一、线性回归模型

1、线性回归定义

线性回归模型定义：

假设给定数据集$D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$，其中$\boldsymbol{x}_i = (x_{i1}; x_{i2}; \ldots; x_{id}), y_i \in \mathbb{R}$，线性回归就是试图学的一个线性模型尽可能的准确的预测实际输出值。

通俗的讲就是求属性和结果之间的线性关系。线性回归模型的函数表达式可以用下面的式子来表达：

$$f(x) = w_1 x_1 + w_2 x_2 + \cdots + w_n x_n + b$$

当然也可以用向量的形式来表达：

$$f(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x} + b$$

从上面的式子来，可以得出线性回归模型就是要求得一组最优的w_i和b来确定线性模型，使这个模型无限逼近现有的数据x_i和结果$f(x_i)$之间的关系。

- x：特征
- y：标签
- w：权重

2、模型定义

设定线性回归模型：

利用上述单特征线性模型来解决这个问题。首先这个数据的输入就一个，就是小明的学习时间。输出是考试的得分。因此该模型可以确定为：

$$f(x) = wx + b$$

但是为了方便后面的理解和计算将该模型化简一下，我们只用一个w来表达输入和输出之间的关系（尽管这样不太严谨，仅仅是为了方便后面的计算），因此现在的模型可以简化为：

$$f(x) = wx$$

设定误差函数：

现在用一个公式计算输出和真实值间的误差：

$$\text{loss} = (f(x) - y)^2 = (wx - y)^2$$

当然数据是有很多的，要计算所有数据真实值和输出之间的误差和并计算出平均值，这个函数为均方误差函数，也是线性回归模型的损失函数。

$$J(x) = \frac{1}{2m} \sum_{i=1}^{m} (f(x_i) - y_i)^2$$

误差函数：1/2m*误差函数

3、权值w的确定

（1）穷举法

现在我们需要找到一个方法来帮助我们找到一个合理的w。现在我们用一个最笨的方法来进行w的求解，这个方法就是**穷举法。**

每日的学习时间	考试得分
1	2
2	4
3	6
4	?

Loss值计算公式：

$$\text{loss} = (f(x) - y)^2 = (wx - y)^2$$

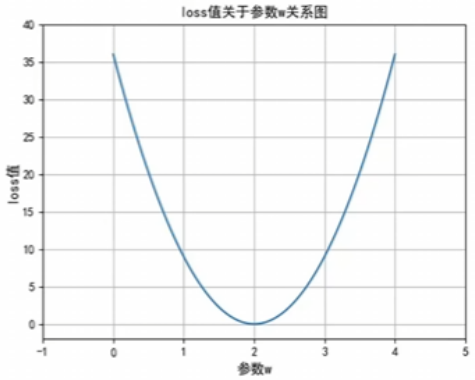
x (Hours)	Loss (w=0)	Loss (w=1)	Loss (w=2)	Loss (w=3)	Loss (w=4)
1	4	1	0	1	4
2	16	4	0	4	16
3	36	9	0	9	36
MSE	18.7	4.7	0	4.7	18.7

穷举法，就是每次通过穷举数值，通过计算`loss`值，来确定权值`w`。

（2）最小二乘法

Loss值计算公式：

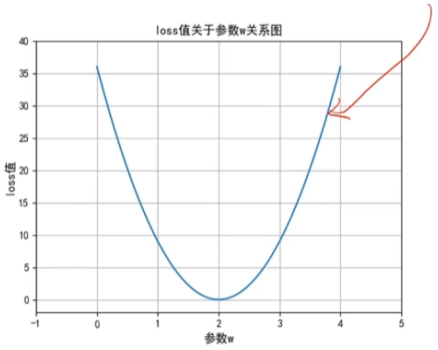
$$\text{loss} = (f(x) - y)^2 = (wx - y)^2$$



第一步：现在知道了`loss`的最小值，只需要对它的式子进行求导，一阶导数为0即可，就是所求点。

线性回归模型参数求解---最小二乘法

解 $w=2$ $w=\frac{y}{x}=w=2$ 1 2 2 4 3 6



求解Loss最小值：

$$\begin{aligned}\frac{\partial \text{Loss}}{\partial w} &= \frac{\partial (wx-y)^2}{\partial w} \\ &= 2(wx-y)\frac{\partial (wx-y)}{\partial w} \\ &= 2(wx-y)x\end{aligned}$$

令倒数为0，求得：

$$w = \frac{y}{x}\ （x\text{不为}0）$$

最小二乘法---向量形式

为了方便理解原理的同时也方便计算，我们将参数b纳入到矩阵$\boldsymbol{w}$中，此时数据特征矩阵$\boldsymbol{x}$则为：

$$\mathbf{X}=\begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1d} & 1 \\ x_{21} & x_{22} & \cdots & x_{2d} & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{md} & 1 \end{pmatrix}$$

线性回归模型的向量表达式是如下式所示：

$$f(\boldsymbol{x})=\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b$$

矩阵$\boldsymbol{w}$为：

$$w=\begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_m \\ b \end{pmatrix}$$

得到线性回归模型的向量表达式如下式所示：

$$f(\boldsymbol{X})=\boldsymbol{X}\boldsymbol{w}$$

第二步：损失函数向量表示形式：

$$J(w)=\frac{1}{2}(f(x)-Y)^2$$
$$=\frac{1}{2}(Xw-Y)^2$$
$$=\frac{1}{2}(Xw-Y)^T(Xw-Y)$$
$$=\frac{1}{2}(w^TX^T-Y^T)(Xw-Y)$$
$$=\frac{1}{2}(w^TX^TXw-Y^TXw-w^TX^TY+Y^TY)$$

第三步：对损失函数进行数学求导

$$J(w)=\frac{1}{2}(w^{\mathrm{T}}X^{\mathrm{T}}Xw-Y^{\mathrm{T}}Xw-w^{\mathrm{T}}X^{\mathrm{T}}Y+Y^{\mathrm{T}}Y)$$

现在针对$J(w)$求导数，首先要知道如下的知识点：

$$\frac{\partial AB}{\partial B}=A^T,\quad \frac{\partial A^TB}{\partial A}=B,\quad \frac{\partial X^TAX}{\partial X}=2AX$$

按这个规律对函数进行求导：

$$\frac{\partial J(w)}{\partial w}=\frac{1}{2}\left(\frac{\partial w^{\mathrm{T}}X^{\mathrm{T}}Xw}{\partial w}-\frac{\partial Y^{\mathrm{T}}Xw}{\partial w}-\frac{\partial w^{\mathrm{T}}X^{\mathrm{T}}Y}{\partial w}\right)$$
$$=\frac{1}{2}[2X^{\mathrm{T}}Xw-(Y^{\mathrm{T}}X)^{\mathrm{T}}-(X^{\mathrm{T}}Y)]$$
$$=\frac{1}{2}[2X^{\mathrm{T}}Xw-2(X^{\mathrm{T}}Y)]$$
$$=X^{\mathrm{T}}Xw-X^{\mathrm{T}}Y$$

令导数为$\frac{\partial J(w)}{\partial w}$=0，解得：

$$X^{\mathrm{T}}Xw-X^{\mathrm{T}}Y=0$$
$$X^{\mathrm{T}}X=X^{\mathrm{T}}Y$$
$$W=(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}Y$$

附加步骤：最小二乘法矩阵求导

$$J(w)=\frac{1}{2}(w^{\top}X^{\top}Xw-Y^{\top}Xw-w^{\top}X^{\top}Y+Y^{\top}Y)$$

现在针对$J(w)$求导数，首先要知道如下的知识点：

$$\frac{\partial AB}{\partial B}=A^{T},\quad \frac{\partial A^{T}B}{\partial A}=B,\quad \frac{\partial X^{T}AX}{\partial X}=2AX$$

按这个规律对函数进行求导：

$$\begin{aligned}\frac{\partial J(w)}{\partial w}&=\frac{1}{2}\left(\frac{\partial w^{\top}X^{\top}Xw}{\partial w}-\frac{\partial Y^{\top}Xw}{\partial w}-\frac{\partial w^{\top}X^{\top}Y}{\partial w}\right)\\&=\frac{1}{2}[2X^{\top}Xw-(Y^{\top}X)^{\top}-(X^{\top}Y)]\\&=\frac{1}{2}[2X^{\top}Xw-2(X^{\top}Y)]\\&=X^{\top}Xw-X^{\top}Y\end{aligned}$$

令导数为$\frac{\partial J(w)}{\partial w}$=0，解得：

$$X^{\top}Xw-X^{\top}Y=0$$
$$X^{\top}X=X^{\top}Y$$
$$W=(X^{\top}X)^{-1}X^{\top}Y$$

梯度下降法

$(X^{\top}X)^{-1}$

缺点：并不是所有的矩阵，都存在可逆矩阵，所以最小二乘法存在缺陷。

（3）梯度下降法

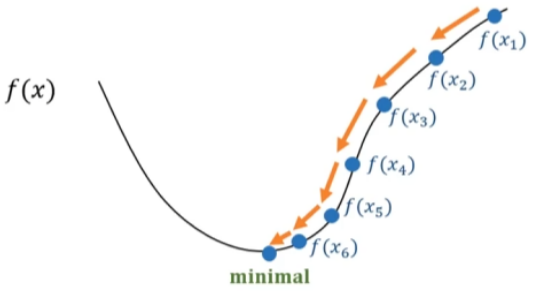
第一步：设置学习率

第二步：求解梯度

第三步：梯度参数更新

梯度下降法参数更新的计算公式就如下所示：

$$w=w-a\frac{\partial J(w)}{\partial w}$$



根据梯度下降法公式，如果最开始将学习率设置为0.1，那么可以的掉最终的w，但如果将学习率设置为0.5，就会出现距离正确值越来越远的情况。

$w - \alpha \frac{\partial}{\partial w}$

b

现在假设有一个损失函数为下式：

$$J(w)=4w^{2}$$

首先需要随机初始化$\boldsymbol{w}$,假设$\boldsymbol{w}=\mathbf{4}$，同时设定学习率为0.1。损失函数的导数如下式所示：

$$w_0=4, a=0.1, \frac{\partial J(w)}{\partial w}=8w$$

第一次$\boldsymbol{w}$更新的过程如下式计算：

$$\begin{aligned}w_1&=w_0-0.1\times\frac{\partial J(\omega)}{\partial w}\\&=4-0.1\times 8\times 4\\&=0.8\end{aligned}$$

=== 成功 ===

后续$\boldsymbol{w}$更新的过程如下式：

$$W_2=0.8-0.1\times 8\times 0.8=0.16$$
$$W_2=0.16-0.1\times 8\times 0.16=0.032$$
$$W_4=0.032-0.1\times 8\times 0.032=0.0064$$

假设此时设置的学习率为0.5：

$$w_0=4$$
$$w_1=4-05\times 8\times 4=-12$$
$$w_2=-12-05\times 8\times(-12)=32$$
$$w_3=32-0.5\times 8\times 32=96$$

=== 失败 ===

现在我们回到文章开头的小明学习成绩和学习时间关系的例子中来，看看怎么通过梯度下降法来进行最优$\boldsymbol{w}$的求解，这里还要注意的是，这里每次计算的梯度，是所有的数据的对应的梯度的平均值，计算公式如下所示：

$$\frac{J(w)}{\partial w} = \frac{1}{m}\frac{\partial \sum_{i=1}^{m}(f(x_i)-y_i)^2}{\partial w}$$

按照这个公式去计算的话，假设初始$\boldsymbol{w}=\boldsymbol{4}$，同时设定学习率为0.01

$$w_0 = 4$$

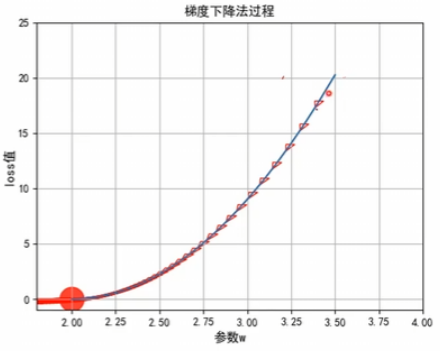
$$w_1 = 4 - 0.01 \times \frac{1}{m}\frac{\partial J(w)}{\partial w} = 3.813$$

$$w_2 = 3.644$$

$$w_3 = 3.490$$

$$\vdots$$

$$w_{99} = 2.000111$$



二、代码

```
"""
    # 1、实战内容
    线性回归，使用梯度下降法对小明的成绩进行求解。
    # 2、实战步骤
    第一步：
        数据，学习时间与成绩得分之间的关系。
    第二步：
        模型。
    第三步：
        损失函数。
    第四步：
        梯度求导。
    第五步：
        利用梯度更新参数。
    第六步：
        设置训练轮次。
    第七步：
        使用更新之后的参数，进行推理预测。
    # 3、个人理解
        （1）对于线性回归模型，咱们需要定义一个它的具体公式，研究x和y之间的关系，y一般等于w*x，这里可以将x理解为权值。
        （2）我们需要做的，就是找到这个权值w。
        ① 首先，如何对权值w进行更新？我们可以使用梯度进行更新，每次w进行一个梯度跳跃。
        ② 其次，因为我们并不知道“w进行一个梯度的条约是否正确，这个梯度是不是我们要找到梯度？”，
        因此我们可以将根据当前轮的w_now，计算出当前轮的预测数值y_pred=w_now*x，然后将“(预测值-真实值的平方)/len”作为当前轮次的损失值，如果损失值越来越小，那么就证明我们的梯度迭代方向是正确的。
        ③ 最后，我们只需要保证前两个步骤是正确的，那么按照梯度，不断迭代w就可以。
        （3）计算预测数值，并输出预测数值。
"""
# [第一步]，定义数据
x_data = [1, 2, 3]
y_data = [2, 4, 6]
# 初始化参数w
w = 4


# [第二步]，定义线性回归模型
```

```python
def forward(x):
    return x * w


# [第三步]，定义损失函数
def cost(xs, ys):
    """
        将“预测值-真实值”的平方累加之后的结果/len作为第index轮的损失值
    """
    cost_value = 0
    for x, y in zip(x_data, y_data):
        # 计算x对应的预测值y_pred
        y_pred = forward(x)
        # 计算损失cost_value
        cost_value += (y_pred - y) ** 2
    # 返回损失平均数值
    return cost_value / max(len(x_data), len(y_data))


# [第四步]，定义计算梯度的公式
def gradient(xs, ys):
    """
        主要是利用第三步之中的损失函数，来不断更新梯度。
        梯度主要是用来更新w的。
    """
    # 初始化梯度
    gradient_value = 0
    for x, y in zip(x_data, y_data):
        gradient_value += 2 * x * (x * w - y)
    return gradient_value / max(len(x_data), len(y_data))


# [第五步]，利用梯度进行更新参数
for index in range(100):
    """
        利用损失值的不断减小，不断对梯度进行更新。
    """
    # 得到第index次的损失值
    cost_val = cost(x_data, y_data)
    # 计算梯度
    grad_val = gradient(x_data, y_data)
    # 参数更新
    w = w - 0.01 * grad_val
    print(f"训练轮次[{index}]===损失：{cost_val},梯度：{grad_val}，w：{w}")

print(f"训练4小时，得分为{forward(4)}")
# 训练4小时，得分为8.000444482757587
```