逻辑回归

从线性回归到逻辑回归

- 之前说的线性回归是预测一个连续的数值，比如金融机构根据一个人的工资、住房、年龄等特征预测放贷量（借多少钱）。
- 线性回归：$y = w_1 x_1 + w_2 x_2 + \ldots + w_n x_n + b$

- 而还有一类问题是预测类别，比如，金融机构根据一个人的工资、住房、年龄等特征预测是/否给这个人放贷。
- 这种预测类别的问题就要用到逻辑回归。

sigmoid函数

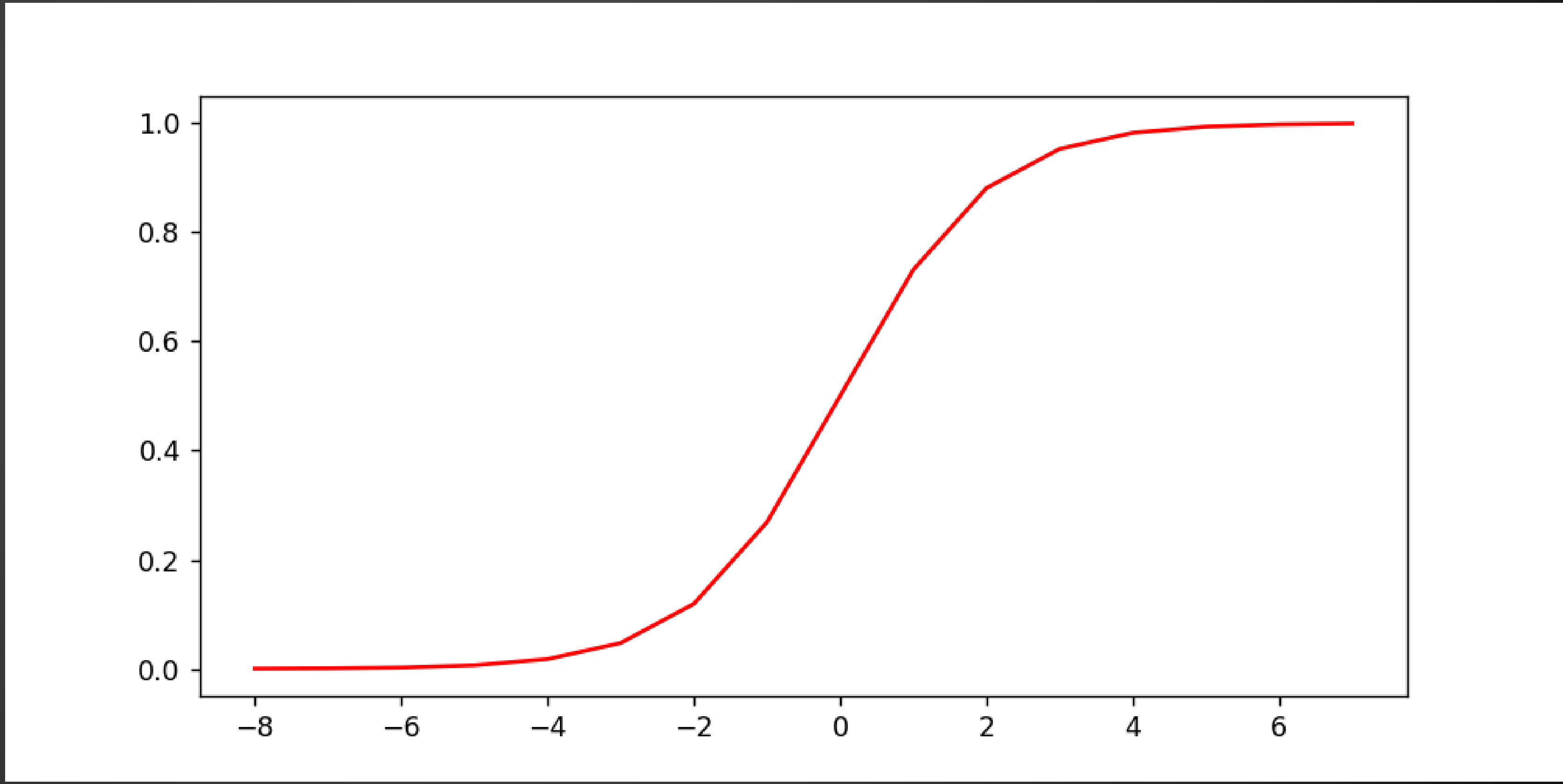
$$f(x) = \frac{1}{1+e^{-x}}, f(x) \in (0,1)$$

基本性质:

当$x \to +\infty, f(x) \to 1$

当$x \to -\infty, f(x) \to 0$

当$x = 0, f(x) = \frac{1}{2}$



逻辑回归基本原理

特征输入：X → 线性回归：回归方程 → 输出：y（数值型）

回归方程：$y = w_1x_1 + w_2x_2 + \ldots + w_nx_n + b$

特征输入：X → 线性回归：输出y（数值型） → 将y代入sigmoid函数 → 输出：$f(y)$ (0,1)，即概率

sigmoid函数：$f(x) = \frac{1}{1+e^{-x}}, f(x) \in (0,1)$

将y代入sigmoid函数：$f(y) = \frac{1}{1+e^{-y}}$，若$f(y) > 0.5$，标记为1，若$f(y) < 0.5$，则标记为0

逻辑回归主要步骤

1. 连续转为离散：sigmoid函数
2. 写出损失函数：似然函数
3. 利用梯度下降求解损失函数：梯度下降

第一步：连续转为离散

多元线性回归方程：

$$h_w(x_1, x_2, ..., x_n) = w_1x_1 + w_2x_2 + ... + w_nx_n + b$$

为了方便后面的运算，将其写成下面的形式：

$$h_w(x_1, x_2, ..., x_n) = w_0x_0 + w_1x_1 + w_2x_2 + ... + w_nx_n$$

其中，$w_0 = b, x_0 = 1$

接着，将其写成向量的形式

$$\vec{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad \vec{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix}$$

将其写为矩阵的乘法。

$$\begin{bmatrix} w_0, w_1, w_2, \cdots, w_n \end{bmatrix} \times \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

$$h_w(x) = w^T x = w_1 x_1 + w_2 x_2 + \ldots + w_n x_n + b$$

将线性回归函数$h_w(x) = w^T x$代入sigmoid函数，得

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

前面已经得到

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

假设预测结果只有两个类别：1和0，$y=1$的概率为p，$y=0$的概率为$1-p$

$$\begin{cases} P(y=1|x;w) = f(h_w(x)) = \frac{1}{1+e^{-w^Tx}} = p \\ P(y=0|x;w) = 1-p \end{cases}$$

为了方便计算，将以上式子统一为以下形式：

$$P(y|x;w) = p^y(1-p)^{1-y}$$

在上式中，当$y=1$时，结果是p，当$y=0$时，结果是$1-p$。

第二步：写出损失函数

假设现在采集到了m个样本，其似然函数为：

$$L(w) = P = P(y_1|x_1;w)P(y_2|x_2;w)\cdots P(y_m|x_m;w) = \Pi_{i=1}^{m} p^{y_i}(1-p)^{1-y_i}$$

但是，由于相乘的式子不好计算，所以通过取对数，将乘法变成加法，得到对数似然函数

$$lnL(w) = ln(\Pi_{i=1}^{m} p^{y_i}(1-p)^{1-y_i}) = \sum_{i=1}^{m} lnp^{y_i}(1-p)^{1-y_i}$$

$$= \sum_{i=1}^{m}(y_i lnp + (1-y_i)ln(1-p))$$

其中，$p = \frac{1}{1+e^{-w^T x}}$

接着要求这个似然函数的最大值，此时如果直接用梯度来求解最大值，就叫作梯度上升了。

为了跟其他机器学习问题统一，即最小化损失函数，引入函数$l(w)=-\frac{1}{m}lnL(w)$作为损失函数，此时将求解最大值问题转化为了求解最小值问题，所以可以利用梯度下降来求解这个最小值。

说明：除以m表示平均损失。
于是，损失函数为：

$$\begin{aligned} l(w) &= -\frac{1}{m}lnL(w) \\ &= -\frac{1}{m}\sum_{i=1}^{m}(y_i lnp + (1-y_i)ln(1-p)) \end{aligned}$$

其中，$p=\frac{1}{1+e^{-w^Tx}}$

第三步：应用梯度下降求解损失函数

首先，计算损失函数$l(w)$的梯度。

$$
\begin{aligned}
\frac{\partial l(w)}{\partial w_j} &= \frac{\partial}{\partial w_j}\left(-\frac{1}{m} lnL(w)\right) \\
&= -\frac{1}{m}\frac{\partial lnL(w)}{\partial w_j} \\
&= -\frac{1}{m}\sum_{i=1}^{m}\left(y_i\frac{1}{p}\frac{\partial p}{\partial w_j} + (-1)(1-y_i)\frac{1}{1-p}\frac{\partial p}{\partial w_j}\right)
\end{aligned}
$$

其中，$j = 0, 1, 2, \cdots, n$，$w_0 = b$

上式中提到了p关于参数w_j的导数

由于$p = \frac{1}{1+e^{-w^T x}}$，是一个复合函数，所以下面先求$\frac{\partial p}{\partial w_j}$

下面求$\frac{\partial p}{\partial w_j}$，$p=\frac{1}{1+e^{-w^Tx}}$

$$
\begin{aligned}
\frac{\partial p}{\partial w_j} &= \frac{\partial}{\partial w_j}\left(\frac{1}{1+e^{-w^Tx}}\right) \\
&= (-1)\times\frac{1}{(1+e^{-w^Tx})^2}\times e^{-w^Tx}\times(-x_j^{(i)}) \\
&= \frac{1}{(1+e^{-w^Tx})^2}\times e^{-w^Tx}\times x_j^{(i)} \\
&= \frac{1}{1+e^{-w^Tx}}\times\frac{e^{-w^Tx}}{1+e^{-w^Tx}}\times x_j^{(i)} \\
&= p(1-p)x_j^{(i)}
\end{aligned}
$$

总之，得到一个结论：$\frac{\partial p}{\partial w_j}=p(1-p)x_j^{(i)}$

我们已经有了$\frac{\partial l(w)}{\partial w_j}=-\frac{1}{m}\sum_{i=1}^{m}(y_i\frac{1}{p}\frac{\partial p}{\partial w_j}+(-1)(1-y_i)\frac{1}{1-p}\frac{\partial p}{\partial w_j})$和$\frac{\partial p}{\partial w_j}=p(1-p)x_j^{(i)}$

于是，接着之前的计算：

$$
\begin{aligned}
\frac{\partial l(w)}{\partial w_j} &= -\frac{1}{m}\sum_{i=1}^{m}(y_i\frac{1}{p}\frac{\partial p}{\partial w_j}+(-1)(1-y_i)\frac{1}{1-p}\frac{\partial p}{\partial w_j}) \\
&= -\frac{1}{m}\sum_{i=1}^{m}\{y_i\frac{1}{p}p(1-p)x_j^{(i)}+(-1)(1-y_i)\frac{1}{1-p}p(1-p)x_j^{(i)}\} \\
&= -\frac{1}{m}\sum_{i=1}^{m}\{y_i(1-p)x_j^{(i)}-(1-y_i)px_j^{(i)}\} \\
&= -\frac{1}{m}\sum_{i=1}^{m}(y_i-p)x_j^{(i)}
\end{aligned}
$$

将$p=\frac{1}{1+e^{-w^Tx}}$代回，得

$$\frac{\partial l(w)}{\partial w_j}=-\frac{1}{m}\sum_{i=1}^{m}(y_i-\frac{1}{1+e^{-w^Tx}})x_j^{(i)}$$

接下来，通过梯度下降求解参数的值，给参数设置一个初始值，然后通过不断更新参数使损失函数减小。

参数更新的表达式为：

$$w_i=w_i-\alpha\frac{\partial}{\partial w_i}l(w_0,w_1,...,w_n)=w_i+\alpha\frac{1}{m}\sum_{i=1}^{m}(y_i-\frac{1}{1+e^{-w^Tx}})x_j^{(i)}$$

说明：这里使用的是批量梯度下降。

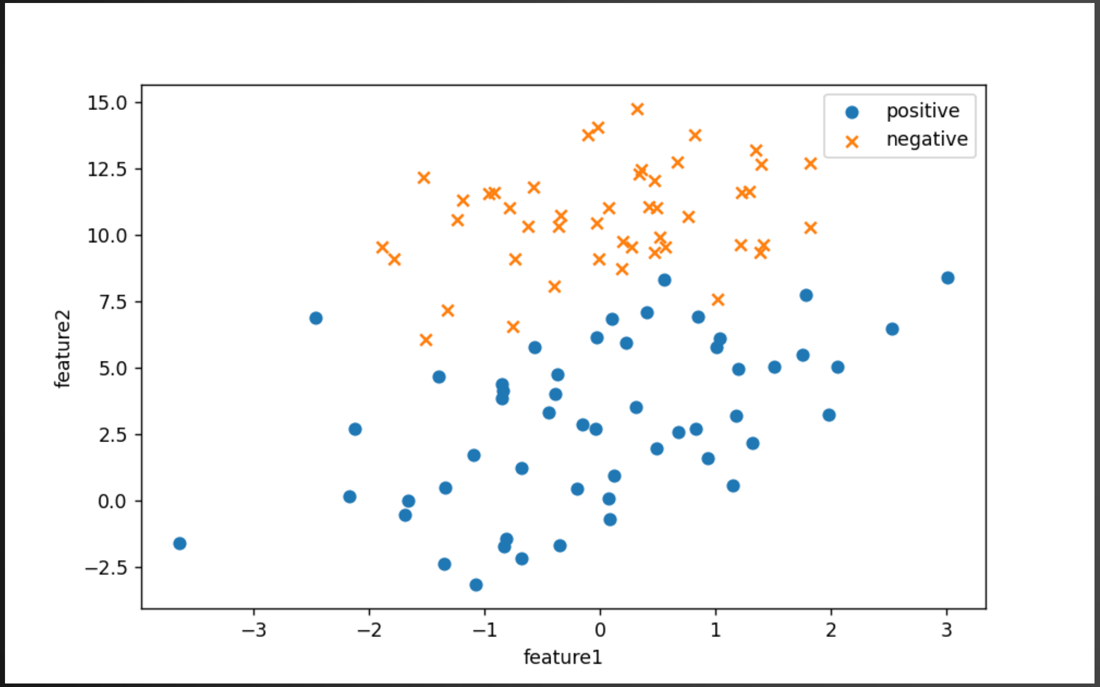
接下来通过计算工具，例如Python进行多次迭代。

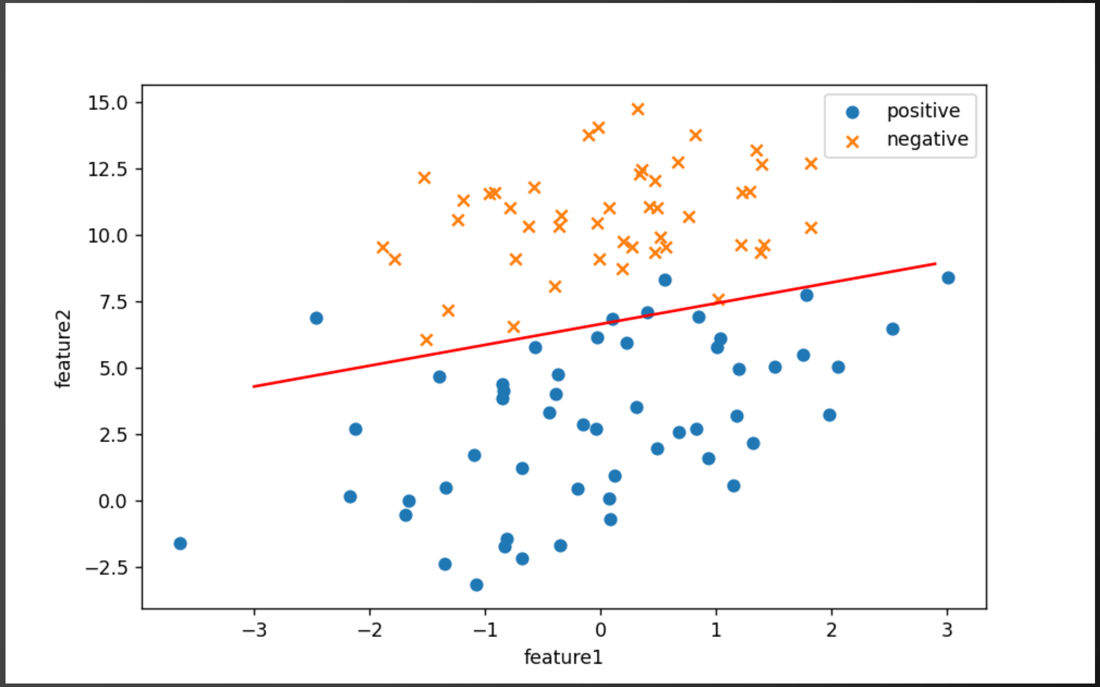
总结

1. 连续转为离散：sigmoid函数
2. 写出损失函数：似然函数，要求最大值，加负号，转化为求最小值
3. 利用梯度下降求解损失函数：梯度下降

梯度下降求解逻辑回归：代码实现

任务目标：使用逻辑回归将两类样本点分开





梯度下降求解逻辑回归：代码实现

代码主要步骤：

1. 数据读取，并绘制散点图
2. 将线性回归代入sigmoid函数
3. 定义损失函数
4. 求解梯度
5. 梯度下降：参数更新
6. 计算精度
7. 逻辑回归结果可视化
8. 应用sklearn中的逻辑回归工具库

案例：企业员工是否离职预测

- 某公司需要根据员工的一些数据预测员工是否会离职。
- 样本数据：一份CSV文件，共有14999个样本。9个特征变量，1个类别变量。

序号	字段名称	中文名称	字段描述
1	satisfaction_level	对公司的满意度	数值型，0-1
2	last_evaluation	最近一次考核分数	数值型，0-1
3	number_project	项目数	数值型，个数
4	average_montly_hours	平均每月工作时长	数值型，小时数
5	time_spend_company	工作年限	数值型，年
6	Work_accident	是否有过工作事故	类别，0：没有，1：有过
7	**left**	**是否离职**	**类别标签y，0：未离职，1：离职**
8	promotion_last_5years	过去5年是否晋升	类别，0：没有，1：有
9	sales	岗位类别	字符型，10种岗位类别
10	salary	薪资水平	字符型，三级：高、中、低

案例：企业员工是否离职预测

代码实现主要步骤：

1. 读取数据、认识数据
2. 数据探索及预处理
3. 逻辑回归建模
4. 模型评估：精度
5. 数据缩放