由此样本构造一个函数 $T(X_1,X_2,\ldots,X_n)$ ，不依赖任何未知参数，则称此函数为一个统计量。例如， 具体到正态分布，这个是非常符合直觉的，因为决定正态分布总共就两个参数：总体均值μ以及总体方σ^2，知道了这两个，总体的情况也就知道了。
- 常用统计量：

(1) $\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$ 是样本的均值，它反映出总体数学期望的信息；

(2) $S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})^2$ 是样本方差，反映总体的方差信息；

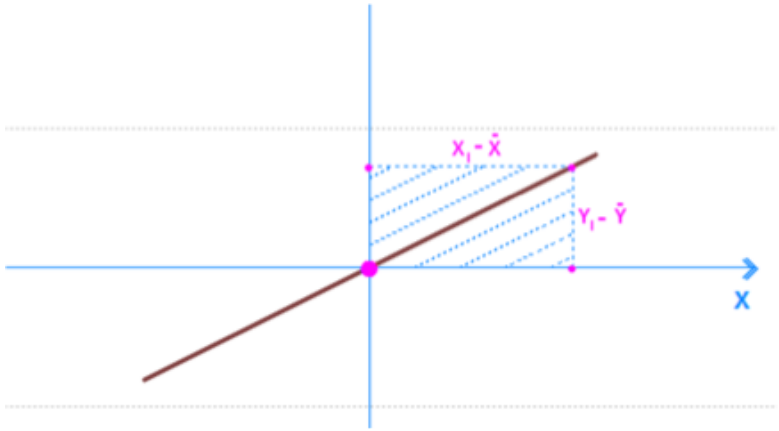
(3) $V = S/\bar{X}$ 是样本变异系数，反映总体变异系数的信息；

除开期望与方差，我们还有很多在实践中可以用来描述分布的矩。例如：偏度，标准3阶矩，即 $E[(\frac{X-\mu}{\sigma})^3]$。它的正负及大小用来衡量分布的不对称性。偏度为正代表分布为右偏态，即右侧的尾部更长，分布的主体集中在左侧。峰度，标准4阶矩，即 $E[(\frac{X-\mu}{\sigma})^4]$。它用来描述随机变量分布的峰态。



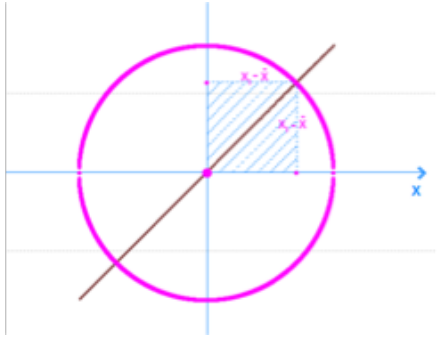
方差和标准差

- N个矩形平均面积就是协方差：协方差反映了点偏离中心点的总程度



$$Cov(X,Y) = \frac{1}{N}\sum(X_i - \bar{X})(Y_i - \bar{Y})$$

- 标准差是一维数据在一维上的平均偏离程度



$$Var(X) = \frac{1}{N}\sum(X_i - \bar{X})^2$$

相关系数

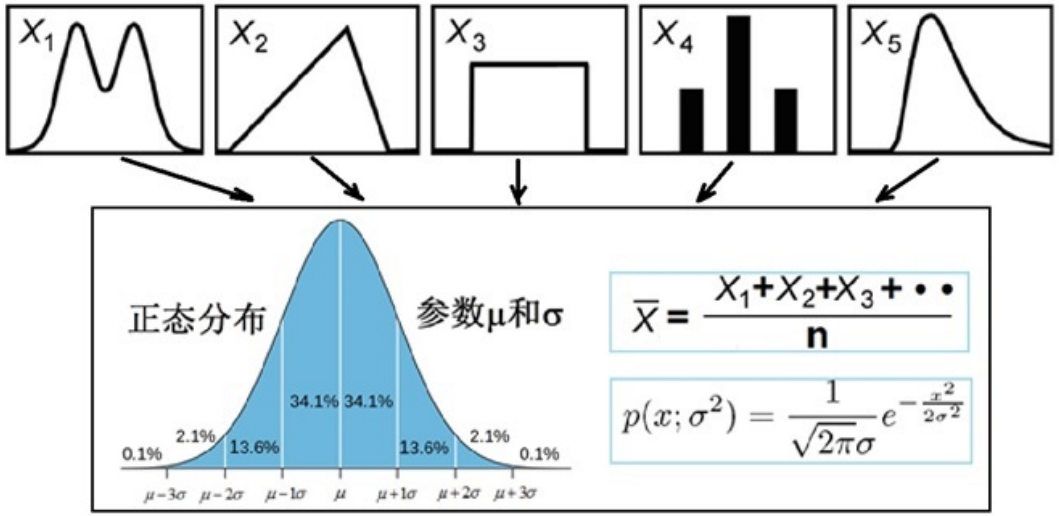
- X_i 和 Y_i 分别在一维上的平均偏离程度是其标准差。X_i 和 Y_i 标准差的乘积表示二维的平均偏离程度。相关系数就是用X、Y的协方差除以X的标准差和Y的标准差

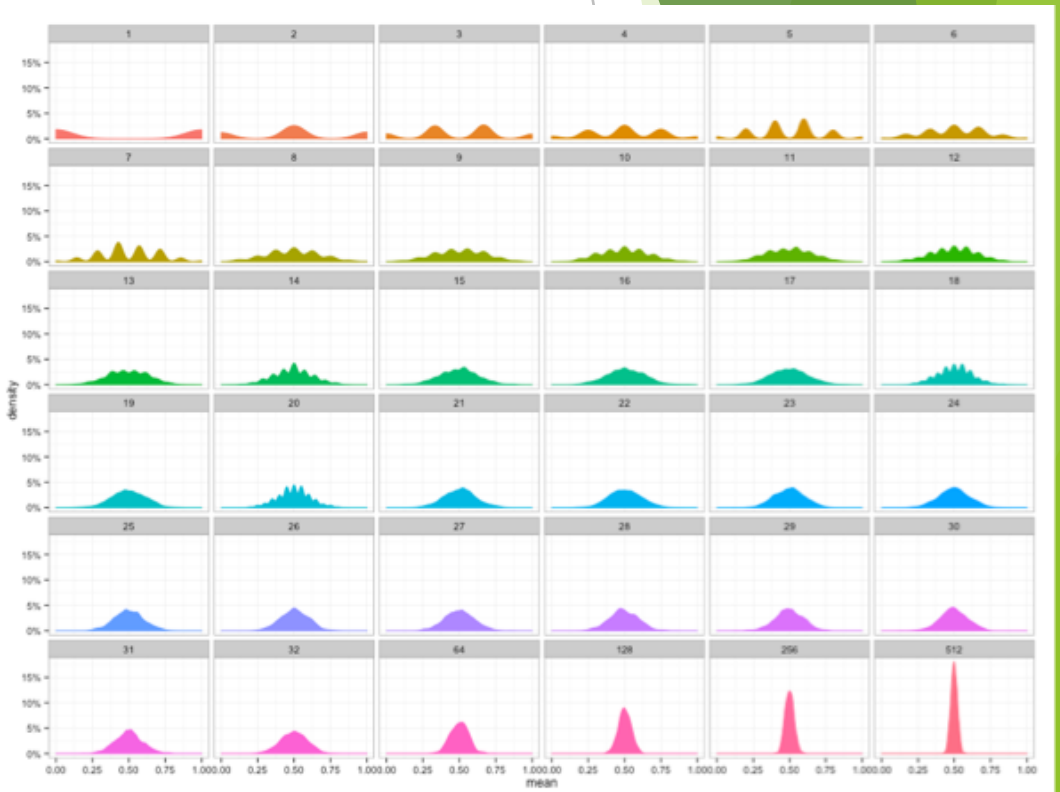
$$\sqrt{\frac{1}{N}\sum(X_i-\bar{X})^2}\times\sqrt{\frac{1}{N}\sum(Y_i-\bar{Y})^2}$$

$$Corr(X,Y)=\frac{\sum(X_i-\bar{X})(Y_i-\bar{Y})}{\sqrt{\sum(X_i-\bar{X})^2}\sqrt{\sum(Y_i-\bar{Y})^2}}$$

常用基本概念

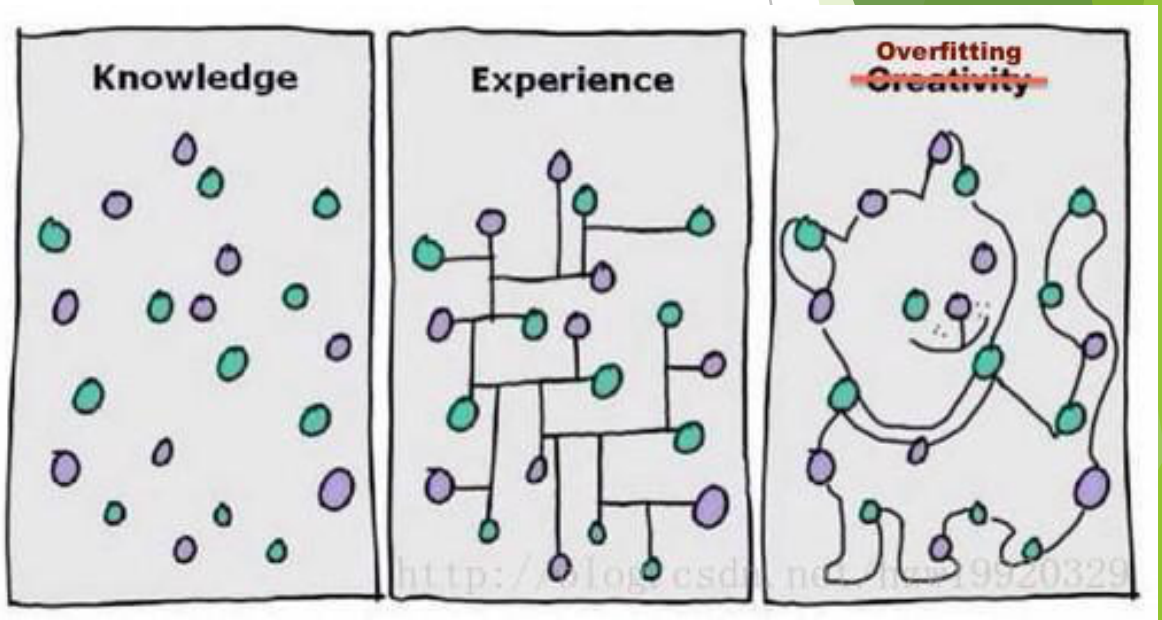
- 大数定理
- 中心极限定理





多项式回归重点

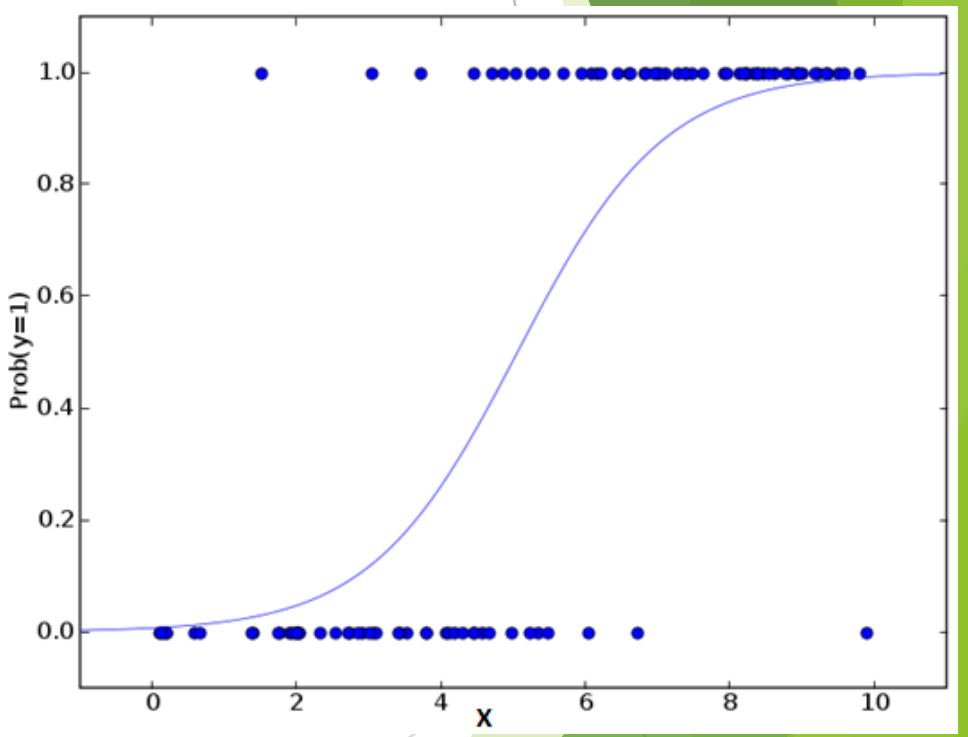
- 很多情况下，我们为了降低误差，经常会抵制不了使用多项式回归的诱惑，但事实是，我们经常会造成过拟合。所以要经常的把数据可视化，观察数据与模型的拟合程度。
- 特别是要看曲线的结尾部分，看它的形状和趋势是否有意义。高的多项式往往会产生特别古怪的预测值。



如果一个回归，它的自变量指数超过1，则称为多项式回归。

逻辑回归

- 在分类问题中使用的非常多
- 逻辑回归因其应用非线性log转换方法，使得其不需要自变量与因变量之间有线性关系
- 为防止过拟合和低拟合，我们应该确保每个变量是显著的。应该使用逐步回归方法去估计逻辑回归
- 逻辑回归需要大样本量，因为最大似然估计在低样本量的情况下表现不好
- 要求没有共线性
- 如果因变量是序数型的，则称为序数型逻辑回归
- 如果因变量有多个，则称为多项逻辑回归



岭回归

- 岭回归的假设与最小二乘法回归的假设相同除了假设正态性。
- 它把系数的值收缩了，但是不会为0.
- 正则化方法是使用了l2正则.

$$= \underset{\beta \in \mathbf{R}^p}{\operatorname{argmin}} \underbrace{\|y - X\beta\|_2^2}_{\text{Loss}} + \lambda \underbrace{\|\beta\|_2^2}_{\text{Penalty}}$$

当碰到数据有多重共线性时，我们就会用到岭回归。所谓多重共线性，简单的说就是自变量之间有高度相关关系。在多重共线性中，即使是最小二乘法是无偏的，它们的方差也会很大。通过在回归中加入一些偏差，岭回归酒会减少标准误差。

Lasson回归

- 其假设与最小二乘回归相同除了正态性
- 其能把系数收缩到0，使得其能帮助特征选择
- 这个正则化方法为l1正则化
- 如果一组变量是高度相关的，lasso会选择其中的一个，然后把其他的都变为0

$$= \underset{\beta \in \mathbb{R}^p}{\operatorname{argmin}} \underbrace{\|y - X\beta\|_2^2}_{\text{Loss}} + \lambda \underbrace{\|\beta\|_1}_{\text{Penalty}}$$

和岭回归类似，Lasso(least Absolute Shrinkage and Selection Operator)也是通过惩罚其回归系数的绝对值。Lasso回归和岭回归不同的是，Lasso回归在惩罚方程中用的是绝对值，而不是平方。这就使得惩罚后的值可能会变成0.

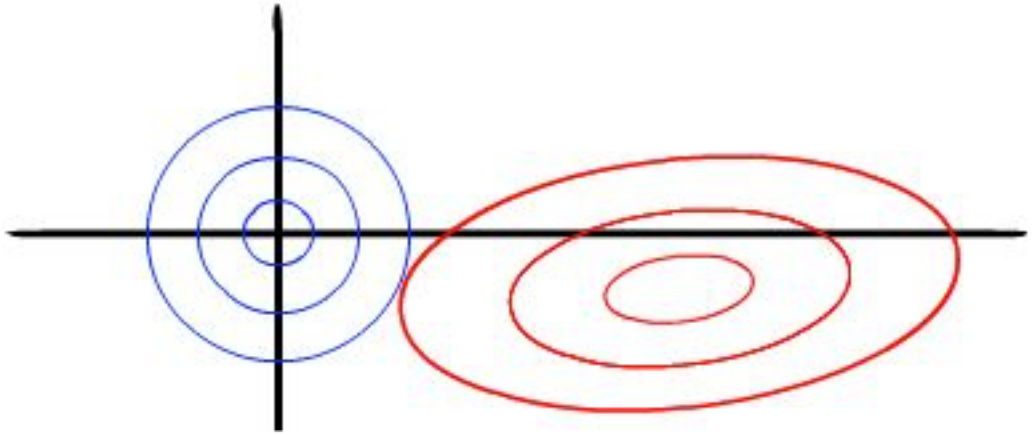
L2正则化

$$f(w) = \sum_{i=1}^{m}(y_i - x_i^T w)^2 + \lambda \sum_{i=1}^{n} w_i^2$$

$$f(w) = \sum_{i=1}^{m}(y_i - x_i^T w)^2$$

$$s.t. \sum_{i=1}^{n} w_j^2 \leq t$$

其中t为某个阈值。



岭回归的几何意义

以两个变量为例，残差平方和可以表示为 w_1, w_2 的一个二次函数，是一个在三维空间中的抛物面，可以用等值线来表示。而限制条件 $w_1^2 + w_2^2 < t$ ， 相当于在二维平面的一个圆。这个时候等值线与圆相切的点便是在约束条件下的最优点，如下图所示，

1. 当岭参数 $\lambda = 0$ 时，得到的解是最小二乘解
2. 当岭参数 λ 趋向更大时，岭回归系数 w_i 趋向于0，约束项 t 很小

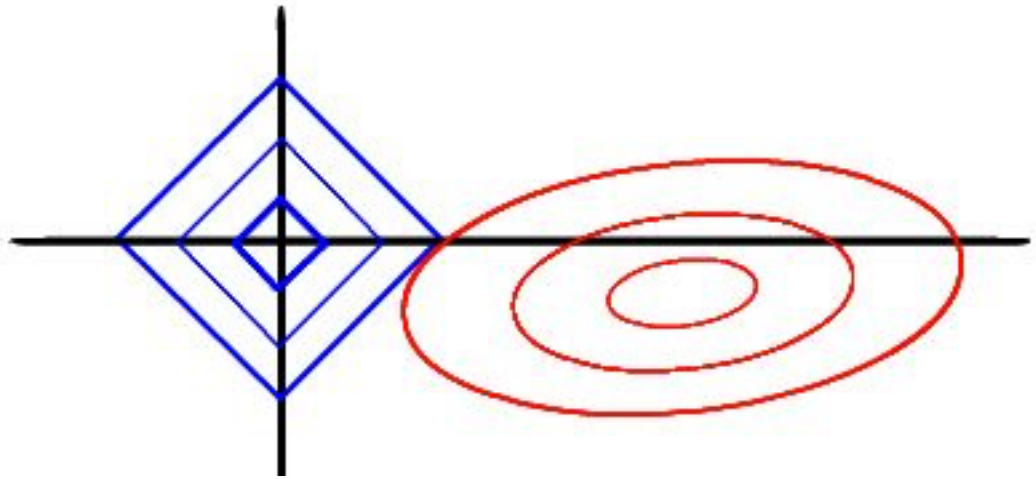
L1正则化

$$f(w) = \sum_{i=1}^{m}(y_i - x_i^T w)^2 + \lambda \sum_{i=1}^{n} w_i$$

$$f(w) = \sum_{i=1}^{m}(y_i - x_i^T w)^2$$

$$s.t. \sum_{i=1}^{n} |w_i| \leq t$$

其中t为某个阈值。



岭回归限定了所有回归系数的平方和不大于 t，在使用普通最小二乘法回归的时候当两个变量具有相关性的时候，可能会使得其中一个系数是个很大正数，另一个系数是很大的负数。通过岭回归的 $\sum_{i=1}^{n} w_i \leq t$ 的限制，可以避免这个问题。

相比圆，方形的顶点更容易与抛物面相交，顶点就意味着对应的很多系数为0，而岭回归中的圆上的任意一点都很容易与抛物面相交很难得到正好等于0的系数。这也就意味着，lasso起到了很好的筛选变量的作用。

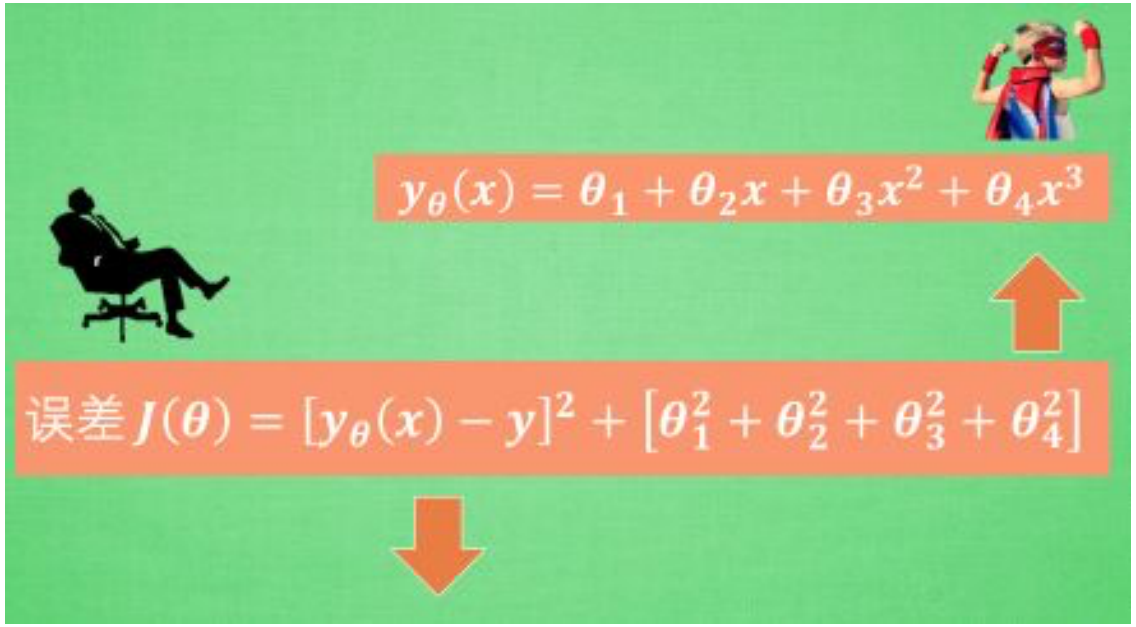
ElasticNet回归

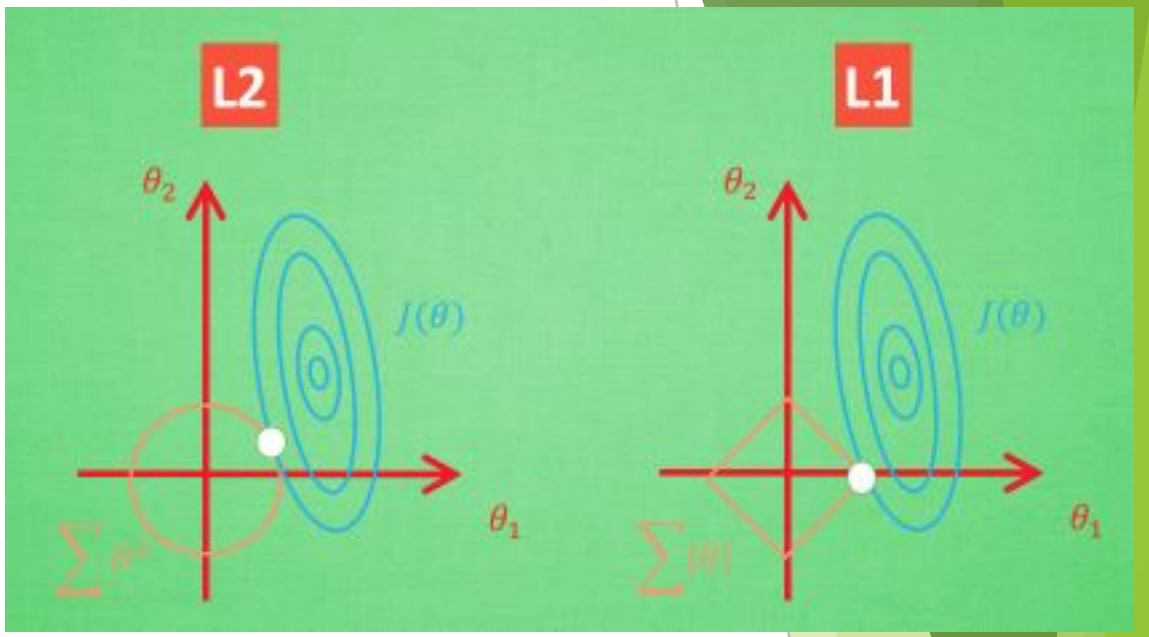
- 在选择变量的数量上没有限制
- 双重收缩对其有影响

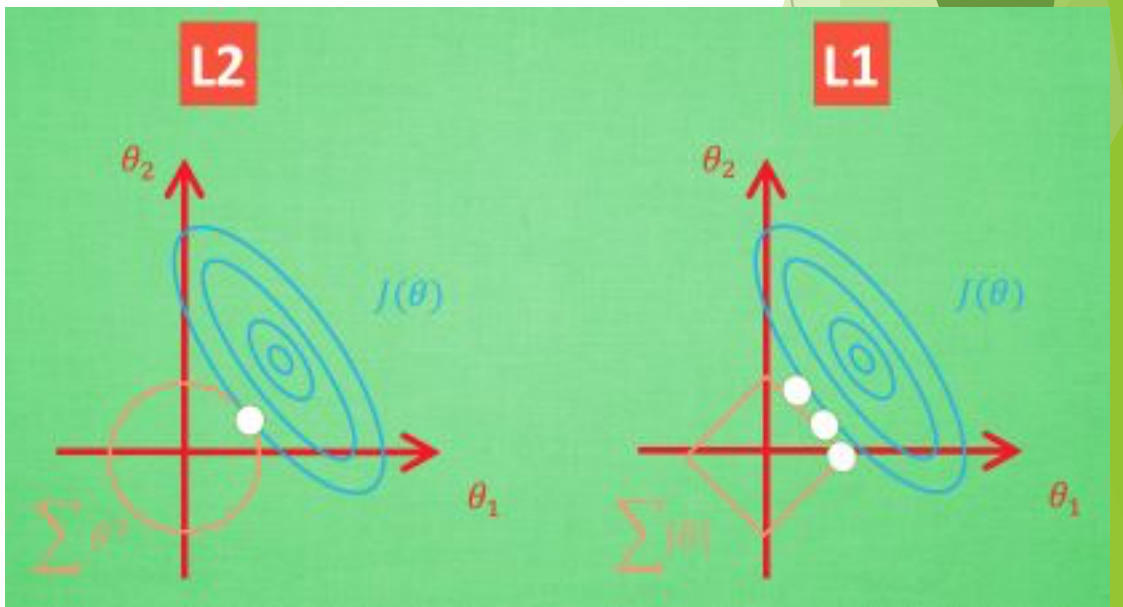
$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}}(\|y - X\beta\|^2 + \lambda_2\|\beta\|^2 + \lambda_1\|\beta\|_1).$$

ElasticNet回归是Lasso回归和岭回归的组合。它会事先训练L1和L2作为惩罚项。当许多变量是相关的时候，Elastic-net是有用的。Lasso一般会随机选择其中一个，而Elastic-net则会选在两个。 与Lasso和岭回归的利弊比较，一个实用的优点就是Elastic-Net会继承一些岭回归的稳定性。

L1和L2正则化



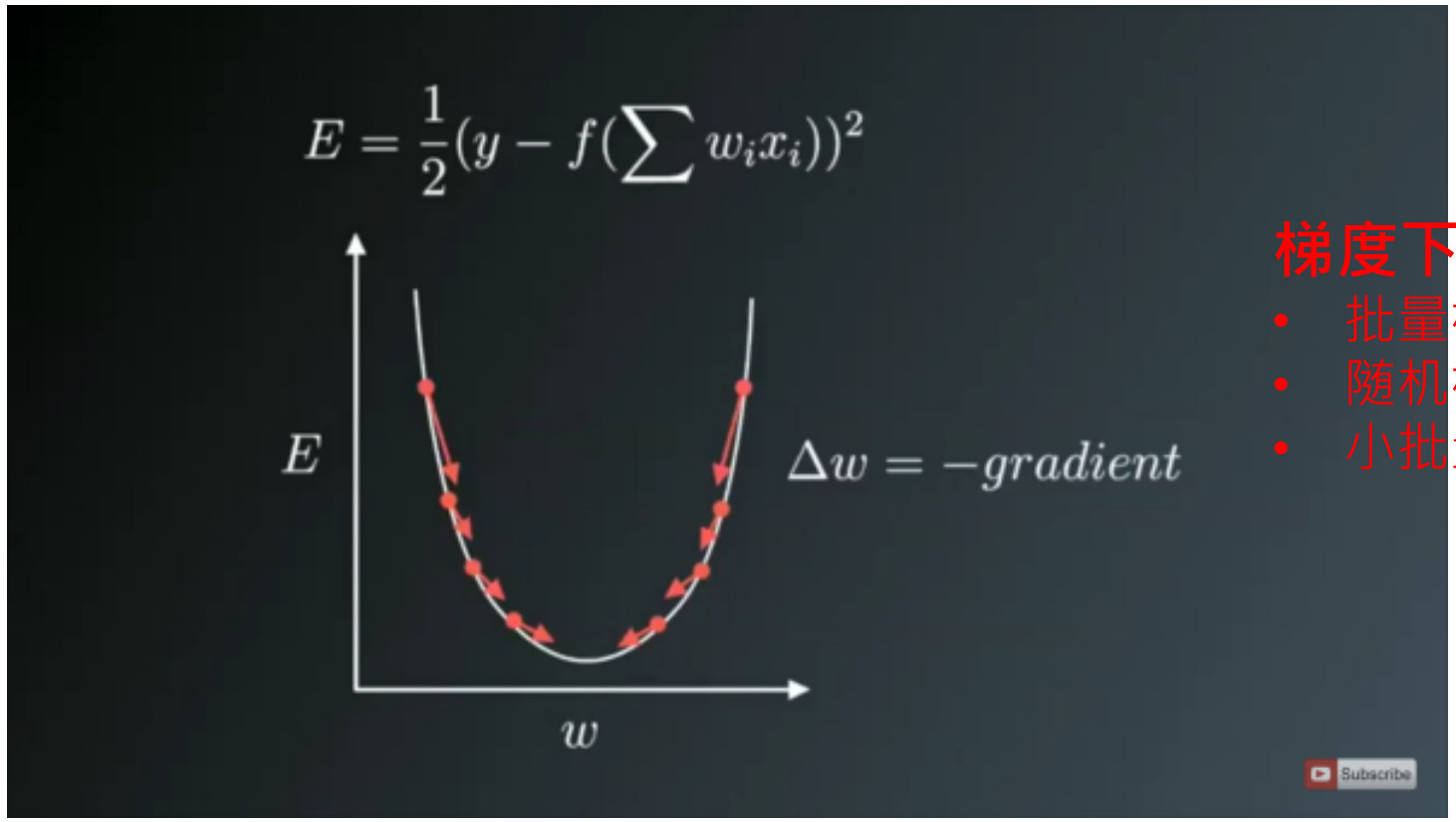




正则化的方程是在黄线上产生的额外误差(也能理解为惩罚度)，在黄圈上的额外误差也是一样.

为了控制这种正规化的强度，我们会加上一个参数 lambda，并且通过 交叉验证 cross validation 来选择比较好的 lambda. 这时，为了统一化这类型的正则化方法，我们还会使用 p 来代表对参数的正则化程度.

梯度下降(gradient descent）



梯度下降的变体：

- 批量梯度下降法（batch gradient descent）
- 随机梯度下降法（stochastic gradient descent）
- 小批量梯度下降法（mini-batch gradient descent）

梯度下降(gradient descent）

- **批量梯度下降法（BGD）**
  - 批量梯度下降法，是梯度下降法最常用的形式，具体做法也就是在更新参数时使用所有的样本来进行更新
- **随机梯度下降法（SGD）**
  - SGD和BGD不同点在于SGD每次迭代的时候并不是将所有数据点累加起来，而是随机选取一个点来迭代更新
- **小批量梯度下降法（MBGD）**
  - 更新函数不再像SGD那样只用一个数据点来迭代，MBGD每次迭代选择n(n<m)个数据点

$$\theta_j = \theta_j - \alpha \frac{\partial J(\theta)}{\partial \theta_j} = \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} x_j \left(h_\theta(x) - y\right)$$

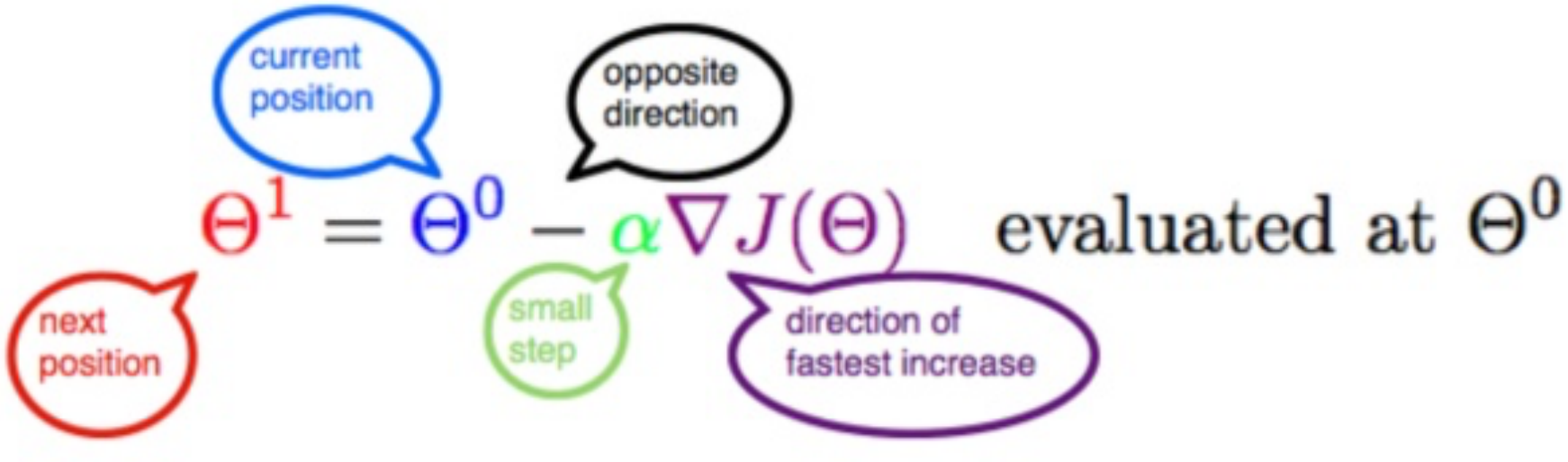
$$\theta_j = \theta_j - \alpha x_j^i \left(h_\theta\left(x^i\right) - y\right)$$

$$\theta_j = \theta_j - \alpha \frac{1}{n} \sum_{i=1}^{n} x_j^n \left(h_\theta\left(x^n\right) - y^n\right)$$

方法	BGD	SGD	MBGD
优点	全局最优，易于并行实现	速度快	两者折中
缺点	速度慢	噪声大，不易于并行实现	两者折中

梯度下降算法 – 参数

- □在梯度下降算法中被称作为学习率或者步长，意味着计算过程中可以通过□来控制每一步走的距离。步长太大，会导致迭代过快，甚至有可能错过最优解。步长太小，迭代速度太慢，很长时间算法都不能结束。所以算法的步长需要多次运行后才能得到一个较为优的值。



多重共线性解决方法：逐步选择

- 当自变量之间的关系较为复杂，对于变量的取舍不易把握时，我们还可以利用逐步回归的方法进行变量筛选，以解决自变量多重共线性的问题。逐步回归法从共线性的自变量中筛选出对因变量影响较为显著的若干个变量，把对因变量贡献不大的自变量排除在模型之外，从而建立最优的回归子集，不仅克服了共线性问题，而且使得回归方程得到简化。
- 常用方法：
  - 进入法
  - 移除法
  - 前进法
  - 后退法
  - 逐步回归法

多重共线性解决方法 - 1

- Enter（进入法）
  - 将所选自变量强制性引入模型中进行拟合，不涉及变量筛选的问题，为默认选项。
- Remove（移除法）
  - 将指定的自变量强制性移除模型。Remove方法的第一步是利用Enter法构建回归方程，第二步再用Remove法将指定的自变量移除模型。该方法常与其他筛选变量的方法联合使用。

多重共线性解决方法 - 2

- Forward selection（前进法）
  - 即回归方程中的自变量从无到有，由少到多逐个引入来构建模型的一种方法。这里需要提到一个新的概念--偏回归平方和，简单来说就是在模型已经含有其他自变量的基础上，加入一个新的自变量后，引起的对于回归模型贡献的增加量，或者删除某个自变量后，引起的对于回归模型贡献的减少量。
  - 前进法的优点是可以自动去掉高度相关的自变量，但也有一定的局限性，前进法在自变量选择的过程中，只在自变量引入模型时考察其是否有统计学意义，并不考虑在引入模型后每个自变量P值的变化，后续变量的引入可能会使先进入方程的自变量变得无统计学意义。
- Backward elimination（后退法）
  - 后退法与前进法相反，即先建立全变量模型，然后逐步剔除无统计学意义的自变量，以此构建回归模型的一种方法。如果说前进法是选拔员工，那么后退法就相当于公司裁员，每一次裁掉一个对公司贡献最小且无显著性意义的员工（剔除标准Pout>0.1），然后对剩下的员工再次进行评估，裁掉一个贡献最小的员工，以此类推不断有员工被裁掉，直到公司认为即使再裁掉其他员工，也不会额外减少对公司的贡献，此时裁员停止，以上即为后退法的基本流程。
  - 后退法的优点是考虑了自变量的组合作用，但是当自变量数目较多或者自变量间高度相关时，可能得不出正确的结论。

多重共线性解决方法 - 3

- Stepwise（逐步回归法）
  - 逐步回归法，是在前进法和后退法的基础上，进行双向筛选变量的一种方法。其本质是前进法，也就是说公司（Y）每引入一个员工（X）后，都要重新对已经进入公司的每个员工的贡献进行评估和检验，如果原有的员工由于后续引入新员工后，其贡献变得不再有显著性，则会将其裁掉，以确保公司里每一个员工的贡献都是有意义的。
  - 这个过程反复进行，直到既没有不显著的自变量引入回归方程，也没有显著的自变量从回归方程中剔除为止，从而得到一个最优的回归方程。逐步回归法结合了前进法和后退法的优点，因此被作为自变量筛选的一种常用的方法。

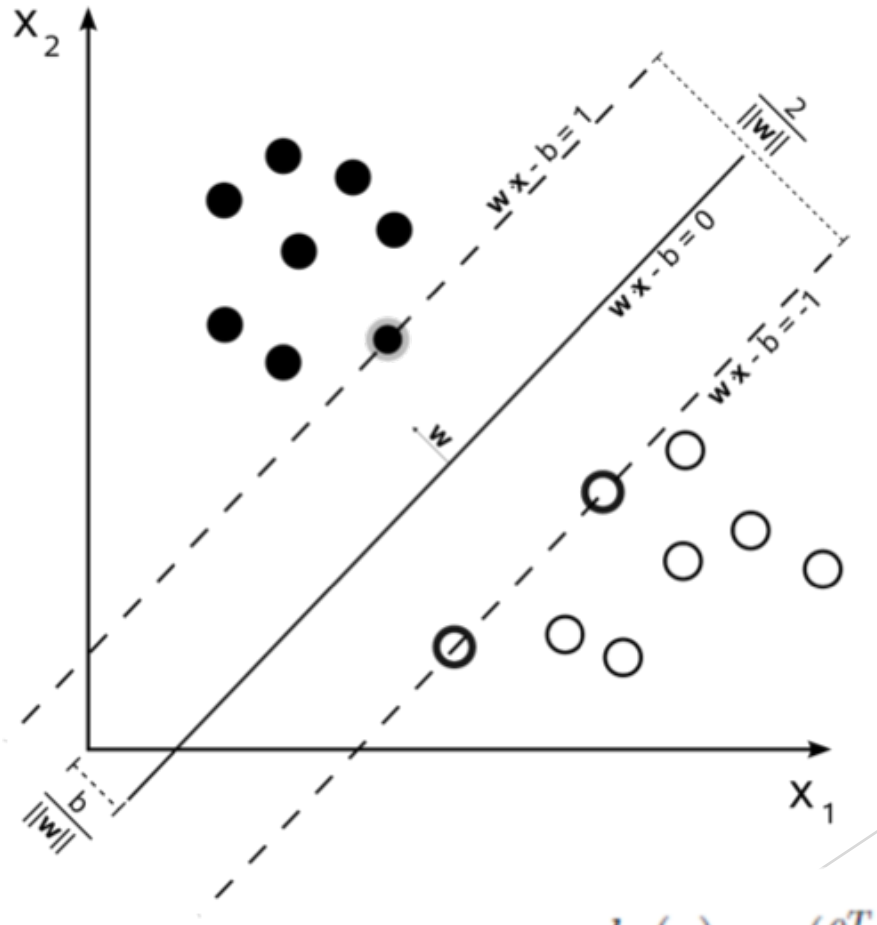
SVM支持向量机

- 支持向量机(Support Vector Machine)是Cortes和Vapnik于1995年首先提出的，它在解决小样本、非线性及高维模式识别中表现出许多特有的优势，并能够推广应用到函数拟合等其他机器学习问题中
- 所谓支持向量机，顾名思义，分为两个部分了解:
  - 支持向量(简单来说，就是支持 or 支撑平面上把两类类别划分开来的超平面的向量点)
  - “机（machine,机器）”是一个算法。在机器学习领域，常把一些算法看做是一个机器，如分类机(当然，也叫做分类器)，而支持向量机本身便是一种监督式学习的方法，它广泛的应用于统计分类以及回归分析中。
- 支持向量机(SVM)是90年代中期发展起来的基于统计学习理论的一种机器学习方法，通过寻求结构化风险最小来提高学习机泛化能力，实现经验风险和置信范围的最小化，从而达到在统计样本量较少的情况下，亦能获得良好统计规律的目的。

SVM为 -1和1分类问题

$$h_{w,b}(x) = g(w^T x + b)$$

$$g(z) = \begin{cases} 1, & z \geq 0 \\ -1, & z < 0 \end{cases}$$



SVM方法本质:从最大间隔出发（目的本就是为了确定法向量w），转化为求对变量w和b的凸二次规划问题。

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}},$$

Logistic回归目的是从特征学习出一个0/1分类模型

Maximum Margin Classifier

- SVM 通过使用最大分类间隙Maximum Margin Classifier 来设计决策最优分类超平面
- 最大间隔能获得最大稳定性与区分的确信度，从而得到良好的推广能力(超平面之间的距离越大，分离器的推广能力越好，也就是预测精度越高，不过对于训练数据的误差不一定是最小的

函数间隔Functional margin $\delta=y(wx+b)=|g(x)|$

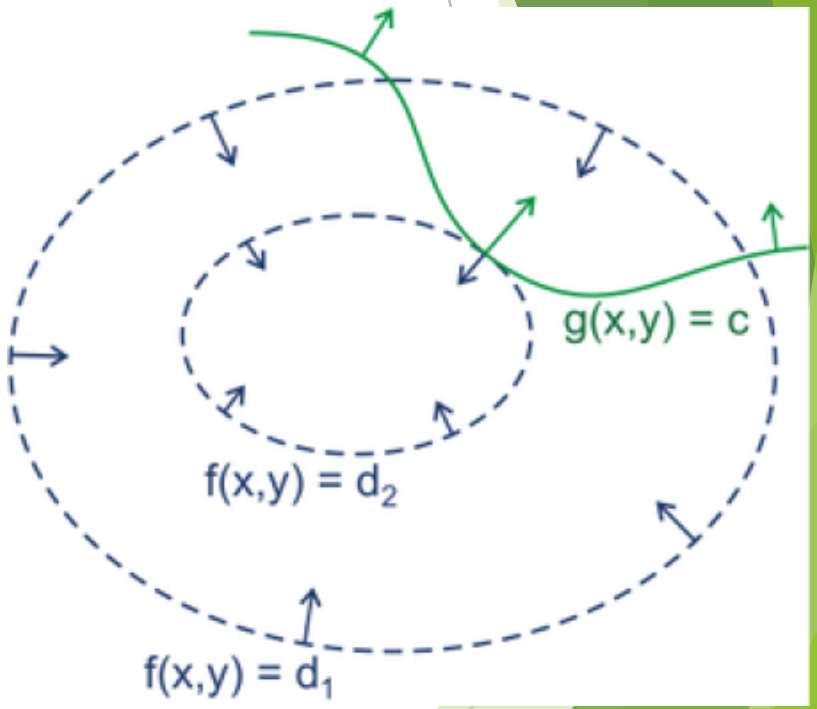
几何间隔Geometrical margin $\delta_{几何}=\frac{1}{\|w\|}|g(x)|$

$$\min \quad \frac{1}{2}\|w\|^2$$

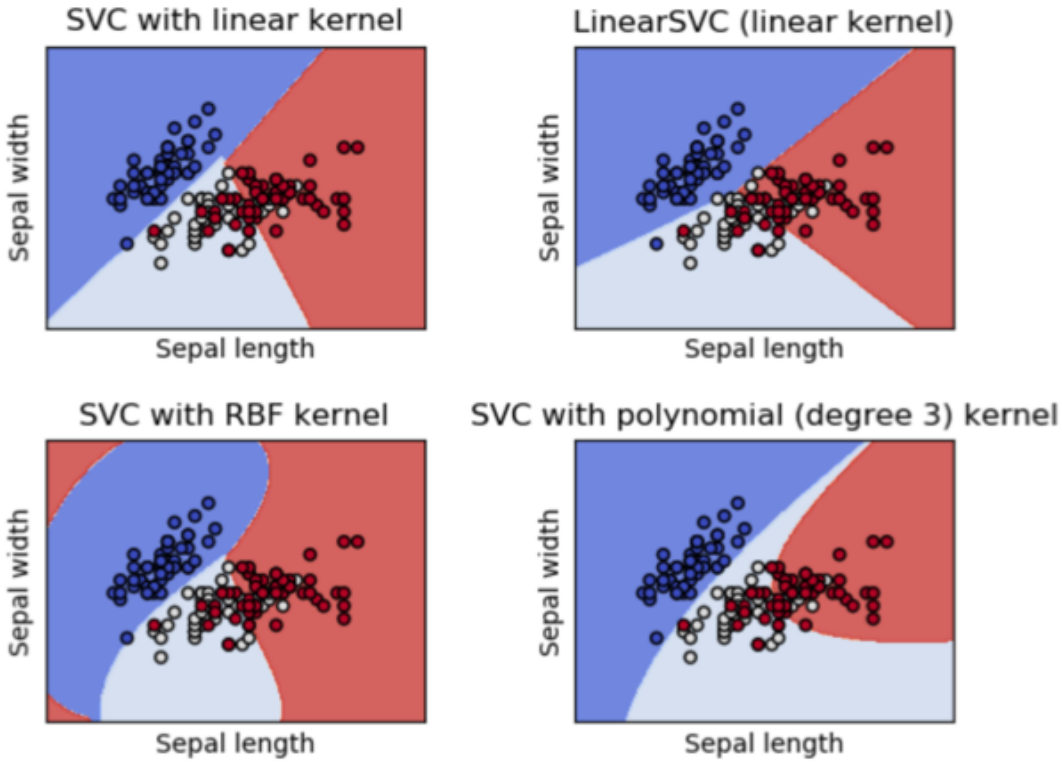
$$\text{subject to} \quad y_i[(wx_i)+b]-1\geq 0 \quad (i=1,2,\ldots,l)$$ (l是样本数)

拉格朗日乘数法

- 绿线标出的是约束g(x,y)=c的点的轨迹。
- 蓝线是f(x,y)的等高线。
- 箭头表示斜率，和等高线的法线平行。
- 从梯度的方向上来看，显然有d1>d2。绿色的线是约束，也就是说，只要正好落在这条绿线上的点才可能是满足要求的点。如果没有这条约束，f(x,y)的最小值应该会落在最小那圈等高线内部的某一点上。而现在加上了约束，最小值点应该在f(x,y)的等高线正好和约束线相切的位置，因为如果只是相交意味着肯定还存在其它的等高线在该条等高线的内部或者外部，使得新的等高线与目标函数的交点的值更大或者更小，只有到等高线与目标函数的曲线相切的时候，可能取得最优值。
- 如果我们对约束也求梯度∇g(x,y)，则其梯度如图中绿色箭头所示。很容易看出来，要想让目标函数f(x,y)的等高线和约束相切，则他们切点的梯度一定在一条直线上(f和g的斜率平行)。



SVM调用



sklearn.svm.SVC

class `sklearn.svm.` **SVC** (*C=1.0, kernel='rbf', degree=3, gamma='auto', coef0=0.0, shrinking=True, probability=False, tol=0.001, cache_size=200, class_weight=None, verbose=False, max_iter=-1, decision_function_shape='ovr', random_state=None*) [source]

C-Support Vector Classification.

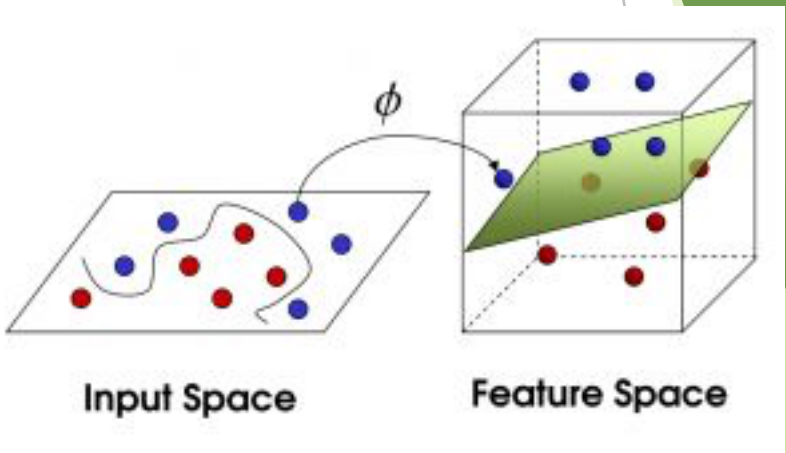
The implementation is based on libsvm. The fit time complexity is more than quadratic with the number of samples which makes it hard to scale to dataset with more than a couple of 10000 samples.

The multiclass support is handled according to a one-vs-one scheme.

For details on the precise mathematical formulation of the provided kernel functions and how gamma, coef0 and degree affect each other, see the corresponding section in the narrative documentation: Kernel functions.

SVM模型核函数

- 尝试不同的核函数
- 把C调大
- 调核函数的参数，以高斯核为例，方差调小。



$$g(x) = < w, x > + b$$

$$= < \sum_{i=1}^{n} (\alpha_i y_i x_i), x > + b$$

$$g(x) = \sum_{i=1}^{n} \alpha_i y_i < x_i, x > + b$$

$$g(x) = \sum_{i=1}^{n} \alpha_i y_i K(x_i, x) + b$$

$$\min_{w,b,\zeta} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \zeta_i$$

$$\text{subject to } y_i(w^T \phi(x_i) + b) \geq 1 - \zeta_i,$$

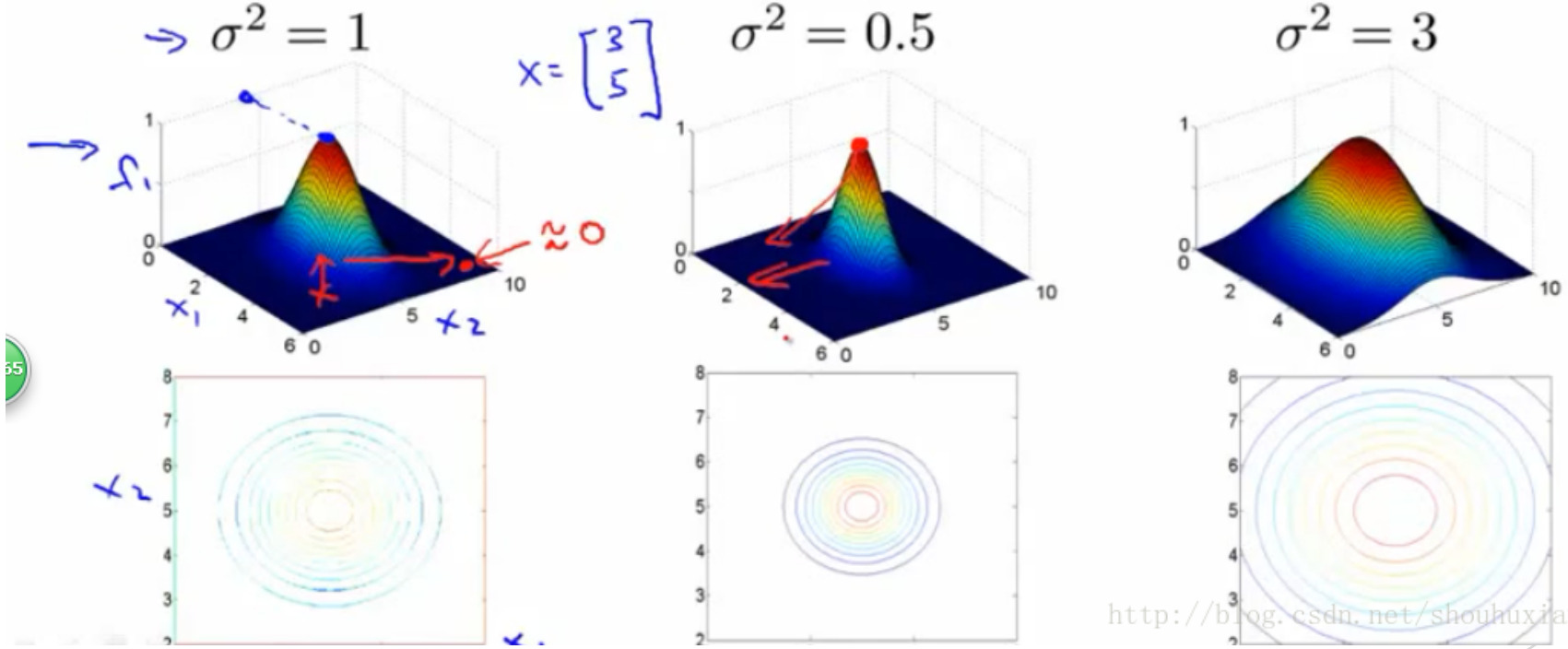
$$\zeta_i \geq 0, i = 1, \ldots, n$$

SVM核函数作用

Example:

$$l^{(1)} = \begin{bmatrix} 3 \\ 5 \end{bmatrix}, \quad f_1 = \exp\left(-\frac{\|x - l^{(1)}\|^2}{2\sigma^2}\right)$$

$\sigma^2 = 1$ $\sigma^2 = 0.5$ $\sigma^2 = 3$



对于高斯核来说，方差sigma 平方是一个可以作为改变的参数

核函数选择

- SVM关键是选取核函数的类型，主要有线性内核，多项式内核，径向基内核(RBF)，sigmoid核。
- Linear核：主要用于线性可分的情形。参数少，速度快，对于一般数据，分类效果已经很理想了。
- RBF核函数可以将一个样本映射到一个更高维的空间，而且线性核函数是RBF的一个特例。 RBF kernel可以处理非线性的情况, 主要用于线性不可分的情形。参数多，分类结果非常依赖于参数。有很多人是通过训练数据的交叉验证来寻找合适的参数，不过这个过程比较耗时。sigmoid kernel又在某些参数下和RBF很像.
- polynomial kernel的参数比RBF多，而参数越多模型越复杂
- 不行换高斯核

只要是满足了Mercer条件的函数，都可以作为核函数

支持向量机特点

- 支持向量机使用一个名为核函数的技巧，来将非线性问题变换为线性问题，其本质是计算两个观测数据的距离。支持向量机算法所寻找的是能够最大化样本间隔的决策边界，因此又被称为大间距分类器。
- 使用线性核函数的支持向量机类似于逻辑回归，但更具稳健性。因而在实践中，支持向量机最大用处是用非线性核函数来对非线性决策边界进行建模。

•优点：支持向量机能对非线性决策边界建模，又有许多可选的核函数。在面对过拟合时，支持向量机有着极强的稳健性，尤其是在高维空间中。

•缺点：不过，支持向量机是内存密集型算法，选择正确的核函数就需要相当的技巧，不太适用较大的数据集。在当前的业界应用中，随机森林的表现往往要优于支持向量机。

PCA & SVD几何意义

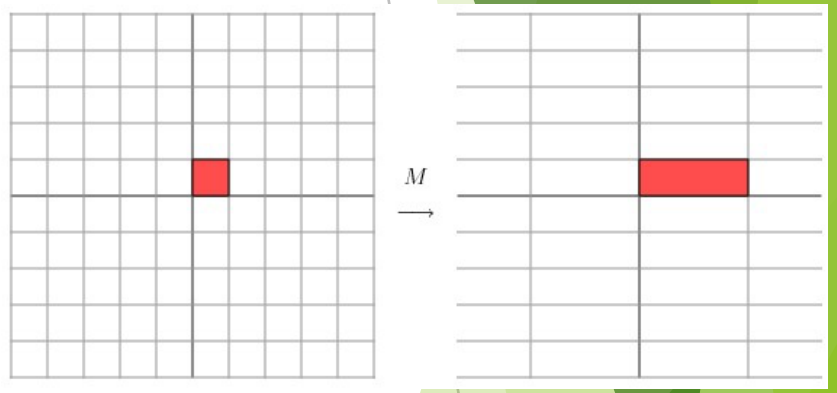
Guangrui Qian

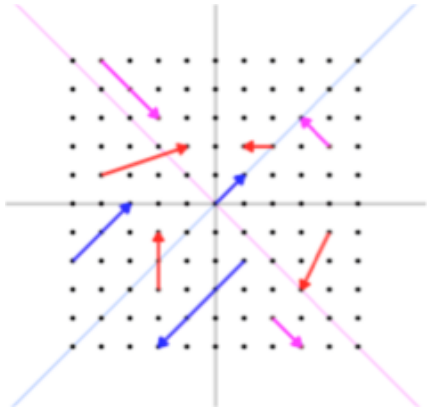
奇异值分解(SVD) – 线性变换几何意义

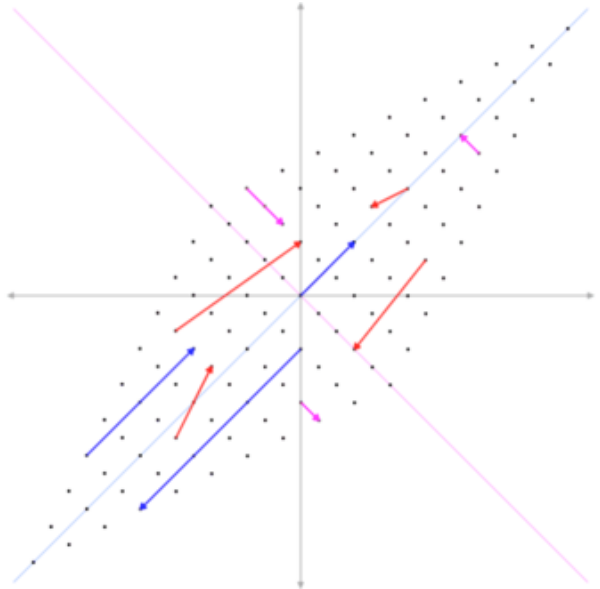
通过SVD对数据的处理，我们**可以使用小得多的数据集来表示原始数据集**，这样做实际上是去除了噪声和冗余信息，以此达到了优化数据、提高结果的目的。

线性变换：

$$M = \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 3x \\ y \end{bmatrix}.$$



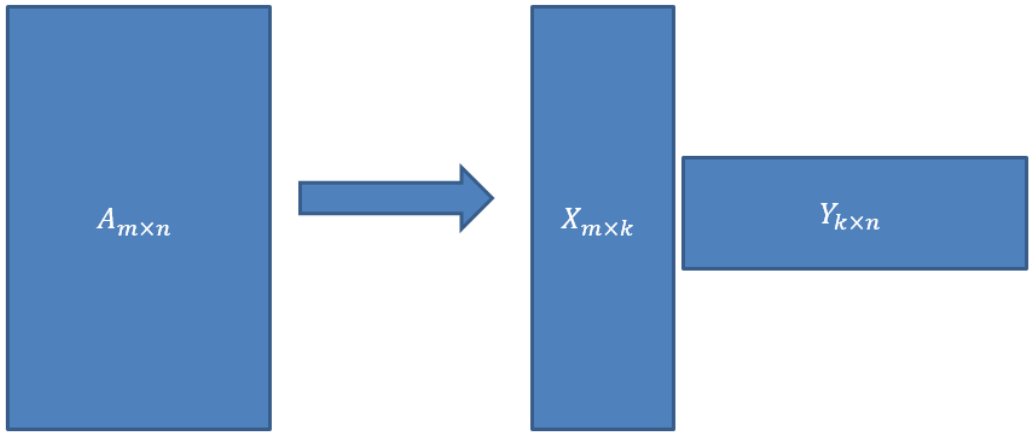




$$A = V\Sigma V^{-1}$$

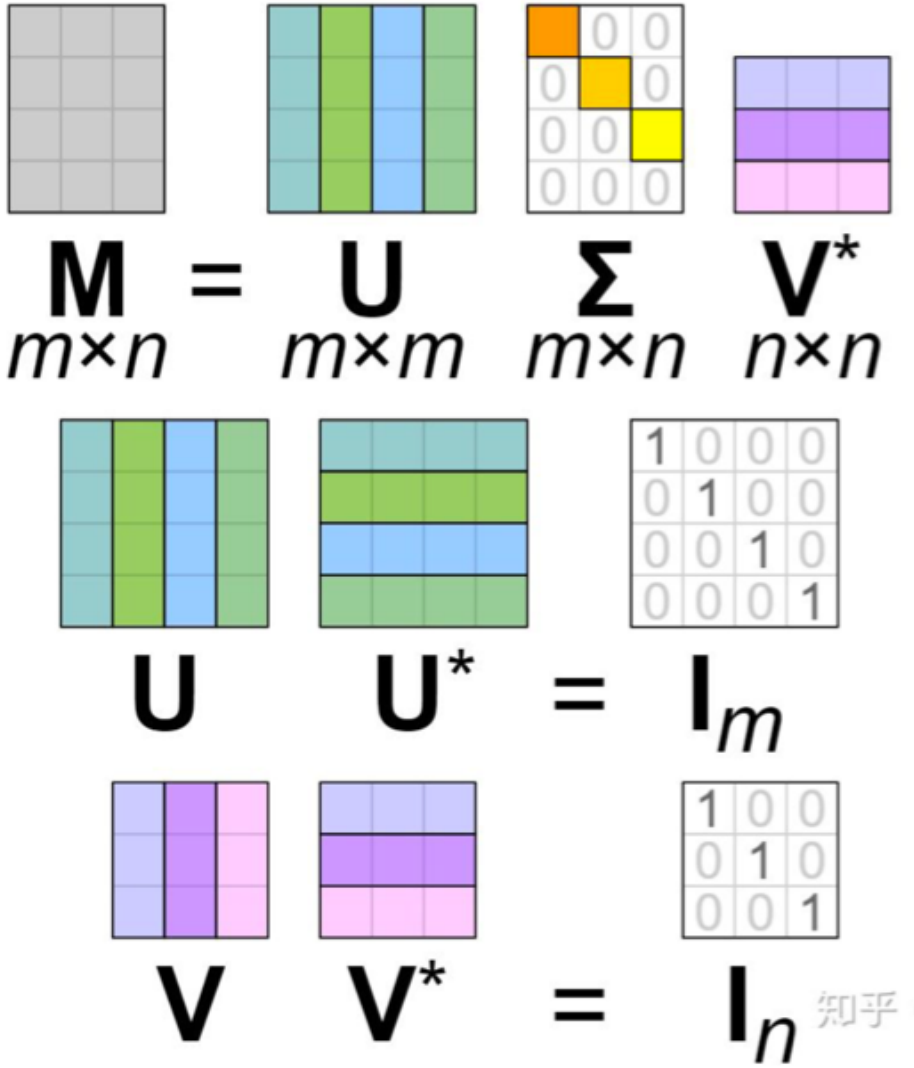
奇异值分解(SVD)

可以用**SVD**来证明对任意**M*N**的矩阵均存在如下分解：



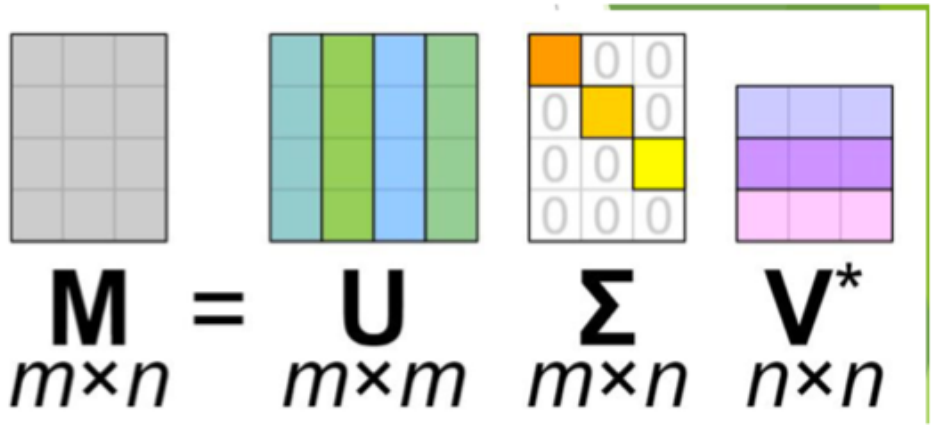
其中k=Rank(A)

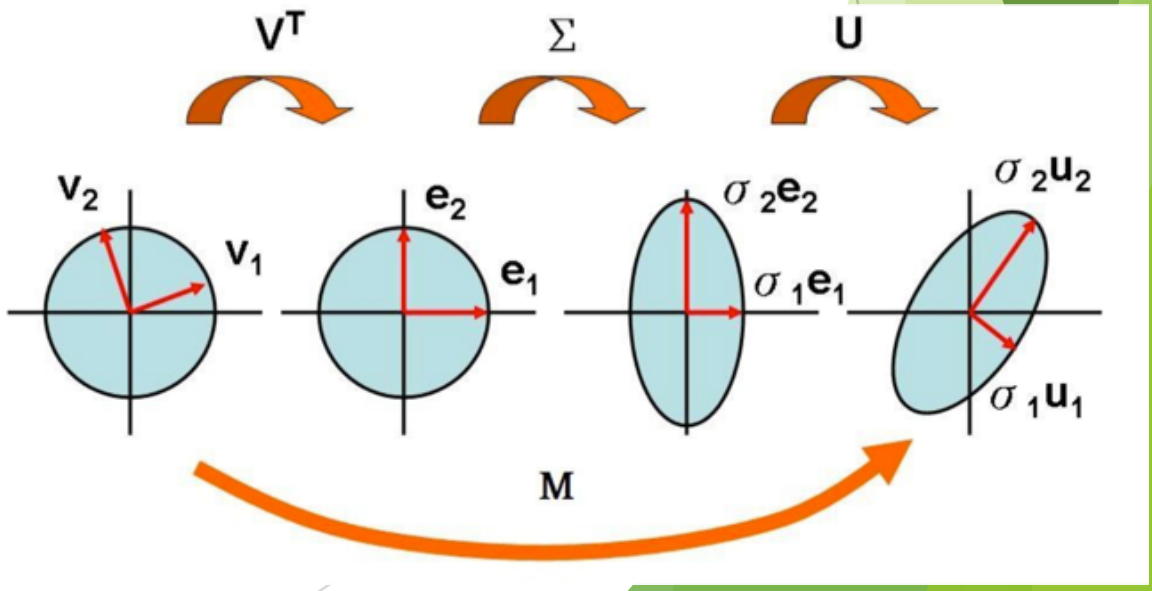
这个可以应用在数据降维压缩上！在数据相关性特别大的情况下存储X和Y矩阵比存储A矩阵占用空间更小！



SVD分解示例

- 线性代数里重要的一种分解形式，其矩阵的特殊含义可以用来做处理线性相关。
- 如在自然语言处理中，对新闻的分类，就可以采用SVD的方法，而且已取得不错的效果。把新闻中的核心词，用一个向量进行表示，每条新闻一个向量，组成一个矩阵，对矩阵进行SVD分解。如：可以用一个大矩阵A来描述这一百万篇文章和五十万词的关联性。这个矩阵中，每一行对应一篇文章，每一列对应一个词。



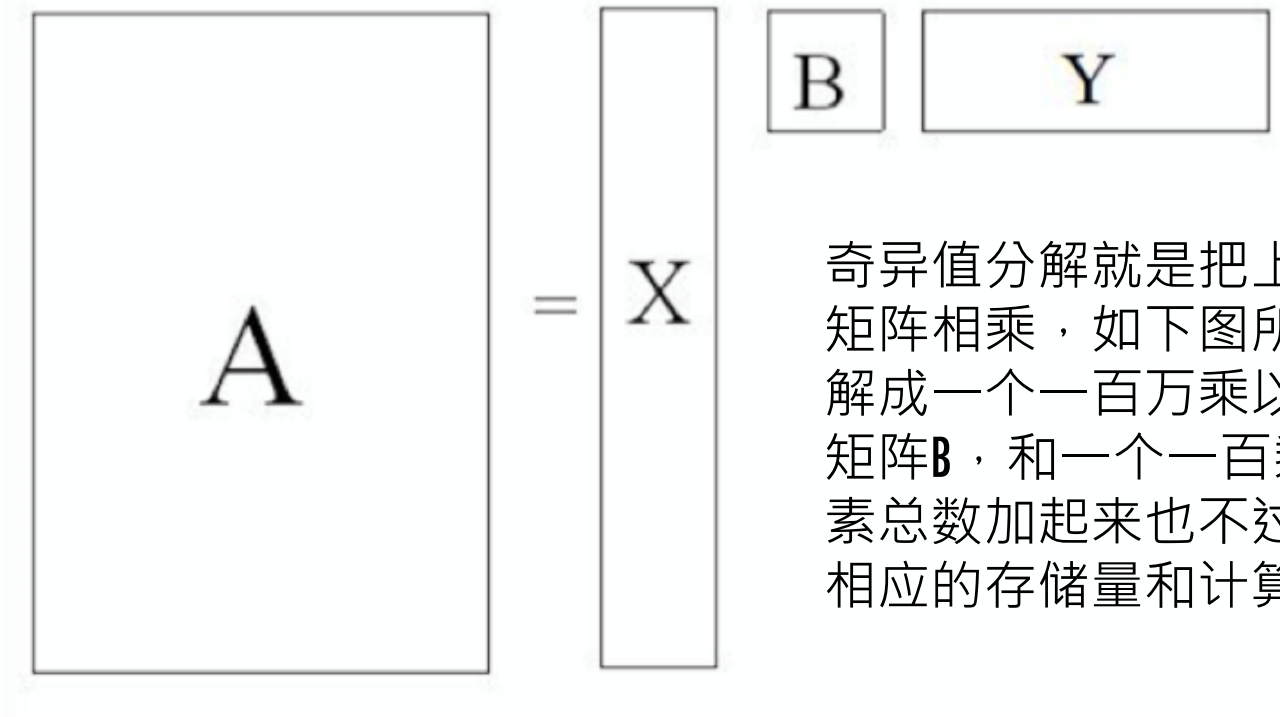


SVD新闻分类问题 - 1

$$A=\begin{pmatrix} a_{11} & \cdots & a_{1j} & \cdots & a_{1n} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{i1} & \cdots & a_{ij} & \cdots & a_{in} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{m1} & \cdots & a_{mj} & \cdots & a_{mn} \end{pmatrix}$$

在图中，M=1,000,000，N=500,000。第 i 行，第 j 列的元素，是字典中第 j 个词在第 i 篇文章中出现的加权词频 (比如，TF/IDF)。读者可能已经注意到了，这个矩阵非常大，有一百万乘以五十万，即五千亿个元素。

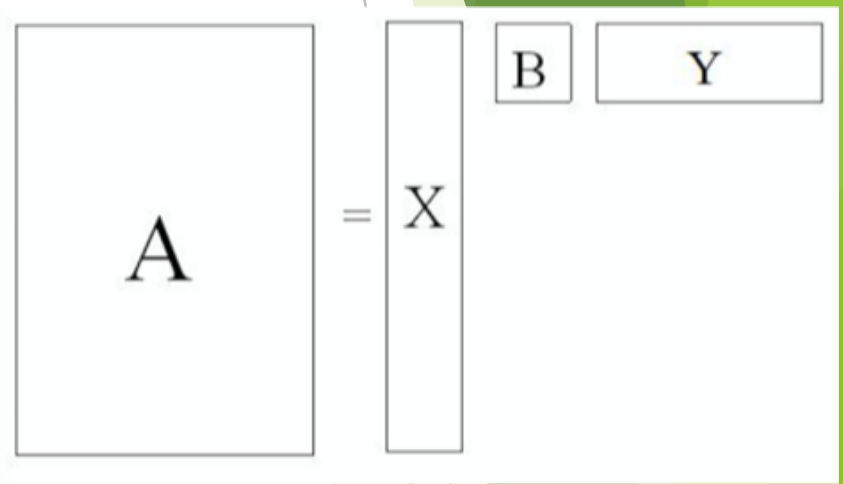
SVD新闻分类问题 - 2



奇异值分解就是把上面这样一个大矩阵，分解成三个小矩阵相乘，如下图所示。比如把上面的例子中的矩阵分解成一个一百万乘以一百的矩阵X，一个一百乘以一百的矩阵B，和一个一百乘以五十万的矩阵Y。这三个矩阵的元素总数加起来也不过1.5亿，仅仅是原来的三千分之一。相应的存储量和计算量都会小三个数量级以上。

SVD新闻分类问题 - 3

- **三个矩阵有非常清楚的物理含义：**
  - 第一个矩阵X中的每一行表示意思相关的一类词，其中的每个非零元素表示这类词中每个词的重要性（或者说相关性），数值越大越相关。
  - 第三个矩阵Y中的每一列表示同一主题一类文章，其中每个元素表示这类文章中每篇文章的相关性。
  - 第二个矩阵B则表示类词和文章之间的相关性。因此，我们只要对关联矩阵A进行一次奇异值分解，我们就可以同时完成了近义词分类和文章的分类。（同时得到每类文章和每类词的相关性）。

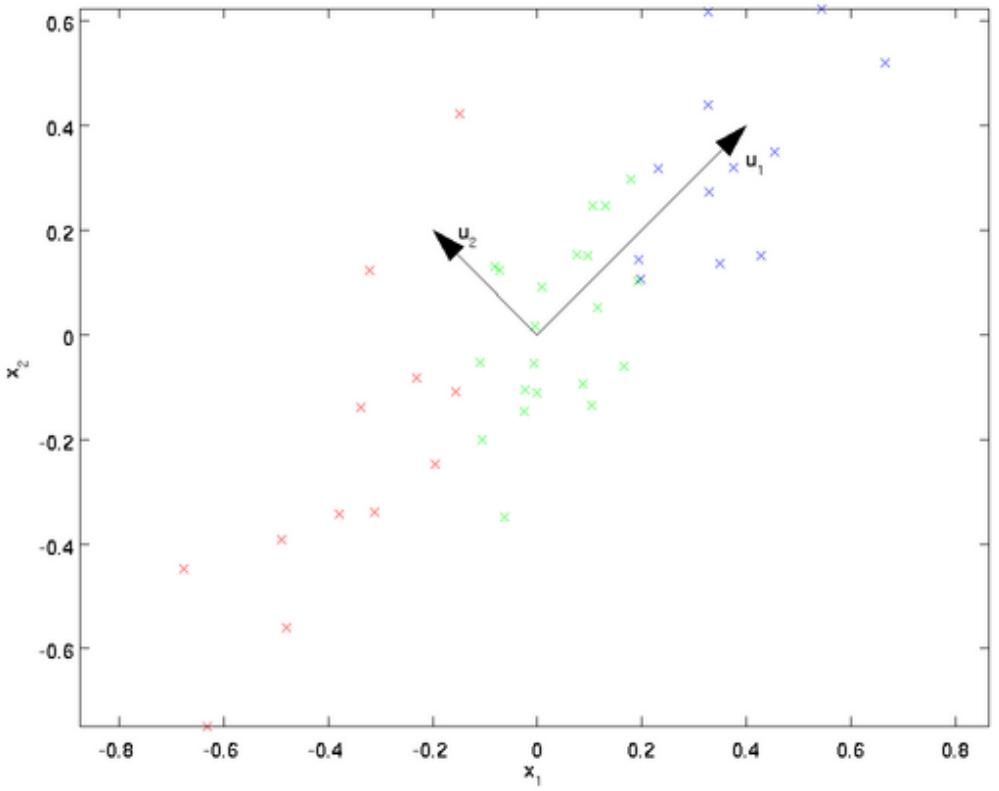


PCA算法

- 主成分分析（Principal components analysis，以下简称PCA）是最重要的降维方法之一。在数据压缩消除冗余和数据噪音消除等领域都有广泛的应用。
- PCA顾名思义，就是找出数据里最主要的方面，用数据里最主要的方面来代替原始数据。具体的，假如我们的数据集是n维的，共有m个数据

$$(x^{(1)},x^{(2)},\ldots,x^{(m)})$$

我们希望将这m个数据的维度从n维降到n'维，希望这m个n'维的数据集尽可能的代表原始数据集。



PCA几何含义展示

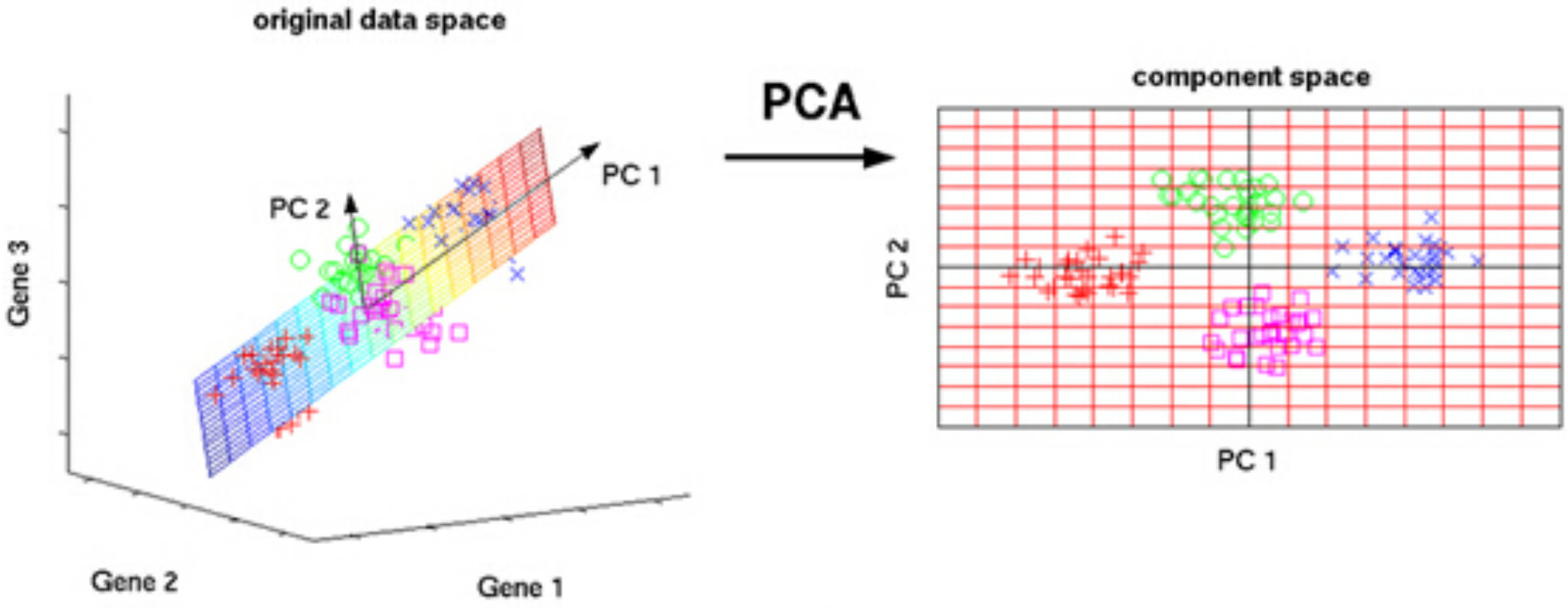


图 1

PCA算法总结

- 作为一个非监督学习的降维方法，它只需要特征值分解，就可以对数据进行压缩，去噪。因此在实际场景应用很广泛。为了克服PCA的一些缺点，出现了很多PCA的变种。

- PCA算法优点：
  - 仅仅需要以方差衡量信息量，不受数据集以外的因素影响。
  - 各主成分之间正交，可消除原始数据成分间的相互影响的因素。
  - 计算方法简单，主要运算是特征值分解，易于实现。

- PCA算法主要缺点：
  - 主成分各个特征维度的含义具有一定的模糊性，不如原始样本特征的解释性强。
  - 方差小的非主成分也可能含有对样本差异的重要信息，因降维丢弃可能对后续数据处理有影响。

PCA算法示例

PCA将n个特征降维到k个，可以用来进行数据压缩，例如100维的向量最后可以用10维来表示，那么压缩率为90%。

得到的均值图像mean_face：

前19个最大主元对应的“特征脸”：

PCA算法示例

测试：

测试用样本：

四副测试样本的重建误差分别为：

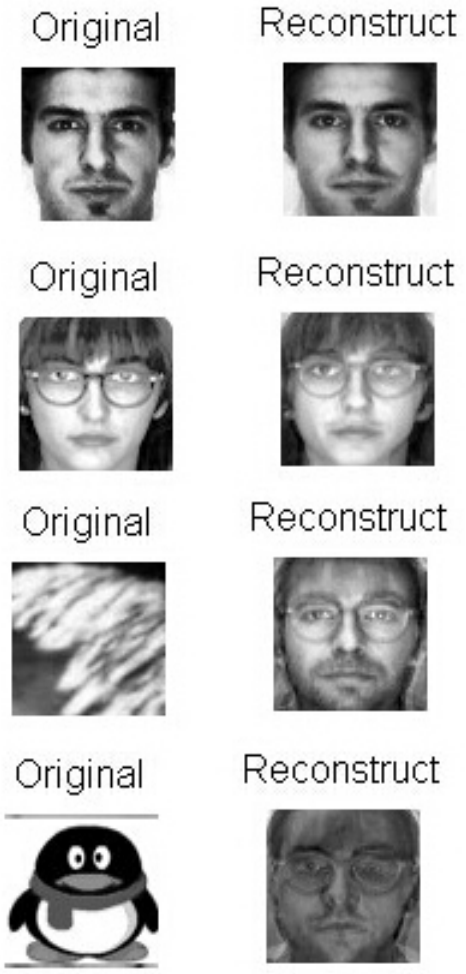
1.4195e+003

1.9564e+003

4.7337e+003

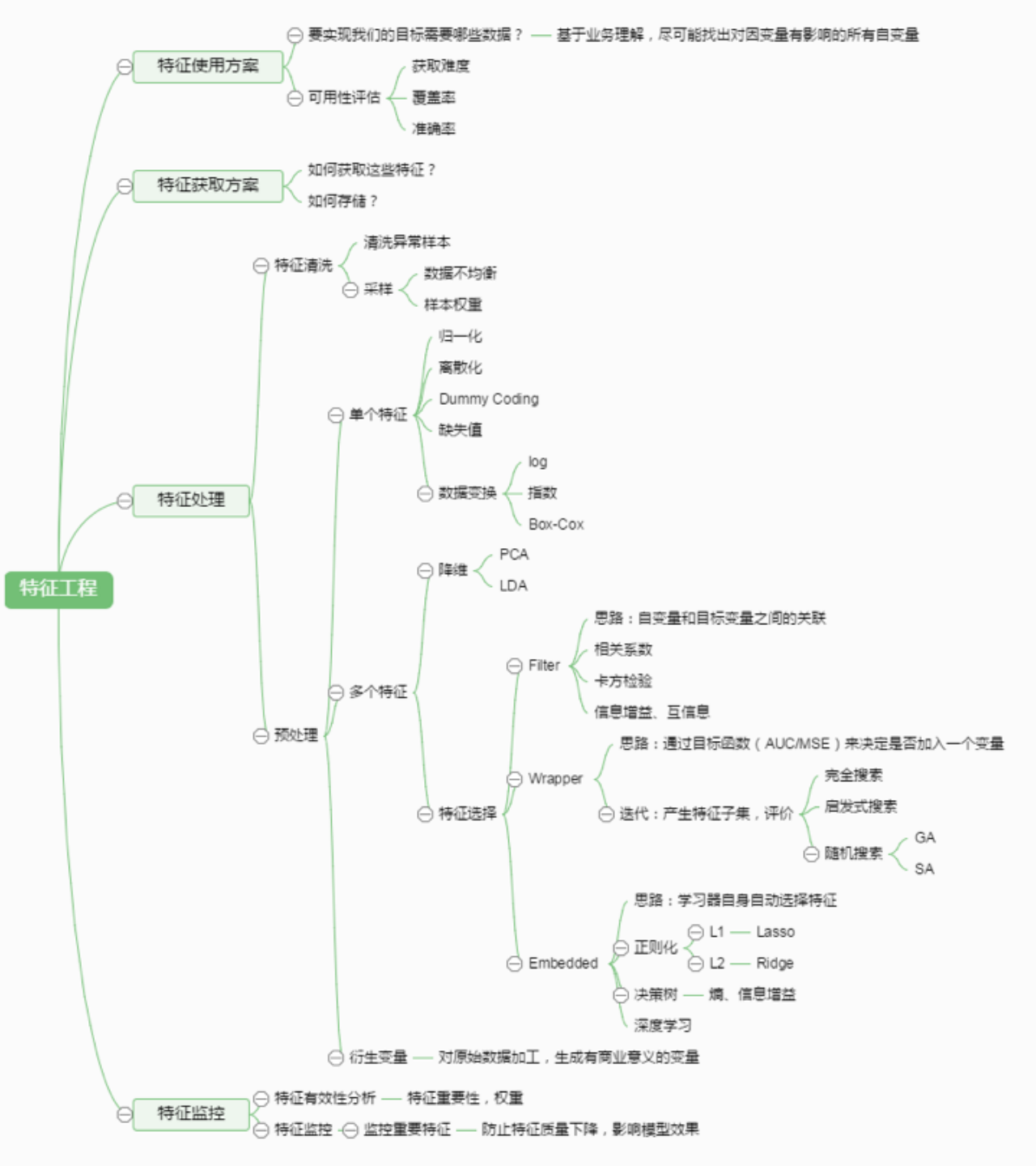
7.0103e+003

重建出的测试样本与原样本的对比：



特征工程

- 数据和特征决定了机器学习的上限，而模型和算法只是逼近这个上限而已
- 特征处理是特征工程的核心部分，包括数据预处理，特征选择，降维等。
- sklearn提供了较为完整的特征处理方法，



特征工程的重要性

- 我们提取得特征中有冗余特征，对模型的性能几乎没有帮助。
- 我们提取的特征中有些可以列为噪声（或者可以称为老鼠屎），对模型的性能不仅没有帮助，还会降低模型的性能。
- feature selection 的本质就是对一个给定特征子集的优良性通过一个特定的评价标准(evaluation criterion)进行衡量．通过特征选择，原始特征集合中的冗余（redundant）特征和不相关（irrelevant）特征被除去。而有用特征得以保留。

特征工程主要流程

- **画图**
- **预处理**
- **处理已有特征**
  - 类别特征、数值特征、时间特征、地理特征、用户特征
  - 缺失值处理
- **创造新特征**
  - 数值特征的简单变换
  - 类别特征与数值特征的组合
  - 用基因编程创造新特征（转换、回归）
  - 用决策树创造新特征

Sklearn常用特征工程方法

类名	功能	说明
StandardScaler	数据预处理（无量纲化）	标准化，基于特征矩阵的列，将特征值转换至服从标准正态分布
MinMaxScaler	数据预处理（无量纲化）	区间缩放，基于最大最小值，将特征值转换到[0, 1]区间上
Normalizer	数据预处理（归一化）	基于特征矩阵的行，将样本向量转换为“单位向量”
Binarizer	数据预处理（二值化）	基于给定阈值，将定量特征按阈值划分
OneHotEncoder	数据预处理（哑编码）	将定性数据编码为定量数据
Imputer	数据预处理（缺失值计算）	计算缺失值，缺失值可填充为均值等
PolynomialFeatures	数据预处理（多项式数据转换）	多项式数据转换
FunctionTransformer	数据预处理（自定义单元数据转换）	使用单变元的函数来转换数据
VarianceThreshold	特征选择（Filter）	方差选择法
SelectKBest	特征选择（Filter）	可选关联系数、卡方校验、最大信息系数作为得分计算的方法
RFE	特征选择（Wrapper）	递归地训练基模型，将权值系数较小的特征从特征集合中消除
SelectFromModel	特征选择（Embedded）	训练基模型，选择权值系数较高的特征
PCA	降维（无监督）	主成分分析法
LDA	降维（有监督）	线性判别分析法

数据预处理

- 不属于同一量纲：即特征的规格不一样，不能够放在一起比较。无量纲化可以解决这一问题。
- 信息冗余：对于某些定量特征，其包含的有效信息为区间划分，例如学习成绩，假若只关心“及格”或不“及格”，那么需要将定量的考分，转换成“1”和“0”表示及格和未及格。二值化可以解决这一问题。
- 定性特征不能直接使用：某些机器学习算法和模型只能接受定量特征的输入，那么需要将定性特征转换为定量特征。最简单的方式是为每一种定性值指定一个定量值，但是这种方式过于灵活，增加了调参的工作。通常使用哑编码的方式将定性特征转换为定量特征：假设有N种定性值，则将这一个特征扩展为N种特征，当原始特征值为第i种定性值时，第i个扩展特征赋值为1，其他扩展特征赋值为0。哑编码的方式相比直接指定的方式，不用增加调参的工作，对于线性模型来说，使用哑编码后的特征可达到非线性的效果。
- 存在缺失值：缺失值需要补充。
- 信息利用率低：不同的机器学习算法和模型对数据中信息的利用是不同的，之前提到在线性模型中，使用对定性特征哑编码可以达到非线性的效果。类似地，对定量变量多项式化，或者进行其他的转换，都能达到非线性的效果。

哑编码(OneHotEncoder)

- 哑变量（Dummy Variable），又称为虚拟变量、虚设变量或名义变量，从名称上看就知道，它是人为虚设的变量，通常取值为0或1，来反映某个变量的不同属性。对于有n个分类属性的自变量，通常需要选取1个分类作为参照，因此可以产生n-1个哑变量。
- 哑编码（哑变量 dummy variable）是因为大部分算法是基于向量空间中的度量来进行计算的，为了使非偏序关系的变量取值不具有偏序性，并且到圆点是等距的。举例来说，如果定义颜色变量“红=1，黄=2，蓝=3”，相当于在向量空间中定义了“红＜黄＜蓝”，这与事实是不符的，并且每个值到圆点的距离是不同的，这会影响到基于向量空间度量算法的效果。
- **One-hot是对低维高信息量的特征在高维中进行打散，使之在模型中更容易被优化/学习。也是ML中常用套路“先升维”的一个方案。**

举一个例子，如职业因素，假设分为学生、农民、工人、公务员、其他共5个分类，其中以“其他职业”作为参照，此时需要设定4哑变量X1-X4，如下所示：

X1=1，学生；X1=0，非学生；

X2=1，农民；X2=0，非农民；

X3=1，工人；X3=0，非工人；

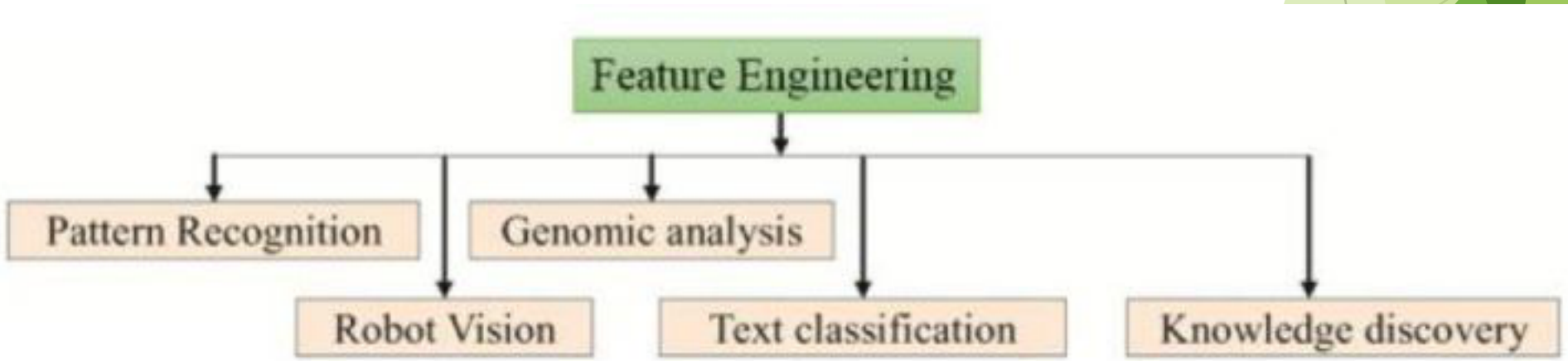
X4=1，公务员；X4=0，非公务员；

那么对于每一种职业分类，其赋值就可以转化为以下形式：

	X1	X2	X3	X4
学生	1	0	0	0
农民	0	1	0	0
工人	0	0	1	0
公务员	0	0	0	1

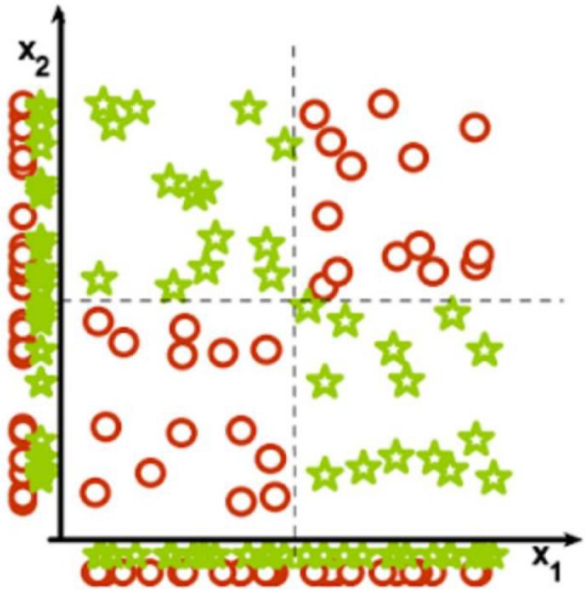
特征工程

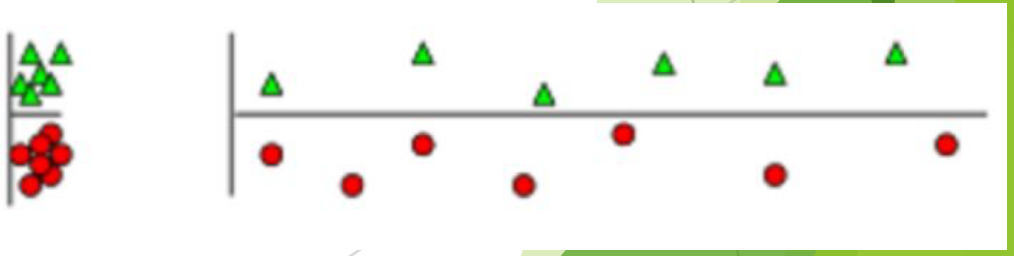
- 特征选择（Feature Selection,FS）和特征抽取（Feature Extraction, FE）是特征工程（Feature Engineering）的两个重要的方面。他们之间最大的区别就是是否生成新的属性。FS仅仅对特征进行排序（Ranking）和选择，FE更为复杂，需要重新认识事物，挖掘新的角度，创新性的创立新的属性，而目前深度学习这么火，一个很重要的原因是缩减了特征提取的任务。不过，目前特征工程依然是各种机器学习应用领域的重要组成部分。
- 处理已有特征、创造新的特征



为什么需要特征选择

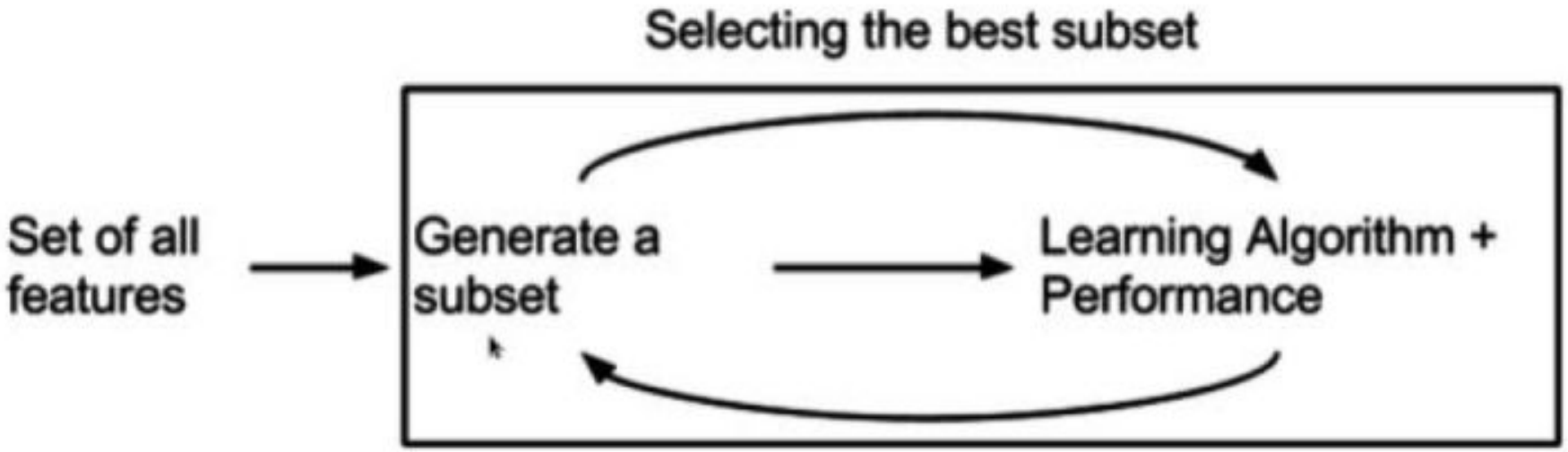
- 特征选择与分类器性能的关系
  - 一般说来，当固定一个分类器的话，所选择的特征数量和分类器的效果之间会有一个曲线，在某个x（1<=x<=n）的地方，会达到最优。
- 特征少了会导致无法区分的情况发生
- 特征多了也不行
- 降低了模型的复杂度，节省了大量计算资源以及计算时间
- 提高了模型的泛化能力





引入了横轴的特征

特征选择

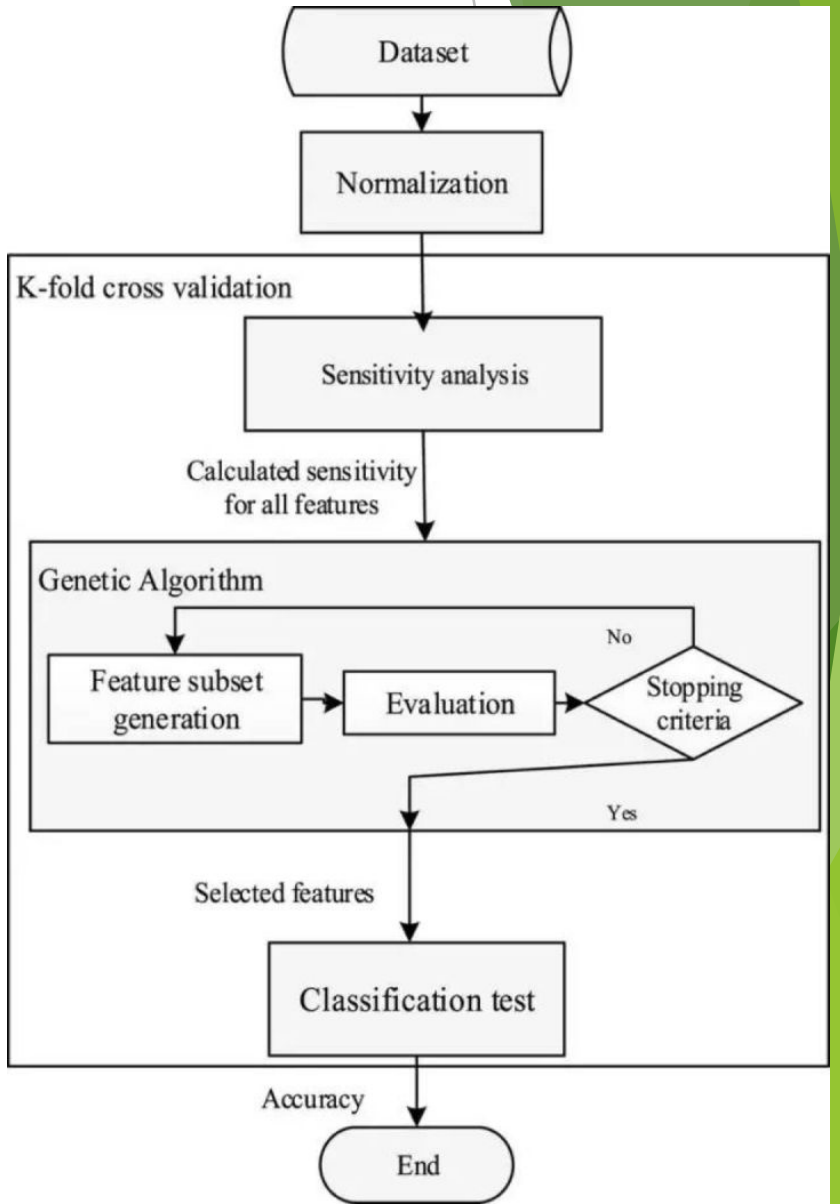


特征选择需要找到一个最优的特征子集， 那么特征选择的的一般流程就是， 找一个集合，然后针对某个学习算法，测试效果如何， 一直循环直到找到最优集合为止。

特征选择搜索常用手段

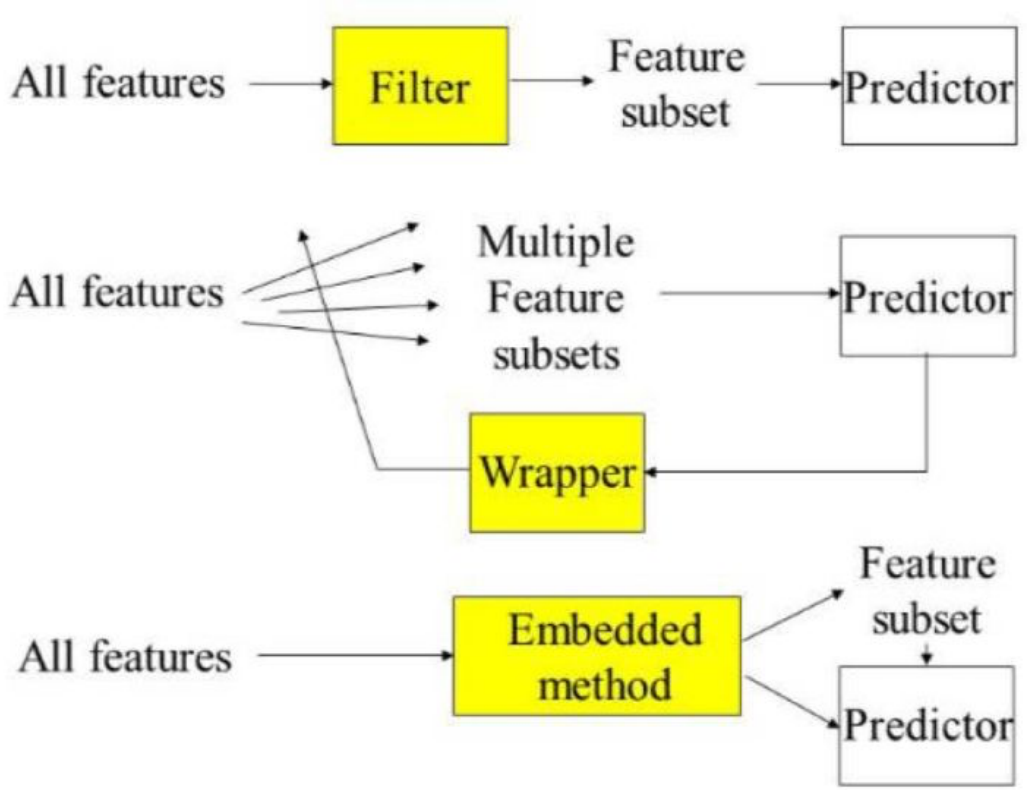
- 穷举法（Exhaustive）：暴力穷尽
- 贪心法（Greedy Selection）：线性时间
- 模拟退火（Simulated Annealing）：随机尝试找最优
- 基因算法（Genetic Algorithm）：组合深度优先尝试
- 邻居搜索（Variable Neighbor Search）：利用相近关系搜索

特征数量小于10个
和Cross-Validation进行评价结合



特征选择经典三刀

- 过滤式（Filter）
  - 快速无比，效果各异
- 包裹式（Wrapper）
  - 需要一定的技巧
- 嵌入式（Embedded）
  - 模式单调，快速并且效果明显，但是如何参数设置，需要深厚的背景知识。



Filter 过滤式 - 1

- 顾名思义，就是要基于贪心的思想， 把需要的特征筛/滤出来。 一般说来， 基于贪心就需要对特征进行打分。 而这个打分可以基于领域知识， 相关性， 距离，缺失， 稳定性 等等。

单一特征选择

- **Welch's t-Test**： 来判断两个属性的分布的均值方差距离

$$t = \frac{\overline{X}_1 - \overline{X}_2}{\sqrt{\frac{s_1^2}{N_1} + \frac{s_2^2}{N_2}}}$$

- **Fish-Score**： 和**Welch's t-Test**类似， 计算两个分布的距离， 均值只差和方差之和的距离。

$$S_i = \frac{\sum_{k=1}^{K} n_j(\mu_{ij} - \mu_i)^2}{\sum_{k=1}^{K} n_j \rho_{ij}^2},$$

- **Chi-Squared test**： 计算类别离散值之间的相关性

$$S_i = \frac{\sum_{k=1}^{K} n_j(\mu_{ij} - \mu_i)^2}{\sum_{k=1}^{K} n_j \rho_{ij}^2},$$

- **Information Gain**：计算两个划分的一致性

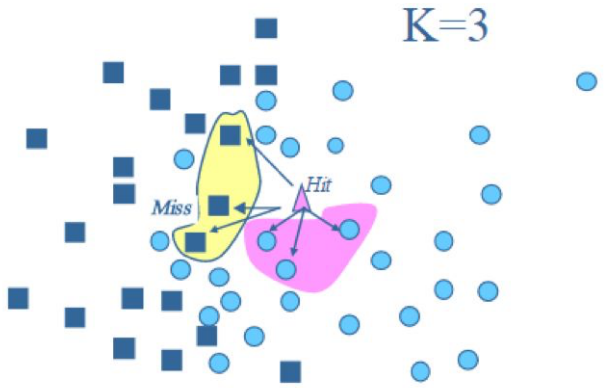
$$\chi^2 = \sum_{i=1}^{r} \sum_{j=1}^{c} \frac{(O_{i,j} - E_{i,j})^2}{E_{i,j}}$$

Filter 过滤式 - 2

多特征选择

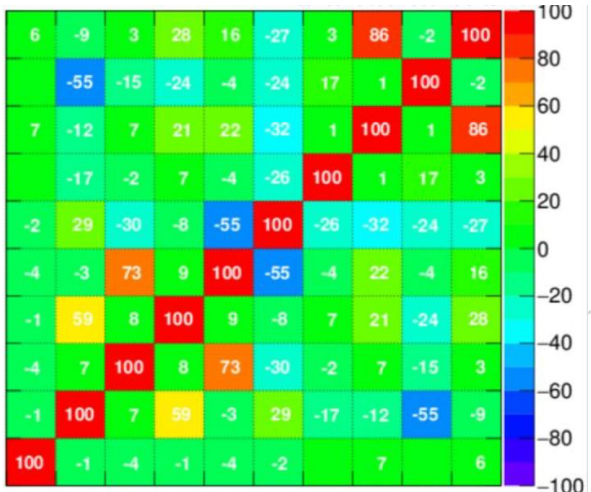
- **Relief-F**： 根据随机选择的样本点，来计算属性之间的相关性。

$$S_i = \frac{1}{2}\sum_{k=1}^{\ell} d(\mathbf{X}_{ik} - \mathbf{X}_{iM_k}) - d(\mathbf{X}_{ik} - \mathbf{X}_{iH_k}),$$



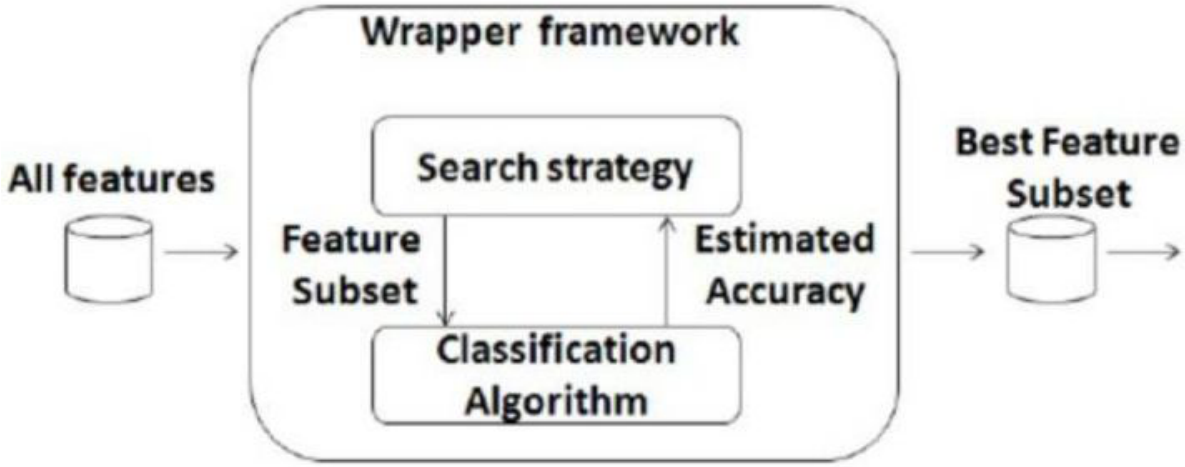
- **Correlation Feature Selection (CFS)**： 利用属性之间的相关性， 进行选择。

$$\mathrm{CFS} = \max_{S_k}\left[\frac{r_{cf_1} + r_{cf_2} + \cdots + r_{cf_k}}{\sqrt{k + 2(r_{f_1f_2} + \cdots + r_{f_if_j} + \cdots + r_{f_kf_1})}}\right].$$



Wrapper 包裹式

- 就是先选定特定算法，然后再根据算法效果来选择特征集合。 一般会选用普遍效果较好的算法， 例如Random Forest， SVM， kNN等等。
  - Forward Selection： 挑出一些属性， 然后慢慢增大挑出的集合。
  - Backward Elimination： 删除一些属性，然后慢慢减小保留的集合。



Embedded 嵌入式

- 利用正则化思想，将部分特征属性的权重变成零。常见的正则化有L1的Lasso，L2的Ridge和混合的Elastic Net。其中L1的算子有明显的特征选择的功能。
- 比较简单的就是会自动进行特征选择，而且一次性就搞定了，速度也不错，难点就是损失函数的选择和缩放参数的选择。