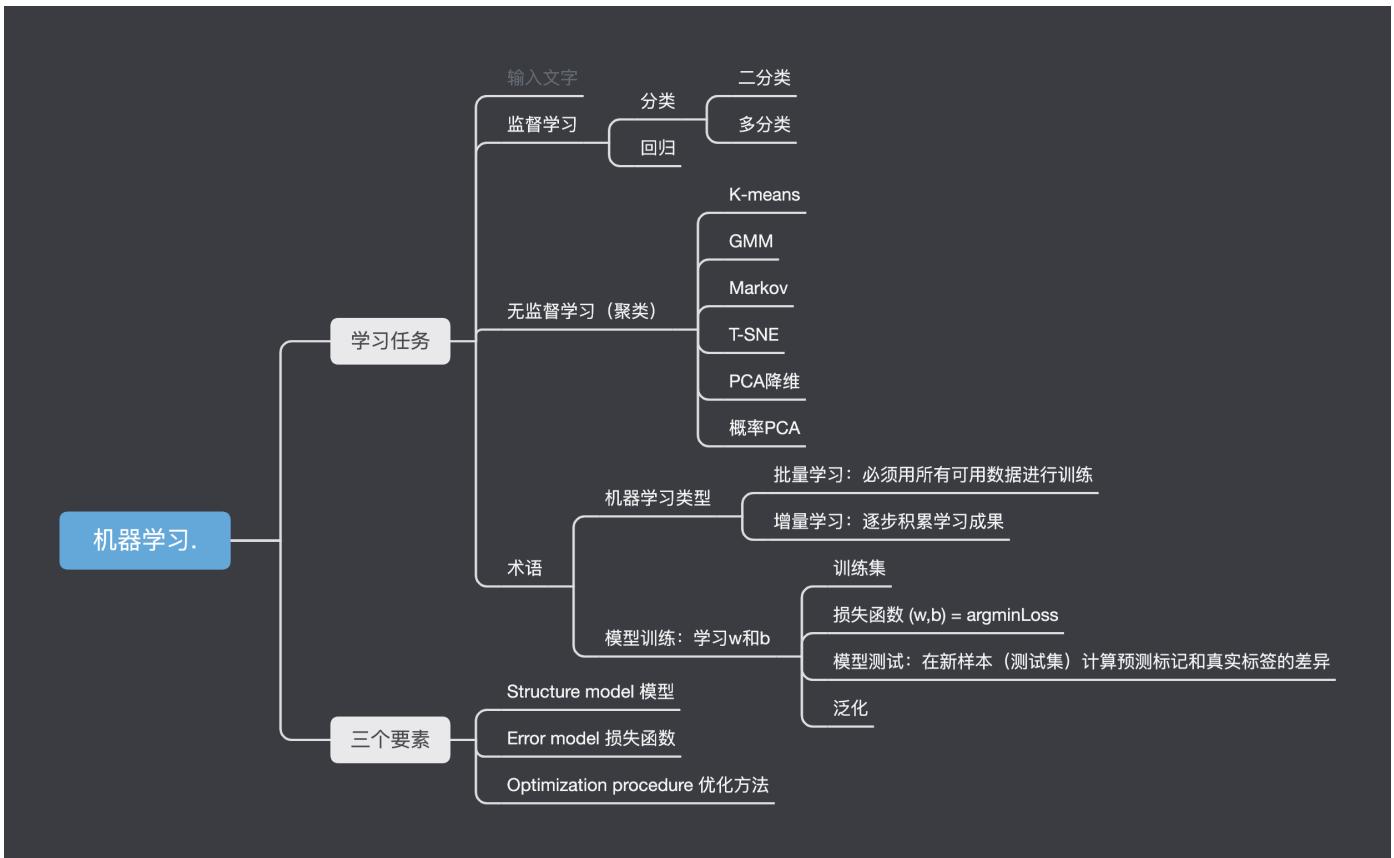


机器学习概论



多元线性回归

函数模型

函数形式

$$f(x) = \theta_0 + \theta_1 x_1 + \cdots + \theta_p x_p$$

向量形式：

通常一个向量指的都是列向量，向量的转置是行向量

$$f(x) = \sum_{i=0}^p \theta_i x_i = \boldsymbol{\theta}^T \mathbf{x} = \mathbf{x}^T \boldsymbol{\theta} = [(x_0 = 1), x_1, x_2, \dots, x_p] \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_p \end{bmatrix}$$

损失函数：最小均方误差MSE：

$$J(\boldsymbol{\theta}) = \frac{1}{2} \sum_{i=1}^n (\mathbf{x}_i^T \boldsymbol{\theta} - y_i)^2$$

线性回归模型：求解损失函数的最小值

$$\boldsymbol{\theta}^* = \operatorname{argmin} J(\boldsymbol{\theta})$$

加入数据后的模型

n组数据 样本量为n时 $\begin{cases} X: n \times p \\ P: p \times 1 \end{cases}$

预测值：

$$\hat{Y} = X\theta = \begin{bmatrix} X_1^T \theta \\ X_2^T \theta \\ \vdots \\ X_n^T \theta \end{bmatrix} = \begin{bmatrix} X_{11} & X_{12} & \dots & X_{1p} \\ X_{21} & X_{22} & \dots & X_{2p} \\ \vdots & & & \\ X_{n1} & X_{n2} & \dots & X_{np} \end{bmatrix} \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_p \end{bmatrix} = n \times 1 \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

实际值label (n组数据n个label)：

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

模型求解

梯度下降法

Gradient Decent

$$\theta := \theta - \alpha \nabla_{\theta} J(\theta)$$

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

其中算子：梯度是偏导数的自然扩展

$$\nabla_{\theta} J = \begin{bmatrix} \frac{\partial J}{\partial \theta_0} \\ \vdots \\ \frac{\partial J}{\partial \theta_p} \end{bmatrix}$$

求损失函数的偏导：

$$\begin{aligned} & \frac{\partial}{\partial \theta_j} (x_i^T \theta - y_i)^2 \\ &= \frac{\partial}{\partial \theta_j} \left(\sum_{j=0}^p x_{i,j} \theta_j - y_i \right)^2 \quad x_i = (x_{i,0}, \dots, x_{i,p})^T \\ &= \left(\sum_{j=0}^p x_{i,j} \theta_j - y_i \right) \frac{\partial}{\partial \theta_j} \left(\sum_{j=0}^p x_{i,j} \theta_j - y_i \right) \\ &= (f(x_i) - y_i) x_{i,j} \end{aligned}$$

正规方程法

$$\begin{aligned} J(\theta) &= \frac{1}{2} \|Y - X\theta\|^2 \\ &= \frac{1}{2} (X\theta - Y)^T (X\theta - Y) \\ &= \frac{1}{2} (\theta^T \underbrace{X^T X}_{B} \theta - 2Y^T X\theta + Y^T Y) \end{aligned}$$

注解：

$$\begin{aligned} \frac{\partial \mathbf{x}^T \mathbf{B} \mathbf{x}}{\partial \mathbf{x}} &= (\mathbf{B} + \mathbf{B}^T) \mathbf{x} \\ \frac{\partial \mathbf{x}^T \mathbf{a}}{\partial \mathbf{x}} &= \frac{\partial \mathbf{a}^T \mathbf{x}}{\partial \mathbf{x}} = \mathbf{a} \end{aligned}$$

我们令 $B = X^T X, B^T = B \Rightarrow (B + B^T)\theta = 2B\theta$

$$\begin{aligned} \nabla_{\theta} J(\theta) &= \frac{\partial J(\theta)}{\partial \theta} = \frac{\frac{1}{2} (\theta^T X^T X \theta - 2Y^T X \theta + Y^T Y)}{\partial \theta} = X^T X \theta - (Y^T X)^T = X^T X \theta - X^T Y = 0 \\ &\Rightarrow X^T X \theta = X^T Y \theta^* = (X^T X)^{-1} X^T \\ &\Rightarrow \theta^* = (X^T X)^{-1} X^T Y \end{aligned}$$

随机梯度下降法

Mini-batch GD

每次只用训练集中的一个数据，把数据分为若干个批，按批来更新参数。一个批中的一组数据共同决定了本次梯度的方向，下降起来就不容易跑偏，减少了随机性。

一个batch形成一个epoch分批次训练

全局最优解

当 $J(\theta)$ 是凸函数（凹函数和凸函数统称凸函数）时，二阶导数大于0, $X^T X$ 为半正定矩阵

$$\nabla_{\theta}^2 J(\theta) = X^T X$$

当训练样本的数目 n 大于训练样本的维度 ($p+1$ 个属性, 特征) $X^T X$ 通常可逆，表明改矩阵是正定矩阵，求的参数是全局最优解。不可逆时，可以接出多个参数解。可使用正则化给出一个“归纳偏好”解。

评估方法

留出法

随机挑选一部分标记数据作为测试集(空心点)，其余的作为训练集(实心点)，计算回归模型，使用测试集对模型评估：MSE = 2.4，测试集不能太大，也不能太小。 $2 \leq n:m \leq 4$

交叉验证法

性能度量

线性回归模型：平方和误差

在测试集上报告 MSE(mean square error) 误差

$$J_{\text{train}}(\theta) = \frac{1}{2} \sum_{i=1}^n (\mathbf{x}_i^T \theta - y_i)^2$$

$$\theta^* = \operatorname{argmin} J_{\text{train}}(\theta) = (X_{\text{train}}^T X_{\text{train}})^{-1} X_{\text{train}}^T \vec{y}_{\text{train}}$$

$$J_{\text{test}} = \frac{1}{m} \sum_{i=n+1}^{n+m} (\mathbf{x}_i^T \theta^* - y_i)^2 = \frac{1}{m} \sum_{i=n+1}^{n+m} \varepsilon_i^2$$

分类任务：错误率与精度

错误率是分类错误的样本数占样本总数的比例

精度是分类正确的样本数占样本总数的比例

对二分类问题：

查准率： $P = \frac{TP}{TP+FP}$

查全率： $R = \frac{TP}{TP+FN}$

F1:

$$F1 = \frac{2 \times P \times R}{P + R} = \frac{2 \times TP}{\text{样例总数} + TP - TN}$$

基于非线形基的线性回归

多项式回归

LR-逻辑回归

Structural model

逻辑函数 (logistic/sigmoid function)

$$y = \frac{1}{1 + e^{-z}} = \frac{1}{1 + e^{-\theta x}}$$

Error model

损失函数 Loss function

$$P(y=1 \mid x; \theta) = f_{\theta}(x) = \frac{1}{1+e^{-\theta^T * x}}$$

$$P(y=0 \mid x; \theta) = 1 - f_{\theta}(x) = \frac{e^{-\theta^T * x}}{1+e^{-\theta^T * x}}$$

求参数方法--对 θ 极大似然估计，使得y发生的概率最大

$$L(\theta) = \prod_{i=1}^n P(y_i \mid x_i; \theta) = \prod_{i=1}^n (f_{\theta}(x_i))^{y_i} (1 - f_{\theta}(x_i))^{1-y_i}$$

转化为对数函数

$$\ln L(\theta) = \sum_{i=1}^n (y_i \ln(f_{\theta}(x_i)) + (1 - y_i) \ln(1 - f_{\theta}(x_i)))$$

$$= \sum_{i=1}^n \left((1 - y_i) (-\theta^T * x_i) - \ln(1 + e^{-\theta^T * x_i}) \right)$$

梯度上升

$$\theta := \theta + \alpha \nabla_{\theta} \ln(L(\theta)) \Leftrightarrow \theta_j := \theta_j + \frac{\partial \ln(L(\theta))}{\partial \theta_j}$$

求梯度

$$\nabla_{\theta} \ln(L(\theta)) = \sum_{i=1}^n \left[-(1 - y_i) \cdot x_i - \frac{1}{1 + e^{-\theta^T x_i}} \left(e^{-\theta^T x_i} \right) (-x_i) \right]$$

$$= \sum_{i=1}^n \left(-1 + y_i + \frac{e^{-\theta^T x_i}}{1 + e^{-\theta^T x_i}} \right) x_i$$

$$= \sum_{i=1}^n (y_i - f_{\theta}(x_i)) x_i \Leftrightarrow \frac{\partial}{\partial \theta_j} \ln(L(\theta)) = \sum_{i=1}^n (y_i - f_{\theta}(x_i)) x_{i,j}$$

代入梯度的参数更新

$$\theta := \theta + \alpha \nabla_{\theta} \ln(L(\theta)) \Rightarrow \theta := \theta + \alpha \sum_{i=1}^n (y_i - f_{\theta}(x_i)) x_i$$

和线性回归的对比

和线形回归模型看似一样，但是f不同，逻辑回归解决的是二分类问题

	逻辑回归	线性回归
输出		
	线形二分类	线性拟合

NN-神经网络

Structural model

逻辑回归的二阶段表示

$$z = b + \sum_{p \times 1} x_i w_{1 \times p} + b_{1 \times 1}$$

$$\hat{y} = \text{sigmoid}(z) = \frac{e^z}{1+e^z}$$

神经元

神经元=线性组合(z, 接收信号)+非线性激活(sigmoid, 输出非线性决策面)

$$z_t = W_1^T \mathbf{x}$$

多神经元

神经网络包含多个神经元，输入x与多个神经元相连。

一个隐藏层的神经网络

$$\begin{aligned} z_1 &= W_1^T \mathbf{x} \\ h_1 &= \text{sigmoid}(z_1) \\ z_2 &= w_2^T h_1 \\ \hat{y} &= \text{sigmoid}(z_2) \end{aligned}$$

W表示X的第j个元素与向量Z的第i个元素之间的链接权重

$$\begin{aligned} W &= \begin{bmatrix} W_{11} & W_{21} & W_{31} & W_{41} \\ W_{12} & W_{22} & W_{32} & W_{42} \\ W_{13} & W_{23} & W_{33} & W_{43} \end{bmatrix} \\ W^T &= \begin{bmatrix} W_{11} & W_{12} & W_{13} \\ W_{21} & W_{22} & W_{23} \\ W_{31} & W_{32} & W_{33} \\ W_{41} & W_{42} & W_{43} \end{bmatrix} \end{aligned}$$

隐含层h

没有隐含层就只需要一个列向量，因为有隐含层所以需要W矩阵
每一层计算就是线性组合+非线形激活

非线形激活函数

引入非线性激活函数的目的是得到非线性决策面，非线形激活函数可以逼近任何复杂的函数，不论网络多深，线形函数只能输出线性决策面。

非线形激活函数

Relu效果最好，因为有部分导数为0，有些为1，为0的部分可以让有些神经元停止学习，起到dropout的作用，可以有效防止过拟合。

binary step

$$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$$

Logistic

$$f(x) = \frac{1}{1 + e^{-x}}$$

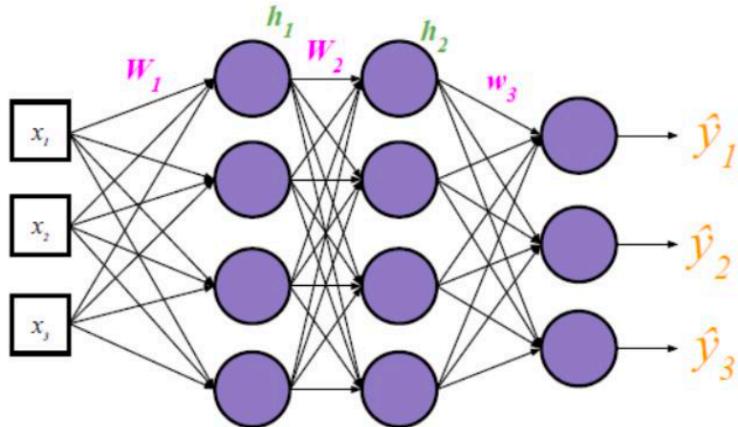
Tanh

$$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$$

ReLU

$$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$$

多分类神经网络



$$z_1 = \mathbf{W}_1^T \mathbf{x}$$

$$h_1 = \text{sigmoid}(z_1)$$

$$z_2 = \mathbf{W}_2^T h_1$$

$$h_2 = \text{sigmoid}(z_2)$$

$$z_3 = w_3^T h_2$$

$$\hat{y} = \text{sigmoid}(z_3)$$

h_1 表示hidden layer 1 output

Hidden layer(隐层)的个数大于1的神经网络，称为深度神经网络

Error model

非正确预测导致的代价

Loss function

交叉熵函数 (cross entropy loss)

二分类损失

逻辑回归中，使用对数似然度量损失(每个样本属于其真实 标记的概率越大越好)

$$\begin{aligned} E = \text{loss} &= -\log P(Y = \hat{y} \mid \mathbf{X} = \mathbf{x}) \\ &= -y \log(\hat{y}) - (1 - y) \log(1 - \hat{y}) \end{aligned}$$

多分类损失

Softmax函数

(柔性 最大值):将输出值转化成概率。

$$\hat{y}_i = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}} = P(y_i = 1 \mid \mathbf{x})$$

y_j 为one-hot 向量,真实标签位置1其他位置为0

$$E = \text{loss} = - \sum_{j=1..K} y_j \log \hat{y}_j$$

回归损失

与分类网络不同：输出层（最后一层）不再包含sigmoid函数

二次代价函数

$$\begin{aligned} E = \text{Los } s &= \frac{1}{2} \|y - \hat{y}\|^2 \\ &= \frac{1}{2} \sum_{j=1}^K (y_j - \hat{y}_j)^2 \end{aligned}$$

模型建模

优化参数目标:寻找使损失达到最小的神经网络权重

$$\mathbf{W}^* = \underset{\mathbf{W}}{\operatorname{argmin}} E(\hat{y}; \mathbf{W})$$

如何学习实现目标的神经网络权重 W --梯度下降

$$W_L(t+1) = W_L(t) - \eta \frac{\partial E}{\partial W_L(t)}$$

反向传播

求偏导从而应用梯度下降

1. 重复应用微积分的链式法则
2. 局部最小化目标函数
3. 要求网络所有的“块”(blocks)都是可微的

正向计算--节点

反向求导--边 链式法则从后往前求

反向传播--回归实例

回归损失函数为二次代价函数

$$E = loss = \frac{1}{2} (y - \hat{y})^2$$

反向传播--二分类实例

二分类损失函数为交叉熵损失函数

$$\text{Loss} = -y \ln(\hat{y}) - (1 - y) \ln(1 - \hat{y})$$

通过梯度下降 最小化Loss

$$\begin{aligned} w_2(t+1) &= w_2(t) - \eta \frac{\partial E}{\partial w_2(t)} \\ W_1(t+1) &= W_1(t) - \eta \frac{\partial E}{\partial W_1(t)} \end{aligned}$$

函数关于一个矩阵求偏导-->对每一个元素求偏导, W_{11}^1 表示输入 x 的第 j 个元素到第一个隐层的第 i 个神经元的权重

$$\begin{aligned} E &= -y \ln(\hat{y}) - (1 - y) \ln(1 - \hat{y}) \\ \hat{y} &= \frac{e^{z_2}}{1 + e^{z_2}} \\ z_2 &= \mathbf{w}_2^T \mathbf{h}_1 \\ \mathbf{h}_1 &= \frac{e^{z_1}}{1 + e^{z_1}} \\ z_1 &= W_1^T \mathbf{x} \\ \frac{\partial E}{\partial W_1} &= \frac{\partial E}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z_2} \cdot \left| \frac{\partial z_2}{\partial \mathbf{h}_1} \cdot \frac{\partial \mathbf{h}_1}{\partial z_1} \cdot \frac{\partial z_1}{\partial W_1} \right| \end{aligned}$$

Hadamard (哈达玛)乘积 /schur 乘积

假设 s 和 t 是两个同样维度的向量, 使用 $s \circ t$ (或 $s \odot t$) 来表示按元素的乘积: $(s \odot t) = s_j t_j$

$$\begin{bmatrix} 1 \\ 2 \end{bmatrix} \odot \begin{bmatrix} 3 \\ 4 \end{bmatrix} = \begin{bmatrix} 1 * 3 \\ 2 * 4 \end{bmatrix} = \begin{bmatrix} 3 \\ 8 \end{bmatrix}$$

反向传播的局部性

反向传播的一般情形

第 l 层第 j 个神经元和第 $l - 1$ 层神经元之间关系

$$z_j^l = \sum_{k=1} w_{jk}^{l-1} h_k^{l-1} + b_j^{l-1}$$

反向传播的一般情形

一些定义

$$\delta_j^l : \quad \delta_j^l \equiv \frac{\partial E}{\partial z_j^l}, \text{ 称为在第 } l \text{ 层第 } j \text{ 个神经元的误差}$$

$$\begin{aligned} z_j^l &= \sum_{k=1} w_{jk}^{l-1} h_k^{l-1} + b_j^{l-1} \\ h_j^l &= \sigma(z_j^l) \\ \sigma(x) &= \frac{1}{1+e^{-x}} \end{aligned}$$

矩阵表达形式--代价函数

$$E = \frac{1}{2} \|\mathbf{y} - \mathbf{h}^L\|^2 = \frac{1}{2} \|\mathbf{y} - \hat{\mathbf{y}}\|^2$$

第 l 层第 j 个神经元和第 $l-1$ 层神经元之间的关系:

$$z_j^l = \sum_{k=1} w_{jk}^{l-1} h_k^{l-1} + b_j^{l-1}, \quad h_j^l = \sigma(z_j^l)$$

🐮 反向传播四个方程

BP1

输出层（最后一层，即为 L 层）误差的方程

$$E = \frac{1}{2} \|\mathbf{y} - \mathbf{h}^L\|^2 = \frac{1}{2} \sum_{j=1}^K (h_j^L - y_j)^2$$

第 L 层第 j 个神经元的误差

$$\delta_j^L = \frac{\partial E}{\partial z_j^L} = \frac{\partial E}{\partial h_j^L} \frac{\partial h_j^L}{\partial z_j^L} = (h_j^L - y_j) \sigma'(z_j^L)$$

向量表达形式:

$$\delta_L = (\mathbf{h}^L - \mathbf{y}) \odot \sigma'(\mathbf{z}^L)$$

BP2

每一层的误差，使用下一层的误差 δ^{l+1} 表示当前层的误差 δ^l :

$$\delta^l = \sigma'(\mathbf{z}^l) \odot (\mathbf{W}^l \delta^{l+1})$$

BP3

代价函数关于偏置b的偏导

BP4

代价函数关于权重的偏导

Summary

向量形式：

$$\begin{aligned} \text{(BP1)} \quad & \delta^L = (\mathbf{h}^L - y) \odot \sigma'(\mathbf{z}^L) \\ \text{(BP2)} \quad & \delta^l = \sigma'(\mathbf{z}^l) \odot (\mathbf{W}^l \delta^{l+1}) \\ \text{(BP3)} \quad & \frac{\partial E}{\partial b^{l-1}} = \delta^l \\ \text{(BP4)} \quad & \frac{\partial E}{\partial W^{l-1}} = \mathbf{h}^{l-1} (\delta^l)^T \end{aligned}$$

数学形式：

$$\begin{aligned} BP1 : \quad & \delta_j^L = (h_j^L - y_j) \sigma'(z_j^L) \\ BP2 : \quad & \delta_j^l = \sum_{k=1} \delta_k^{l+1} w_{kj}^l \sigma'(z_j^l) \\ BP3 : \quad & \frac{\partial E}{\partial b_j^{l-1}} = \delta_j^l \\ BP4 : \quad & \frac{\partial E}{\partial w_{jk}^{l-1}} = h_k^{l-1} \delta_j^l \end{aligned}$$

反向传播算法

1. 输入x：为输入层设置对应的激活值h1
2. 前向传播：线性组合+非线形激活
3. 输出层误差和反向误差传播：BP1和BP2
4. 输出：误差函数的梯度由BP3和BP4给出

模型改进

改进损失函数

对数似然

损失函数对比

交叉熵VS二次代价函数

	二次代价函数	交叉熵
函数表达式	$E = \frac{1}{2}(y - \hat{y})^2$	$E = -y \ln \hat{y} - (1 - y) \ln(1 - \hat{y})$
对参数w偏导	$\frac{\partial E}{\partial w} = (\hat{y} - y)\sigma'(z)x = (\sigma(z) - y)\sigma'(z)x$	$\frac{\partial E}{\partial w} = (\hat{y} - y)x = (\sigma(z) - y)x$
对参数b的求导	$\frac{\partial E}{\partial b} = (\hat{y} - y)\sigma'(z) = (\sigma(z) - y)\sigma'(z)$	$\frac{\partial E}{\partial b} = (\hat{y} - y) = (\sigma(z) - y)$

权重初始化建议

随机初始化:使用Numpy的 np.random.randn函数生成均值为0, 标准差为1的高斯分布。

改进:对于任意l层, 使用均值为0, 标准差为1 的高斯分布随机分布初始化权重参数 W^{l-1}, b^{l-1} 。此时中间变量 z^l 服从均值为0, 标准差为1的高斯分布。

减少过拟合:dropout

1. 随机地删除网络中的一半的隐藏神经元, 同时让输入层和输出层的神经元保持不变。
2. 把输入x通过修改后的网络前向传播, 然后把得到的损失结果通过修改的网络反向传播。在mini-batch 上执行完这个过程后, 在没有被删除的神经元上更新对应的参数(w, b)
3. 继续重复上述过程:
 - 恢复被删掉的神经元(此时被删除的神经元保持原样, 而没有被删除的神经元已经有所更新)
 - 从隐藏层神经元中随机选择一个一半大小的子集临时删除掉(备份被删除神经元的参数)。
 - 对一小批训练样本, 先前向传播然后反向传播损失并更新参数(w, b) (没有被删除的那一部分参数得到更新, 删除的神经元参数保持被删除前的结果)。

缓解梯度消失:ReLU

当 z 是负数的时候, 梯度为0, 神经元停止学习(类似于 dropout作用, 减少过拟合);当 z 大于0时, 梯度为1, 可以缓解下溢问题

SVM-支持向量机

线形可分-SVM

约束优化问题

1. 目标函数 $\min f(x)$
2. 变量
3. 约束条件

$$\text{s.t. } \begin{aligned} g_j(x) &= 0, \quad j = 1, 2, \dots, n \\ h_i(x) &\leq 0, \quad i = 1, 2, \dots, m \end{aligned}$$

求解方法---拉格朗日乘数法

$$L(x, \lambda, \alpha) = f(x) + \sum_j a_j g_j(x) + \sum_i \lambda_i h_i(x)$$

$$\frac{\partial L}{\partial x} = \nabla f(x^*) + \sum_i a_j \nabla g_j(x^*) + \sum_i \lambda_i \nabla h_i(x^*) = 0$$

$$\begin{aligned}\nabla_{\mathbf{x}} L &= \frac{\partial L}{\partial \mathbf{x}} = \nabla f + \lambda \nabla g = \mathbf{0} \\ \nabla_{\lambda} L &= \frac{\partial L}{\partial \lambda} = g(\mathbf{x}) = 0\end{aligned}$$

计算 L 对 x 与 λ 的偏导数并设为零，可得最优解的必要条件

如何理解KKT?

Karush-Kuhn-Tucker (KKT) 条件是非线性规划最佳解的必要条件。KKT条件将Lagrange乘数法所处理涉及等式的约束优化问题推广至不等式

$$\begin{aligned}\nabla f(x^*) + \sum_j a_j \nabla g_j(x^*) + \sum_i \lambda_i \nabla h_i(x^*) &= 0 \\ g_j(x^*) &= 0 \\ h_i(x^*) &\leq 0, \lambda_i \geq 0, \lambda_i h_i(x^*) = 0\end{aligned}$$

对偶问题

$$\max_{\alpha_i \geq 0} \min_w L(w, \alpha)$$

$$f_0(w) = \max_{\alpha_i \geq 0} L(w, \alpha)$$

$$f_0(w) > L(w, \alpha)$$

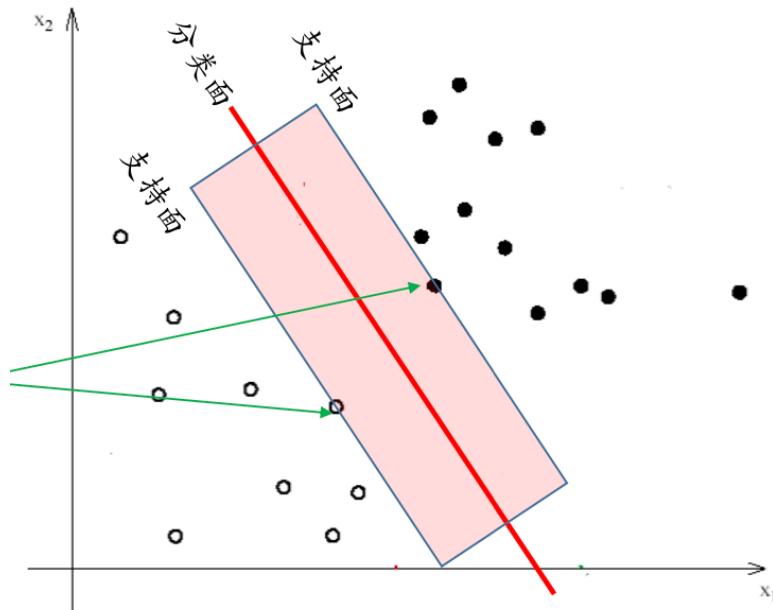
简单的例子

$$\begin{aligned}\min_u u^2 \\ \text{s.t. } u \geq b\end{aligned}$$

使用拉格朗日乘数法将其转化为

$$\begin{aligned}L &= u^2 + \lambda(u - b) \\ \frac{\partial L}{\partial u} &= 2u + \lambda \\ u - b &= 0 \\ u &= b \\ \lambda &= 2b\end{aligned}$$

Margin模型



分类面:

$$w^T x + b = 0$$

+1 支持面:

$$w^T x + b = 1$$

-1 支持面:

$$w^T x + b = -1$$

向量 w 与支持面、分类面正交

$$\left. \begin{array}{l} w^T x_1 + b = 1 \\ w^T x_2 + b = -1 \end{array} \right\} \Rightarrow w^T (x_1 - x_2) = 0$$

使用 w 和 b 对 M 建模

$$\left. \begin{array}{l} w^T x^+ + b = +1 \\ w^T x^- + b = -1 \end{array} \right\} \Rightarrow w^T (x^+ - x^-) = 2$$

得到两个支撑面最大间隔

$$\text{margin } M = \|x^+ - x^-\| = \frac{2}{\|w\|}$$

分类模型

目标函数: 间隔最大 (二次函数)

$$\begin{aligned} \max \left(\frac{2}{\|w\|} \right) &\Leftrightarrow \min (\|w\|^2) \\ \min_{w,b} w^T w / 2 \end{aligned}$$

约束:线形约束

$$\begin{cases} \mathbf{w}^T \mathbf{x}_i + b \geq 1 & y_i = 1 \\ \mathbf{w}^T \mathbf{x}_i + b \leq -1 & y_i = -1 \end{cases}$$

约束可合并为:

$$y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1$$

SVM--进阶

线形不可分SVM-软-SVM

新的优化问题

$$\min_{w,b} w^T w / 2 + C \sum_{i=1}^n \epsilon_i$$

约束:

$$\begin{aligned} y_i (\mathbf{w}^T \mathbf{x}_i + b) &\geq 1 - \epsilon_i \\ \epsilon_i &\geq 0 \end{aligned}$$

软-SVM对偶问题

$$\begin{aligned} \max_{\alpha} \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \\ \sum_i \alpha_i y_i = 0 \\ C \geq \alpha_i \geq 0, \forall i \end{aligned}$$

用拉格朗日乘数法转华为无约束问题:

$$\begin{aligned} L &= \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n \epsilon_i - \sum_{i=1}^n \alpha_i (y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1 + \epsilon_i) - \sum_{i=1}^n \mu_i \epsilon_i \\ \frac{\partial L}{\partial w} &= 0 \\ \frac{\partial L}{\partial \alpha} &= 0 \\ \frac{\partial L}{\partial \mu} &= 0 \end{aligned}$$

支持向量有两类:

1. 支持面上的点
2. 违背硬约束样本点

非线形-核SVM

模型：

1. 利用非线性映射把原始数据映射到高维空间中 $\phi(x)$
2. 目标函数

$$\begin{aligned} & \min w^T w / 2 \\ & \text{s.t. } y_i (w^T \phi(x_i) + b) \geq 1 \end{aligned}$$

多分类SVM

1. 一对多one-versus-rest

一种正样本，多种负样本

会出现数据不平衡，分类面偏置

$$\hat{y} \leftarrow \operatorname{argmax}_k w_k x + b_k$$

改进：期望正类和负类之间的错误达到平衡

$$\begin{aligned} & \min w^T w / 2 + C \left(\frac{N_+}{N_+} \sum_{i:y_i=+1} \epsilon_i + \frac{N_-}{N_-} \sum_{i:y_i=-1} \epsilon_i \right) N = N_+ + N_- \\ & \text{s.t. } y_i (w^T x_i + b) \geq 1 - \epsilon_i \\ & \epsilon_i \geq 0 \end{aligned}$$

2. 多个1V1 one-versus-one 训练 $\frac{m(m-1)}{2}$ 个分类器

样本量较少，分类器数量更多，测试成本高

SVR 支持向量回归

结论：

利用KKT条件：

$$\begin{cases} \alpha_i (y_i - f(x_i) - \epsilon - \xi_i) = 0 \\ \hat{\alpha}_i (f(x_i) - y_i - \epsilon - \hat{\xi}_i) = 0 \\ \xi_i (C - \alpha_i) = 0 \\ \hat{\xi}_i (C - \hat{\alpha}_i) = 0 \end{cases}$$

1. 当 $0 < \alpha_i < C$, x_i 落在间隔带上边界
2. 当 $\alpha_i = C$ 时, x_i 落在间隔带上边界外侧
3. 当 $0 < \alpha_i < C$, x_i 落在间隔带下边界
4. 当 $\alpha_i = C$ 时, x_i 落在间隔带下边界外侧
5. 当 $\alpha_i = \hat{\alpha}_i = 0$ 时, 点落在间隔带内侧

特征选择和稀疏学习

特征选择

过滤式选择

(Filter method)

单变量(Univariate)过滤方法:Signal-to-noise ratio (S2N)

$$S2\ N = \frac{|\mu_+ - \mu_-|}{\sigma_+ + \sigma_-}$$

多变量(Multivariate)过滤方法:Relief

给定训练集 $x_1, y_1, \dots, x_n, y_n$,

1、对每个样本 x_i , 在同类样本中找最近邻 $x_{i,hit}$;在异类样本中寻找最近邻 $x_{i,miss}$

2、计算对应于属性 j 的统计量

$$\delta_j = \sum_i -\text{diff}\left(x_i^j, x_{i,hit}^j\right)^2 + \text{diff}\left(x_i^j, x_{i,miss}^j\right)^2$$

3、若 δ_j 大于指定阈值 τ , 选择属性 j ;或者指定一个 k 值, 选择统计量最大的前 k 个特征

包裹式选择

(Wrapper method)

将所有属性作为一个集合, 每次从中选出部分作为训练特征。

NP难问题

寻找最优子集

验证集: 选超参数

嵌入式选择--正则化

(Embedded method)

L1正则化

$$E = \frac{1}{2n} \sum_x \|\mathbf{y}^x - \mathbf{h}^{x,L}\|^2 + \frac{\eta}{2n} \sum_l \|w^l\|_1$$

L2正则化

$$E = \frac{1}{2n} \sum_x \|y^x - h^{x,L}\|^2 + \frac{\eta}{2n} \sum_l \|w^l\|_2^2$$

混合正则化

$$E = \frac{1}{2n} \sum_x \|y^x - h^{x,L}\|^2 + \frac{\beta}{2n} \sum_l \|W^l\|_1 + \frac{\eta}{2n} \sum_l \|W^l\|_2^2$$

对偏置b不进行正则化，只对权重w进行正则化，假设 $\sum_{i=1}^n x_i = 0$,

$$\beta_0 = \frac{1}{n} \sum_{i=1}^n y_i$$

$$\min_{\beta, \beta_0} \sum_{i=1}^n (y_i - \beta^T x_i - \beta_0)^2 + \lambda \sum_{j=1}^p |\beta_j|^q$$

L2-正则化

$q=2$, 使用L2范数正则化称为ridge regression, 岭回归

$$\begin{aligned} & \min_{\beta, \beta_0} \sum_{i=1}^n (y_i - \beta^T x_i - \beta_0)^2 + \lambda \sum_{j=1}^p |\beta_j|^q \text{化为有约束形式: } (\text{why? 为什么要这样化}) \\ & \min_{\beta, \beta_0} \sum_{i=1}^n (y_i - \beta^T x_i - \beta_0)^2 \\ & \text{s. t. } \|\beta\|^2 \leq t \end{aligned}$$

求解方法：化为矩阵形式，利用正规方程法对其进行求解

$$L = \|Y - X\beta\|^2 + \lambda \|\beta\|^2$$

对权重参数求偏导，二范数的矩阵表示

$$\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2} = \sqrt{\mathbf{X}^T \mathbf{X}}$$

$$\frac{\partial L}{\partial \beta} = 2X^T(X\beta - Y) + 2\lambda\beta = 0$$

$$\Rightarrow X^T X \beta - X^T Y + \lambda \beta = 0$$

$$\Rightarrow (X^T X + \lambda I) \beta = X^T Y$$

求解得到：

$$\Rightarrow \beta = (X^T X + \lambda I)^{-1} X^T Y$$

(*) SVD 奇异值分解

用SVD解释岭回归: $X = UDV^T$, U 为 $n \times p$, V 为 $p \times p$ 正交矩阵, D 为对角阵, 满足 $d_1 \geq d_2 \geq \dots \geq d_p \geq 0$

$$\begin{aligned}
\mathbf{X}\hat{\beta}^{\text{ls}} &= \mathbf{X}(\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} \\
&= \mathbf{U} \mathbf{U}^T \mathbf{y} \\
\mathbf{X}\hat{\beta}^{\text{ridge}} &= \mathbf{X}(\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y} \\
&= \mathbf{U} \mathbf{D} (\mathbf{D}^2 + \lambda \mathbf{I})^{-1} \mathbf{D} \mathbf{U}^T \mathbf{y} \\
&= \sum_{j=1}^p \mathbf{u}_j \frac{d_j^2}{d_j^2 + \lambda} \mathbf{u}_j^T \mathbf{y}
\end{aligned}$$

L1-正则化

Least Absolute Shrinkage and Selection Operator, Lasso回归

$\mathbf{q}=1$, w 变成0, 自动放弃特征, 起到特征选择, 防止过拟合的方法, 可以使得特征矩阵稀疏。

$$\begin{aligned}
&\min_{\boldsymbol{\beta}, \beta_0} \sum_{i=1}^n (y_i - \boldsymbol{\beta}^T \mathbf{x}_i - \beta_0)^2 \\
&\text{s.t. } \|\boldsymbol{\beta}\|_1 \leq t
\end{aligned}$$

等价拉格朗日表达形式, 用拉格朗日乘数法, 化有条件极值为无条件极值:

$$\min_{\boldsymbol{\beta}, \beta_0} \sum_{i=1}^n (y_i - \boldsymbol{\beta}^T \mathbf{x}_i - \beta_0)^2 + \lambda \|\boldsymbol{\beta}\|_1$$

L1约束使得解关于 y 非线性, 而且不能像岭回归那样可以得到封闭解。

1. 闭式解需要满足正交性 $X^T X$

2. 一般方法Lasso回归求解(一般情形): 坐标下降法(CoordinateDescent)

相当于每次迭代都只是更新x的一个维度, 即把该维度当做变量, 剩下的n-1个维度当作常量, 通过最小化f(x)来找到该维度对应的新的值。坐标下降法就是通过迭代地构造序列 x^0, x^1, x^2, \dots 来求解问题, 即最终点收敛到期望的局部极小值点

Lasso回归

$$\min_{\boldsymbol{\beta}, \beta_0} \sum_{i=1}^n (y_i - \boldsymbol{\beta}^T \mathbf{x}_i - \beta_0)^2 + \lambda \|\boldsymbol{\beta}\|_1$$

每次针对一个属性进行更新, 为什么 β_0 下面的推到没了

$$L = \sum_{i=1}^n \left(y_i - \sum_{j=1}^p x_{i,j} \beta_j \right)^2 + \lambda \sum_{j=1}^p |\beta_j|$$

完全平方式展开, L对权重参数求导

$$\frac{\partial L}{\partial \beta_k} = 2a_k + 2b_k \beta_k + \lambda \text{sign}(\beta_k), \text{ where } a_k = \sum_{i=1}^n x_{i,k} \left(\sum_{j \neq k} x_{i,j} \beta_j - y_i \right), b_k = \sum_{i=1}^n x_{i,k}^2$$

利用 $\frac{\partial L}{\partial \beta_k} = 0$, 得到

$$\beta_k = \begin{cases} -\frac{1}{b_k} \left(a_k - \frac{\lambda}{2} \right), & a_k > \frac{\lambda}{2} \\ 0, & -\frac{\lambda}{2} < a_k < \frac{\lambda}{2} \\ -\frac{1}{b_k} \left(a_k + \frac{\lambda}{2} \right), & a_k < -\frac{\lambda}{2} \end{cases}$$

稀疏表示字典学习

字典学习：给定数据集 x_1, x_2, \dots, x_n

$$\min_{B, \alpha_i} \sum_{i=1}^n \left(\|x_i - B\alpha_i\|^2 + \lambda \|\alpha_i\|_1 \right)$$

- 其中 $B \in R^{p \times k}$ 为字典矩阵， k 为字典的词汇量(通常由用户指定)， $\alpha_i \in R^k$ 是样本 $x_i \in R^p$ 的稀疏表示。

求解方法：交替优化（控制变量法）

1. 固定 B ，优化 α ---Lasso 回归问题-----坐标下降法求解
2. 固定 α ，优化 B -----线形优化-----正规方程法求解

集成学习

思想：民主决策，少数服从多数

好的集成：个体要有差异，个体精度不能太低：好而不同

集成的有效性：

$$H(\mathbf{x}) = \text{sign} \left(\sum_{i=1}^T h_i(\mathbf{x}) \right)$$

分类错误率随着 T 的增大呈指数下降

$$\begin{aligned} P(H(\mathbf{x}) \neq f(\mathbf{x})) &= \sum_{k=0}^{\lfloor T/2 \rfloor} \binom{T}{k} (1-\epsilon)^k \epsilon^{T-k} \\ &\leq \exp \left(-\frac{1}{2} T (1-2\epsilon)^2 \right) \end{aligned}$$

重采样

自适应权重重置和组合

1. 随机采样：bagging
2. 带权采样：boosting

串行式

特点：强依赖

代表性：boosting---权值逐渐变大

Boosting

算法步骤

1. 给所有训练样例赋予相同的权重
2. 训练第一个基本分类器
3. 对分类错误的测试样例提高其权重
4. 用调整过的带权训练集训练第二个基本分类器
5. 重复上述过程
6. 对所有的基分类器进行加权组合

$$H_M(x) = \text{sign} \left(\sum_{m=1}^M \alpha_m h_m(x) \right)$$

h_m 是基分类器， w_n^m 表示样本权重，n为样本数量，m为个体分类器数

- 对于分类错误的样本--提高其权重

Ada boosting

考试会考

模型

二分类问题：

N个训练样本： $x_n (n = 1, \dots, N)$

每个训练样本的标签为 $y_n \in \{-1, +1\}$ ， $h_m(x) \in \{-1, +1\}$

算法步骤：

1. 初始化每个训练样本的权重 $w_n^{(1)} = 1/N$ ，均匀分配
2. 第一个基分类器开始训练，通过最小化误差函数 $\min L_m = \sum_{n=1}^N w_n^{(m)} I(h_m(x_n) \neq y_n)$ ，I为指示函数，训练分类器 h_m 的参数
3. 计算加权的分类错误率 $\epsilon_m = \frac{\sum_{n=1}^N w_n^{(m)} I(h_m(x_n) \neq y_n)}{\sum_{n=1}^N w_n^{(m)}}$ ，错误率逐渐降低
4. 计算分类器权重 $\alpha_m = \ln \frac{1-\epsilon_m}{\epsilon_m}$ ，分类权重逐渐增大
5. 更新样本权重 $w_n^{(m+1)} = w_n^{(m)} \exp(\alpha_m I(h_m(x_n) \neq y_n))$
6. 使用 $H_M(x) = \text{sign}(\sum_{m=1}^M \alpha_m h_m(x))$

Error model

并行式

不存在强依赖

决策树

贝叶斯分类器

x样本，c为标签

判别式模型：

生成式模型：

GAN：对抗生成网络

DCGAN：

模型

基于贝叶斯公式的后验概率：

$$\begin{aligned} P(C = c_i \mid \mathbf{X} = \mathbf{x}) &= \frac{P(\mathbf{X} = \mathbf{x} \mid C = c_i)P(C = c_i)}{P(\mathbf{X} = \mathbf{x})} \\ &\propto P(\mathbf{X} = \mathbf{x} \mid C = c_i)P(C = c_i) \\ &\text{for } i = 1, 2, \dots, L \end{aligned}$$

贝叶斯分类

$$\underset{c_j \in C}{\operatorname{argmax}} P(x_1, x_2, \dots, x_p \mid c_j)P(c_j)$$

朴素贝叶斯分类

朴素条件：对已知类别，假设所有属性互相对立

$$\begin{aligned} P(X_1, X_2, \dots, X_p \mid C) &= P(X_1 \mid X_2, \dots, X_p, C)P(X_2, \dots, X_p \mid C) \\ &= P(X_1 \mid C)P(X_2, \dots, X_p \mid C) \\ &= P(X_1 \mid C)P(X_2 \mid C) \cdots P(X_p \mid C) \end{aligned}$$

朴素贝叶斯分类器模型（联合转化为连乘）：

$$\arg \max_{c_j \in C} P(c_j) \prod_{i=1}^P P(x_i \mid c_j)$$

需要估计：

- 先验 $P(C = c_j)$
- 每个属性的条件概率 $P(x_i | c_j)$

避免0概率问题

若某个属性值在训练集中没有与某个类同时出现过，则基于频率的概率估计将为零。

修正：在分母上+属性取值数目，分子加上类的个数

$$\hat{P}(X_i = x | C = c_j) = \frac{N(X_i = x | C = c_j) + 1}{N(C = c_j) + |X_i|}$$

高斯朴素贝叶斯分类器

高斯分布

$$\mathcal{N}(\mathbf{x} | \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^p/2} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \exp \left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right\}$$

一维 Gaussian：

均值和方差的极大似然估计值分别是样本的均值及样本的方差

$$\mu = \frac{1}{n} \sum_{i=1}^n x_i, \quad \sigma^2 = \frac{1}{n} \sum_{i=1}^n (x_i - \mu)^2$$

多维 Gaussian：

$$\mu = \frac{1}{n} \sum_{i=1}^n x_i, \quad \Sigma = \frac{1}{n} \sum_{i=1}^n (x_i - \mu)(x_i - \mu)^T$$

高斯朴素贝叶斯分类器

$$\begin{aligned} \underset{C}{\operatorname{argmax}} P(C | X) &= \underset{C}{\operatorname{argmax}} P(X, C) = \underset{C}{\operatorname{argmax}} P(X | C)P(C) \\ \hat{P}(x_i | C = c_j) &= \frac{1}{\sqrt{2\pi}\sigma_{ij}} \exp \left(-\frac{(x_i - \mu_{ij})^2}{2\sigma_{ij}^2} \right) \end{aligned}$$

x 样本，c 为标签

由 X 得到高斯分布的均值和方差，后代入高斯分布

$$P(\text{密度} | \text{好瓜}) = P(\text{密度} | \text{好瓜}) * P(\text{含糖率} | \text{好瓜})$$

朴素：用所有样本

非朴素：联合等于两个乘积

共有 $L \times (p + p \times (p + 1)/2)$ 个参数

$\sum : p \times p$

$\mu : p$

朴素高斯必要性：估计的参数量减少

逻辑回归决策面： $\theta^T X = 0$

高斯贝叶斯决策面：

分到l类和k类的概率相等

$$\log P(c_k | x) - \log P(c_l | x) = 0$$

用贝叶斯公式展开

$$\log \frac{P(C_k | X)}{P(C_l | X)} = \log \frac{P(X | C_k)}{P(X | C_l)} + \log \frac{P(C_k)}{P(C_l)}$$

其中

$$\begin{aligned} \log P(x | c_k) &= -\frac{1}{2}(x - \mu_k)^T \sum_k^{-1} (x - \mu_k) - \log |\Sigma_k|^{\frac{1}{2}} \\ \log \frac{P(c_k | x)}{P(c_l | x)} &= \frac{1}{2}(x - \mu_l)^T \Sigma_l^{-1} (x - \mu_l) - \frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k) + \log \frac{|\Sigma_l|^{\frac{1}{2}}}{|\Sigma_k|^{\frac{1}{2}}} + \log \frac{\pi_k}{\pi_l} \end{aligned}$$

假设每一类的协方差矩阵均相同，

$$\sum_k = \sum, \forall k$$

$$\Sigma_j = \begin{bmatrix} \sigma_{1j}^2 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_{pj}^2 \end{bmatrix}$$

决策函数可以从x的二次转为一次函数

$$\begin{aligned} &= \log \frac{\pi_k}{\pi_l} + \frac{1}{2}(x - \mu_l)^T \Sigma^{-1} (x - \mu_l) - \frac{1}{2}(x - \mu_k)^T \Sigma^{-1} (x - \mu_k) \\ &= \frac{\log \frac{\pi_k}{\pi_l} + \frac{1}{2}\mu_l^T \Sigma^{-1} \mu_l - \frac{1}{2}\mu_k^T \Sigma^{-1} \mu_k}{b} + x^T \frac{\Sigma^{-1}(\mu_k - \mu_l)}{a} \end{aligned}$$

决策边界：

$$x^T a + b = 0$$

要估计的参数个数： $L \times (p + p \times (p + 1)/2)$

若 $a_0 + \sum_{i=1}^p a_i x_j > 0$, 将x的标签置为c1, 否则将其标签置为c2

LDA决策面

通过假设每一类具有的相同协方差矩阵，得到一种经典的线性学习方法：线性判别分析（Linear Discriminant Analysis, LDA）

线形决策面

$$\begin{aligned} &= \log \frac{\pi_k}{\pi_l} + \frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_l)^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}_l) - \frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}_k) \\ &= \log \frac{\pi_k}{\pi_l} + \frac{1}{2} \boldsymbol{\mu}_l^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_l - \frac{1}{2} \boldsymbol{\mu}_k^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_k + \mathbf{x}^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_k - \boldsymbol{\mu}_l) \end{aligned}$$

参数估计

先验：

$$\hat{P}(C = C_j) = \frac{N(C = c_j)}{N}$$

均值：第j个高斯分布的均值

$$\boldsymbol{\mu}_j = \frac{1}{N(C = c_j)} \sum_{X \in c_j} X$$

方差：

$$\boldsymbol{\Sigma} = \frac{1}{N} \sum_{c_j \in C} \sum_{X \in c_j} (X - \boldsymbol{\mu}_j)(X - \boldsymbol{\mu}_j)^T$$

求决策边界

已知相应的参数：

$$\begin{aligned} * \pi_1 = \pi_2 &= 0.5 \\ * \boldsymbol{\mu}_1 = (0, 0)^T, \boldsymbol{\mu}_2 &= (2, -2)^T \\ * \boldsymbol{\Sigma} &= \begin{pmatrix} 1.0 & 0.0 \\ 0.0 & 0.5625 \end{pmatrix} \end{aligned}$$

则代入上述公式求得决策边界：

$$* \text{Decision boundary: } 5.56 - 2.00x_1 + 3.56x_2 = 0.0$$

Loss function

逻辑回归	LDA
无 x 的分布假设：	假设 x 服从高斯分布

逻辑回归-判别式

$$L = P(c \mid x; \theta) = (f_\theta(x))^c \left(1 - \frac{f_\theta}{\theta}(x)\right)^{1-c}$$

$$\theta^{(0)} \quad \theta^{(1)} := \theta^{(0)} + \alpha^+ \left. \frac{\partial^T - x}{\partial \theta} \right|_{\theta^{(0)}}$$

LDA--生成式

$$P(x \mid c_k) \sim M(\mu_k, \Sigma_k)$$

$$= \frac{1}{(2\pi)^{\frac{p}{2}} |\Sigma_k|^{\frac{1}{2}}} \exp -\frac{1}{2} (\mathbf{x} - \mu_k)^T \Sigma_k^{-1} (\mathbf{x} - \mu_k)$$

其中 $\Sigma_k = \Sigma, \forall k.$ $\mu_k = \frac{1}{n} \sum_i x_i$

决策边界 线性 $\sum_k = \frac{1}{h} \bar{2}$

高斯朴素贝叶斯决策面

K-NN分类算法

K-近邻 (K-nearest neighbors)

对一个未知样本进行分类：

1. 计算未知样本与标记样本的距离
2. 确定 k 个近邻
3. 使用近邻样本的标签确定目标的标签：

例如，将其划分到 k 个样本中 出现最频繁的类

没有模型，没有error model

KNN回归

马尔可夫链

Markov模型 (链)

贝叶斯网

有向无环图

特征变量之间的



一步转移

School of Software Engineering

□ $A_{ij} = P(X_{t+1} = j | X_t = i)$ 指定了从状态 i 到状态 j 的一步转移概率。

$$p(X_t=a), \dots, p(X_t=s)$$

□ 给定 t 时刻的状态分布向量 $v_t = (P(X_t = 1), \dots, P(X_t = K))$, 则

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_K \end{bmatrix}$$

$$\begin{aligned} v_{t+1}(j) &= P(X_{t+1} = j) \\ &= \sum_{i=1}^K P(X_t = i) P(X_{t+1} = j | X_t = i) = \sum_{i=1}^K v_t(i) A_{ij} = v_t A(:, j) \end{aligned}$$

□ 写成矩阵形式 $\underbrace{v_{t+1}}_{\text{Step}} = \underbrace{v_t A}_{\sum_i v_{t+1}(i)}$

18

1步转移

$$\begin{aligned} v_{t+1}(j) &= P(X_{t+1} = j) \\ &= \sum_{i=1}^K P(X_t = i) P(X_{t+1} = j | X_t = i) = \sum_{i=1}^K v_t(i) A_{ij} = v_t A(:, j) \end{aligned}$$

n步转移

$$P(X_1 = i_1, \dots, X_T = i_T) = \pi_{i_1} \prod_{t=2}^T A_{i_{t-1} i_t}$$

平稳分布

平稳分布：对于一个Markov链，给定初始状态分布 $v_1 = \pi = P(X_1 = 1, \dots, P(X_1 = K)$ ，利用状态转移公式 $v_{t+1} = v_t A$ ，经过一定次数的迭代之后，若能达到 $v = v A$ 则称Markov链达到了平稳分布。一旦进入平稳分布，在之后的任意时刻的概率分布永远为 v ，马尔可夫链处于稳定状态 稳定状态： v 经过 A 转移后仍然是 v

应用

1. 句子补全
2. 网页排序：page-rank

damping look 解决断链问题

page-rank

PageRank认为某个网页的重要性有两个因素决定：指向网页的链接数量以及输出网页的链接数量。

超链接的个数和质量

数量假设和质量假设

若网页 j 到网页 i 有边，则令 $L_{ij} = 1$ ，否则 $L_{ij} = 0$ 。因此， j 的输出链接为，出链的个数（有向图出度）

$$c_j = \sum_{i=1}^N L_{ij}$$

指向网页的链接越多权重越大，而输出网页的链接越多权重越小

$$p_i = (1 - d) + d \sum_{j=1}^N \left(\frac{L_{ij}}{c_j} \right) p_j$$

p_i 为网页重要性， c_j 表示网页 j 对网页 i 的重要性的程度

参数估计计算

MLE最大似然估计 for Markov chain

HMM隐马尔可夫链

HMM三个问题

1. 评估问题：概率计算问题

估计模型下观测序列出现的概率

2. 解码问题：状态预测问题

给定模型参数和一个观测序列，推断隐状态序列

3. 学习问题：参数估计问题

给定多个观测数据 Y ，估计一组参数

□ HMM: $\lambda = \{A, B, \pi\}$ 。令 $Q = \{q_1, \dots, q_T\}$ 和 $Y = \{y_1, \dots, y_T\}$ 。HMM的三个基本问题如下:

1、概率计算问题: 给定观测序列 Y 和模型参数 λ , 估计模型下观测序列 Y 出现的概率 $P(Y|\lambda)$

2、状态预测问题: 给定模型参数 λ 和一个观测序列 $Y = \{y_1, \dots, y_T\}$, 推断隐状态序列: $\arg \max_Q P(Q|Y, \lambda)$

3、学习问题: 给定多个观测数据 Y , 估计一组参数 λ 使得 $P(Y|\lambda)$ 最大, 即极大似然估计参数:

常规方法: 遍历--复杂度指数级

非监督学习

--压缩思想

1. 纵向结构--聚类
2. 横向结构--降维度

线形:

非线形:

聚类

clustering---簇内距小, 簇间距大

簇的定义

数据表示: 向量空间

相似性/距离: 欧氏距离

簇的个数: 数据驱动, 自己识别出来

聚类算法: 划分聚类算法, 层次聚类算法

算法的收敛性: 收敛速度

层次式聚类算法

1. 自顶向上: 聚合
2. 自顶向下: 分裂

★划分式聚类算法

1. K-means
2. GMM (高斯混合模型)

K-means

模型

算法步骤：

输入：数据N个样本，簇的个数指定为K

1. 初始化：随机选择K个数据点作为相应的簇中心{}

2. 迭代：

1. 对每一个样本进行归簇，距离哪个聚类中心最近，则讲其归为哪一族

$$x_j \in C_i \Leftrightarrow \min_{t=1, \dots, K} \{\|x_j - \mu_t\|\} = \|x_j - \mu_i\|$$

$$2. \text{重新计算每个簇的均值 (簇中心)} \quad \mu_i = \frac{1}{|C_i|} \sum_{x_j \in C_i} x_j$$

3. 终止迭代：簇中心不发生改变时

输出：

目标函数：簇内样本到簇中心的平方和距离最小

$$\operatorname{argmin}_{C, \mu} \sum_{i=1}^K \sum_{x_j \in C_i} \|x_j - \mu_i\|_2^2$$

非凸函数，NP-hard

解决之道：迭代优化（交替优化：固定一组变量值去优化另一组变量值）

• 初始化K个簇中心： $\mu = \{\mu_1, \mu_2, \dots, \mu_K\}$ • 迭代进行以下优化

• 更新簇成员：固定 μ ，优化 C • 更新簇中心：固定 C ，优化 μ

算法复杂度：

迭代次数：假设迭代 l 步算法收敛。因此总的计算复杂度

为 $O(l K n p)$

由于 K 和 l 通常都远远小于 n ，可认为是关于 n 的线性复杂度

初值对算法的影响：

通过启发式方法选择好的初值：例如要求种子点之间有较大的距离 □ 尝试多个初值，选择平方误差和最小的一组聚类结果

聚类数目K的影响：

手肘法：目标函数的值和 k 的关系图是一个手肘的形状，而这个肘部 对应的k值就是数据的最佳聚类数。k=2时，对应肘部，故选择 k值为2

局限性

不适合对形状不是超维椭圆体(或超维球体)的数据

K-means延伸

层次-kmeans

乘积量化：就是划分数据集，然后分别划分出聚类中心，然后向量乘积

128*n

n个样本训练256个cluster：聚类中心

划分样本集做完笛卡尔乘积之后的维数和用原始样本求中心的维数一样吗？

PCA

标准PCA

线形PCA

三种建模思想

PCA 求解角度

1. 最大投影方差
2. 最小投影距离
3. 奇异值分解(SVD)

最大投影方差

信息（方差）能尽可能大的保持

最小投影距离

投影数据与原数据的之间的最小平方距离尽可能小

目标函数：

$$\mathbf{w}_1 = \arg \max_{|\mathbf{w}|=1} \frac{1}{m} \sum_{i=1}^m \left\{ (\mathbf{w}^T \mathbf{x}_i)^2 \right\} \quad \text{Var}(X) = E \{ [X - E(X)]^2 \}$$

$$L = -w^\top Aw + \lambda(w^\top w - 1)$$

$$\frac{\partial L}{\partial w} = -2 \cdot Aw + 2\lambda w = 0 \Rightarrow Aw = \lambda w \Rightarrow w^\top Aw = \lambda$$

w1协方差矩阵的最大特征值对应的特征向量

$$\max_{W \in R^{p \times k}} \text{tr} \left(W^T \left(\frac{1}{m} XX^T \right) W \right), W^T W = I_k$$

PCA主方向 = 数据协方差矩阵的特征向量 • 更大特征值对应的特征向量更加重要
降维结果

$$Z = W^T X$$

重构结果

$$\hat{X} = WZ = WW^T X$$

目标:计算数据k个主方向

- 第一步:数据居中
- 第二步:计算居中数据的协方差矩阵
- 第三步:计算协方差矩阵最大k个特征值对应的特征向量, 组成矩阵
- 输出降维结果
- 问题:
 - 第k个主成分的方差是多少?
 - k选择多大

$$\begin{aligned} & w_k^\top \left(\frac{1}{M} \sum_{i=1}^M x_i x_i^T \right) w_k \\ &= w_k^\top \lambda_k w_k = \lambda_k w_k^\top w_k = \lambda_k \end{aligned}$$

K的选择

奇异值分解SVD

$$A = U\Sigma V^T$$

PCA应用-数据预处理

数据白化(Whitening)操作

使用PCA, 可以同时去除变量之间的线性关系以及对数据进行归一化:

- 假设数据的协方差矩阵为S

$$S = \frac{1}{m} \sum_{i=1}^m (x_i - \bar{x})(x_i - \bar{x})^T$$

- 利用 $W^T SW = \Lambda$ 定义一个变换

$$y_i = \Lambda^{-\frac{1}{2}} W^T (x_i - \bar{x})$$

则 y 的均值为0， 协方差为单位矩阵。

概率PCA

核PCA

步骤

1. 输入
2. 构造Gram矩阵
3. 对高维数据去中心化
4. 对K进行特征分解
5. 计算 x 的低维表示

LLE

Locally Linear embedding 局部线性嵌入

LLE关注于降维时保持样本局部的线性特征，由于LLE在降维时保持了样本的局部特征

1. 找最近邻：欧氏距离
2. 重构：重构系数之和=1

$$\varepsilon(W) = \sum_{i=1}^N \left\| x_i - \sum_{j=1}^k W_{ij} x_{ij} \right\|^2 = \sum_{i=1}^N \left\| \sum_{j=1}^k W_{ij} (x_i - x_{ij}) \right\|^2$$

W_{ij} 求解方法：拉格朗日乘数法

$$L(W_i) = W_i^T Z_i W_i + \lambda (W_i^T \mathbf{1}_{k \times 1} - 1)$$
$$2Z_i W_i + \lambda \mathbf{1}_{k \times 1} = 0, \text{ 即 } W_i = -\frac{\lambda}{2} Z_i^{-1} \mathbf{1}_{k \times 1}$$

3. 低维嵌入 