



中国科学技术大学
University of Science and Technology of China

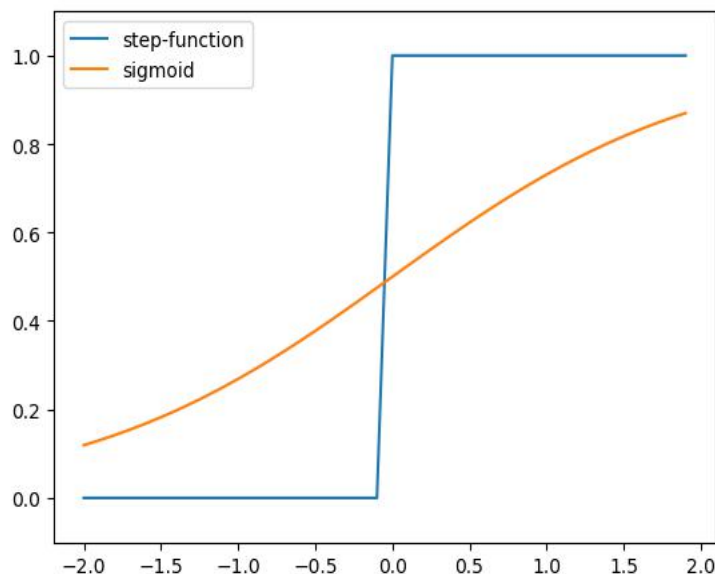
LR实验

时间：11/3/2020

机器学习中监督式学习方法主要包含两大类：

- 1) 变量呈现为离散值，一般用于分类——**分类模型**；
- 2) 变量呈现为连续性数值，一般用于数值型预测——**回归模型**。

举例来说，我们在不同的任务中目的是不相同，对于LR(logistics regression) 模型，我们得到的计算结果是0-1之间的一个数字。比如衡量客户购买某个商品的可能性，我们将LR的计算结果视为概率。同样的，如果在可能性区间中设置一个阈值(0.5)，当大于0.5时，认为客户会购买该商品；小于0.5则不会购买该商品。



左边图中黄色线即为logistics regression的拟合函数，称之为sigmoid函数

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

sigmoid函数是一个s形曲线，就像是阶跃函数的温和版，阶跃函数在0和1之间是突然的起跳，而sigmoid有个平滑的过渡。

多元回归方程形式：

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p$$

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta}$$

实际上，在很多问题中自变量的数目不止一个，我们用上式中矩阵形式表示，多元函数中线性组合形式。

$$P(Y = 1) = \frac{1}{1 + e^{-X\boldsymbol{\beta}}}$$

$$Z = X \cdot \boldsymbol{\beta}, \text{ 构造LR函数}$$

似然函数和最大似然估计：

数理统计中，似然函数是一种关于统计模型中参数的函数，与概率描述不同的是：概率一般是指定参数后，预测即将发生事件的可能性；在抑制某些观测所得的结果后，对有关概率模型的参数估计称之为似然估计。

最大似然函数估计：似然函数取得最大值表示相应的参数能够使得统计模型的参数估计最为合理。一般先选择一个似然函数，假设数据的分布满足正态分布函数的特性。应用中一般会取似然函数的对数作为计算最大值的函数。需要注意的一点是，似然函数的最大值不一定唯一，也不一定存在。

最大似然估计:

以二分类问题为例, y 的可能结果为0或1, 概率函数可以表示为:

$$P(y) = P(y = 1)^y P(y = 0)^{1-y}$$

加上自变量(特征量 x), 参数值(β):

$$P(y|x, \beta) = P(y = 1|x, \beta)^y [1 - P(y = 1|x, \beta)]^{1-y}$$

同时将sigmoid函数 $P(Y=1) = \frac{1}{1+e^{-X\beta}}$ 代入:

$$\mathcal{L}(\beta) = \prod_{i=1}^n P(y_i|x_i, \beta) = \prod_{i=1}^n \left(\frac{1}{1+e^{-x_i\beta}} \right)^{y_i} \left(1 - \frac{1}{1+e^{-x_i\beta}} \right)^{1-y_i}$$

最后取对数:

$$\log \mathcal{L}(\beta) = \sum_{i=1}^n \left([y_i \cdot \log(\frac{1}{1+e^{-x_i\beta}})] + [(1-y_i) \cdot \log(1 - \frac{1}{1+e^{-x_i\beta}})] \right)$$

损失函数：

损失函数是用于衡量预测值与实际值的偏移程度，即预测的错误程度。若不考虑过拟合的情况，当模型的损失函数值越小，则认为模型的预测能力越好。

Eg：线性回归中的平方损失函数：

$$Q = \sum_1^n (y_i - \hat{y}_i)^2 = \sum_1^n (y_i - x_i \beta)^2$$

在LR模型中，损失函数称之为最大似然损失函数，即似然函数取对数，在取得相反数：

$$J(\beta) = -\log \mathcal{L}(\beta) = -\sum_{i=1}^n [y_i \log P(y_i) + (1 - y_i) \log(1 - P(y_i))]$$

损失函数的偏导：

1. Sigmoid函数计算其导数：

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

2. 对数据集中每一个 x_i ，有多个特征的分量：

$$\frac{1}{1 + e^{-x_i\beta}} = f(x_i\beta) \quad \text{代入损失函数}$$

3. 对损失函数中的参数 β 求导：

$$\frac{\partial f(x_i\beta)}{\partial \beta_j} = f(x_i\beta) \cdot (1 - f(x_i\beta)) \cdot x_{ij}$$

4. 对数据集中所有的样本求和：

$$\frac{\partial J(\beta)}{\partial \beta_j} = -\sum_{i=1}^n (y_i - f(x_i\beta)) \cdot x_{ij} = \sum_{i=1}^n \left(\frac{1}{1 + e^{-x_i\beta}} - y_i \right) \cdot x_{ij}$$

梯度上升/下降法:



中国科学技术大学
University of Science and Technology of China

梯度下降优化的伪代码:

确定训练Epochs T ; 初始化 β (W , b); 学习率 α :

循环 T 个Epoch:

遍历所有的训练样本 Q (共 m 个样本):

对于每一个样本 X_i

分别计算训练数据的权重向量 dw , db 累加

$$W = W - \alpha * (dw/m)$$

$$b = b - \alpha * (db/m)$$

```
while  $\|\nabla \ell(\hat{w})\| > \delta$  do  
     $\hat{w}_{t+1} \leftarrow \hat{w}_t - \alpha \nabla \ell(\hat{w})$   
end while
```

梯度下降法原理

随机梯度上升法：



中国科学技术大学
University of Science and Technology of China

梯度下降优化的伪代码：

确定训练Epochs T ；初始化 β (W , b)；学习率 α ：

循环 T 个Epoch：

遍历所有的训练样本 Q (共 m 个样本)：

随机选择一个样本 X_i

分别计算训练数据的权重向量 dw ， db 累加

$$W = W - \alpha * (dw/m)$$

$$b = b - \alpha * (db/m) \quad m=1$$

牛顿法:



牛顿法伪代码:

确定训练Epochs T ; 初始化 β (W , b);

循环 T 个Epoch:

遍历所有的训练样本 Q (共 m 个样本):

求目标函数一阶导数

求目标函数二阶导数 (海森矩阵)

更新 W , b

```
while  $\|\nabla \ell(\hat{w})\| > \delta$  do  
     $\hat{w}_{t+1} \leftarrow \hat{w}_t - (\nabla^2 \ell(\hat{w}))^{-1} \nabla \ell(\hat{w})$   
end while
```

牛顿法中权重更新公式
(海森矩阵)

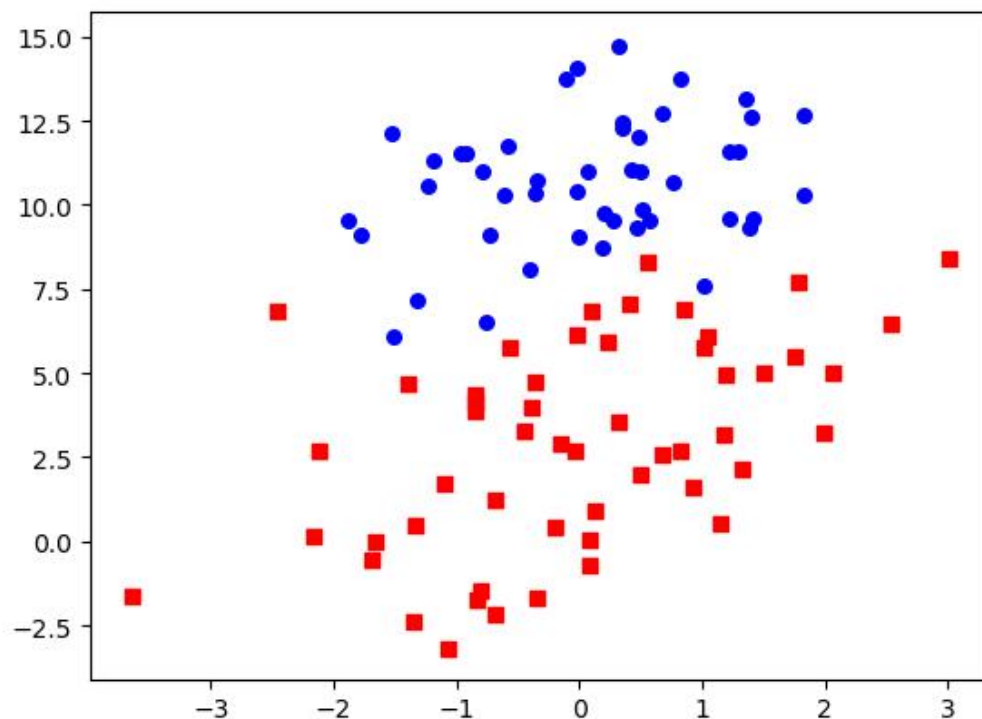
数据集：



中国科学技术大学
University of Science and Technology of China

分类数据集：

训练集图示：



Horse-colic数据集，数据集来自2010年1月11日UCI机器学习数据库。如左图所示，我们只选择其中100个样本，每个样本中包含两个特征。

实验要求:







数据集划分成训练集和测试集:

Train_data: 数据集大小为70

Test_data: 数据集大小为30

1. 使用梯度上升法或随机梯度上升法优化LR模型, 数据存放在data文件夹中。

 test_data.npy	2020/9/24 20:12	NPY 文件	1
 test_target.npy	2020/9/24 20:12	NPY 文件	1
 train_data.npy	2020/9/24 20:12	NPY 文件	2
 train_target.npy	2020/9/24 20:12	NPY 文件	1

2. 数据load方法

```
x_train = np.load("./Data/LR/train_data.npy")
y_train = np.load("./Data/LR/train_target.npy")
x_test = np.load("./Data/LR/test_data.npy")
y_test = np.load("./Data/LR/test_target.npy")
```

3. Baseline

测评指标: 精度值, 正确预测占整体的比例

训练集精度: 0.9

测试集精度: 0.85

实验要求:



中国科学技术大学
University of Science and Technology of China

LR模型分类示意图:

