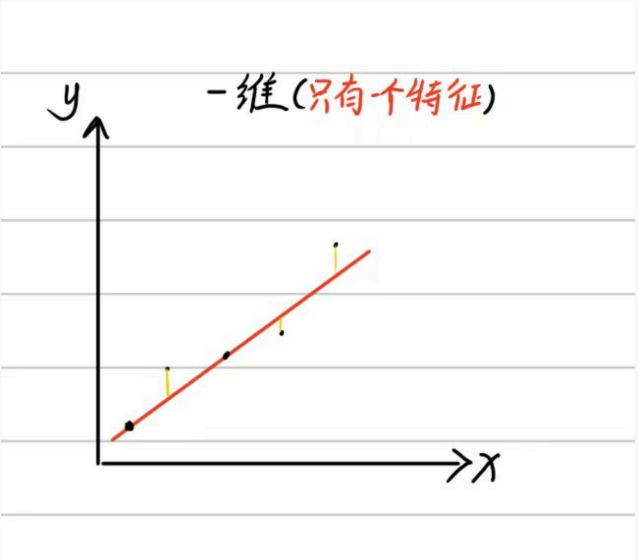
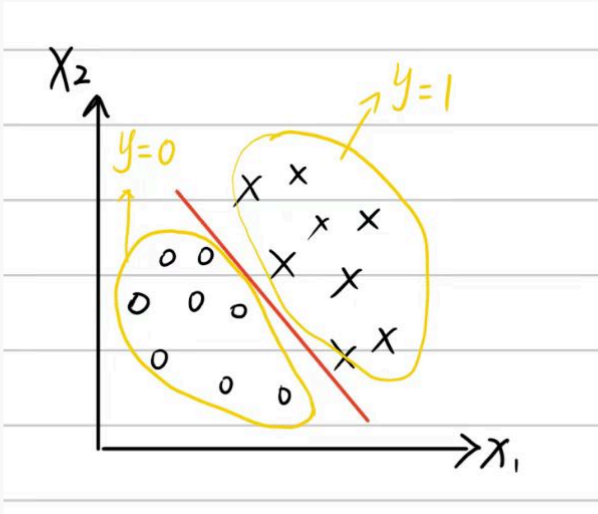
线性逻辑回归模型

一、线性逻辑回归模型定义

实际上就是一个**使用线性回归进行二分类的模型**，在机器学习中主要有**回归任务和分类任务两大类**，回归任务预测的是连续的变量，分类问题的输出是有限个离散变量。

下面两幅图展示了线性回归模型和逻辑回归模型的区别：

	
(a)线性回归模型	(b)线性逻辑回归模型

观察上图，线性回归模型横轴是特征（上图假设是一维），纵轴是预测目标的值，其**实际上是找到一条直线尽可能的逼近坐标上的点，也就是找到一组关系尽可能的逼近原有X-Y之间的关系**，而线性逻辑回归横轴和纵轴都是特征，其**实际上是想找到一条直线可以将不同类别（叉和圆的坐标点）分离开来**（称它为边界直线），对于任意的特征组X1,X2都有一个点对应在这个平面上，将其带入边界直线若值大于0则表示在边界线的上侧，小于0则表示在边界线的下侧，同时绝对值越大，属于该类别的可能性就越大，此时如果有一个函数可以将这些值转换到0-1之间，就可以将其看作属于正类别y=1的概率，通常会认为大于0.5就属于正类别，当然这个阈值是可以调节的。

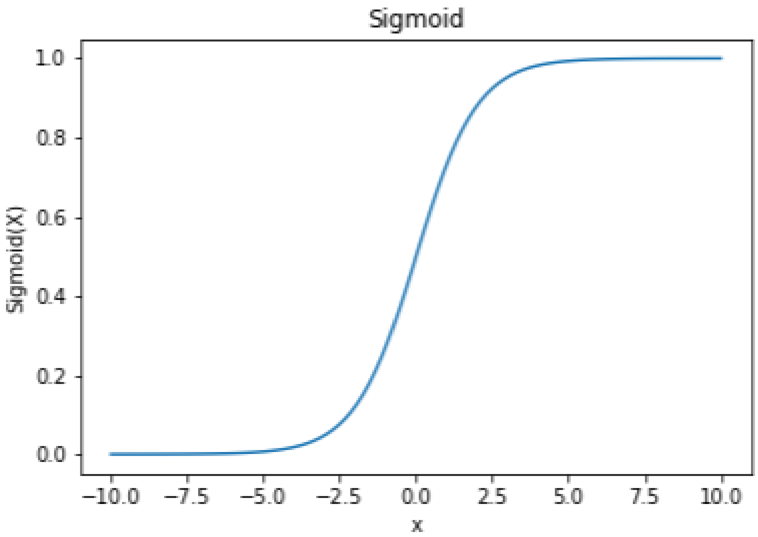
二、逻辑回归模型原理

1.sigmoid函数（S型函数）

sigmoid函数的定义如下：

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

其函数图像如下所示：



观察图像可以看到其定义域是任意实数，值域是[0,1],因此其作用就是**将输入的变量Z映射到0-1之间。**

2.模型假设函数

显然，我们可以通过sigmoid函数将值转换到0-1之间，我们将边界直线的表达式：

$$Z = W^T X + b \tag{2}$$

带入sigmoid函数就可以得到逻辑回归模型的公式：

$$g(X) = \frac{1}{1 + e^{-W^T X + b}} \tag{3}$$

因此对应预测结果为正例概率的表达式为：

$$P(y = 1|X) = \frac{1}{1 + e^{-W^T X + b}} \tag{4}$$

对应的预测结果负例概率表达式就是：1-P(y=1|X)，将正反例概率合并在一起如下式所示：

$$P(y|X) = P(y|X)^y [1 - P(y = 1|X)]^{1-y} \tag{5}$$

3.损失函数

我们需要**找到最优参数使得预测出的结果全部正确的概率最大**，简单来讲就是所有的样本的预测正确的概率相乘得到数值是最大的，这样我们就得到了损失函数如下所示：

$$L(w) = \prod_{i=1}^{m} P(1|x_i)^{y_i} [1 - P(1|x_i)]^{1-y_i} \tag{6}$$

我们用$\sigma(ww^T x_i + b)$表示sigmoid函数，同时为了计算方便**对其取对数，使得连乘变为连加**得到新的损失函数如下所示：

$$l(w) = \sum_{i=1}^{m} (y_i ln(\sigma(w^T x_i + b)) + (1 - y_i) ln(1 - \sigma(w^T x_i + b))) \tag{7}$$

在机器学习中我们通常式寻找损失函数的最低点，因此我们对上式取相反数，同时为了避免累加导致结果的不收敛，我们进行取平均，最终得到损失函数如下：

$$J(w) = -\frac{1}{m}\sum_{i=1}^{m}(y_i ln(\sigma(w^T x_i + b)) + (1-y_i) ln(1-\sigma(w^T x_i + b))) \tag{8}$$

4.参数求解

使用梯度更新法计算最优参数，就需要求解出损失函数对w和b的偏导，下面给出具体的计算过程：

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

则 $$\sigma'(z) = \frac{e^{-z}}{(1+e^{-z})^2} = \frac{1+e^{-z}-1}{(1+e^{-z})^2} = \sigma(z) - \sigma^2(z) = \sigma(z)[1-\sigma(z)]$$

$$z = w^T x_i + b \qquad \frac{\partial z}{\partial w} = x_i$$

则 $$\frac{\partial \sigma(w^T x_i + b)}{\partial w} = \sigma(w^T x_i + b)[1-\sigma(w^T x_i + b)]\, x_i$$

$$\frac{\partial J(w,b)}{\partial w} = -\frac{1}{m}\sum_{i=1}^{m} \frac{\partial\{y_i \ln\sigma(w^T x_i + b) + (1-y_i)\ln[1-\sigma(w^T x_i + b)]\}}{\partial w}$$

$$= -\frac{1}{m}\sum_{i=1}^{m} \frac{y_i}{\sigma(w^T x_i + b)}\frac{\partial\sigma(w^T x_i + b)}{\partial w} + \frac{(1-y_i)}{1-\sigma(w^T x_i + b)}\frac{\partial[-\sigma(w^T x_i + b)]}{\partial w}$$

$$= -\frac{1}{m}\sum_{i=1}^{m} [[1-\sigma(w^T x_i + b)] x_i y_i + \sigma(w^T x_i + b) x_i (y_i - 1)]$$

$$= -\frac{1}{m}\sum_{i=1}^{m} x_i y_i - x_i \sigma(w^T x_i + b)$$

$$= \frac{1}{m}\sum_{i=1}^{m} x_i[\sigma(w^T x_i + b) - y_i]$$

$$\frac{\partial z}{\partial b} = 1 \qquad \frac{\partial \sigma(w^T x_i + b)}{\partial w} = \sigma(w^T x_i + b)[1-\sigma(w^T x_i + b)]$$

与对w求偏导少了x_i，因此 $$\frac{\partial J(w,b)}{\partial b} = \frac{1}{m}\sum_{i=1}^{m}[\sigma(w^T x_i + b) - y_i]$$

三、代码实践问题

1.数据读取路径问题：

读取数据默认是从打开工程的文件根目录下搜索的，如果此时工程的根目录与存储数据的excel不在一地方就会报错无法找到文件如下所示：

```
FileNotFoundError: [Errno 2] No such file or directory: 'breast_cancer_data.csv'
```

解决方法：使用相对路径进行文件读取，此时的excel文件与python文件在一个目录下，因此使用os包中的函数获取到excel文件位置再进行读取，具体代码如下：

```
script_dir = os.path.dirname(__file__)#获取python文件路径
print(script_dir)
original_data_path = os.path.join(script_dir, "breast_cancer_data.csv")
original_data = pd.read_csv(original_data_path)
print(original_data)
```