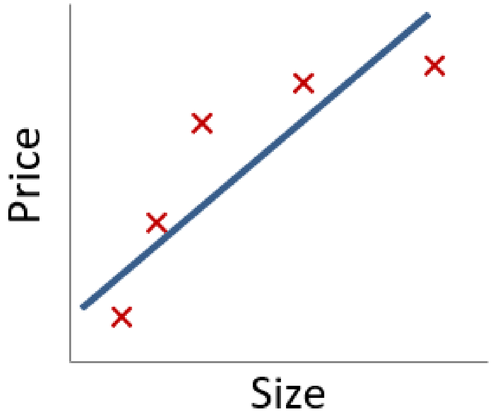
Regularization

正则化

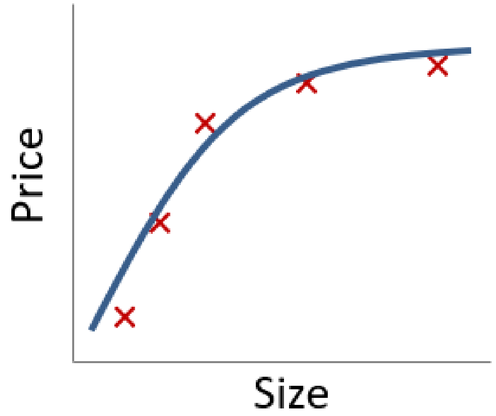
Outline

- 欠拟合与过拟合
- L2正则化——Ridge Regression
- L1正则化——LASSO
- 基于R的实现

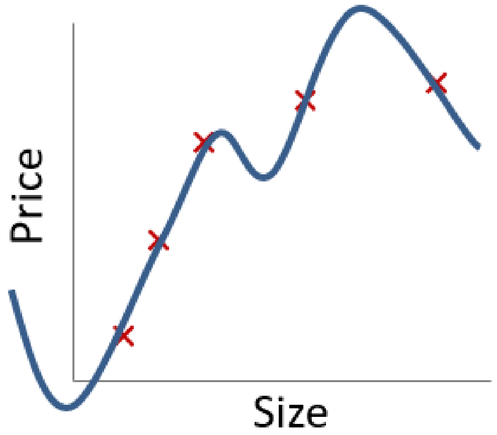
欠拟合与过拟合



$f(x) = \beta_0 + \beta_1 x$



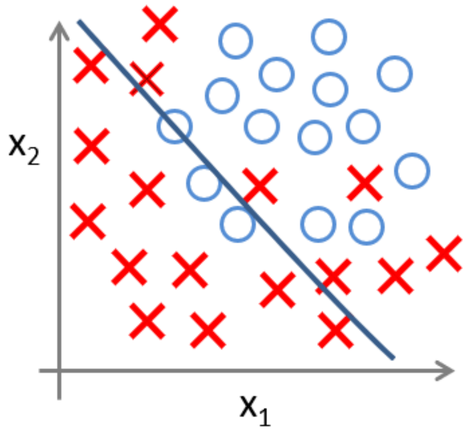
$f(x) = \beta_0 + \beta_1 x + \beta_2 x^2$



$f(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 x^4$

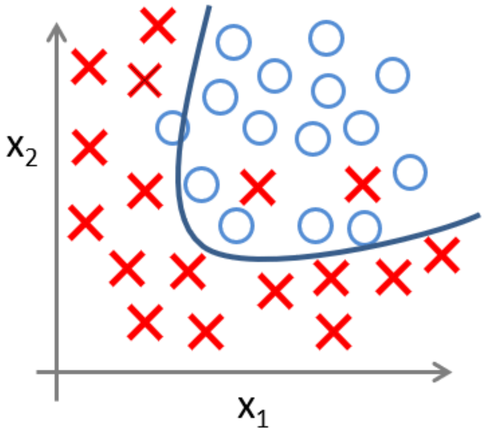
- 欠拟合(Underfitting)：模型过于简单，算法对训练集的拟合效果都很差，更别提在测试集泛化能力。
- 过拟合(Overfitting)：当我们拥有海量的特征，或训练样本很少时，算法对训练集的拟合效果非常好，但却无法准确预测新的样本，泛化能力很差。

欠拟合与过拟合——分类场景

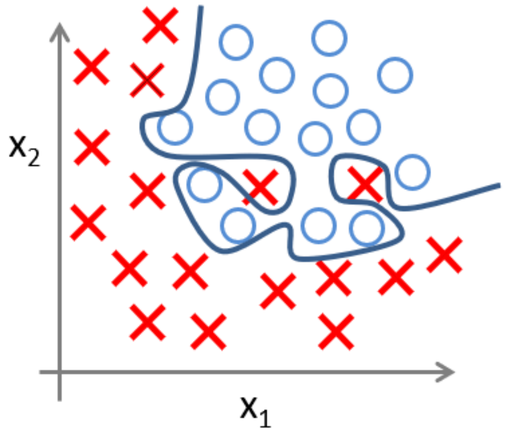


$f(x) = g(\beta_0 + \beta_1 x_1 + \beta_2 x_2)$

g为logistic函数



$f(x)$
$= g(\beta_0 + \beta_1 x_1 + \beta_2 x_2$
$+ \beta_3 x_1^2 + \beta_4 x_2^2 + \beta_5 x_1 x_2)$

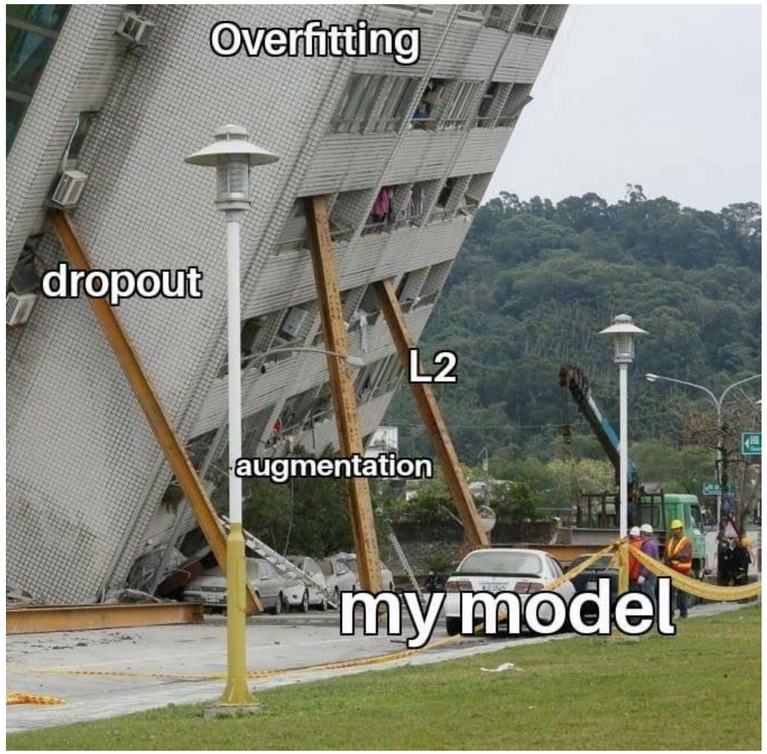


$f(x)$
$= g(\beta_0 + \beta_1 x_1 + \beta_2 x_2$
$+ \beta_3 x_1^2 x_2 + \beta_4 x_1^2 x_2^2 + \beta_5 x_1^2 x_2^3$
$+ \beta_6 x_1^3 x_2 + \cdots)$

过拟合


THE BEST WAY TO
EXPLAIN OVERFITTING


A GOOD EXAMPLE OF
OVERFITTING


Overfitting
dropout
L2
augmentation
my model

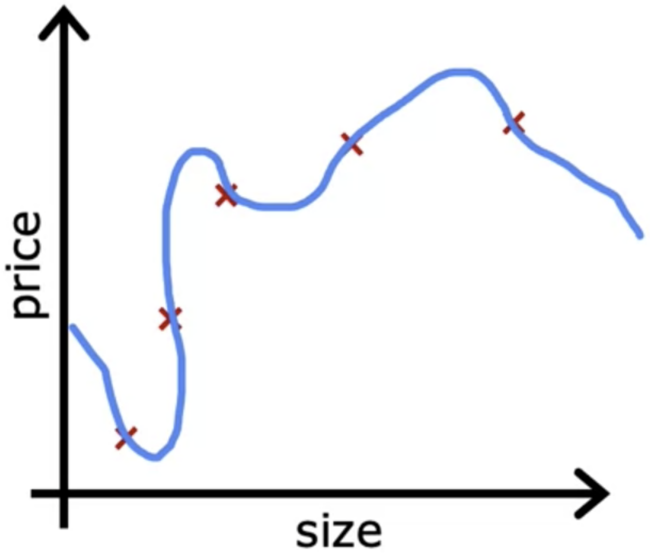
如何解决欠拟合？

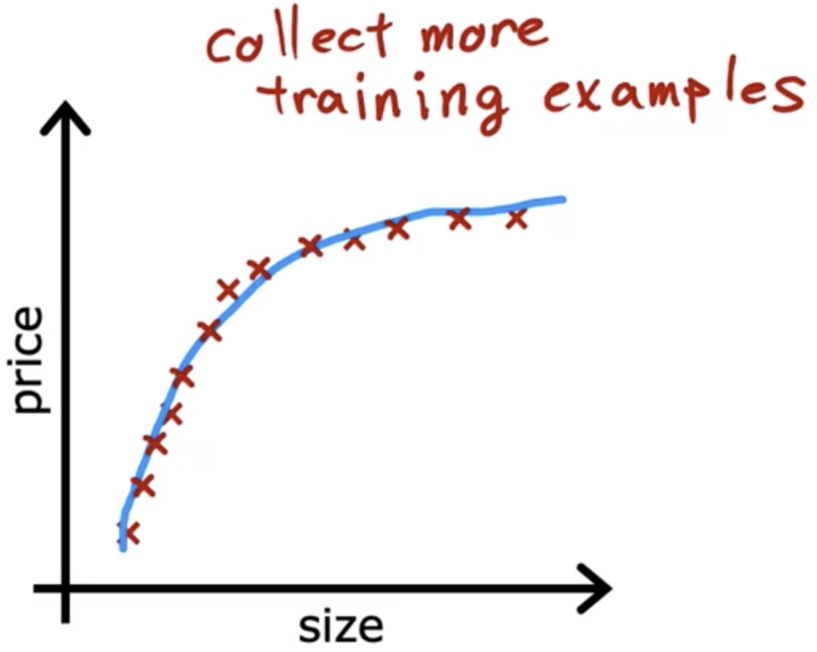
- 增加特征数量
- 增加训练时间
- 增加模型复杂度

如何解决过拟合？

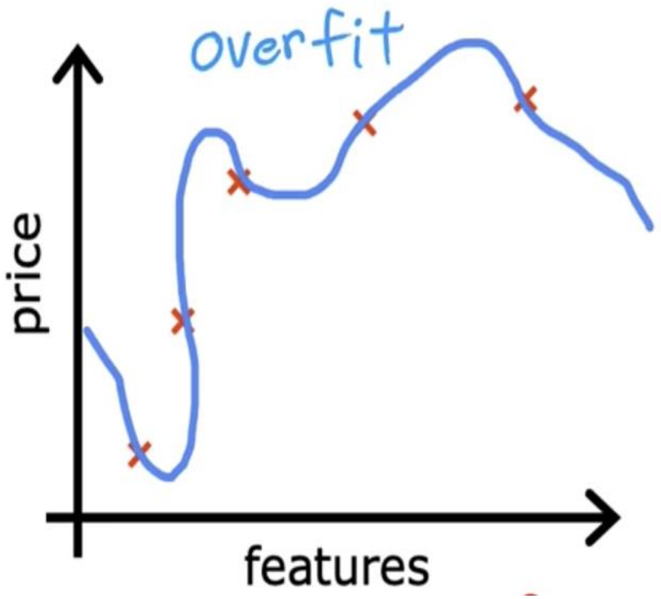
- 增加样本/数据增强
- 提前终止训练
- Dropout
- 正则化

增加样本

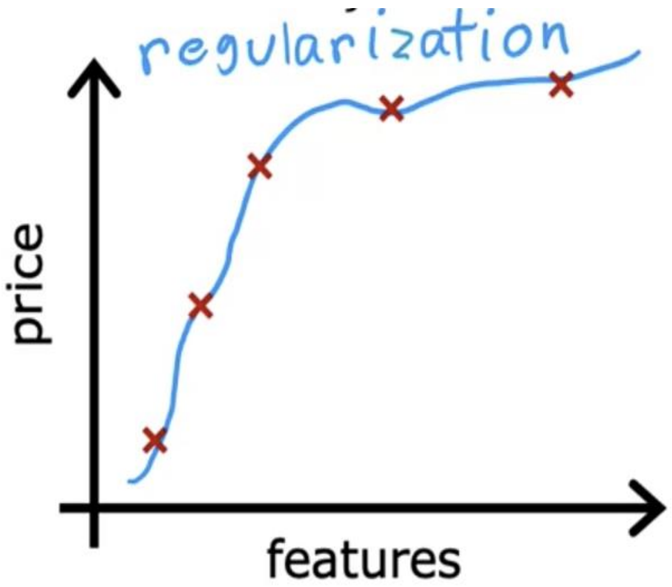




正则化



$f(x) = 100 + 28x - 385x^2 + 39x^3 - 174x^4$



$f(x) = 10 + 13x - 0.23x^2 + 0.000014x^3 - 0.0001x^4$

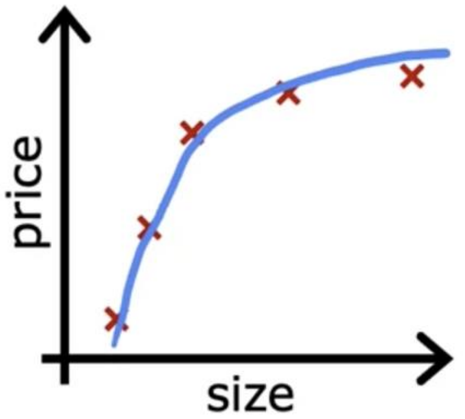
正则化

- 正则化方法基于所有的 d 个特征建立模型。
- 然而，相对于原始估计（如最小二乘估计）来说，参数都朝着零的方向有所缩减。
- 这样的缩减有着减小方差的作用。
- 取决于缩减方式的不同，有些系数最终可能等于零，有着变量选择的效果。

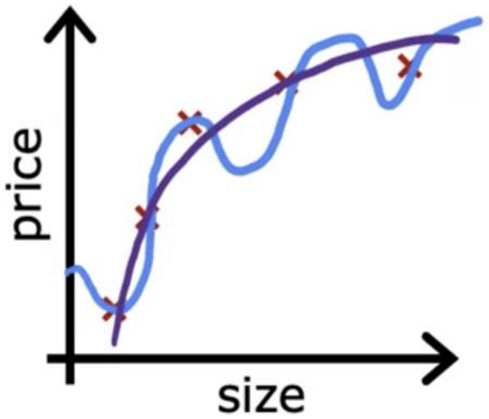
正则化——Intuition

- 为避免过拟合，我们要使β_3和β_4非常小（≈ 0）
- 为此，我们可以定义一个新的损失函数：

$$\frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)}-\boldsymbol{\beta}^{T}\boldsymbol{x}^{(i)}\right)^2+1000\beta_3^2+1000\beta_4^2$$



$f(x)=\beta_0+\beta_1x+\beta_2x^2$



$f(x)=\beta_0+\beta_1x+\beta_2x^2+\beta_3x^3+\beta_4x^4$

- 本质上，对β_3^2和β_4^2很大的模型施加了惩罚（penalty）：如果想要最小化损失函数，唯一的可能途径就是β_3和β_4都很小；否则，$1000\beta_3^2+1000\beta_4^2$就会非常大
- 因此，当最小化上述损失函数时，会得到$\beta_3\approx 0$以及$\beta_4\approx 0$。我们有效地几乎完全剔除了x^3和x^4的效应
- 最终的模型近似等于$\beta_0+\beta_1x+\beta_2x^2$，有效地解决了过拟合问题
- 在实际中，可能有茫茫多的特征，不知道该去对哪些具体的特征施加惩罚。怎么办？

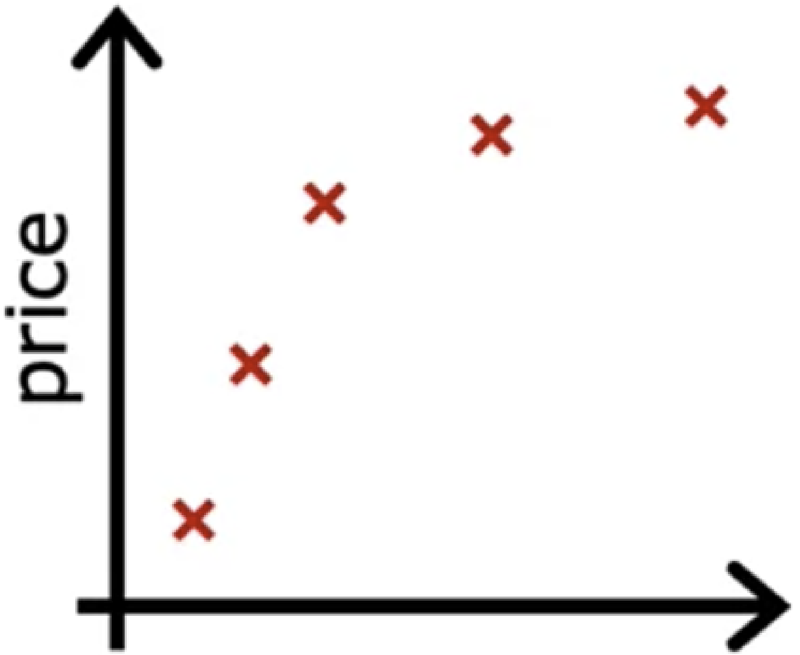
更一般的...

size	bedrooms	floors	age	avg income	...	distance to coffee shop	price
x_1	x_2	x_3	x_4	x_5		x_{100}	y

- 当特征很多的时候，很难知道到底该对哪一个特征施加惩罚。
- 正则化的做法通常是对所有的特征施加惩罚，更精确的说，对$\beta_1, \dots, \beta_d$施加惩罚。
- 例如：$\frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} - \boldsymbol{\beta}^{T}\boldsymbol{x}^{(i)}\right)^2 + \lambda\sum_{j=1}^{d}\beta_j^2$
- 我们通常不对β_0施加惩罚。在实际中，是否对其施加惩罚几乎没有影响。
- 最小化上面的损失函数包含了对两个目标的平衡：
  - 最小化第一项，鼓励算法通过最小化残差平方和去尽量准确地拟合训练数据
  - 最小化第二项，鼓励算法将参数保持在较小的水平，这会减轻过拟合
- $\lambda \geq 0$决定了两个目标之间的平衡，或相对重要性。

λ的作用

- λ决定了两个目标之间的平衡，或相对重要性。
- 考虑模型$f(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 x^4$
- 若$\lambda = 0$，没有任何正则化
- 若$\lambda = 10^{10}$，对正则化项赋予了极大的权重，迫使算法把参数调整为接近于0
- 我们要做的：选取一个恰当的λ，很好的平衡两个目标
- 选取λ：交叉验证



正则化——Overview

- 正则化的思想核心在于调整损失函数 L。我们在原始的损失函数的基础上增加一个**正则项**，目的是对模型参数的一些性质进行惩罚：

$$L_{new}(\boldsymbol{\beta}) = L(\boldsymbol{\beta}) + \lambda \cdot R(\boldsymbol{\beta})$$

- $\lambda \geq 0$ 是一个标量，控制了正则项的权重或重要性。
- 这里的损失函数 L 可以是平方损失函数、逻辑回归损失函数、Softmax回归损失函数…
- 线性回归正则化：$L_{new}(\boldsymbol{\beta}) = \frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} - \boldsymbol{\beta}^{\boldsymbol{T}}\boldsymbol{x}^{(i)}\right)^2 + \lambda \cdot R(\boldsymbol{\beta})$
- 逻辑回归正则化：$L_{new}(\boldsymbol{\beta}) = -\frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)}\log\hat{y}^{(i)} + \left(1-y^{(i)}\right)\log\left(1-\hat{y}^{(i)}\right)\right) + \lambda \cdot R(\boldsymbol{\beta})$
- 使用$L_{new}(\boldsymbol{\beta})$ 拟合模型可以使得最终的模型参数满足我们想要的性质（由$R(\boldsymbol{\beta})$定义）。

练习

- 考虑损失函数：$\frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)}-\boldsymbol{\beta}^{T}\boldsymbol{x}^{(i)}\right)^{2}+\lambda\sum_{j=1}^{d}\beta_{j}^{2}$。增大 λ 会倾向于：

A. 正则化的程度减弱

B. 正则化的程度加强

C. 增大参数$\beta_1,\ldots,\beta_d$的尺度

D. 减小参数$\beta_1,\ldots,\beta_d$的尺度

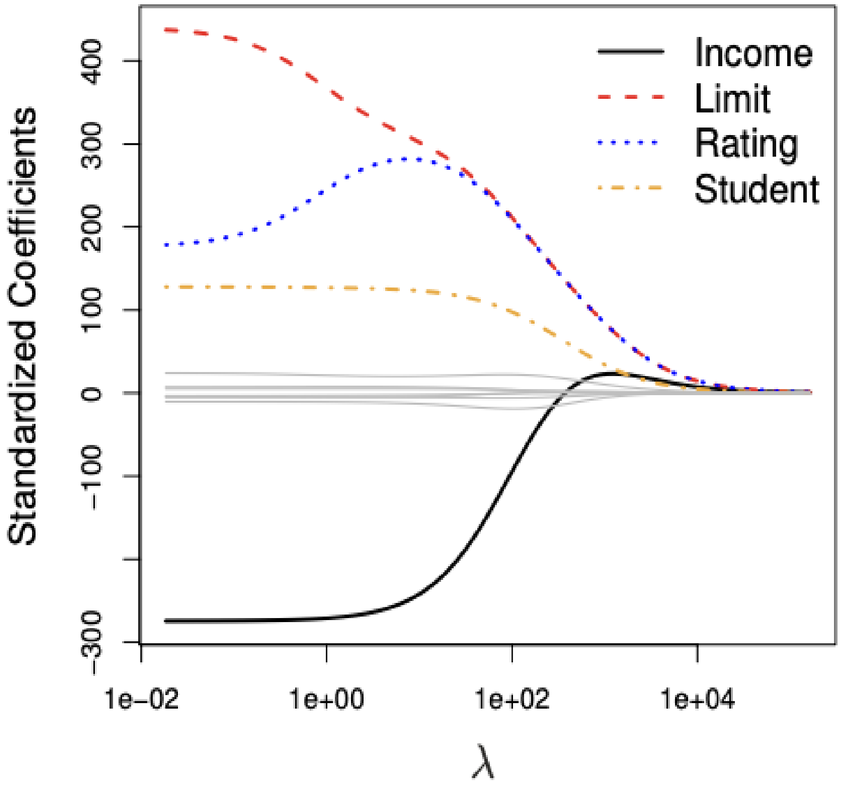
岭回归(Ridge Regression, 1970)

- 我们假设特征已经标准化：$\frac{1}{n}\sum_i x_j^{(i)} = 0$，$\frac{1}{n}\sum_i \left(x_j^{(i)}\right)^2 = 1$
- 此时，$\hat{\beta}_0 = \bar{y} = \frac{1}{n}\sum_i y^{(i)}$，不失一般性我们假设$\bar{y} = 0$，并因此省略$\beta_0$

- $L_{ridge}(\boldsymbol{\beta}) = \frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} - \boldsymbol{\beta^T x^{(i)}}\right)^2 + \lambda\sum_{j=1}^{d}\beta_j^2$
- 又称L2正则化：$\sum_{j=1}^{d}\beta_j^2 = \|\boldsymbol{\beta}\|_2^2$
- 存在显式解：$\hat{\beta}_{ridge} = (\boldsymbol{X^T X} + \lambda \boldsymbol{I})^{-1}\boldsymbol{X^T Y}$，这也是其名字岭回归的由来。

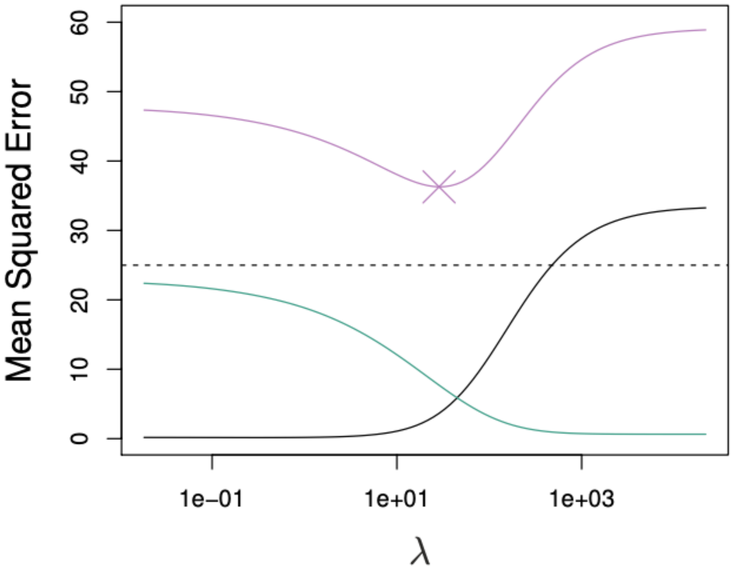
例——Credit Card Balance

- 我们对信用卡账单和很多特征进行岭回归。
- 图中每一条线为每个特征的参数随 λ 的变化情况。
- λ接近0时，岭回归的参数估计和最小二乘几乎一样。
- 随着λ的增大，岭回归的参数整体上会趋向于0，不排除个别参数有局部增大的情况。



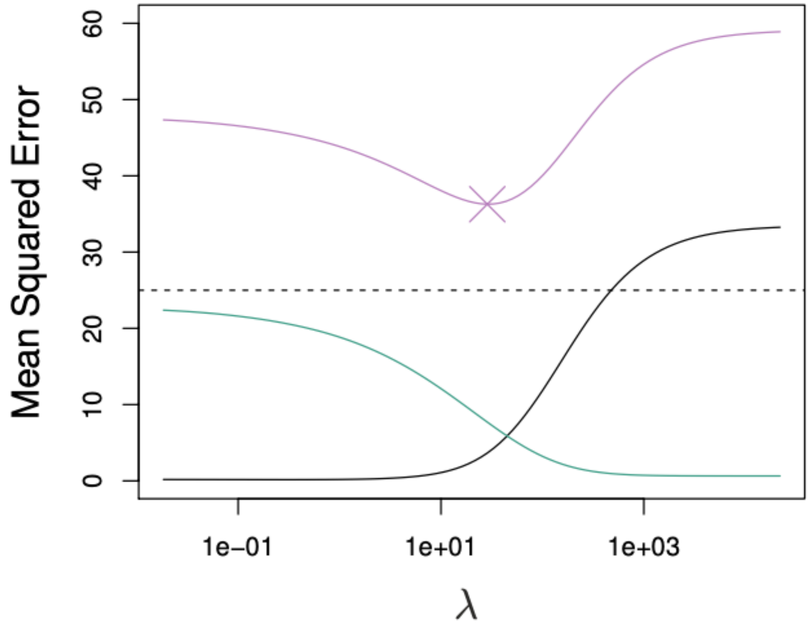
岭回归 & 最小二乘

- 最小二乘估计是无偏估计，而岭回归是有偏估计
- 岭回归的优势——**方差偏差平衡**
- 随着λ的增大，岭回归的flexibility不断下降，随之而来的是更小的方差和更大的偏差。
- 右图是一个人造的例子：$d = 45\ n = 50$
- 我们可以发现，$\lambda = 0$时，偏差（黑线）很小，方差（绿线）很大。
- 随着λ的增大，偏差缓慢上升，方差迅速下降，导致测试误差（粉线）下降。λ过大的时候，方差缓慢下降，偏差迅速上升，导致测试误差上升。



岭回归 & 最小二乘

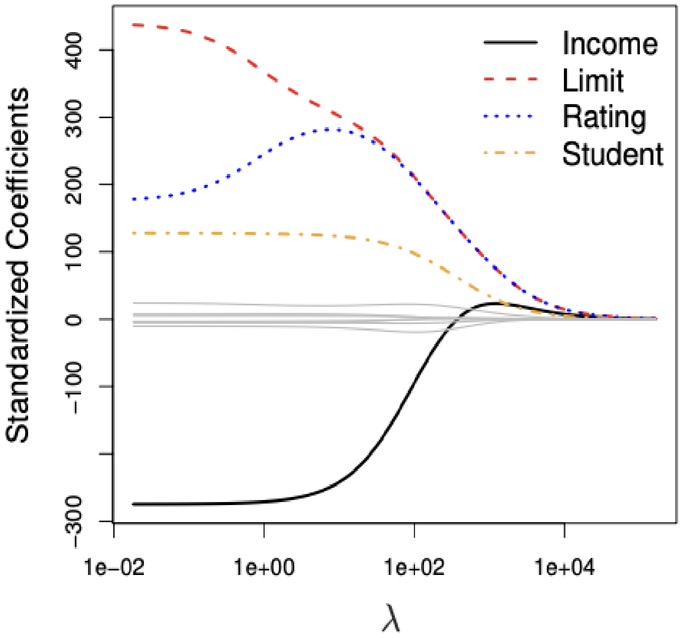
- 总结：岭回归通过正则项，牺牲了偏差，换取了测试误差的下降，从而可以有效缓解过拟合问题。



- 当$d > n$时，$\boldsymbol{X}$不再是满秩矩阵，$\boldsymbol{X}^T\boldsymbol{X}$不再可逆，导致最小二乘估计失效。岭回归也可以有效的解决这一问题。
- 事实上，这是岭回归被提出的初衷。

岭回归的缺点

- 岭回归最终得到的模型仍包括所有的 d 个特征
- 正则项 $\lambda\sum_{j=1}^{d}\beta_j^2$ 会压缩所有的系数趋向于0，但不会设定其中的任何一个等于0（除非$\lambda=\infty$）
- 从测试误差的角度来说无可厚非，但在 d 非常大时会给模型可解释性带来困难
- 例如，我们发现最重要的特征是Income，Limit，Rating，Student。因此我们希望得到一个只有这四个特征的模型。
- 岭回归显然做不到这一点



LASSO(Least Absolute Shrinkage and Selection Operator)

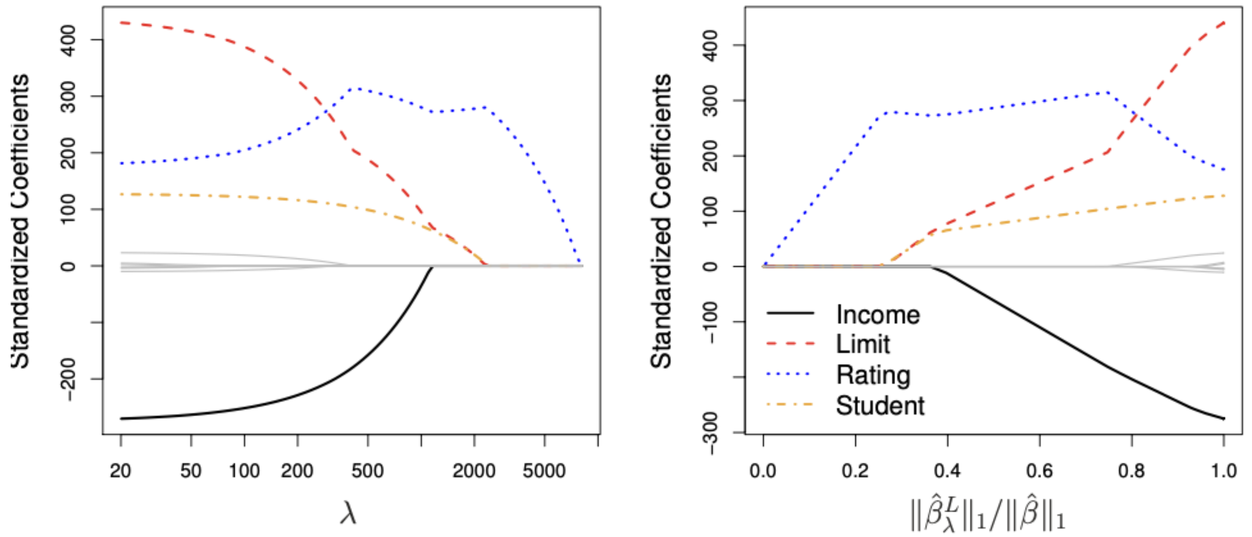
- Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1), 267-288.

$$L_{LASSO}(\boldsymbol{\beta}) = \frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} - \boldsymbol{\beta}^{T}\boldsymbol{x}^{(i)}\right)^2 + \lambda\sum_{j=1}^{d}|\beta_j|$$

- 又称L1正则化： $\sum_{j=1}^{d}|\beta_j| = \|\boldsymbol{\beta}\|_1$
- 与岭回归不同，LASSO使用了绝对值的一阶惩罚函数代替了平方和的二阶函数。虽然只是形式稍有不同，但是得到的结果却大不相同。
- L1正则化约束导致最终的解是$y^{(i)}$ 的非线性函数，因此一般情况下不存在显式解

例——Credit Card Balance

- 随着λ的增大，类似于岭回归，LASSO得到的参数整体上会趋向于0，不排除个别参数有局部增大的情况。
- LASSO与岭回归的**核心区别**：LASSO的很多参数在半路上就归零了，因此有着变量选择的效果，有更强的可解释性。



等价形式

- 岭回归的等价形式：

$$\underset{\boldsymbol{\beta}}{\text{minimize}} \ \frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} - \boldsymbol{\beta}^{T}\boldsymbol{x}^{(i)}\right)^2 \ \text{subject to} \ \sum_{j=1}^{d}\beta_j^2 \leq s$$

- LASSO的等价形式：

$$\underset{\boldsymbol{\beta}}{\text{minimize}} \ \frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} - \boldsymbol{\beta}^{T}\boldsymbol{x}^{(i)}\right)^2 \ \text{subject to} \ \sum_{j=1}^{d}|\beta_j| \leq s$$

- 通过拉格朗日乘子法，可以验证 λ 和 s 之间存在一 一对应关系。

几何解释

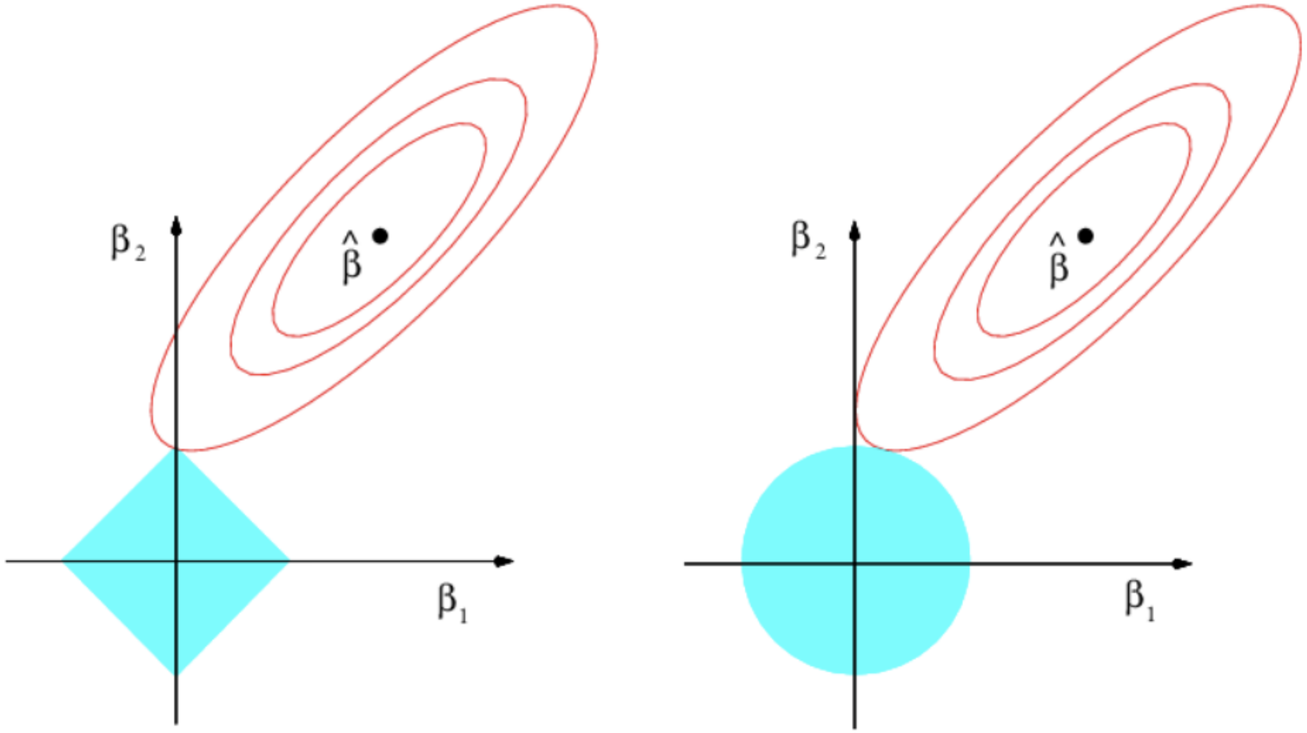
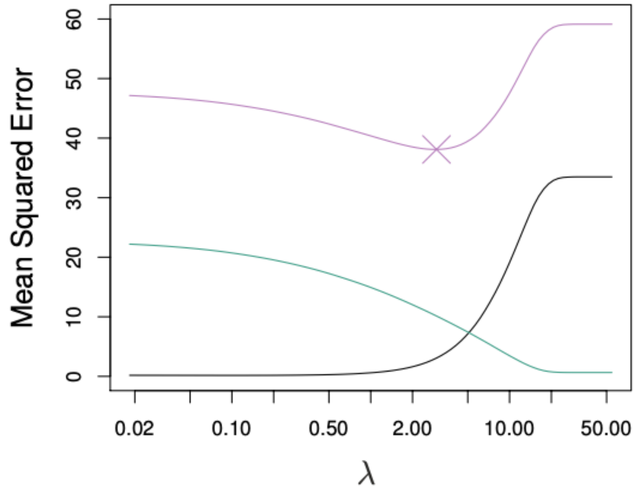
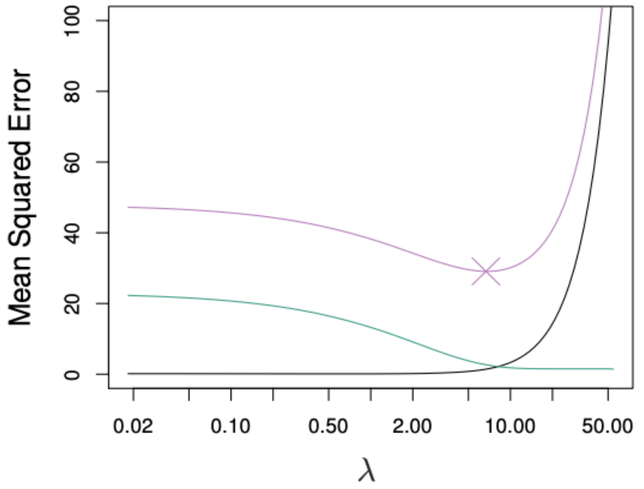


FIGURE 6.7. *Contours of the error and constraint functions for the lasso* (left) *and ridge regression* (right). *The solid blue areas are the constraint regions,* $|\beta_1| + |\beta_2| \leq s$ *and* $\beta_1^2 + \beta_2^2 \leq s$, *while the red ellipses are the contours of the RSS.*

人造例子：续

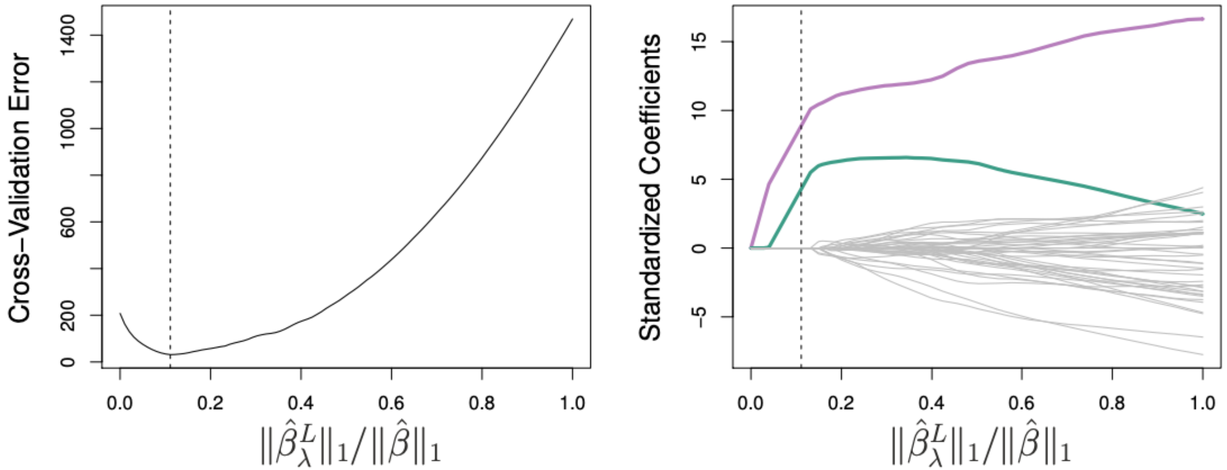
- 右图是一个人造的例子：$d = 45\ n = 50$
- 可以看到，LASSO和岭回归的测试MSE呈现出类似的趋势，但是其最小的测试MSE大于岭回归。
- 这是正常的，因为我们所有的45个特征都假设和响应变量有关。
- 我们换一个例子，45个特征中只有两个和响应变量有关。此时LASSO的最小测试MSE小于岭回归。
- 这两个例子说明了，岭回归和LASSO都无法完全支配对方，他们各有各的优势。
- 在实际中，我们永远无法知道和响应变量有关的特征数量是多少，只能通过测试MSE来判断。





如何选λ?

- 在实际中，应该很小心的选择λ，否则会起到反作用
- 交叉验证Cross-Validation
- 我们选取一些可能的λ，对每一个λ计算CV误差，并找到使得CV误差最小的λ
- 最后，我们用选出的λ对所有的观测重新拟合模型
- 右图展示了CV应用在LASSO和模拟的稀疏数据的情况



ELASTIC-NET

- Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, *67*(2), 301-320.
- 虽然LASSO会执行变量选择，但是对于两个高度共线性的特征来说，通常系数会一个变成0，一个仍保留在模型中。
- 岭回归会同时保留这两个特征，同时压缩两个特征的系数。
- 所以为了应对共线性问题，我们可以结合LASSO和岭回归
- $L_{\text{net}}(\boldsymbol{\beta}) = \frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} - \boldsymbol{\beta}^{\boldsymbol{T}}\boldsymbol{x}^{(i)}\right)^2 + \lambda\left[\alpha\sum_{j=1}^{d}|\beta_j| + (1-\alpha)\sum_{j=1}^{d}\beta_j^2\right]$

本章小节

- 欠拟合与过拟合
- 正则化
- 岭回归
- LASSO

扩展：数值优化方法

Outline

- 数值优化基础简介
- 知识回顾
- ADMM 算法

大 O 符号 I

- 对于两个有着相同定义域的函数 $f(x)$ 与 $g(x)$，我们记 $f(x) = O\big(g(x)\big)$，如果存在一个 M 以及 x_0 使得

$$|f(x)| \leq M|g(x)|, \quad \forall\, x \geq x_0$$

- 一些例子
  - $f(x) = O(1)$表示 $f(x)$ 是有界的.
  - $f(x) = g(x) + O(h(x))$ 表示 $f(x) - g(x) = O(h(x))$.
  - $O(f(x)) + O(f(x)) = O(f(x))$.
  - $O(f(x)) * O(g(x)) = O(f(x)g(x))$.
  - $|O(f(x))|^{\lambda} = O(|f(x)|^{\lambda}),\ \lambda > 0$.

- 一些简单的 0 关系：$x^2 = O(x^2)$, $e^x = 1 + x + O(x^2)$.

小 O 符号 I

- 对于两个有着相同定义域的函数$f(x)$ 与 $g(x)$, 记 $f(x) = o\big(g(x)\big)$, 如果对每一个 M 都能找到一个 x_0 使得

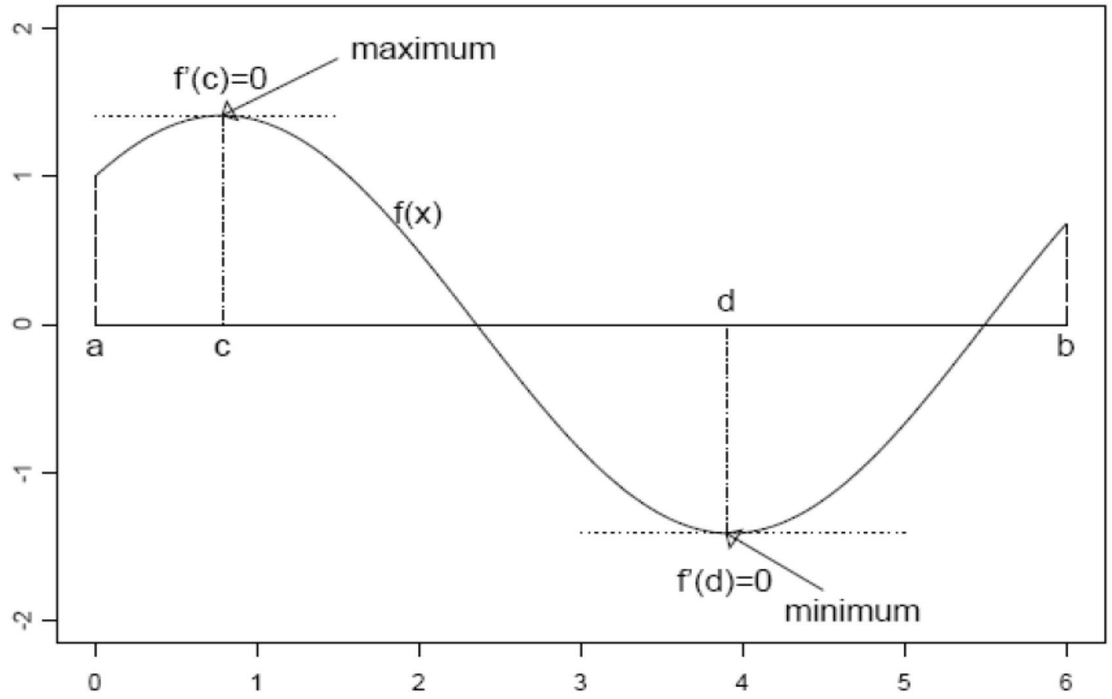
$$|f(x)| \leq M|g(x)|, \forall\, x \geq x_0$$

- 一些例子
  - $\lim\limits_{x\to 0} \dfrac{o\big(g(x)\big)}{g(x)} = 0$
  - 如果 $limx \to \infty f(x) = 0$，那么$f(x) = o(1)$
  - $o(f(x)) + o(f(x)) = o(f(x))$
  - $O(f(x)) * o(g(x)) = o(f(x)g(x))$
  - $|o(f(x))|^{\lambda} = o(|f(x)|^{\lambda}),\ \lambda > 0$

- 一些简单的 0 关系: $x^2 = o(x^3),\ x^2 = o(x!),\ \ln(x) = o(x)$

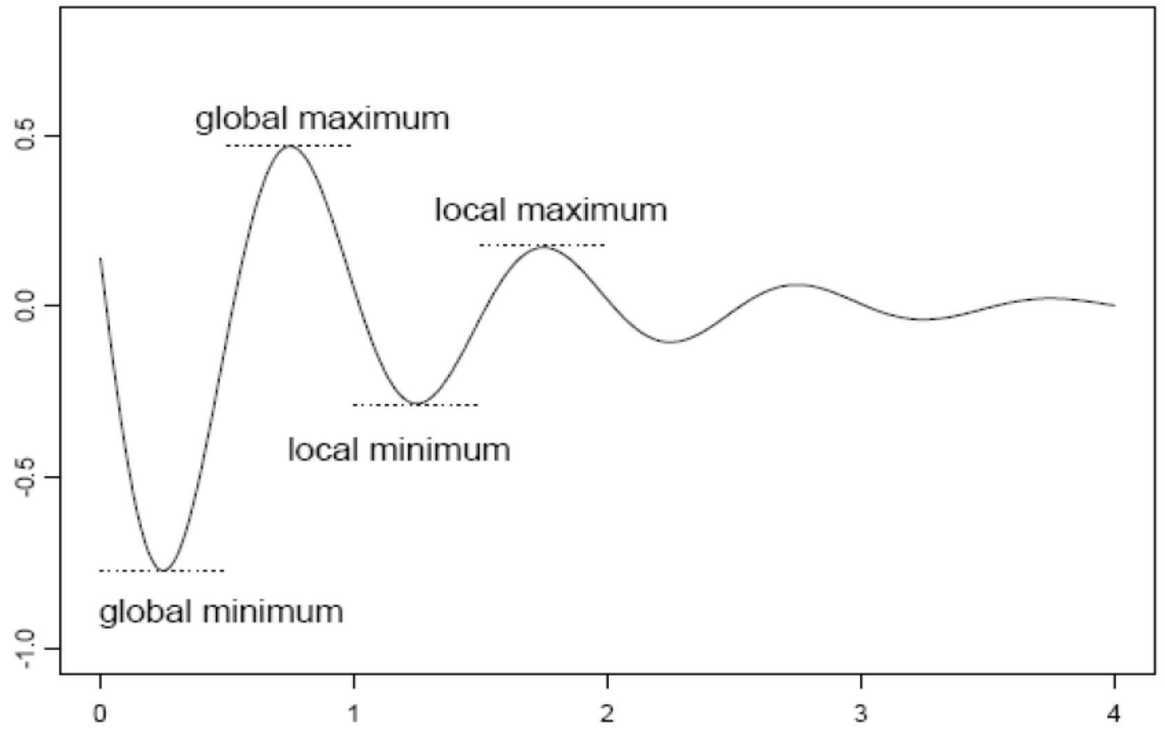
WEIERSTRASS 定理 I

- Weierstrass 定理：一个取值于闭区间 $[a,b]$ 上的连续函数 $f(x)$,其最大最小值存在并且在区间中取得。

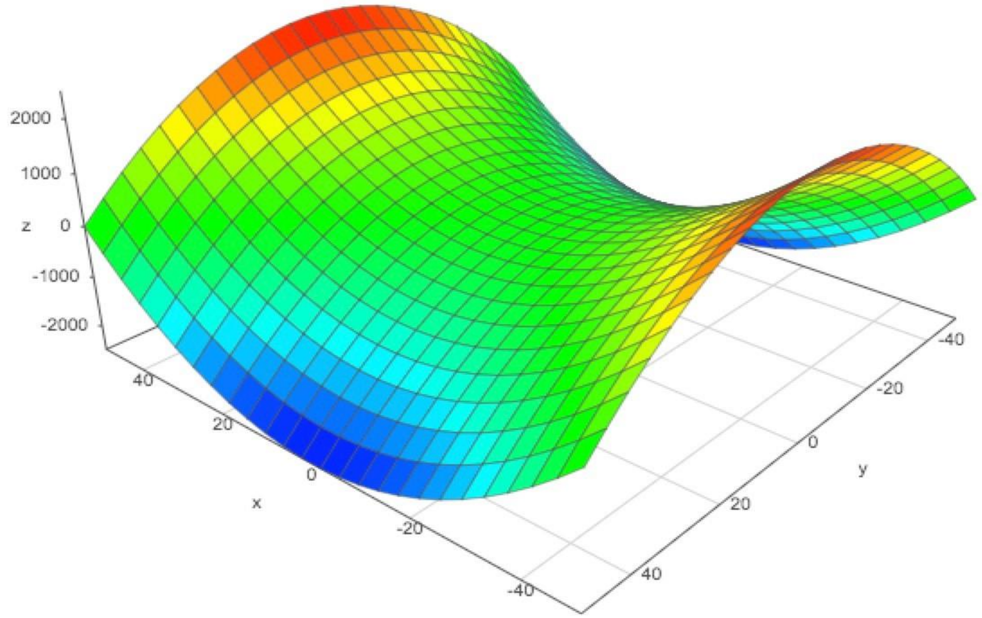


WEIERSTRASS 定理 II

- 驻点 (Stationary point)：上图中，使得 $f'(c) = 0$ 的 c 我们称为驻点（平稳点）。但这里需要注意一点，$f'(c) = 0$ 并不能保证 c 即为我们所要求的全局最优点，它可能是局部最优点(local max or min) 或者为鞍点(Saddlepoint)。



WEIERSTRASS 定理 III



知识回顾 I

- 凸函数 (Convex function): 假设$\mathcal{X}$为一个实数向量空间中的凸集, $f:\mathcal{X}\to\mathcal{R}$表示某一函数
  - 如果 $\forall t\in[0,1],\ x_1,x_2\in\mathcal{X}$, 我们有

$$f(t_1x_1+(1-t)x_2)\leq tf(x_1)+(1-t)f(x_2),\forall x$$

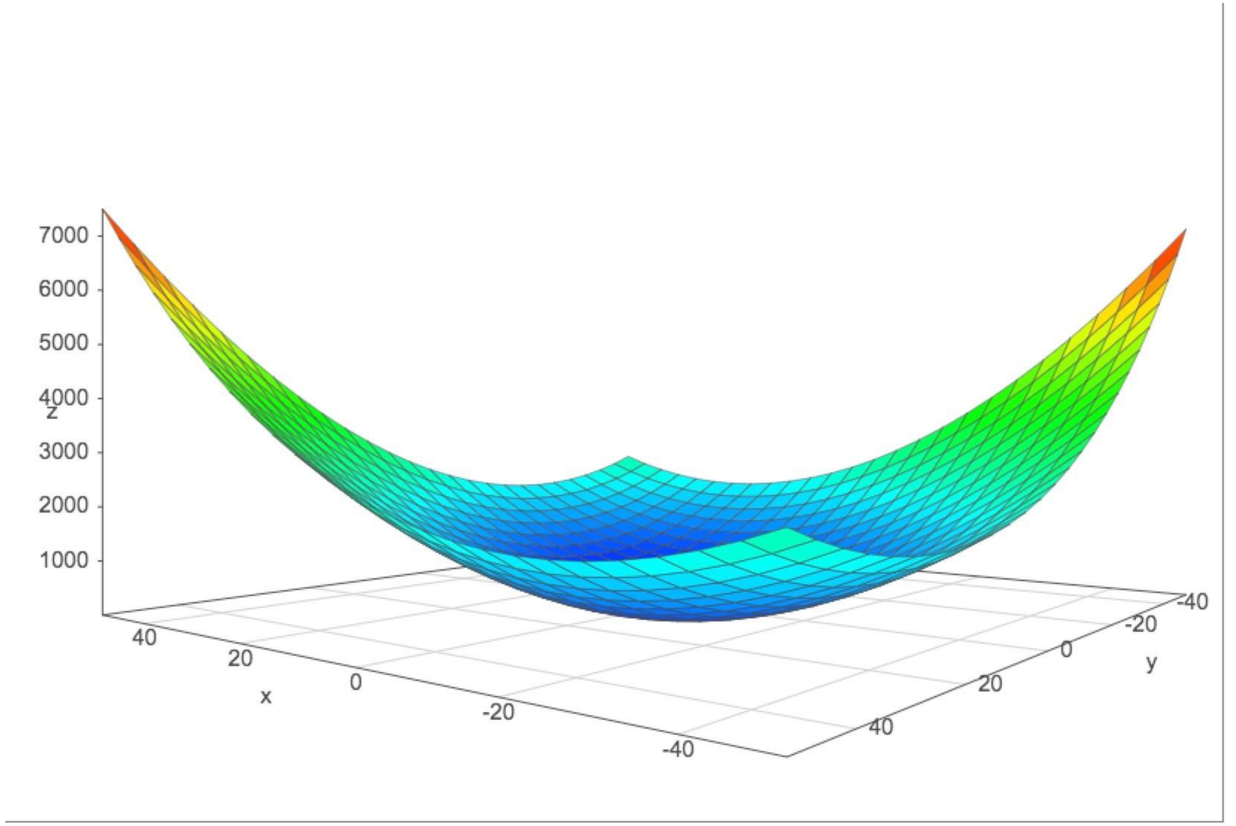
    那么$f(x)$ 被称为凸函数 (Convex)。

  - 如果$\forall t\in[0,1],x_1\neq x_2\in\mathcal{X}$, 我们有

$$f(t_1x_1+(1-t)x_2)< tf(x_1)+(1-t)f(x_2),\forall x$$

    那么$f(x)$ 被称为严格凸函数 (Strictly convex)。

知识回顾 II



知识回顾 III

- 一个凸函数的例子：有关范数的定义为 $||\cdot||: \mathcal{X} \rightarrow \mathcal{R}^{+}$对所有的 $x, u \in \mathcal{X}$ 有
  - $||ax|| = |a|\,||x||, \ \forall a \in \mathcal{R}$
  - $||x + u|| \leq ||x|| + ||u||$
  - $||x|| \geq 0, \ \forall x \in \mathcal{X}, \ ||x|| = 0$当且仅当 $x = 0$

中值定理 I

假设连续函数 $f(x)$ 取值在闭区间 $[a,b]$ 上，并且在 (a,b) 中可导。那么一定可以找到一点 $c \in (a,b)$, 使得

$$f(b) - f(a) = f'(c)(b-a).$$

- 证明思路：构造函数

$$g(x) = f(b) - f(x) + \frac{f(b)-f(a)}{b-a}(x-b)$$

使得其在 $[a,b]$ 上连续，(a,b) 上可导，继而我们有

$$g'(x) = -f'(x) + \frac{f(b)-f(a)}{b-a},\ g(a) = g(b) = 0$$

考虑平稳点 $g'(c) = 0$，即得所求。

泰勒展开 I

- 泰勒展开

假设 $f(x)$ 在点 x_0 附近的区间 (a,b) 中有 $n+1$ 阶可导，那么对于任意 $x \in (a,b)$，我们都 可以将 $f(x)$ 在 x_0 点处泰勒展开

$$f(x) = f(x_0) + \sum_{i=1}^{n} \frac{f^{(i)}(x_0)}{i!}(x - x_0)^i + \frac{(x - x_0)^{n+1}}{(n+1)!} f^{(n+1)}(\xi),$$

对某些 $\xi \in [x_0, x]$。此外，如果 $f^{(n+1)}(\xi)$有界，那么余项可写为

$$R_{n+1}(x) = \frac{(x - x_0)^{n+1}}{(n+1)!} f^{(n+1)}(\xi) = O(|x - x_0|^{n+1}).$$

算法的收敛率 I

- 算法的收敛率（Convergence rate）

对优化算法来讲，我们实际上是想要找到一个序列 $\boldsymbol{\beta}^0, \boldsymbol{\beta}^1, \ldots, \boldsymbol{\beta}^t, \ldots$，使得其能够最终收敛到我们想要得到的参数$\boldsymbol{\beta}^*$上。基于此目的，我们可以定义算法的收敛速率

  - 如果存在一个常数 $c \in [0,1)$ 使得

$$\lim_{\mathrm{t}\to\infty} \frac{\| \boldsymbol{\beta}^{\mathrm{t}+1} - \boldsymbol{\beta}^* \|}{\| \boldsymbol{\beta}^{\mathrm{t}} - \boldsymbol{\beta}^* \|^q} = \mathrm{c},$$

我们就说 $\{\boldsymbol{\beta}^{\mathrm{t}}\}_{\mathrm{t}=1}^{\infty}$收敛到 $\boldsymbol{\beta}^*$的速率为 q。

  - 当 $q = 1,2$ 时， 我们分别称其收敛速率为线性（linearly），二次收敛（quadratically）速率。

收敛标准 I

- 绝对收敛标准： $\| \boldsymbol{\beta}^{(t+1)} - \boldsymbol{\beta}^{(t)} \| < \epsilon$
- 相对收敛标准： $\dfrac{\| \boldsymbol{\beta}^{(t+1)} - x^{(t)} \|}{\| \boldsymbol{\beta}^{(t)} \|} < \epsilon$
- 调整的相对收敛标准： $\dfrac{\| \boldsymbol{\beta}^{(t+1)} - \boldsymbol{\beta}^{(t)} \|}{\| \boldsymbol{\beta}^{(t)} \| + \epsilon} < \epsilon.$

优化问题 I

- ADMM 算法用来解决以下优化问题：

$$\min\ f(x) + g(z)$$
$$s.t.\ \ Ax + Bz = c,$$

- 其中 $x \in \mathcal{R}^p$，$z \in \mathcal{R}^q$，A 是一个 $r \times p$ 实矩阵，B 是一个 $r \times q$实矩阵，c 是一个 r 维向量，且 f 和 g 这里假定为凸函数 (Convex function)。

标准 ADMM 算法 I

- 标准 ADMM 优化问题对应的增广拉格朗日方程为：

$$\mathcal{L}_\rho(x,z,u) = f(x) + g(z) + u^T(Ax + Bz - c) + \frac{\rho}{2} \| Ax + Bz - c \|_2^2.$$

- 具体的优化迭代步骤如下：

$$x^{k+1} := \arg\min_x \mathcal{L}_\rho(x, z^k, u^k) \quad (1)$$

$$z^{k+1} := \arg\min_z \mathcal{L}_\rho(x^{k+1}, z, u^k) \quad (2)$$

$$u^{k+1} := u^k + \rho(Ax^{k+1} + Bz^{k+1} - c) \quad (3)$$

- 理论上，ADMM 算法会迭代至目标函数一阶导数等于零的点。
- 当然这些点未必就一定是取得最小值的地方，然而这是目前基于梯度计算的优化方法的通病（想一想为什么？）。

ADMM 算法的一个变形 I

- 考虑如下变形：

$$\mathcal{L}_\rho(x,z,u) = f(x) + g(z) + \frac{\rho}{2} \| Ax + Bz - c + v \|_2^2 \quad + \quad 常数$$

这里 $v = \frac{1}{\rho} u.$

- 优化迭代步骤更改如下：

$$x^{k+1} := \arg\min_x \mathcal{L}_\rho(x, z^k, v^k) \tag{4}$$

$$z^{k+1} := \arg\min_z \mathcal{L}_\rho(x^{k+1}, z, v^k) \tag{5}$$

$$v^{k+1} := v^k + Ax^{k+1} + Bz^{k+1} - c \tag{6}$$

以 LASSO 为例 I

- 假设收集到样本 $\mathcal{Z}^{\mathrm{n}} = \{(y_i, x_i)\}_{i=1}^{n}$， 其中 y 表示因变量，$x \in \mathcal{R}^p$ 为自变量。

- 考虑如下优化问题

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - x_i^T\boldsymbol{\beta})^2 + \lambda\sum_{j=1}^{p}|\beta_j|. \tag{7}$$

- 将上式写为向量化的形式：

$$\min_{x} \| y - X\boldsymbol{\beta} \|_2^2 + \lambda_n \| \boldsymbol{\beta} \|_1,$$

这里 $X = (x_1, \ldots, x_n)^T \in \mathcal{R}^{n\times p}$ 以及 $\lambda_n = n\lambda$.

问题解析 I

- 针对上面的问题，取

$$f(\boldsymbol{\beta}) = \| y - X\boldsymbol{\beta} \|_2^2, \qquad g(z) = \lambda_n \| z \|_1,$$

以及

$$s.t. \quad \boldsymbol{\beta} - z = 0.$$

- 写出上面问题的增广拉格朗日函数为

$$\mathcal{L}_\rho(\boldsymbol{\beta}, z, v) = \| y - X\boldsymbol{\beta} \|_2^2 + \lambda_n \| z \|_1 + \frac{\rho}{2} \| \boldsymbol{\beta} - z + v \|_2^2.$$

问题解析 I

- 定初值 $\boldsymbol{\beta}^0, z^0, v^0$; 假设当前迭代为第 t 步, $t \geq 0$
- 求解第 $t+1$ 步各估计值的迭代步骤如下：

$$\boldsymbol{\beta}^{t+1} := \arg\min_{\boldsymbol{\beta}} \| y - X\boldsymbol{\beta} \|_2^2 + \frac{\rho}{2} \| \boldsymbol{\beta} - z^t + v^t \|_2^2, \tag{8}$$

$$z^{t+1} := \arg\min_{z} \lambda_n \| z \|_1 + \frac{\rho}{2} \| \boldsymbol{\beta}^{t+1} - z + v^t \|_2^2,$$

$$v^{t+1} := v^t + (\boldsymbol{\beta}^t + 1 - z^{t+1}) \tag{9}$$

- 具体每步的优化如下：

问题解析 I

- 针对(8)，有

$$\boldsymbol{\beta}^{t+1} := \arg\min_{\boldsymbol{\beta}} \| y - X\boldsymbol{\beta} \|_2^2 + \frac{\rho}{2} \| \boldsymbol{\beta} - z^t + v^t \|_2^2$$

- 对上式求导并令为 0，得到

$$\boldsymbol{\beta}^{t+1} = \left(2X^T X + \rho \mathrm{I_p}\right)^{-1} (2Xy + \rho(z^t - v^t))$$

问题解析 I

- 针对(9)，有

$$z^{t+1} = \mathrm{Prox}_{\frac{\lambda_n}{\rho}}(\boldsymbol{\beta}^{t+1} + v^t),$$

- 上式可具体写为

$$(z^{t+1})_{\mathrm{j}} = \mathrm{Prox}_{\frac{\lambda_n}{\rho}}(u_j) = \mathrm{sign}(u_j)(|u_j| - \frac{\lambda_n}{\rho})_+$$

$$= \begin{cases} u_j - \frac{\lambda_n}{\rho}, & if\, u_j > \frac{\lambda_n}{\rho}, \\ 0, & if\, |u_j| \le \frac{\lambda_n}{\rho}, \\ u_j + \frac{\lambda_n}{\rho}, & if\, u_j < -\frac{\lambda_n}{\rho} \end{cases}$$

其中 $u_j = (\boldsymbol{\beta}^{t+1} + v^t)_j$. 上述即为软阀值算子。

- 进而可以得到 $v^{t+1} = v^t + \boldsymbol{\beta}^{t+1} - z^{t+1}$
- 重复上述步骤，直到收敛到指定误差为止。

常用算法 I

- NR 算法
- （随机）梯度下降法
- ADMM 算法
- Proximal 算法
- **坐标下降法**