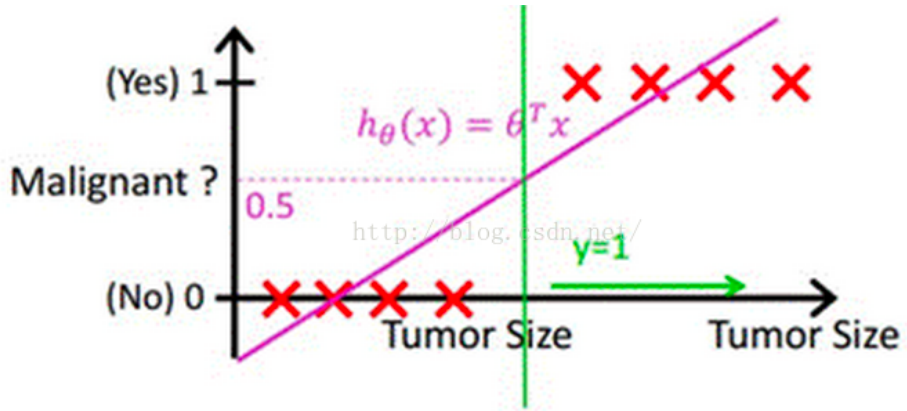
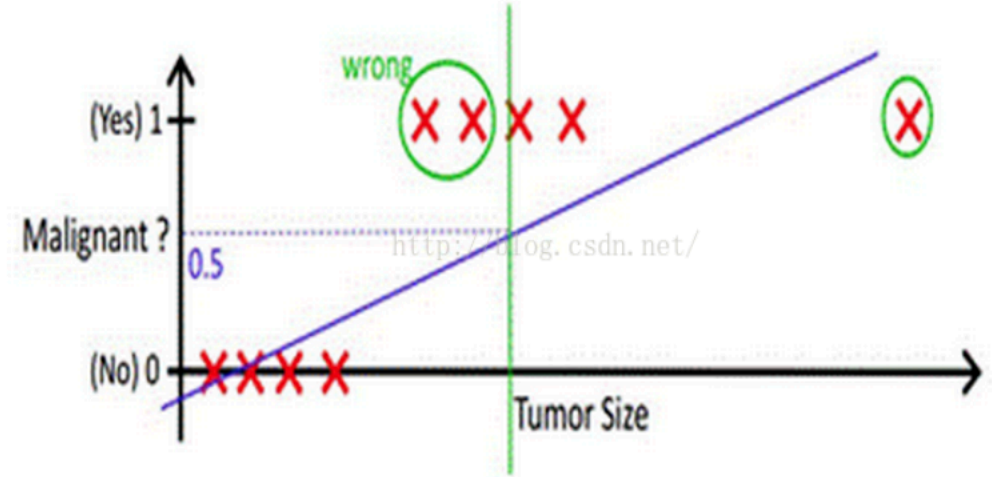
Logistic Regression

逻辑回归

线性回归对于分类的困扰



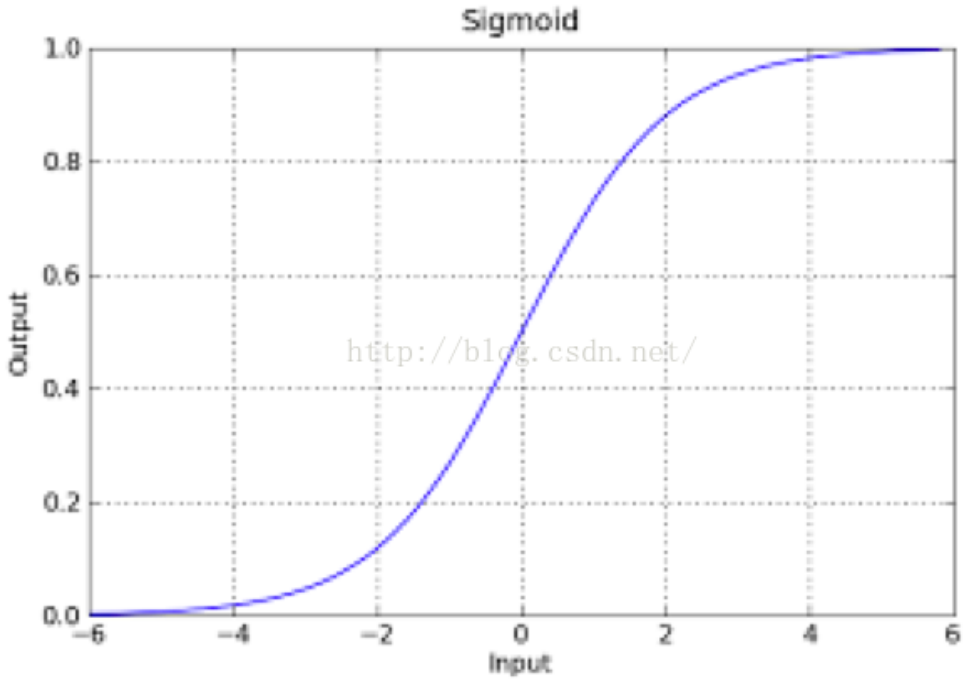


Sigmod函数

- 构造一个函数更好地解决二分类问题
  - 目标:输出一个 (0,1) 的实数代表概率

$$g(z) = \frac{1}{1 + e^{-z}}.$$

  - $Z = w^t X$ 线性回归



实质

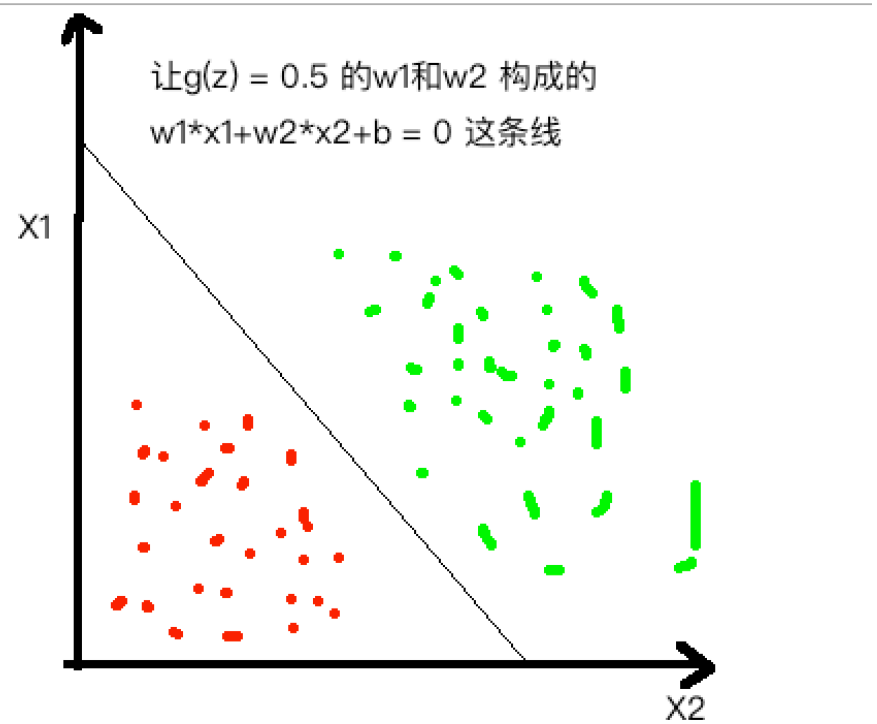
- 我们赋予 $g(z) = \frac{1}{1+e^{-z}}$. 函数输出的含义为 该条数据y=1的概率
- 即 g(z) <0.5 时 判断y=0
- g(z)>=0.5时 判断y=1

z的图像含义

- g(z) = 0.5 时 $w^t X = 0$ 二维平面下是一条直线

- 直线左侧的x通过计算会让
  - Gz<0.5

- 直线右侧的x通过计算会让
  - Gz>0.5



如何确定g(z)?

- 确定了一组w 就确定了 z

- 确定了z 就确定了 g(z) 的输出, 那么配合上已知的x 确定了w就能确定g(z)的最终输出值, 那么g(z) 也可以写成 g(w,x)

最大似然估计

- 根据若干已知的 X,y (训练集) 找到一组w 使得 x作为已知条件下 y 发生的概率最大

- 既然g(w,x)的输出含义为 P(y=1|w,x)
- 那么 P(y=0|w,x) = 1 – g(z)

- 那么就将g(w) 作为未知数, 将训练集上y^{hat}=y的概率计算出来

- 只要让我的g(w,x) 函数在训练集上预测正确的概率最大,我的g(w,x)就是好g(w,x)

w1	w2	w3	w4		
x1	x2	x3	x4	y	预测正确的概率
24	3	2	3	1	g(wx)
4	3	2	2	1	g(wx)
23	1	3	3	1	g(wx)
1	2	5	5	0	1-g(wx)
3	12	1	2	0	1-g(wx)

- P(正确) = $\begin{cases} g(w,xi) & \text{当 } yi=1 \text{ 时} \\ 1-g(w,xi) & \text{当 } yi=0 \text{ 时} \end{cases}$

- 对于每一条数据预测正确的概率
  - P(正确) = $(g(w,xi))^{y^i} * (1-g(w,xi))^{1-y^i}$

- 全部预测正确的概率= 每一条数据预测正确的概率相乘
- P(全部正确) = $\prod_{i=1}^{n} p_i$(正确)

- 记P(全部正确) 为 $L(\theta)$ 这里的θ为之前的w

$$L(\theta) = \prod_{i=1}^{m} (h_\theta(x^{(i)}))^{y^{(i)}} (1 - h_\theta(x^{(i)}))^{1-y^{(i)}}$$

- 现在想找到一组θ(就是前面说的w) 使得上面的概率函数有最大值
- 由于ln函数时单调递增的 ln(L(θ)) 最大时 L(θ) 也最大

$$l(\theta) = \log L(\theta)$$

$$= \sum_{i=1}^{m} \left(y^{(i)} \log h_\theta(x^{(i)}) + \left(1 - y^{(i)}\right) \log(1 - h_\theta(x^{(i)})) \right)$$

定义损失函数

- 使 $l(\theta)$ 最大的θ 生产出来的 g(θ,x) 全预测对的概率最大
- 损失函数: 某个函数结果越小 生成的模型越好
- 那么我们定义 - $l(\theta)$ 为逻辑回归的损失函数

$$J_{\log}(w) = \sum_{i=1}^{m} -y_i Log(p(x_i; w)) - (1 - y_i) Log(1 - p(x_i; w))$$

整理思路

- 损失函数最小 -> $l(\theta)$ 最大 -> 在训练集上全部正确的概率最大 -> 达成目标

- 问题转化为 找到一组使损失函数最小的w

$$J_{\log}(w) = \sum_{i=1}^{m} -y_i Log(p(x_i; w)) - (1 - y_i) Log(1 - p(x_i; w))$$

梯度下降

- 想要通过梯度下降优化 L(w) 到最小值需要几步?

  - 1. 随机产生w0
  - 2. wk+1 = wk + $\lambda * -\frac{\partial L(wk)}{\partial (wk)}$
  - 3. 迭代足够多轮 的 wk+1 就是能使L(w) 最小的w

说干就干

$$\theta_j := \theta_j - \alpha \frac{\delta}{\delta_{\theta_j}} J(\theta)$$

$$\frac{\delta}{\delta_{\theta_j}} J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} \left(y_i \frac{1}{h_\theta(x_i)} \frac{\delta}{\delta_{\theta_j}} h_\theta(x_i) - (1 - y_i) \frac{1}{1 - h_\theta(x_i)} \frac{\delta}{\delta_{\theta_j}} h_\theta(x_i) \right)$$

$$= -\frac{1}{m} \sum_{i=1}^{m} \left(y_i \frac{1}{g(\theta^T x_i)} - (1 - y_i) \frac{1}{1 - g(\theta^T x_i)} \right) \frac{\delta}{\delta_{\theta_j}} g(\theta^T x_i)$$

$$= -\frac{1}{m} \sum_{i=1}^{m} \left(y_i \frac{1}{g(\theta^T x_i)} - (1 - y_i) \frac{1}{1 - g(\theta^T x_i)} \right) g(\theta^T x_i)(1 - g(\theta^T x_i)) \frac{\delta}{\delta_{\theta_j}} \theta^T x_i$$

$$= -\frac{1}{m} \sum_{i=1}^{m} (y_i(1 - g(\theta^T x_i)) - (1 - y_i) g(\theta^T x_i)) x_i^j$$

$$= -\frac{1}{m} \sum_{i=1}^{m} (y_i - g(\theta^T x_i)) x_i^j$$

$$= \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i) x_i^j$$

结论

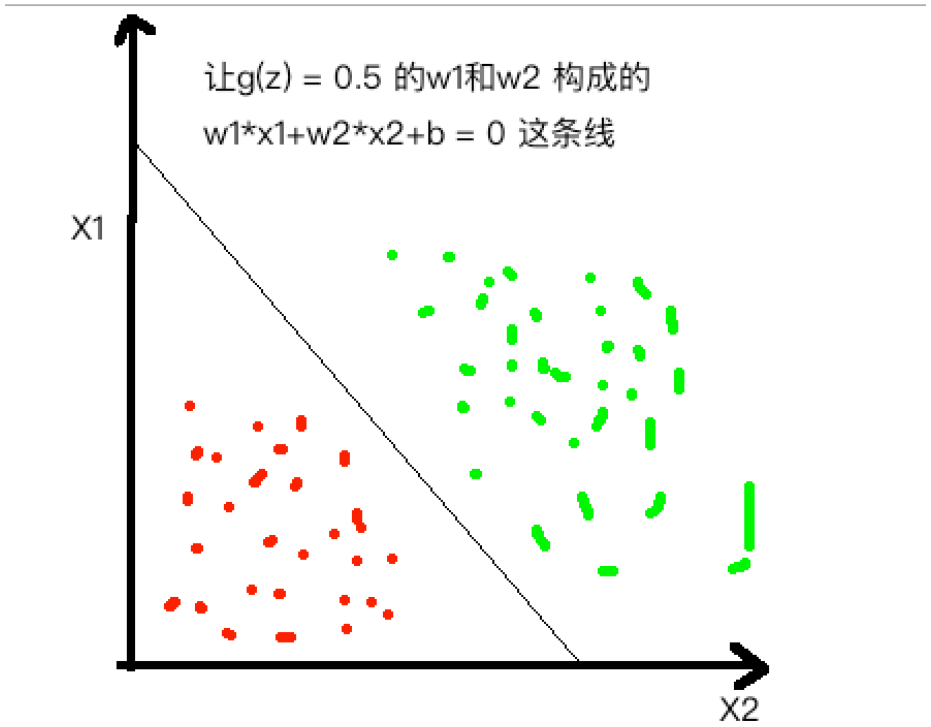
w1	w2	w3	w4		
x1	x2	x3	x4	y	预测正确的概率
24	3	2	3	1	g(wx)
4	3	2	2	1	g(wx)
23	1	3	3	1	g(wx)
1	2	5	5	0	1-g(wx)
3	12	1	2	0	1-g(wx)

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i) x_i^j - \frac{\lambda}{m} \theta_j$$

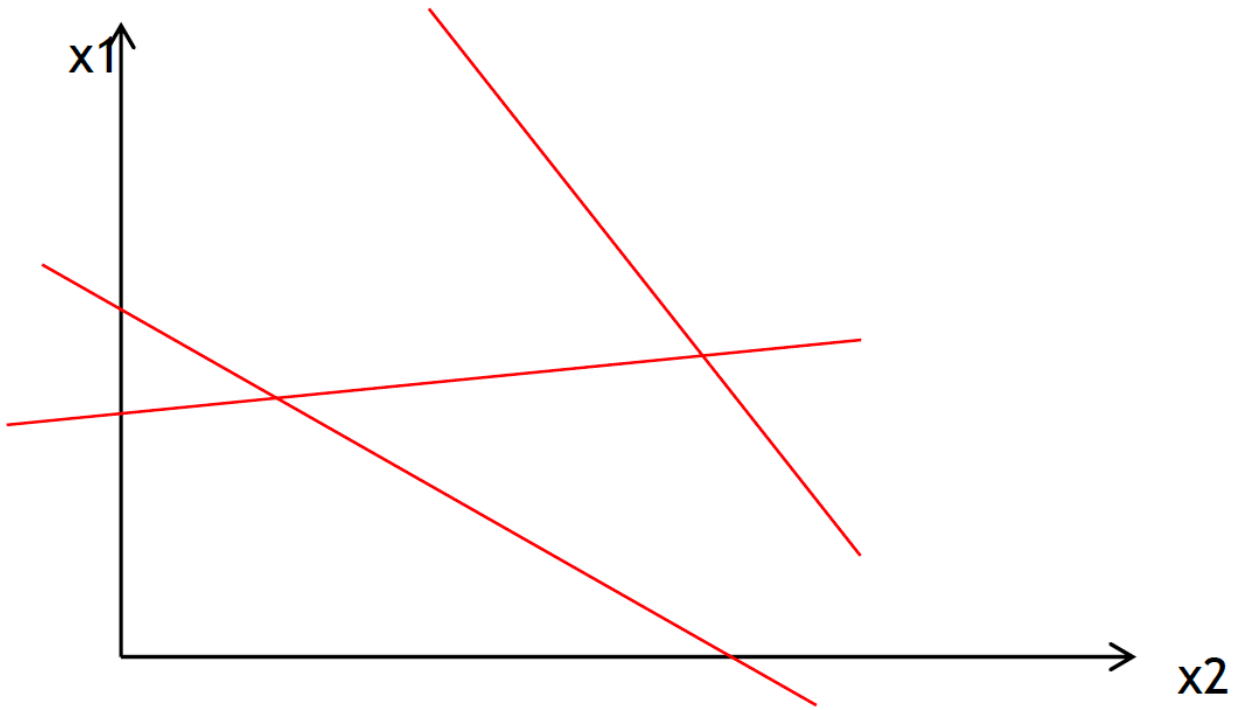
逻辑回归的优化

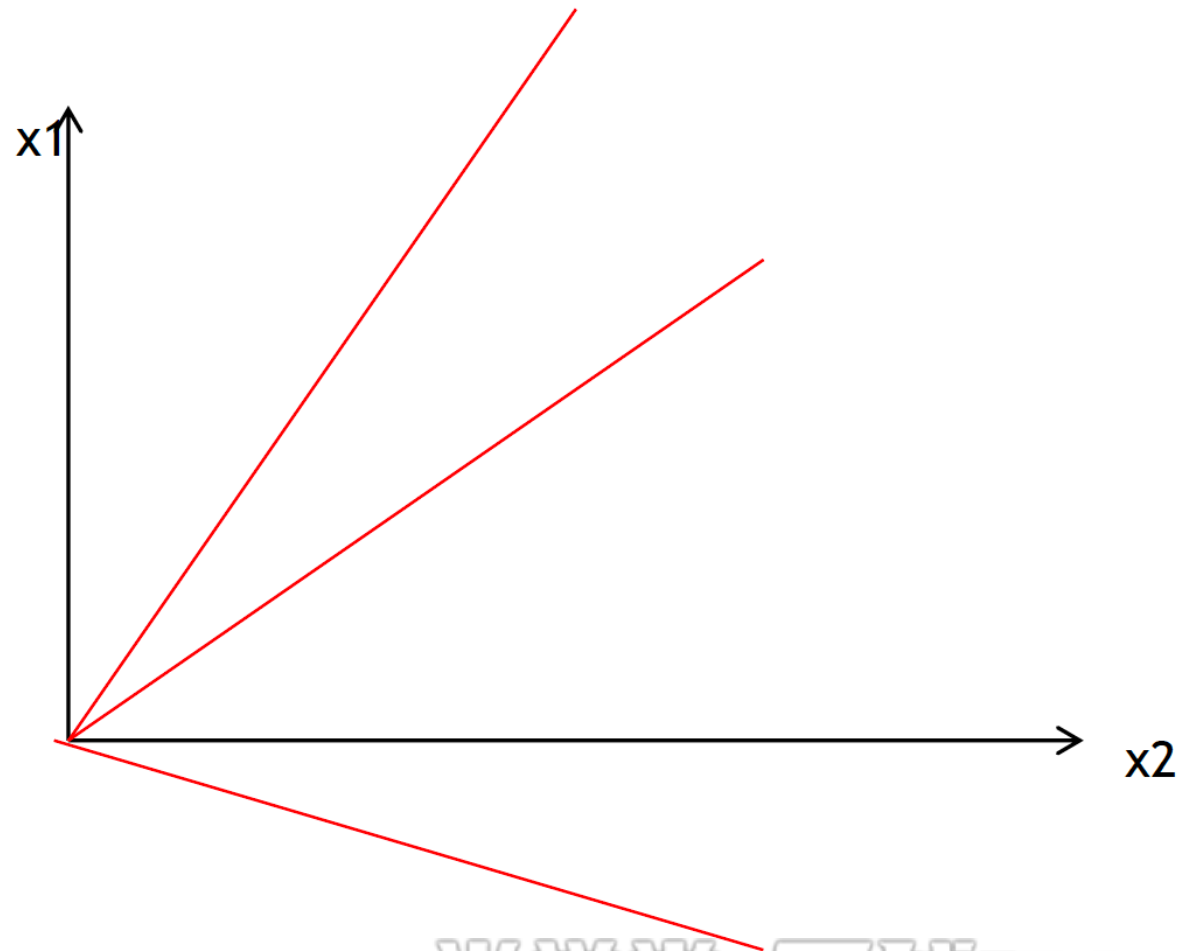
- W0(分界线的截距)

- 手动为数据集的x增加一列全1
  此时的w0就是截距

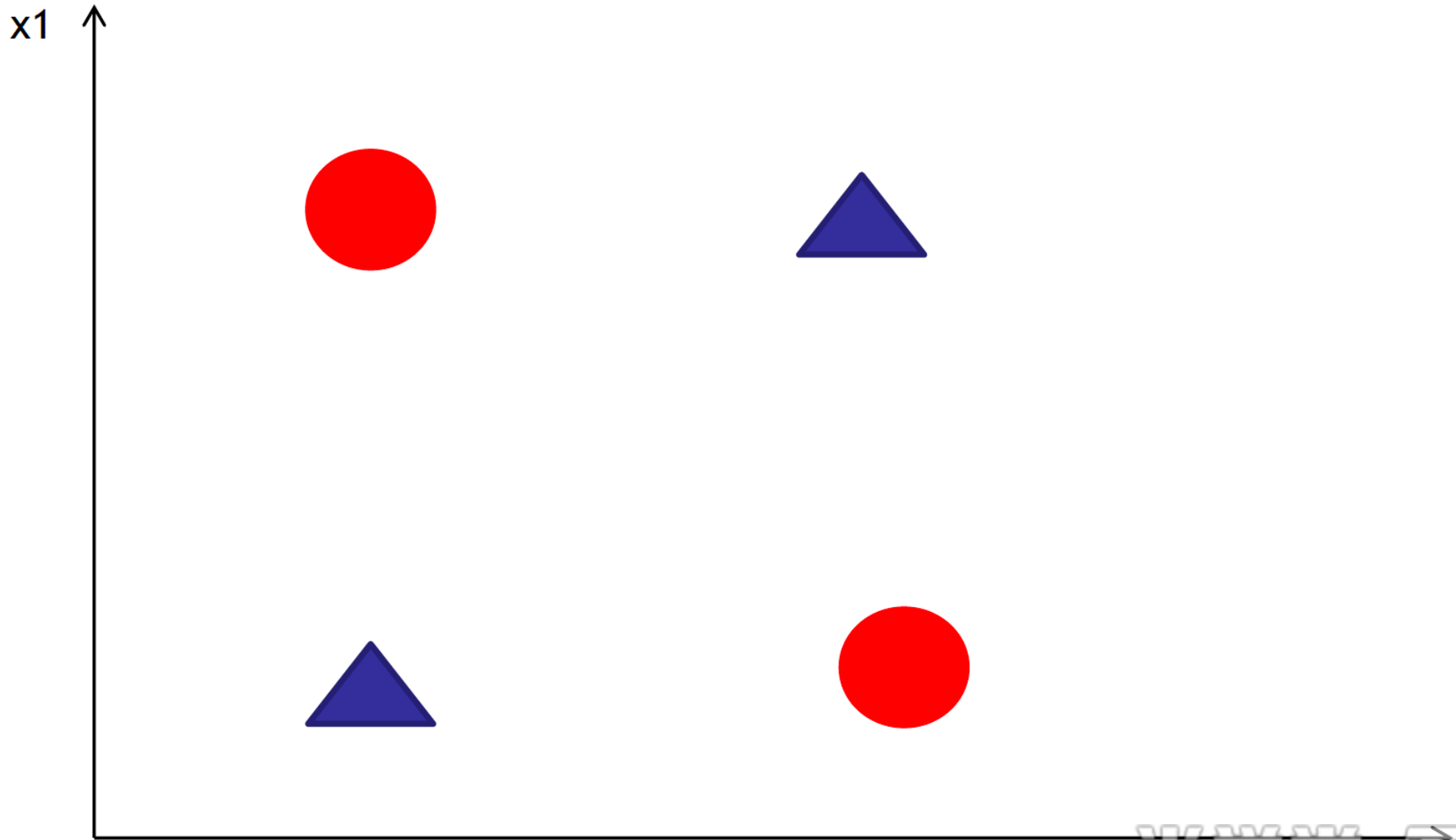


有w0 vs 无w0

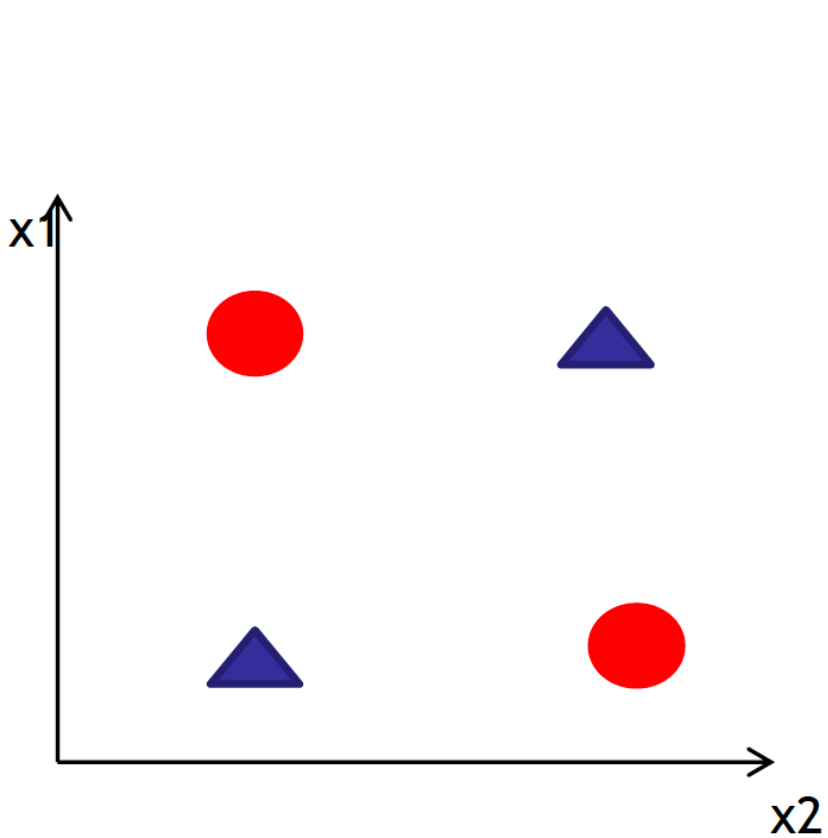


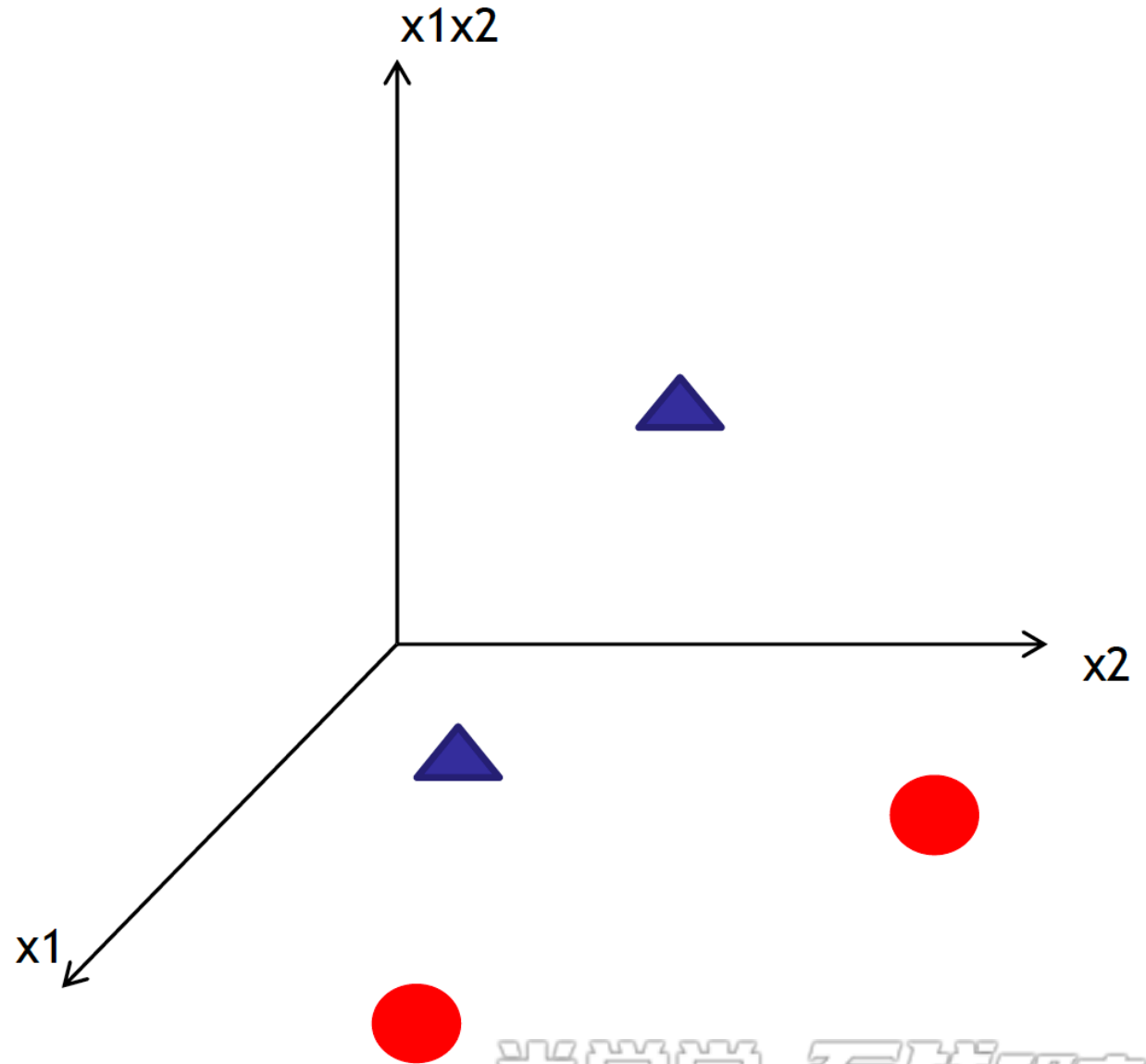


一种特殊情况：线性不可分



一种解决方案：映射至高维





改变阈值 0.5

- 根据需求的变通 去除固定阈值0.5

- 癌症病人的判断？
- 假如病人是癌症：
  - 判断成不是癌症
- 假如病人是非癌症
  - 判断是癌症

- 0.3
- 虽然整体的错误率变大了，但是规避了一些不能接受的风险

L1 L2 正则

- 复习线性回归中的 L1 L2正则

- L1 与 L2 的特性
  - Ridge　整体变小
  - Lasso　稀疏编码
    - 副产品 降维

- 牺牲正确率来提高模型的推广能力
  - 换句话说 牺牲测试集内的正确率换取验证集的正确率

多分类问题

- 将多分类转变为多个2分类
- 改变训练集 将多分类改为2分类

N分类转为n个二分类

- 修改数据的lable
- 训练N个逻辑回归模型
- 根据输出的概率结果输出
- 注意事项: 样本不均衡

样本不均衡问题

- 假如一个数据集 正负例样本比例为 1:100
  - 训练处的模型会倾向于将所有例子判为负例

- 解决方法:
  - 重采样 对多的欠采样, 对少的重采样
  - 人工创造新样本: 属性随机采样组成新的数据
  - 使用决策树算法

多分类问题 softmax

- 如果设计一个模型用于处理10分类问题 那么概率输出应该是什么样的?
- 输入一条数据 输出10个概率 选择概率最大的那个作为分类结果

$$h_\theta(x^{(i)}) = \begin{bmatrix} p(y^{(i)}=1|x^{(i)};\theta) \\ p(y^{(i)}=2|x^{(i)};\theta) \\ \vdots \\ p(y^{(i)}=k|x^{(i)};\theta) \end{bmatrix} = \frac{1}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}} \begin{bmatrix} e^{\theta_1^T x^{(i)}} \\ e^{\theta_2^T x^{(i)}} \\ \vdots \\ e^{\theta_k^T x^{(i)}} \end{bmatrix}$$

构建预测函数

$$h_\theta(x^{(i)}) = \begin{bmatrix} p(y^{(i)}=1|x^{(i)};\theta) \\ p(y^{(i)}=2|x^{(i)};\theta) \\ \vdots \\ p(y^{(i)}=k|x^{(i)};\theta) \end{bmatrix} = \frac{1}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}} \begin{bmatrix} e^{\theta_1^T x^{(i)}} \\ e^{\theta_2^T x^{(i)}} \\ \vdots \\ e^{\theta_k^T x^{(i)}} \end{bmatrix}$$

- 关键在于: $\theta = \begin{bmatrix} —\theta_1^T— \\ —\theta_2^T— \\ \vdots \\ —\theta_k^T— \end{bmatrix}$ 有了这一组θ, h(x)就可以使用了

损失函数

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=1}^{k} 1\left\{y^{(i)} = j\right\} \log \frac{e^{\theta_j^T x^{(i)}}}{\sum_{l=1}^{k} e^{\theta_l^T x^{(i)}}}\right]$$

- 有了损失函数就可以交给 sgd 或者 l-bfgs 来进行最小化 我们就可以求得一个 $\theta = \begin{bmatrix} —\theta_1^T— \\ —\theta_2^T— \\ \vdots \\ —\theta_k^T— \end{bmatrix}$ 使得模型表现最好

$$\theta_j := \theta_j - \alpha \nabla_{\theta_j} J(\theta) (j = 1, \ldots, k) \qquad \nabla_{\theta_j} J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[x^{(i)}\left(1\{y^{(i)} = j\} - p(y^{(i)} = j | x^{(i)}; \theta)\right)\right]$$

Softmax有趣的特点

$$
\begin{aligned}
p(y^{(i)} = j | x^{(i)}; \theta) &= \frac{e^{(\theta_j - \psi)^T x^{(i)}}}{\sum_{l=1}^{k} e^{(\theta_l - \psi)^T x^{(i)}}} \\
&= \frac{e^{\theta_j^T x^{(i)}} e^{-\psi^T x^{(i)}}}{\sum_{l=1}^{k} e^{\theta_l^T x^{(i)}} e^{-\psi^T x^{(i)}}} \\
&= \frac{e^{\theta_j^T x^{(i)}}}{\sum_{l=1}^{k} e^{\theta_l^T x^{(i)}}}.
\end{aligned}
$$

每组θ向量减去同一个向量, 不会影响最终预测结果

Softmax 与逻辑回归的关系

- 当类别数 k=2 时

$$h_\theta(x) = \frac{1}{e^{\theta_1^T x} + e^{\theta_2^T x^{(i)}}} \begin{bmatrix} e^{\theta_1^T x} \\ e^{\theta_2^T x} \end{bmatrix}$$

$$h(x) = \frac{1}{e^{\vec{0}^T x} + e^{(\theta_2-\theta_1)^T x^{(i)}}} \begin{bmatrix} e^{\vec{0}^T x} \\ e^{(\theta_2-\theta_1)^T x} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{1+e^{(\theta_2-\theta_1)^T x^{(i)}}} \\ \frac{e^{(\theta_2-\theta_1)^T x}}{1+e^{(\theta_2-\theta_1)^T x^{(i)}}} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{1+e^{(\theta_2-\theta_1)^T x^{(i)}}} \\ 1 - \frac{1}{1+e^{(\theta_2-\theta_1)^T x^{(i)}}} \end{bmatrix}$$

- 此时的softmax回归 就是 参数为(θ2-θ1)的逻辑回归