分类与预测

2021/10/18

目录

分类与预测——回归分析

- 回归分析是通过建立模型来研究变量之间相互关系的密切程度、结构状态及进行模型预测的一种有效工具，在工商管理、经济、社会、医学和生物学等领域应用十分广泛。

- 从19世纪初高斯提出最小二乘估计算起，回归分析的历史已有200多年。

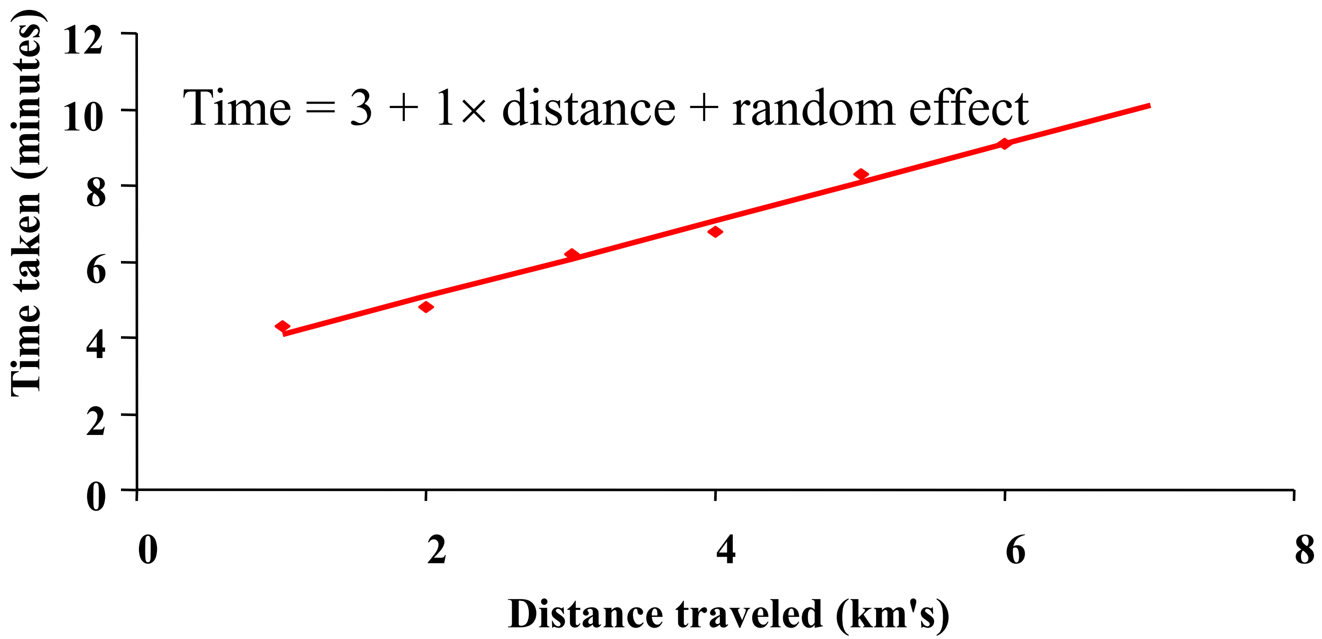
分类与预测——回归分析

- 常用的回归模型如下：

回归模型	适用条件	算法描述
线性回归	因变量与自变量是线性关系	对一个或多个自变量和因变量之间的线性关系进行建模，可用最小二乘法求解模型系数。
非线性回归	因变量与自变量之间不都是线性关系	对一个或多个自变量和因变量之间的非线性关系进行建模。如果非线性关系可以通过简单的函数变换转化成线性关系，用线性回归的思想求解；如果不能转化，用非线性最小二乘方法求解。
Logistic回归	因变量的一般有1-0（是否）两种取值	是广义线性回归模型的特例，利用Logistic函数将因变量的取值范围控制在0和1之间，表示取值为1的概率。
岭回归	参与建模的自变量之间具有多重共线性	是一种改进最小二乘估计的方法。
主成分回归	参与建模的自变量之间具有多重共线性	主成分回归是根据主成分分析的思想提出来的，是对最小二乘法的一种改进，它是参数估计的一种有偏估计。可以消除自变量之间的多重共线性。

回归——Linear Regression线性回归分析

$$Y = \theta_0 + \theta_1 X_1 + \theta_2 X_2 + \cdots + \theta_n X_n + \varepsilon$$



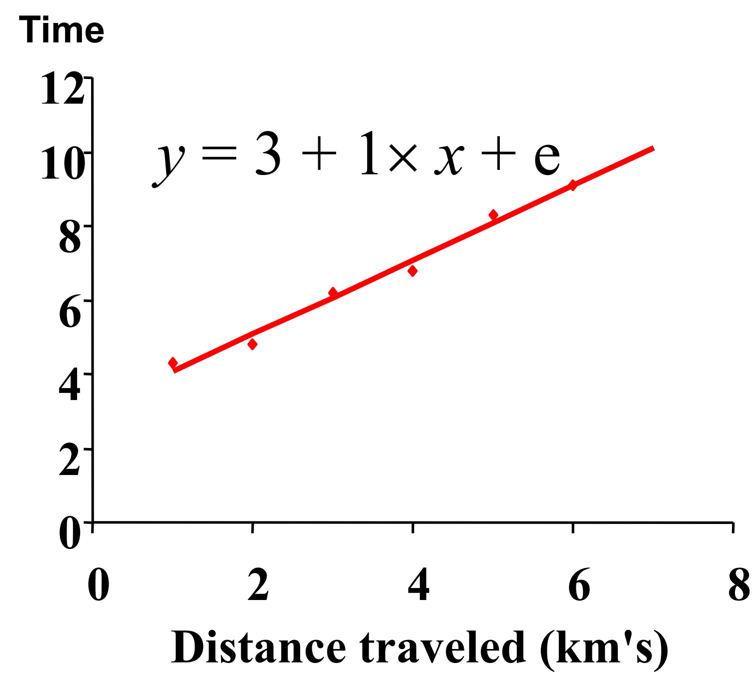
基于旅行距离来预测旅行时间

回归——二元线性回归及最小二乘法

- 基本形式

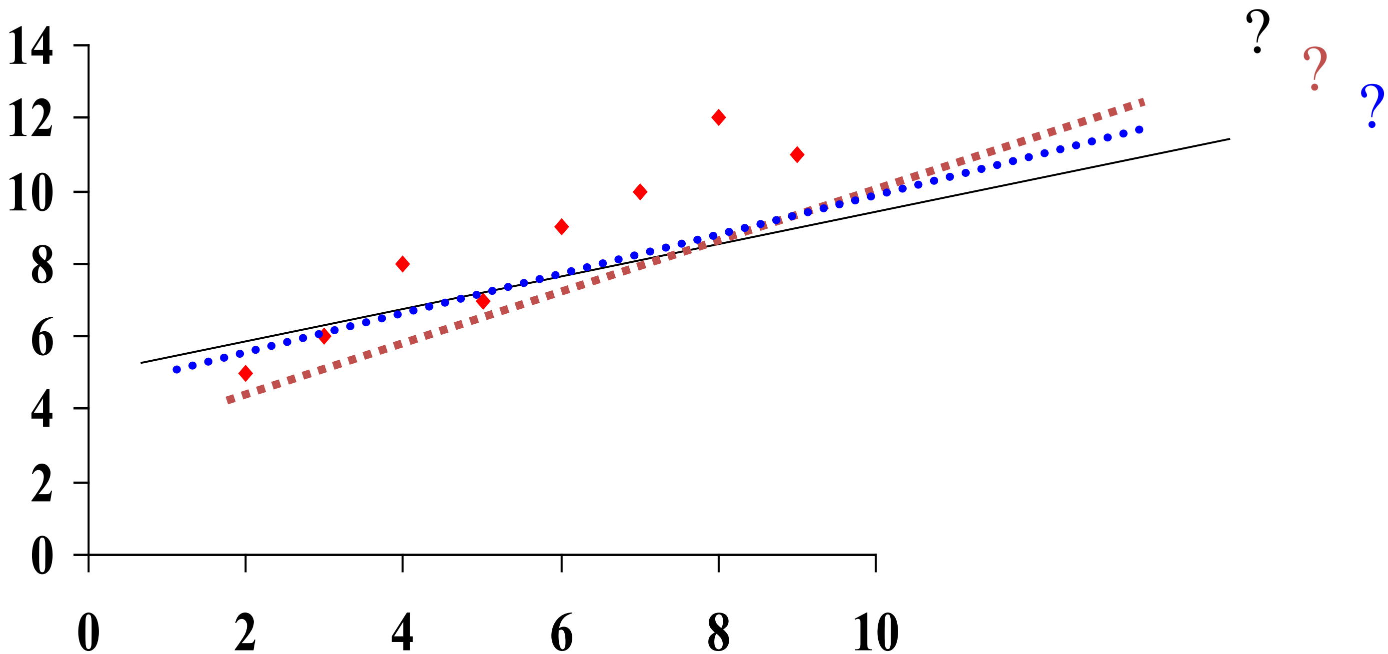
$$y = \theta_0 + \theta_1 x + e$$

- *where*
  - y = the dependent/target variable
  - x = the independent variable
  - θ_o = The y-intercept
  - θ_1 = The slope of the line
  - e = random error term



回归——二元线性回归及最小二乘法

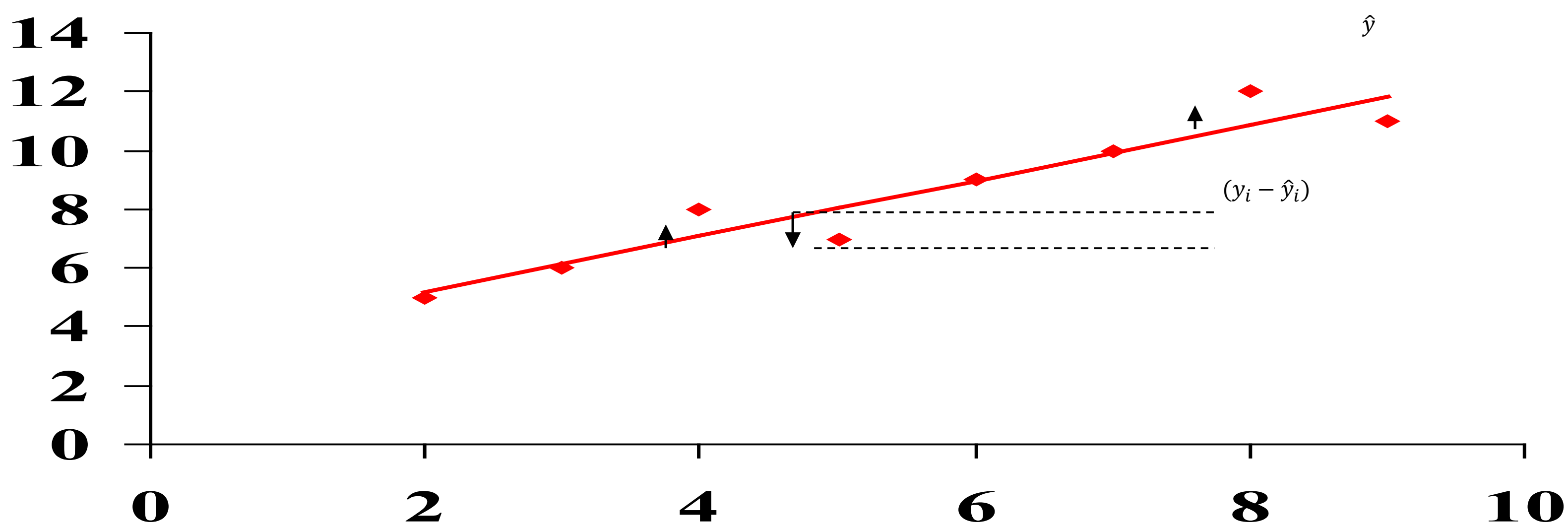
- 估计参数 $$y = \theta_0 + \theta_1 x + e$$



- 估计参数的目的是找到最佳拟合的直线 (the *line of best fit*)

回归——二元线性回归及最小二乘法

- 由于实际数据不一定符合线性函数，找到最佳拟合的直线可以通过最小化误差来实现



回归——二元线性回归及最小二乘法

- 误差（ error ）或残差 （ *residual* ）

$$e_i = (y_i - \hat{y})$$

y_i 为实际值，$\hat{y}$为预测值

- 最优模型通过最小化如下的SSE误差

$$SSE = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - \hat{y})^2$$

- 最优解

$$\hat{\theta}_1 = \frac{SS_{xy}}{SS_x} \qquad SS_{xy} = \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) \qquad SS_x = \sum_{i=1}^{n} (x_i - \bar{x})^2$$

$$\hat{\theta}_0 = \bar{y} - \hat{\theta}_1 \bar{x} \qquad \bar{y} = \frac{\sum_{i=1}^{n} y_i}{n} \qquad \bar{x} = \frac{\sum_{i=1}^{n} x_i}{n}$$

回归——Linear Regression线性回归分析

- Y为预测值，连续变量 $X=(x_1, x_2, ..., x_k)$，之间的关系可以建模为

$$y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + ... + \theta_k x_k + \varepsilon$$

- 给定数据 X及 y, 我们可以估计 $\theta_0, \theta_1, \theta_2, ..., \theta_k$

$$y = X\theta \qquad y = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ . \\ y_n \end{bmatrix} \qquad X = \begin{bmatrix} 1 & x_{1,1} & x_{1,2} & . & . & . & x_{1,k} \\ 1 & x_{2,1} & x_{2,2} & . & . & . & x_{2,k} \\ . & . & . & & . & & . \\ . & . & . & & . & & . \\ . & . & . & & . & & . \\ 1 & x_{i,1} & x_{i,2} & . & . & . & x_{i,k} \\ . & . & . & & . & & . \\ . & . & . & & . & & . \\ . & . & . & & . & & . \\ 1 & x_{n,1} & x_{n,2} & . & . & . & x_{n,k} \end{bmatrix} \qquad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ . \\ . \\ . \\ \theta_k \end{bmatrix}$$

- 损失函数—衡量估计的误差

$$J(\theta) = \frac{1}{2n} \sum_{i=1}^{n} (x_i\theta - y_i)^2 = \frac{1}{2n} \|X\theta - y\|_F^2 = \frac{1}{2n} (X\theta - y)^T (X\theta - y)$$

回归——Linear Regression线性回归分析

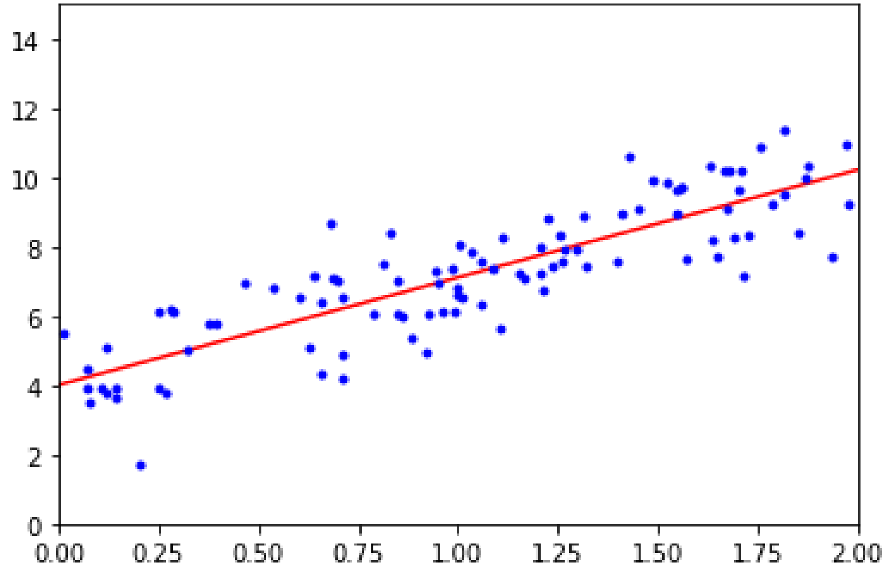
解析解

1. Since $y = X\hat{\theta}$ $X^TX\hat{\theta} = X^Ty$ $\hat{\theta} = (X^TX)^{-1}X^Ty$
2. 由于J(θ)的二阶导X^TX为半正定矩阵，所以Hessian矩阵是半正定的

   对J(θ)求偏导并令其为0， $X^TX\hat{\theta} - X^Ty = 0$，得到的解就是使损失函数最小的解，我们得到

$$\hat{\theta} = (X^TX)^{-1}X^Ty$$

```
import numpy as np
import matplotlib.pyplot as plt
X = 2 * np.random.rand(100, 1)
y = 4 + 3 * X + np.random.randn(100, 1)
#生成模拟数据
X_b = np.c_[np.ones((100, 1)), X]
beta_best =np.linalg.inv(X_b.T.dot(X_b)).dot(X_b.T).dot(y)
print(beta_best)
X_new = np.array([[0], [2]])
X_new_b = np.c_[(np.ones((2, 1))), X_new]
print(X_new_b)
y_predict = X_new_b.dot(beta_best)
print(y_predict)
# 绘制函数图像和测试集点分布
plt.plot(X_new, y_predict, 'r-')
plt.plot(X, y, 'b.')
plt.axis([0, 2, 0, 15])
plt.show()
```



不足：

- X^TX是dxd维的，求逆矩阵的计算复杂度是d的三次方
- 如果X^TX是奇异阵，则无法得到解

回归——Linear Regression线性回归分析

梯度下降（GD）

```
import numpy as np
import matplotlib.pyplot as plt
X = 2 * np.random.rand(100, 1)
y = 4 + 3 * X + np.random.randn(100, 1)
X_b = np.c_[np.ones((100, 1)), X]
n_epochs = 500
t0, t1 = 5, 50
m = 100

def learning_schedule(t):
    return t0 / (t + t1)

beta = np.random.randn(2, 1)
for epoch in range(n_epochs):
    for i in range(m):
        random_index = np.random.randint(m)
        xi = X_b[random_index:random_index+1]
        yi = y[random_index:random_index+1]
        gradients = 2*xi.T.dot(xi.dot(beta)-yi)
        learning_rate = learning_schedule(epoch*m + i)
        beta = beta - learning_rate * gradients

X_new = np.array([[0], [2]])
X_new_b = np.c_[(np.ones((2, 1))), X_new]
print(X_new_b)
y_predict = X_new_b.dot(beta)
print(y_predict)
# 绘制函数图像和测试集点分布
plt.plot(X_new, y_predict, 'r-')
plt.plot(X, y, 'b.')
plt.axis([0, 2, 0, 15])
plt.show()
```

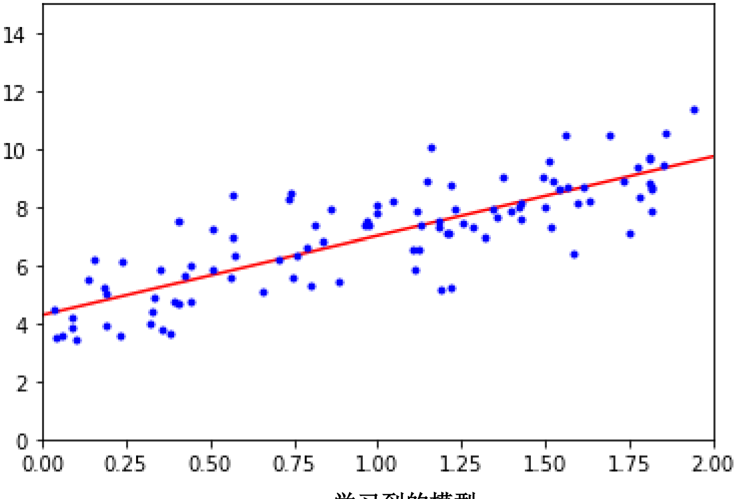
当距离最优解很远时，我们希望学习率大些，加快学习速度，当距离最优解较近是我们希望减小学习率，避免来回震荡

$$\theta = \theta - \alpha \frac{\partial J(\theta)}{\partial \theta}$$

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{\partial}{\partial \theta_j} \frac{1}{2n} \sum_{i=1}^{n} (x_i\theta - y_i)^2 = \frac{1}{n} \sum_{i=1}^{n} (x_i\theta - y_i) x_i^j$$

$$\theta_j = \theta_j - \alpha \frac{\partial J(\theta)}{\partial \theta_j} = \theta_j - \alpha \frac{1}{n} \sum_{i=1}^{n} (x_i\theta - y_i) x_i^j$$

学习率



学习到的模型

回归——Logistic回归

- 线性回归模型是相对简单的回归模型，但是通常因变量和自变量之间呈现某种曲线关系，就要建立非线性回归模型。
- Logistic回归属于概率型非线性回归，分为二分类和多分类的回归模型。对于二分类的Logistic回归，因变量 y 只有“是、否”两个取值，记为1和0。假设在自变量作用下， y 取“是”的概率是*p*，则取“否”的概率是*1-p*，研究的是当y 取“是”发生的概率 *p* 与自变量 的关系。

示例

邮件：垃圾邮件Spam / 非垃圾邮件Not Spam?

在线交易：欺诈 (Yes / No)?

癌症Tumor: 恶性Malignant / 良性Benign ?

0: 负类“Negative Class”

1: 正类“Positive Class”

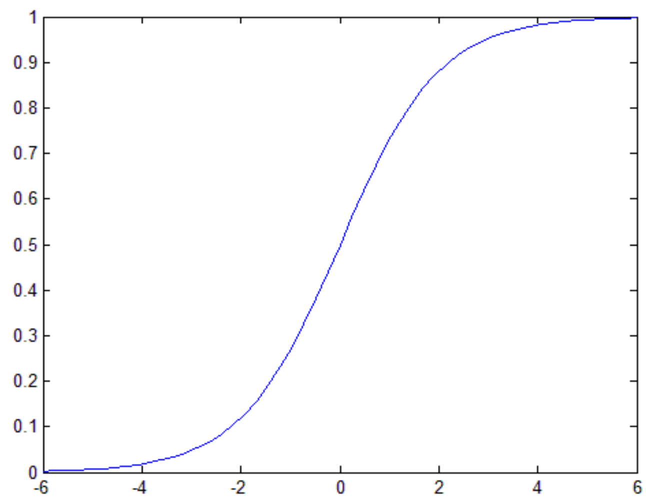
回归——Logistic回归分析及分类

- Logistic函数。

二分类的Logistic回归模型中的因变量的只有1-0（如是和否、发生和不发生）两种取值。假设在 p个独立自变量 $x_1, x_2, \dots, x_p$ 作用下，记y 取1的概率是 $p = P(y = 1|X)$，取0概率是1-p，取 1 和取 0 的概率之比为 $\frac{p}{1-p}$，称为事件的优势比（odds），对odds取自然对数即得Logistic变换：$Logit(p) = Ln\left(\frac{p}{1-p}\right)$

当 p 在（0, 1）之间变化时，odds的取值范围是$(0, +\infty)$，$Logit(p)$ 的取值范围是（$-\infty, +\infty$）

令 $Logit(p) = Ln\left(\frac{p}{1-p}\right) = z$，则 $p = \frac{1}{1 + e^{-z}}$，即为Logistic函数，如下图：



回归——Logistic回归分析

- Logistic回归模型

Logistic回归模型是建立 $Ln\left(\frac{p}{1-p}\right)$ 与自变量的线性回归模型。

Logistic回归模型为：

$$Ln\left(\frac{p}{1-p}\right) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_k x_k + \varepsilon$$

因为 $Ln\left(\frac{p}{1-p}\right)$ 的取值范围是 $(-\infty, +\infty)$，这样，自变量 $\theta_1, \theta_2, \ldots, \theta_k$ 可在任意范围内取值。

记 $h_\theta(x) = \frac{1}{1+e^{\theta_0+\theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_k x_k + \varepsilon}}$ ，得到：

$$p = P(y=1|X) = h_\theta(x) \qquad 1-p = P(y=0|X) = 1-h_\theta(x)$$

Cost function

$$J(\theta) = \frac{1}{\mathrm{n}} \sum_{i=1}^{n} Cost(h_\theta(x_i), y_i)$$
$$= -\frac{1}{\mathrm{n}} \sum_{i=1}^{n} (y_i \log h_\theta(x_i) + (1-y_i)\log(1-h_\theta(x_i)))$$

用类似线性回归的梯度下降方法进行优化

为什么逻辑回归损失函数不用MSE
https://www.jianshu.com/p/af1e5cff21b9

回归——偏差（Bias）和方差（Variance）

在监督学习中，已知样本(x_1, y_1)，…，(x_n, y_n)，要求拟合出一个模型（函数$\hat{f}$），其预测值$\hat{f}(x)$与样本实际值 y 的误差最小。

考虑到样本数据其实是采样，y并不是真实值本身，假设真实模型（函数）是f，则采样值y=$f(x)+\varepsilon$，其中ε代表噪音，其均值为0，方差为 σ^2 。

拟合函数$\hat{f}$ 的主要目的是希望它能对新的样本进行预测，所以，拟合出函数 $\hat{f}$ 后，需要在测试集（训练时未见过的数据）上检测其预测值与实际值y之间的误差。可以采用平方误差函数（mean squared error）来度量其拟合的好坏程度，即 $(y-\hat{f}(x))^2$

$$E\left[(y-\hat{f}(x))^2\right] = \sigma^2 + \mathrm{Var}[\hat{f}(x))] + \left(Bias\left[\hat{f}(x)\right]\right)^2$$

误差的期望值 = 噪音的方差 + 模型预测值的方差 + 预测值相对真实值的偏差的平方

噪声方差：标记值与真实值差平方的期望

方差：使用样本数相同的不同训练集产生的方差

偏差：期望输出与真实标记的差别

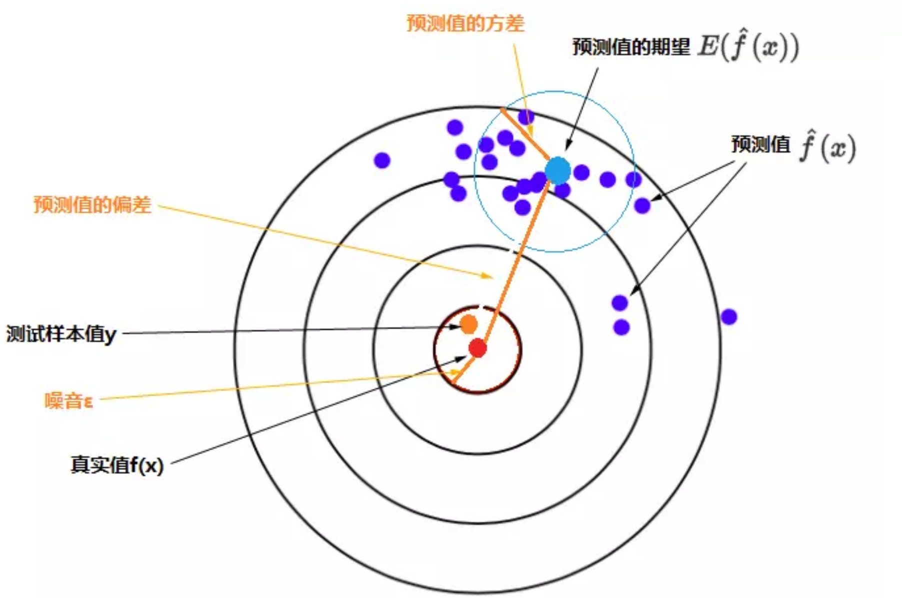
$$Bias\left[\hat{f}(x)\right] = f(x) - E(\hat{f}(x))$$

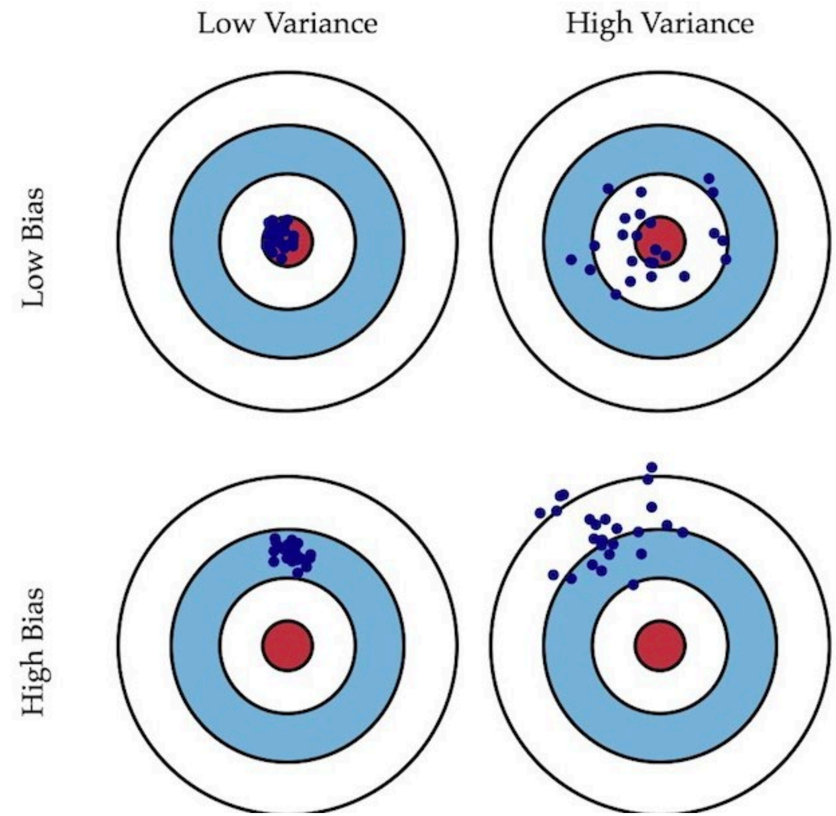
$$\begin{aligned}
E[(y-\hat{f})^2] &= E[y^2 + \hat{f}^2 - 2y\hat{f}] \\
&= E[y^2] + E[\hat{f}^2] - E[2y\hat{f}] \\
&= \left(Var[y] + (E[y])^2\right) + \left(Var[\hat{f}] + (E[\hat{f}])^2\right) - E[2(f+\varepsilon)\hat{f}] \\
&= Var[y] + Var[\hat{f}] + (E[y])^2 + (E[\hat{f}])^2 - E[2f\hat{f} + 2\varepsilon\hat{f}] \\
&= Var[y] + Var[\hat{f}] + f^2 + (E[\hat{f}])^2 - E[2f\hat{f}] - E[2\varepsilon\hat{f}] \\
&= Var[y] + Var[\hat{f}] + f^2 + (E[\hat{f}])^2 - 2fE[\hat{f}] - 2E[\varepsilon]E[\hat{f}] \\
&= Var[y] + Var[\hat{f}] + \left(f^2 + (E[\hat{f}])^2 - 2fE[\hat{f}]\right) \\
&= Var[y] + Var[\hat{f}] + (f - E[\hat{f}])^2 \\
&= \sigma^2 + Var[\hat{f}] + (Bias[\hat{f}])^2
\end{aligned}$$

- 噪声则表示任何学习算法在泛化能力的下界，描述了学习问题本身的难度；
- 方差度量了模型预测值的变化，描述了数据扰动造成的影响；
- 偏差度量了学习算法的期望预测与真实结果的偏离程度，刻画描述了算法本身对数据的拟合能力，也就是训练数据的样本与训练出来的模型的匹配程度。

https://www.jianshu.com/p/8c7f033be58a

回归——偏差（Bias）和方差（Variance）



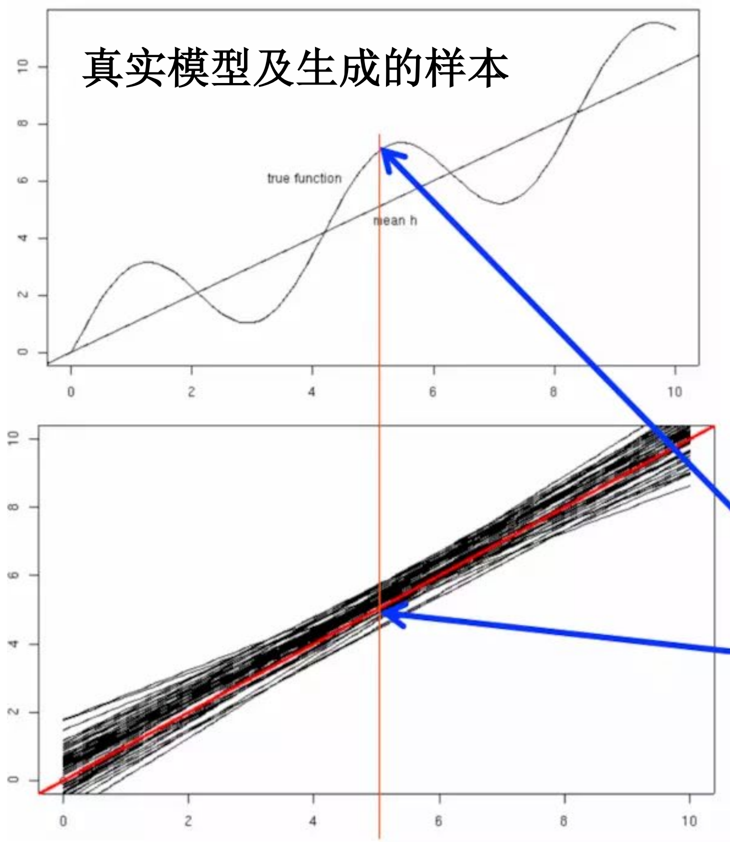


$$
\begin{aligned}
&E\left[(y-\hat{f}(x))^{2}\right] \\
&=\sigma^{2} \\
&\quad+\operatorname{Var}[\hat{f}(x))\,] \\
&\quad+\left(Bias\left[\hat{f}(x)\right]\right)^{2}
\end{aligned}
$$

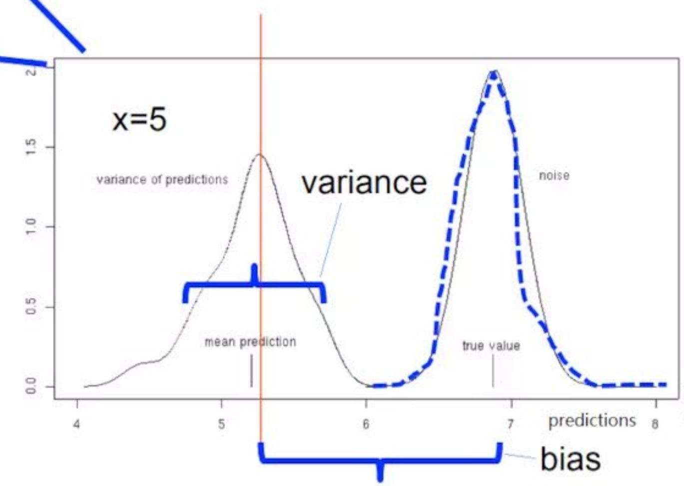
选择相对较好的模型的顺序：方差小，偏差小 ＞ 方差小，偏差大 ＞ 方差大，偏差小 ＞ 方差大，偏差大。

回归——偏差和方差的计算

$$y = x + 2\sin(1.5x) + N(0,0.2)$$



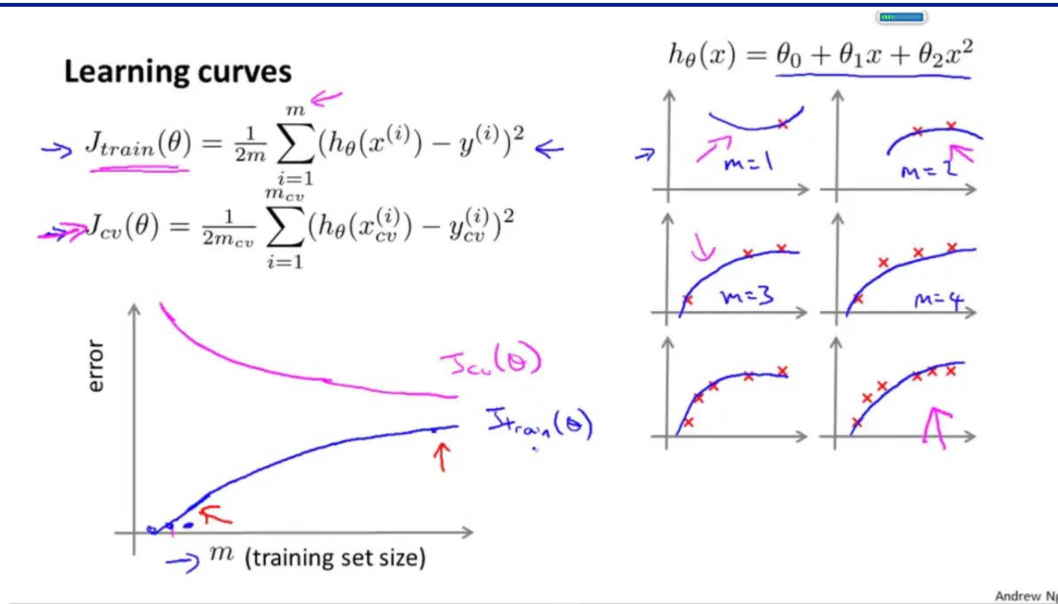
取x=5来计算偏差及方差，如下图所示

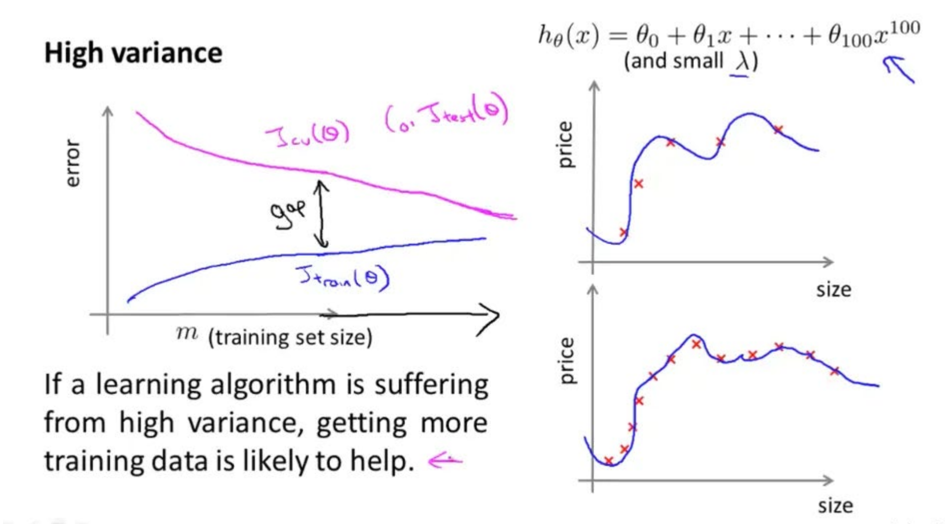


用线性函数来构建模型，每次取20个样本来训练一个线性模型，总共构造50个线性模型

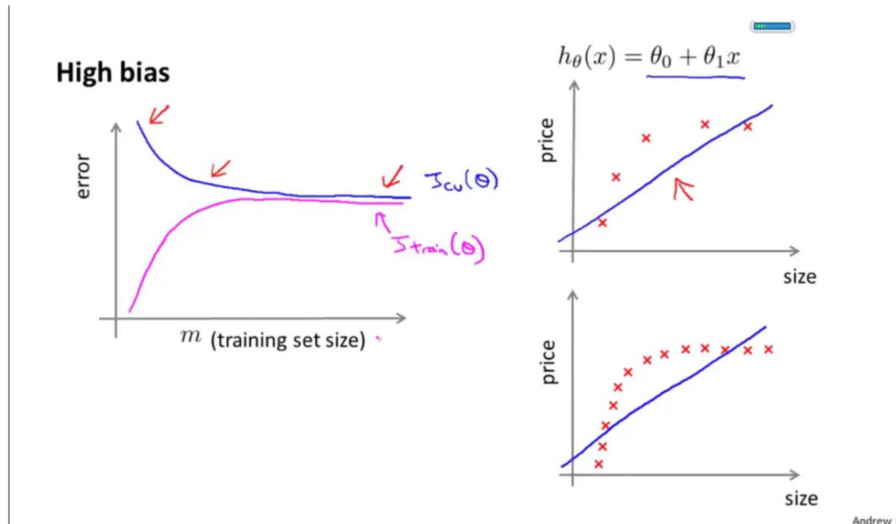
https://www.jianshu.com/p/8c7f033be58a

回归——样本的数量





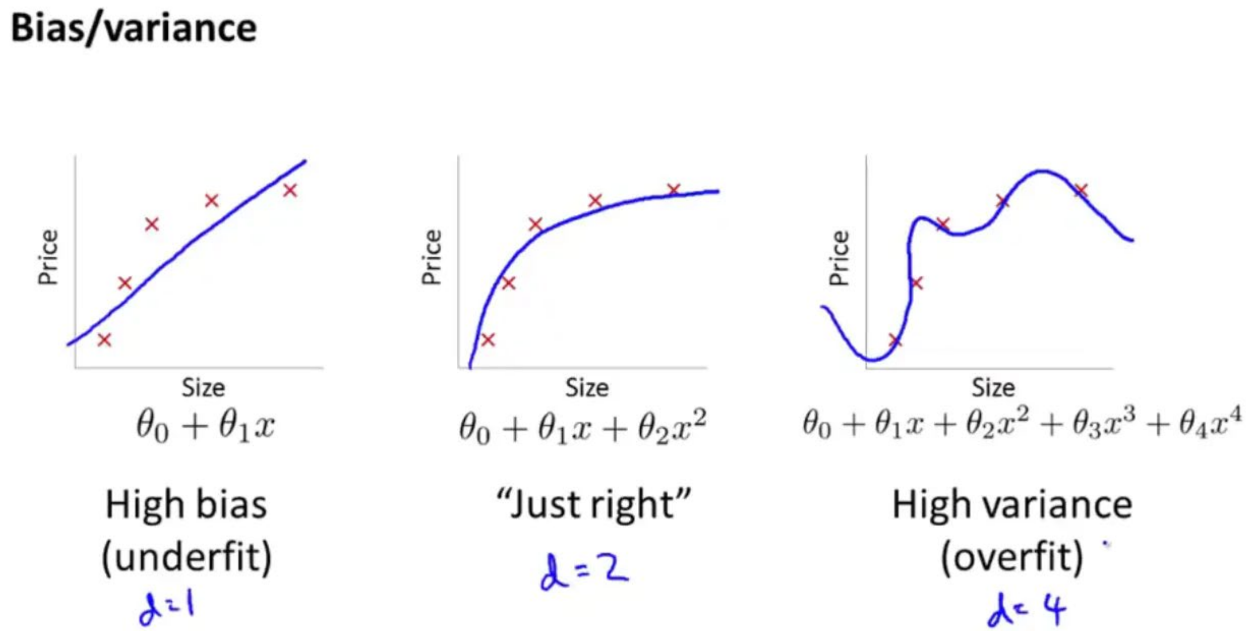
模型方差较高时，增加样本会有帮助



模型偏差较高时，增加样本帮助不大

https://www.jianshu.com/p/8c7f033be58a

回归——Polynomial Regression多项式回归



多项式次数	模型复杂度	方差	偏差	过/欠拟合
低	低	低	高	欠拟合
中	中	中	中	适度
高	高	高	低	过拟合

回归——Ridge Regression岭回归

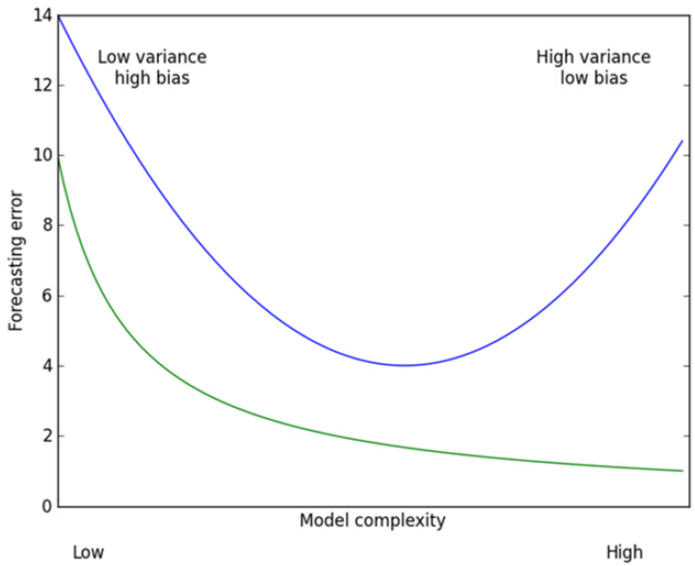
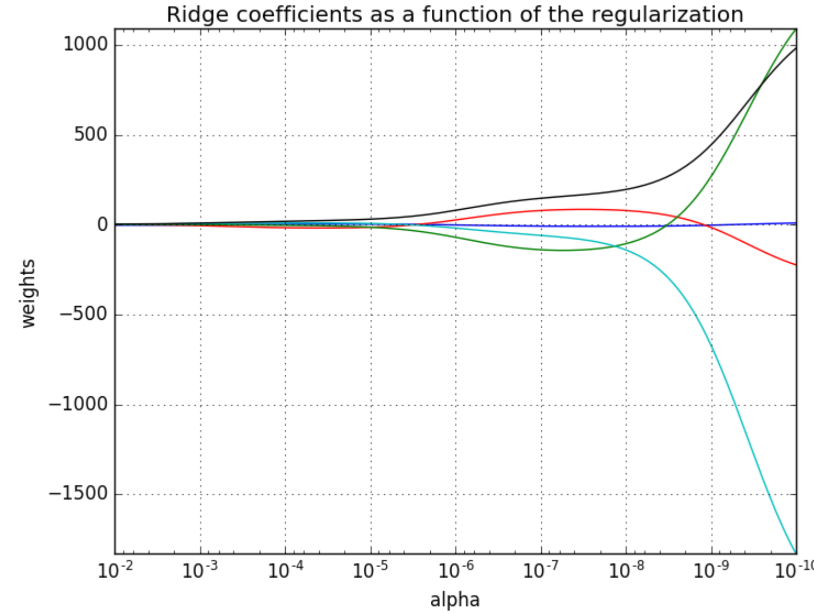


Figure 8.8 The bias variance tradeoff illustrated with test error and training error. The training error is the top curve, which has a minimum in the middle of the plot. In order to create the best forecasts, we should adjust our model complexity where the test error is at a minimum.

$$\min\{||Xw - y||^2 + \alpha ||w||^2\}$$

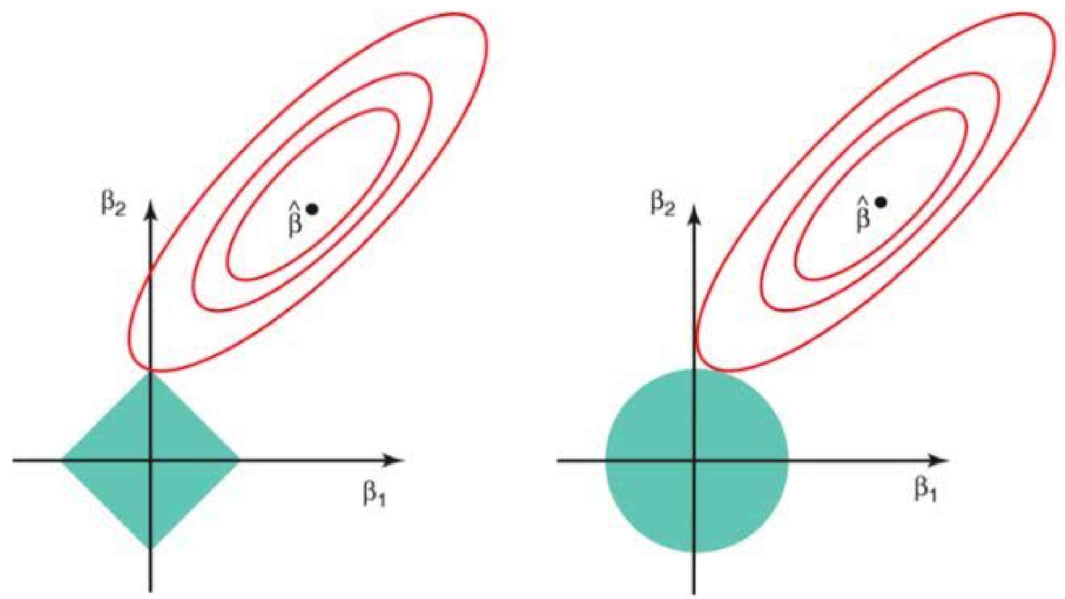


正则化项权重α	模型复杂度	方差	偏差	过/欠拟合
大	低	低	高	欠拟合
中	中	中	中	适度
小	高	高	低	过拟合

回归——Lasso Regression套索回归

Lasso estimate具有shrinkage(收缩方法又称为正则化regularization)和selection(选择部分重要特征)两种功能

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



Credit : An Introduction to Statistical Learning by Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani

重要优点：

（1）这种回归在模型假设方面与岭回归完全一致；

（2）模型可将某些变量对应的系数直接收缩至0，这有助于变量的筛选；

（3）该模型属于L1惩罚；

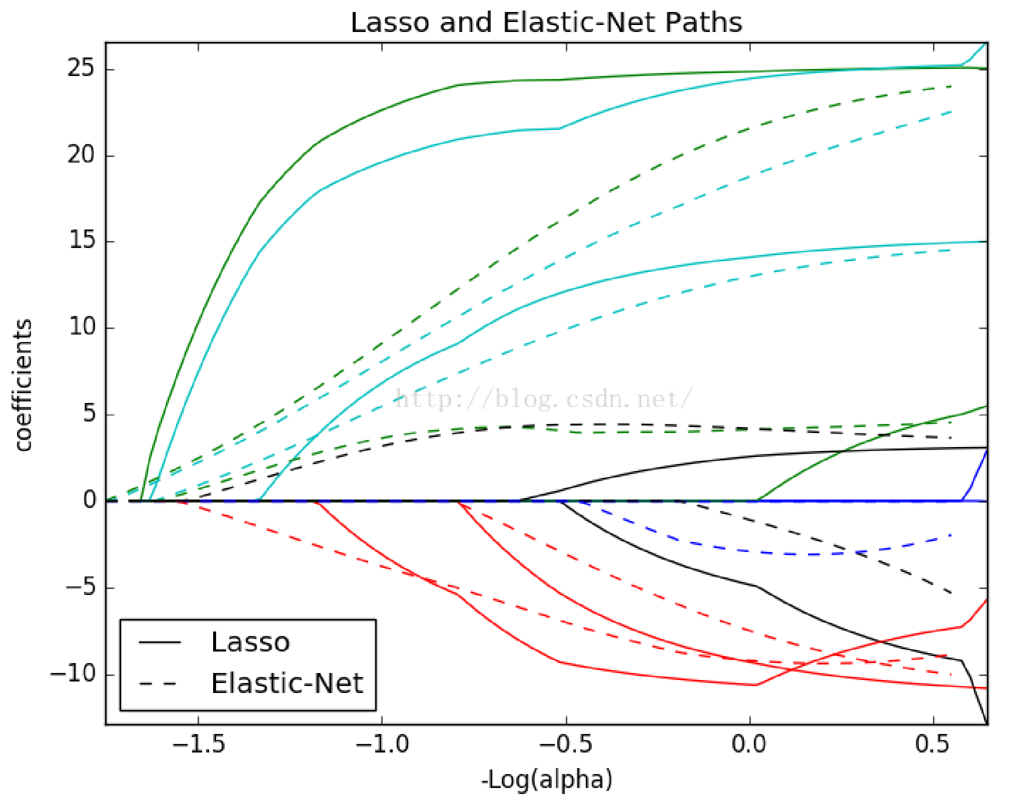
（4）如果存在某组预测因子高度相关的情况，Lasso方法仅会选取它们中的一个，并把所对应的系数收缩至0。

回归——ElasticNet回归

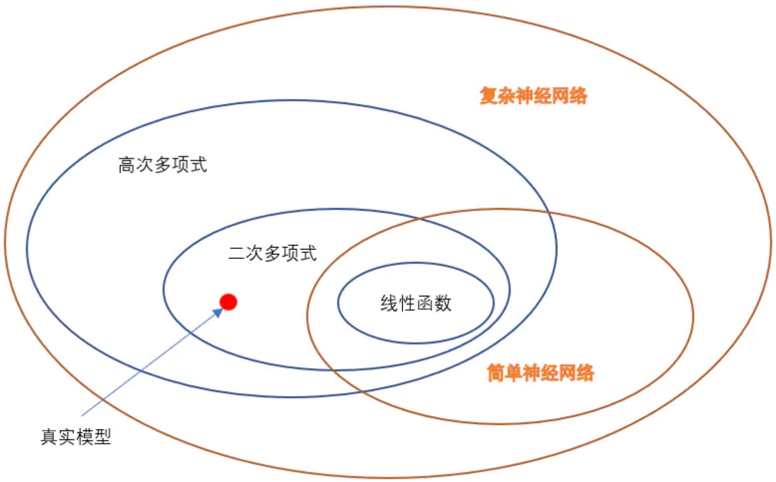
ElasticNet是Lasso和Ridge回归技术的混合体。它使用L1来训练并且L2优先作为正则化矩阵。

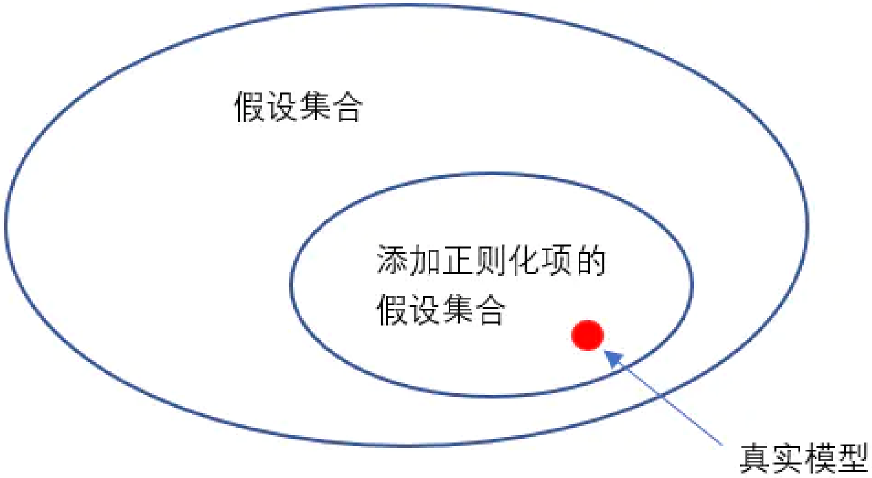
$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}}(\|y - X\beta\|^2 + \lambda_2\|\beta\|^2 + \lambda_1\|\beta\|_1)$$

ElasticNet在我们发现用Lasso回归太过(太多特征被稀疏为0),而岭回归也正则化的不够(回归系数衰减太慢)的时候，可以考虑使用ElasticNet回归来达到一个折衷的效果。



回归——选择假设集合





有监督学习的整个流程，其实就是一个不断缩小假设集合的过程。从大的方面看可以分为两个步骤：

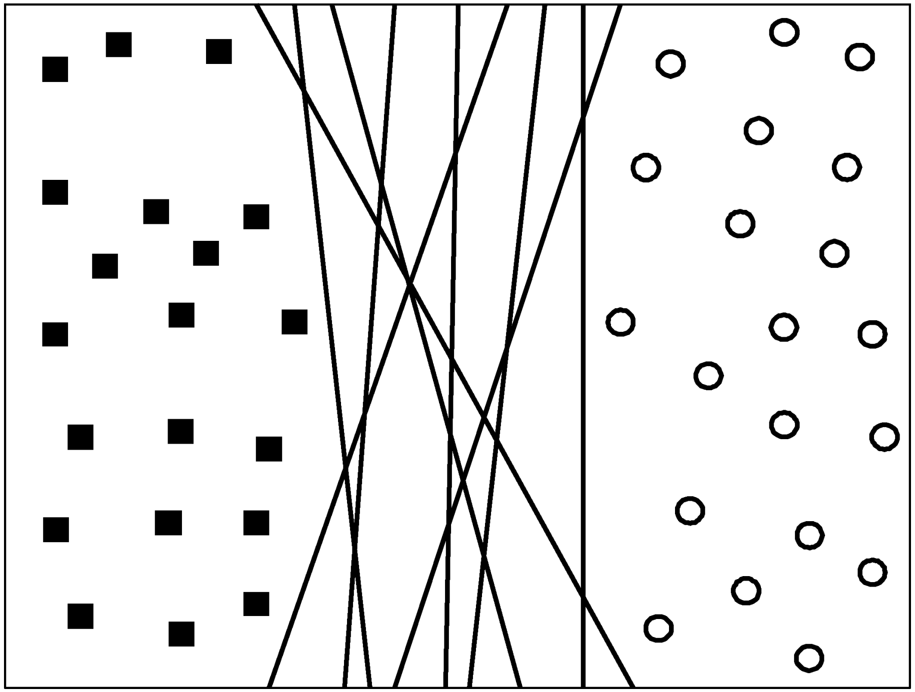
- 选择一个假设集合，包括模型及相关结构、超参数等。选择一些不同的假设集合，然后通过考察它们的偏差方差，对各假设集合的性能进行评估。
- 使用样本数据进行训练，使该模型尽量拟合样本，就是从上面选定的假设集合中找到一个特定的假设（模型）。

https://www.jianshu.com/p/8c7f033be58a

回归——SVM

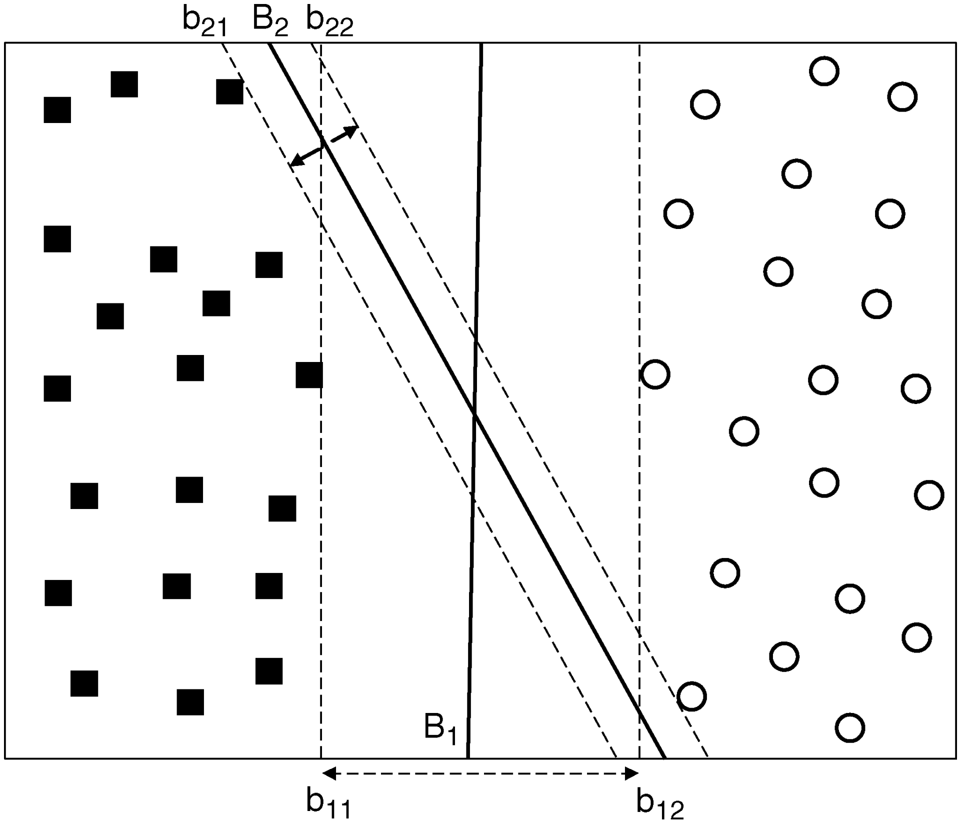
◆ 两个线性可分的类

- 找到这样一个超平面，使得所有的方块位于这个超平面的一侧，而所有的圆圈位于它的另一侧
- 可能存在无穷多个那样的超平面



回归——SVM

- 找这样的超平面，它最大化边缘
  - B1比B2好

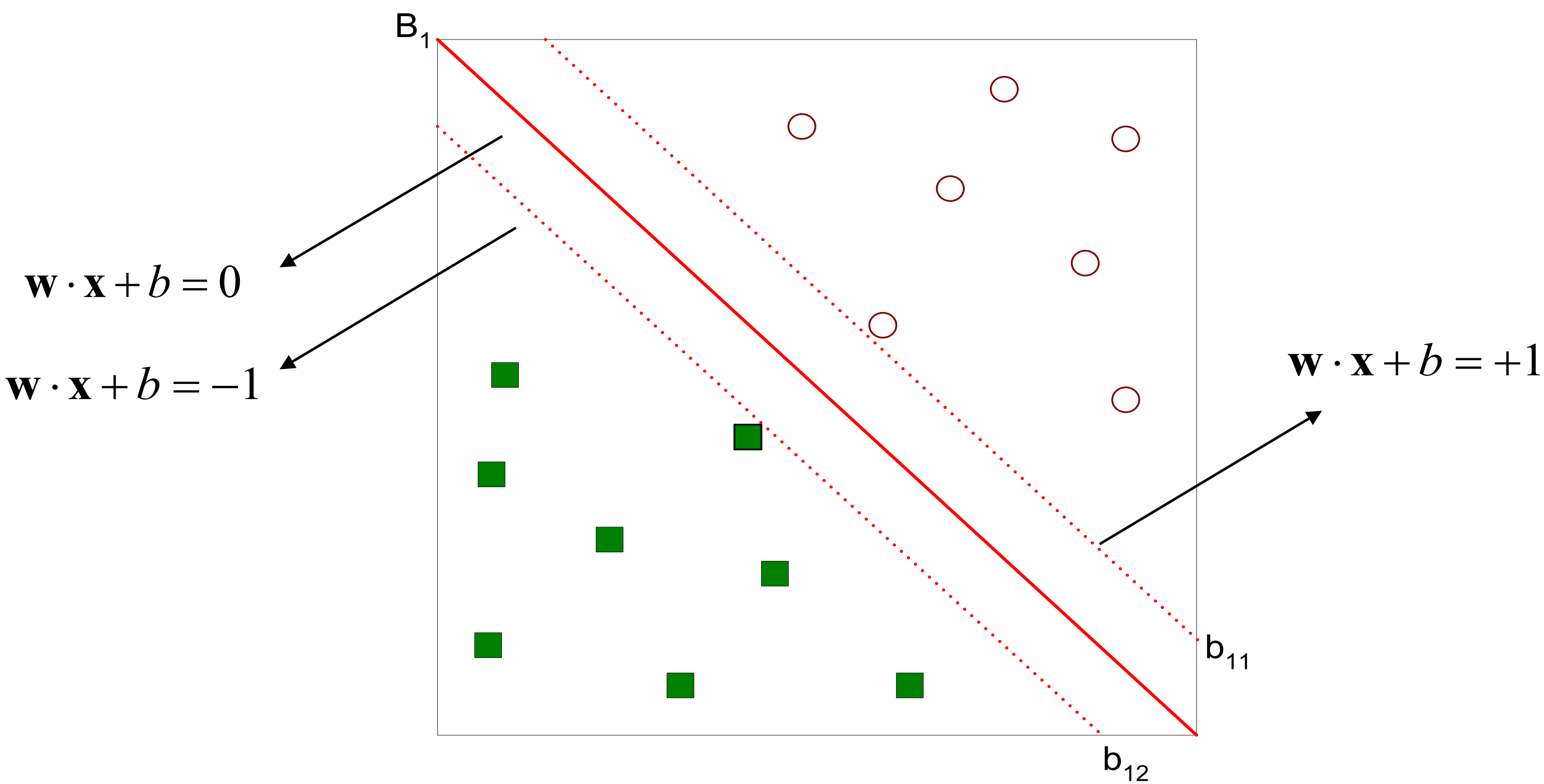


回归——SVM

- 一个线性分类器的决策边界可以写成如下形式：

  $$wx + b = 0$$

  – 其中，w和b是模型的参数



回归——SVM

- 方块的类标号为+1，圆圈的类标号为–1
- z的类标号y

$$y=\begin{cases}1 & \text{if } \mathbf{w}\cdot\mathbf{z}+\mathrm{b}\geq 0\\ -1 & \text{if } \mathbf{w}\cdot\mathbf{z}+\mathrm{b}<0\end{cases}$$

- 调整决策边界的参数w和b，两个平行的超平面b_{i1}和b_{i2}可以表示如下
  - b_{i1}： $w \cdot x + b = 1$
  - b_{i2}： $w \cdot x + b = -1$
- 可以证明, 边缘d

$$d=\frac{2}{\|\mathbf{w}\|}$$

回归——SVM

- w的方向垂直于决策边界
  - 如果x_a和x_b是任意两个位于决策边界上的点，则

$$wx_a + b = 0,\ wx_b + b = 0$$

  - 于是$w(x_b - x_a) = 0$. 由于$x_b - x_a$是决策超平面中任意向量（表示起点x_a指向终点x_b的向量），于是w的方向必然垂直于决策边界
- 令x_1是b_{i1}上的数据点，x_2是b_{i2}上的数据点. 代入b_{i1}和b_{i2}相减得到

$$w.(x_1 - x_2) = 2$$

- 由 $\cos(\mathbf{u},\mathbf{v}) = \dfrac{\mathbf{u}\cdot\mathbf{v}}{\|\mathbf{u}\|\|\mathbf{v}\|}$ 即 $\mathbf{u}\cdot\mathbf{v} = \|\mathbf{u}\|\|\mathbf{v}\|\cos(\mathbf{u},\mathbf{v})$

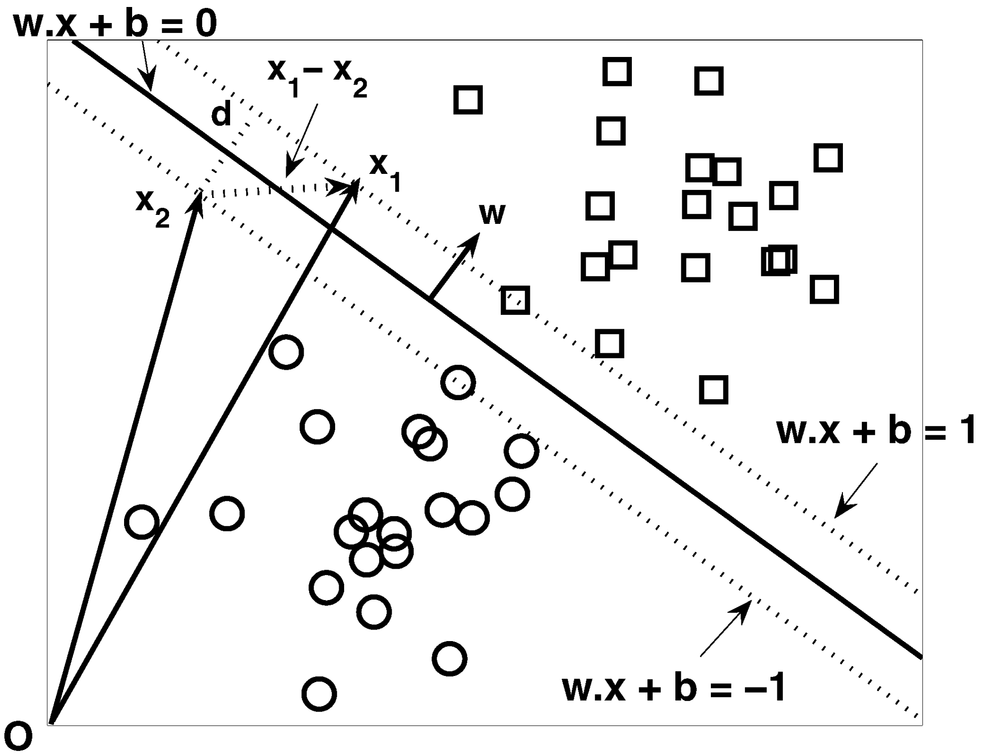
- 令u=w, v= $x_1 - x_2$, 得到

$$||w||||x_1 - x_2||\cos(w, x_1 - x_2)=2$$

回归——SVM

- 但$||x_1-x_2||\cos(w, x_1-x_2)=d$
- 于是$||w||\times d=2$，即

$$d=\frac{2}{\|\mathbf{w}\|}$$



回归——SVM

- SVM的训练阶段从训练数据中估计决策边界的参数w和*b* 最大化边缘*d*，并满足

  $\mathrm{w}x_i + b \geq 1$　如果$y_i = 1$

  $\mathrm{w}x_i + b \leq -1$　如果$y_i = -1$

- 即

  $y_i(\mathrm{w}x_\mathrm{i} + b) \geq 1$

- 最大化*d*等价于最小化

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

约束条件　$y_i(\mathbf{wx}_i + b) \geq 1, \quad i = 1,2,\ldots,N$

- 这是一个凸二次优化问题，可以通过标准的拉格朗日乘子（Lagrange multiplier）方法求解

回归——SVM

- 拉格朗日算子

$$L_p = \frac{1}{2}||\mathrm{w}||^2 - \sum_{i=1}^{N} \lambda_i (y_i(\mathrm{w} \cdot \mathrm{x}_i + b) - 1) \tag{5-38}$$

  – 其中，参数λ_i称为拉格朗日乘子

- 对L_p关于w和b求偏导，并令它们等于零

$$\frac{\partial L_p}{\partial \mathrm{w}} = 0 \Rightarrow \mathrm{w} = \sum_{i=1}^{N} \lambda_i y_i \mathrm{x}_i \tag{5-39}$$

$$\frac{\partial L_p}{\partial_b} = 0 \Rightarrow \sum_{i=1}^{N} \lambda_i y_i = 0 \tag{5-40}$$

- 因为拉格朗日乘子λ_i是未知的，因此仍然不能得到w和b的解

回归——SVM

- 使用Karuch-Kuhn-Tucher（KKT）条件：

$$\lambda_i \geq 0$$

$$\lambda_i\,[y_i(\mathrm{w}\mathrm{x}_i + b) - 1] = 0 \qquad (5.42)$$

  - 除非训练实例满足方程$y_i(\mathrm{w}\mathrm{x}_i + b) = 1$, 否则拉格朗日乘子$\lambda_i$必须为零
  - $\lambda_i > 0$的训练实例位于超平面b_{i1}或b_{i2}上，称为支持向量

- （5.39）和（5.40）代入到公式（5.38）中

$$L_D = \sum_{i=1}^{N} \lambda_i - \frac{1}{2}\sum_{i,j} \lambda_i \lambda_j y_i y_j \mathbf{x_i} \cdot \mathbf{x_j} \qquad (5\text{-}43)$$

$$\text{s.t.} \quad \sum_{i=1}^{N} \lambda_i y_i = 0$$

- 这是L_p的对偶问题(最大化问题)。可以使用数值计算技术,如二次规划来求解λ

回归——SVM

- 解出λ_i后, 用(5.39)求w, 再用(5.42)求b
- 决策边界为

$$\left(\sum_{i=1}^{N}\lambda_i y_i \mathbf{x}_i \cdot \mathbf{x}\right)+b=0$$

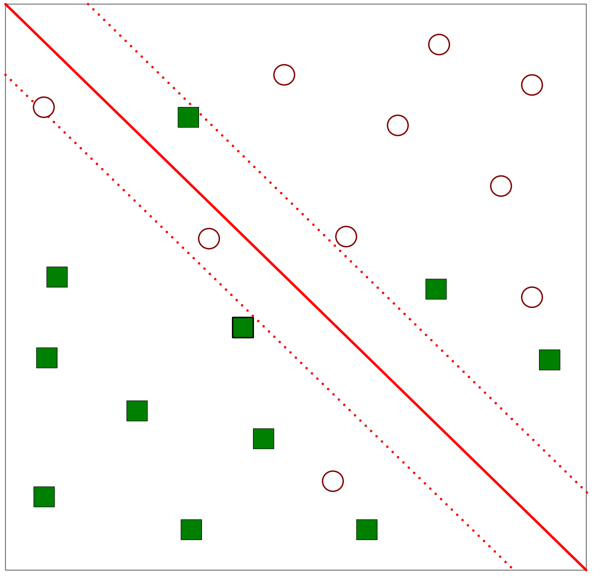
- z可以按以下的公式来分类

$$f(\mathbf{z})=sign(\mathbf{w}\mathbf{z}+b)=sign(\sum_{i=1}^{N}\lambda_i y_i \mathbf{x}_i \cdot \mathbf{z}+b)$$

  – 如果f(z) = 1，则检验实例z被分为到正类，否则分到负类

回归——SVM

- 如果不是线性可分的，怎么办？
  - 软边缘（soft margin）
    - 允许一定训练误差
  - 引入松弛变量ξ_i



$$\min \frac{1}{2}\|\mathbf{w}\|^2 + C(\sum_{i=1}^{N}\xi_i)^k$$

约束条件 $y_i(\mathbf{w}\mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = 1,2,\ldots,N$

  - 其中C和k是用户指定的参数，
    对误分训练实例加罚
  - 取k=1，
    C根据模型在确认集上的性能选择

回归——SVM

- 拉格朗日算子

$$L=\frac{1}{2}\|\mathbf{w}\|^2+C\sum_{i=1}^{N}\xi_i-\sum_{i=1}^{N}\lambda_i\{y_i(\mathbf{w}\cdot\mathbf{x_i}+b)-1+\xi_i\}-\sum_{i=1}^{N}\mu_i\xi_i$$

  – 其中，前面两项是需要最小化的目标函数，第三项表示与松弛变量相关的不等式约束，而最后一项是要求ξ_i的值非负的结果

- KKT条件

$$\xi_i \geq 0,\ \lambda_i \geq 0,\ \mu_i \geq 0$$

$$\lambda_i\{y_i(\mathrm{w}\cdot \mathrm{x}_i + b) - 1 + \xi_i\} = 0$$

$$\mu_i\,\xi_i = 0$$

回归——SVM

- 令L关于w, b和ξ_i的一阶导数为零

$$\frac{\partial L}{\partial w_j} = w_j - \sum_{i=1}^{N} \lambda_i y_i x_{ij} = 0 \Rightarrow w_j = \sum_{i=1}^{N} \lambda_i y_i x_{ij}$$

$$\frac{\partial L}{\partial b} = -\sum_{i=1}^{N} \lambda_i y_i = 0 \Rightarrow \sum_{i=1}^{N} \lambda_i y_i = 0$$

$$\frac{\partial L}{\partial \xi_i} = C - \lambda_i - \mu_i = 0 \Rightarrow \lambda_i + \mu_i = C$$

回归——SVM

- 对偶拉格朗日问题

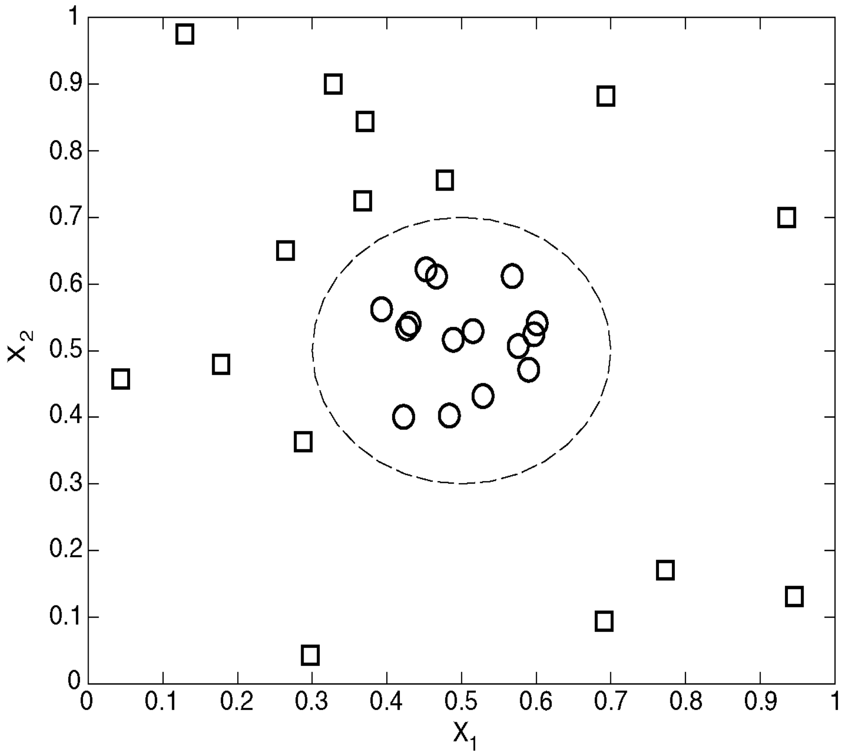
$$
\begin{aligned}
L_D &= \frac{1}{2}\sum_{i,j}\lambda_i\lambda_j y_i y_j \mathbf{x}_i\cdot\mathbf{x}_j + C\sum_i \xi_i \\
&\quad -\sum_i \lambda_i\{y_i(\sum_j \lambda_j y_j \mathbf{x}_i\cdot\mathbf{x}_j + b) - 1 + \xi_i\} \\
&\quad -\sum_i (C-\lambda_i)\xi_i \\
&= \sum_{i=1}^{N}\lambda_i - \frac{1}{2}\sum_{i,j}\lambda_i\lambda_j y_i y_j \mathbf{x_i}\cdot\mathbf{x_j}
\end{aligned}
$$

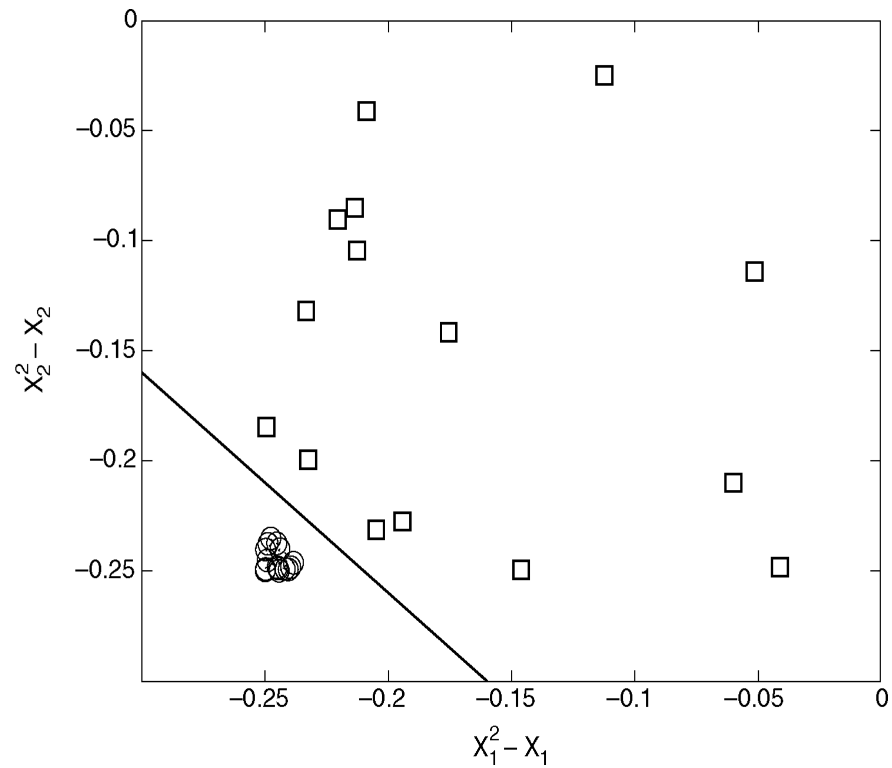
s.t. $\sum_{i=1}^{N}\lambda_i y_i = 0$

– 其形式与可分情况相同, 但是$0\leq\lambda_i\leq C$
– 类似于标准SVM模型的求解过程进行求解

回归——SVM

- 使用非线性变换Φ
- 例: $\Phi:(x_1,x_2)\rightarrow(x_1^2,x_2^2,\sqrt{2}x_1,\sqrt{2}x_2,\sqrt{2}x_1x_2,1)$





□ $x_2^2-x_2+a(x_1^2-x_1)-b>0 \Rightarrow \left(x_2-\frac{1}{2}\right)^2+\left(x_2-\frac{a}{2}\right)^2-const>0$

回归——SVM

- 非线性 SVM的优化问题

$$\min_{\mathbf{w}} \frac{\|\mathbf{w}\|^2}{2}$$

约束条件: $y_i(\mathrm{w}\Phi(\mathrm{x}_i) + b) \geq 1,\ i = 1, 2, \ldots, N$

- 对偶拉格朗日问题

$$L_D = \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$$

- 参数w和b

$$\mathbf{w} = \sum_i \lambda_i y_i \Phi(\mathbf{x}_i)$$

$$\lambda_i \{y_i \sum_j \lambda_j y_j \Phi(\mathbf{x}_j) \cdot \Phi(\mathbf{x}_i) + b) - 1\} = 0$$

回归——SVM

- 实例z分类

$$f(\mathbf{z}) = sign(\mathbf{w}\cdot\Phi(\mathbf{z})+b) = sign\left(\sum_{i=1}^{n}\lambda_j y_j \Phi(\mathbf{x}_i)\cdot\Phi(\mathbf{z})+b\right)$$

- 点积$\Phi(x_i)\cdot\Phi(x_j)$ 计算问题
  - 能否在原空间直接计算?
  - 例:

$$\Phi:(x_1,x_2)\rightarrow(x_1^2,x_2^2,\sqrt{2}x_1,\sqrt{2}x_2,\sqrt{2}x_1x_2,1)$$

$$\begin{aligned}\Phi(\mathbf{u})\cdot\Phi(\mathbf{v}) &= (u_1^2,u_2^2,\sqrt{2}u_1,\sqrt{2}u_2,\sqrt{2}u_1u_2,1)\cdot(v_1^2,v_2^2,\sqrt{2}v_1,\sqrt{2}v_2,\sqrt{2}v_1v_2,1)\\ &= u_1^2v_1^2+u_2^2v_2^2+2u_1v_1+2u_2v_2+2u_1u_2v_1v_2+1\\ &= (\mathbf{u}\cdot\mathbf{v}+1)^2\end{aligned}$$

核技术

- Mercer定理:

  核函数K可以表示为$K(u, v) = \Phi(u)\cdot\Phi(v)$

  核函数的充要条件是K矩阵是半正定的。

- 常用核函数

$$K(\mathrm{x},\mathrm{y}) = e^{-\|\mathrm{x}-\mathrm{y}\|^2/(2\sigma^2)}$$

$$K(\mathrm{x},\mathrm{y}) = \tanh(k\mathrm{x}\cdot\mathrm{y}-\delta)$$