1. 线性回归

1.1 概念

单层的神经网络，其实就是一个神经元，可以完成一些线性的工作，比如拟合一条直线，这用一个神经元就可以实现。当这个神经元只接收一个输入时，就是单变量线性回归，可以在二维平面上用可视化方法理解。当接收多个变量输入时，叫做多变量线性回归，此时可视化方法理解就比较困难了，通常我们会用变量两两组对的方式来表现。

当变量多于一个时，两个变量的量纲和数值有可能差别很大，这种情况下，我们通常需要对样本特征数据做归一化，然后把数据喂给神经网络进行训练，否则会出现“消化不良”的情况。

1.2 单入单出的单层神经网络 - 单变量线性回归

1.2.1 单变量线性回归问题

1.2.1.1 提出问题

实际上这就是线性回归的解题思路：利用已有值，预测未知值。只有一个自变量和一个因变量，因此可以用简单直接的方法来解决问题。但是，当有多个自变量时，这种直接的办法可能就会失效了。假设有三个自变量，很有可能不能够在样本中找到和这三个自变量的组合非常接近的数据，此时我们就应该借助更系统的方法了。

1.2.1.2 一元线性回归模型

回归分析是一种数学模型。当因变量和自变量为线性关系时，它是一种特殊的线性模型。

最简单的情形是一元线性回归，由大体上有线性关系的一个自变量和一个因变量组成，模型是：

$$Y = a + bX + \varepsilon \qquad (1)$$

X 是自变量，Y 是因变量，ε 是随机误差，a 和 b 是参数，在线性回归模型中，a, b 是我们要通过算法学习出来的。

对于线性回归模型，有如下一些概念需要了解：

- 通常假定随机误差 ε 的均值为 0，方差为σ^2（$\sigma^2 > 0$，σ^2 与 X 的值无关）
- 若进一步假定随机误差遵从正态分布，就叫做正态线性模型
- 一般地，若有 k 个自变量和 1 个因变量（即公式1中的 Y），则因变量的值分为两部分：一部分由自变量影响，即表示为它的函数，函数形式已知且含有未知参数；另一部分由其他的未考虑因素和随机性影响，即随机误差
- 当函数为参数未知的线性函数时，称为线性回归分析模型
- 当函数为参数未知的非线性函数时，称为非线性回归分析模型
- 当自变量个数大于 1 时称为多元回归
- 当因变量个数大于 1 时称为多重回归

1.2.2 最小二乘法

1.2.2.1 历史

最小二乘法，也叫做最小平方法（Least Square），**它通过最小化误差的平方和寻找数据的最佳函数匹配。**利用最小二乘法可以简便地求得未知的数据，并使得这些求得的数据与实际数据之间误差的平方和为最小。最小二乘法还可用于曲线拟合。其他一些优化问题也可通过最小化能量或最小二乘法来表达。

1.2.2.2 数学推理结果

线性回归试图学得：

$$z_i = w \cdot x_i + b \qquad (1)$$

使得：

$$z_i \simeq y_i \qquad (2)$$

其中，x_i 是样本特征值，y_i 是样本标签值，z_i 是模型预测值。

均方差(MSE - mean squared error)是回归任务中常用的手段：

$$J = \frac{1}{2m}\sum_{i=1}^{m}(z_i - y_i)^2 = \frac{1}{2m}\sum_{i=1}^{m}(y_i - wx_i - b)^2 \qquad (3)$$

J 称为损失函数。实际上就是试图找到一条直线，使所有样本到直线上的残差的平方和最小。，通过对 w 和 b 求导，再令导数为 0（到达最小极值），就是 w 和 b 的最优解。

结果为：

分子分母都乘以 m：

$$w = \frac{m\sum_{i=1}^{m} x_i y_i - \sum_{i=1}^{m} x_i \sum_{i=1}^{m} y_i}{m\sum_{i=1}^{m} x_i^2 - (\sum_{i=1}^{m} x_i)^2} \tag{13}$$

$$b = \frac{1}{m}\sum_{i=1}^{m}(y_i - wx_i) \tag{14}$$

1.2.3 梯度下降法

1.2.3.1 预设函数（Hypothesis Function）

线性函数：

$$z_i = x_i \cdot w + b \tag{1}$$

1.2.3.2 损失函数（Loss Function）

均方误差：

$$loss_i(w,b) = \tfrac{1}{2}(z_i - y_i)^2 \tag{2}$$

与最小二乘法比较可以看到，梯度下降法和最小二乘法的模型及损失函数是相同的，都是一个线性模型加均方差损失函数，模型用于拟合，损失函数用于评估效果。

区别在于，最小二乘法从损失函数求导，直接求得数学解析解，而梯度下降是利用导数传递误差，再通过迭代方式一步一步（用近似解）逼近真实解。

1.2.3.3 计算z的梯度

根据公式2：

$$\frac{\partial loss}{\partial z_i} = z_i - y_i \tag{3}$$

1.2.3.4 计算 w 的梯度

用 $loss$ 的值作为误差衡量标准，通过求 w 对它的影响，也就是 $loss$ 对 w 的偏导数，来得到 w 的梯度。

根据公式1和公式3：

$$\frac{\partial loss}{\partial w} = \frac{\partial loss}{\partial z_i}\frac{\partial z_i}{\partial w} = (z_i - y_i)x_i \tag{4}$$

1.2.3.5 计算 b 的梯度

$$\frac{\partial loss}{\partial b} = \frac{\partial loss}{\partial z_i}\frac{\partial z_i}{\partial b} = z_i - y_i \tag{5}$$

1.2.4 神经网络法

简单讲述了一下神经网络做线性拟合的原理，即：

1. 初始化权重值
2. 根据权重值放出一个解
3. 根据均方差函数求误差
4. 误差反向传播给线性计算部分以调整权重值
5. 是否满足终止条件？不满足的话跳回2

其实神经网络法和梯度下降法在本质上是一样的，只不过神经网络法使用一个崭新的编程模型，即以神经元为中心的代码结构设计，这样便于以后的功能扩充。

1.2.4 三种形式的比较

三种方法得到的 w 和 b 的值

方法	w	b
最小二乘法	2.056827	2.965434
梯度下降法	1.71629006	3.19684087
神经网络法	1.71629006	3.19684087

这个问题的原始值是可能是 $w=2, b=3$，由于样本噪音的存在，使用最小二乘法得到了 $2.05, 2.96$ 这样的非整数解，但是使用梯度下降和神经网络两种方式，都得到 $1.71, 3.19$ 这样的值，准确程度很低。

最小二乘法可以得到数学解析解，所以它的结果是可信的。梯度下降法和神经网络法实际是一回事儿，只是梯度下降没有使用神经元模型而已。

原因探究如下：

1.2.5 梯度下降的三种形式

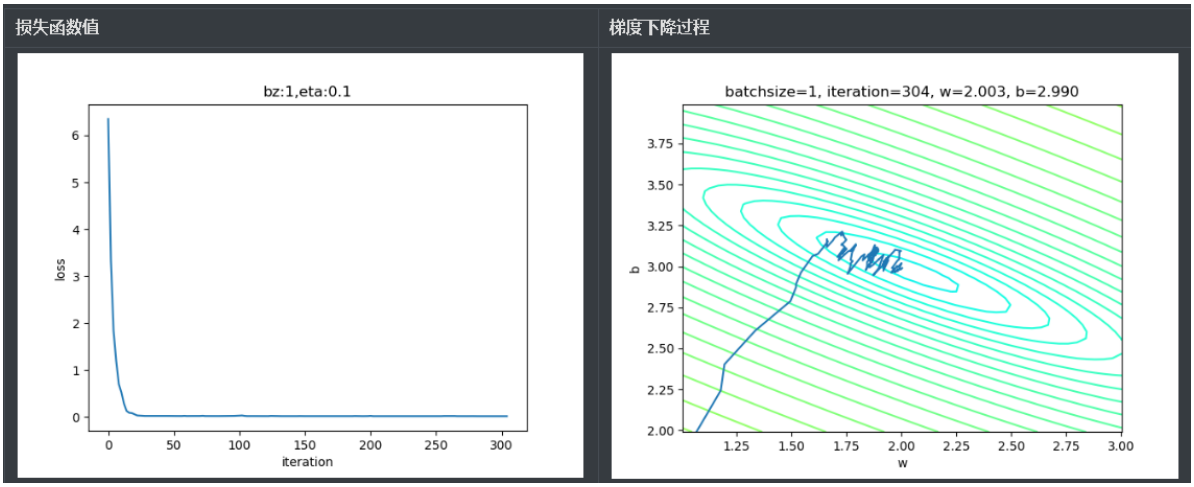
1.2.5.1 单样本随机梯度下降

图示：

特点：

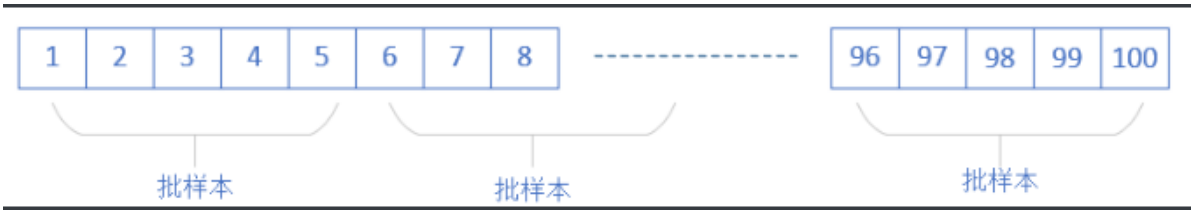
- 训练样本：每次使用一个样本数据进行一次训练，更新一次梯度，重复以上过程。
- 优点：训练开始时损失值下降很快，随机性大，找到最优解的可能性大。
- 缺点：受单个样本的影响最大，损失函数值波动大，到后期徘徊不前，在最优解附近震荡。不能并行计算。

设置`batch_size=1`，即单样本方式：



1.2.5.2 小批量样本梯度下降

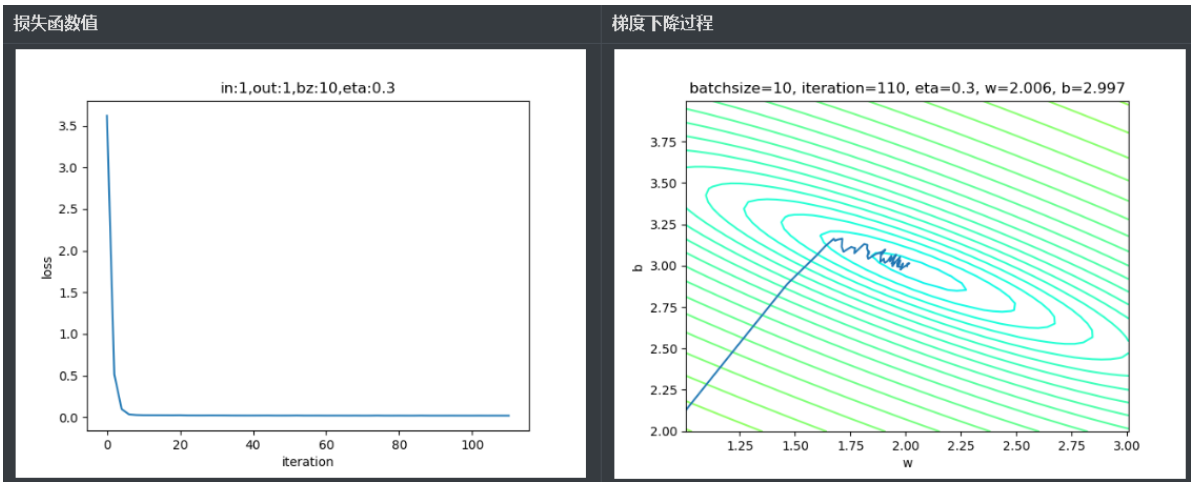
图示：



特点：

- 训练样本：选择一小部分样本进行训练，更新一次梯度，然后再选取另外一小部分样本进行训练，再更新一次梯度。
- 优点：不受单样本噪声影响，训练速度较快。
- 缺点：batch size的数值选择很关键，会影响训练结果。

设置`batch_size=10`：



在实际工程中，我们通常使用小批量梯度下降形式。

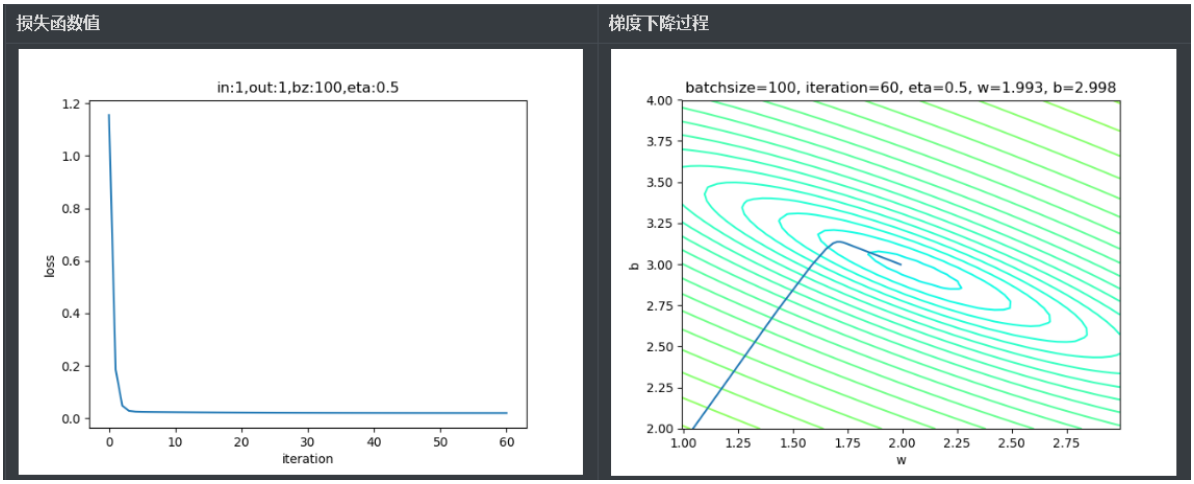
1.2.5.3 全批量样本梯度下降

图示：

特点：

- 训练样本：每次使用全部数据集进行一次训练，更新一次梯度，重复以上过程。
- 优点：受单个样本的影响最小，一次计算全体样本速度快，损失函数值没有波动，到达最优点平稳。方便并行计算。
- 缺点：数据量较大时不能实现（内存限制），训练过程变慢。初始值不同，可能导致获得局部最优解，并非全局最优解。

设置`batch_size=-1`，即是全批量的意思。



1.3 多入单出的单层神经网络 - 多变量线性回归

1.3.1 多变量线性回归问题

1.3.1.1 提出问题

问题：在北京通州，距离通州区中心15公里的一套93平米的房子，大概是多少钱?

样本序号	地理位置	居住面积	价格（万元）
1	10.06	60	302.86
2	15.47	74	393.04
3	18.66	46	270.67
4	5.20	77	450.59
...	...	...	...

- 特征值1 - 地理位置，统计得到：
    - 最大值：21.96公里
    - 最小值：2.02公里
    - 平均值：12.13公里
- 特征值2 - 房屋面积，统计得到：
    - 最大值：119平米
    - 最小值：40平米
    - 平均值：78.9平米
- 标签值 - 房价，单位为百万元：
    - 最大值：674.37
    - 最小值：181.38

- 平均值：420.64

这个数据是三维的，所以可以用两个特征值作为 x 和 y，用标签值作为 z，在 xyz 坐标空间中

1.3.1.2 多元线性回归模型

多元线性回归的函数模型如下：

$y = a_0 + a_1 x_1 + a_2 x_2 + \cdots + a_k x_k$

具体化到房价预测问题，上面的公式可以简化成：

$$z = x_1 \cdot w_1 + x_2 \cdot w_2 + b$$

抛开本例的房价问题，对于一般的应用问题，建立多元线性回归模型时，为了保证回归模型具有优良的解释能力和预测效果，应首先注意自变量的选择，其准则是：

1. 自变量对因变量必须有显著的影响，并呈密切的线性相关；
2. 自变量与因变量之间的线性相关必须是真实的，而不是形式上的；
3. 自变量之间应具有一定的互斥性，即自变量之间的相关程度不应高于自变量与因变量之因的相关程度；
4. 自变量应具有完整的统计数据，其预测值容易确定。

1.3.1.3 解决方案之正规方程

英文名是 Normal Equations。

对于线性回归问题，除了前面提到的最小二乘法可以解决一元线性回归的问题外，也可以解决多元线性回归问题。

对于多元线性回归，可以用正规方程来解决，也就是得到一个数学上的解析解。它可以解决下面这个公式描述的问题：

$$y = a_0 + a_1 x_1 + a_2 x_2 + \cdots + a_k x_k \qquad (1)$$

1.3.1.3.1 简单的推导方法

在做函数拟合（回归）时，我们假设函数 H 为：

$$H(w, b) = b + x_1 w_1 + x_2 w_2 + \cdots + x_n w_n \qquad (2)$$

令 $b = w_0$，则：

$$H(W) = w_0 + x_1 \cdot w_1 + x_2 \cdot w_2 + \cdots + x_n \cdot w_n \qquad (3)$$

公式3中的 x 是一个样本的 n 个特征值，如果我们把 m 个样本一起计算，将会得到下面这个矩阵：

$$H(W) = X \cdot W \qquad (4)$$

公式5中的 X 和 W 的矩阵形状如下：

$$X = \begin{pmatrix} 1 & x_{1,1} & x_{1,2} & \cdots & x_{1,n} \\ 1 & x_{2,1} & x_{2,2} & \cdots & x_{2,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{m,1} & x_{m,2} & \cdots & x_{m,n} \end{pmatrix} \qquad (5)$$

$$W = \begin{pmatrix} w_0 \\ w_1 \\ \vdots \\ w_n \end{pmatrix} \tag{6}$$

然后我们期望假设函数的输出与真实值一致，则有：

$$H(W) = X \cdot W = Y \tag{7}$$

其中，Y的形状如下：

$$Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix} \tag{8}$$

先把等式两边同时乘以 X 的转置矩阵，以便得到 X 的方阵：

$$X^\top XW = X^\top Y \tag{9}$$

简化得到：

$$W = (X^\top X)^{-1} X^\top Y \tag{10}$$

至此可以求出 W 的正规方程。

1.3.1.3.2 运行结果

带入具体的值就可以得到相关的结果，

```
w1= -2.0184092853092226
w2= 5.055333475112755
b= 46.235258613837644
z= 486.1051325196855
```

然后得到房价预测值 $z = 486$ 万元。

至此，得到了解析解。可以用这个做为标准答案，去验证的神经网络的训练结果。

1.3.1.4 解决方案之神经网络

前期准备工作！！！！

1. 样本特征数据标准化
2. 标准化标签值

1.3.1.4.1 为何要样本特征数据标准化

理论层面上，神经网络是以样本在事件中的统计分布概率为基础进行训练和预测的，所以它对样本数据的要求比较苛刻。具体说明如下：

1. 样本的各个特征的取值要符合概率分布，即 $[0, 1]$。

2. 样本的度量单位要相同。我们并没有办法去比较1米和1公斤的区别，但是，如果我们知道了1米在整个样本中的大小比例，以及1公斤在整个样本中的大小比例，比如一个处于0.2的比例位置，另一个处于0.3的比例位置，就可以说这个样本的1米比1公斤要小。
3. 神经网络假设所有的输入输出数据都是标准差为1，均值为0，包括权重值的初始化，激活函数的选择，以及优化算法的设计。
4. 数值问题

   标准化可以避免一些不必要的数值问题。因为激活函数sigmoid/tanh的非线性区间大约在$[-1.7，1.7]$。意味着要使神经元有效，线性计算输出的值的数量级应该在1（1.7所在的数量级）左右。这时如果输入较大，就意味着权值必须较小，一个较大，一个较小，两者相乘，就引起数值问题了。
5. 梯度更新

   若果输出层的数量级很大，会引起损失函数的数量级很大，这样做反向传播时的梯度也就很大，这时会给梯度的更新带来数值问题。
6. 学习率

   如果梯度非常大，学习率就必须非常小，因此，学习率（学习率初始值）的选择需要参考输入的范围，不如直接将数据标准化，这样学习率就不必再根据数据范围作调整。对 w_1 适合的学习率，可能相对于 w_2 来说会太小，若果使用适合 w_1 的学习率，会导致在 w_2 方向上步进非常慢，从而消耗非常多的时间；而使用适合 w_2 的学习率，对 w_1 来说又太大，搜索不到适合 w_1 的解。

1.3.1.4.2 样本特征数据标准化的常用方法

- Min-Max标准化（离差标准化），将数据映射到 $[0,1]$ 区间

$$x_{new} = \frac{x-x_{min}}{x_{max}-x_{min}} \qquad (1)$$

- 平均值标准化，将数据映射到[-1,1]区间

$$x_{new} = \frac{x-\bar{x}}{x_{max}-x_{min}} \qquad (2)$$

- 对数转换

$$x_{new} = \ln(x_i) \qquad (3)$$

- 反正切转换

$$x_{new} = \frac{2}{\pi}\arctan(x_i) \qquad (4)$$

- Z-Score法

把每个特征值中的所有数据，变成平均值为0，标准差为1的数据，最后为正态分布。Z-Score规范化（标准差标准化 / 零均值标准化，其中std是标准差）：

$$x_{new} = \frac{x_i-\bar{x}}{std} \qquad (5)$$

- 中心化，平均值为0，无标准差要求

$$x_{new} = x_i - \bar{x} \qquad (6)$$

- 比例法，要求数据全是正值

$$x_{new} = \frac{x_k}{\sum_{i=1}^{m} x_i} \qquad (7)$$

1.3.1.4.3 总结：

1. X 必须标准化，否则无法训练；
2. Y 值不在 $[0,1]$ 之间时，要做标准化，好处是迭代次数少；
3. 如果 Y 做了标准化，对得出来的预测结果做关于 Y 的反标准化

遇到的一些问题解决收获：

1. 样本不做标准化的话，网络发散，训练无法进行；
2. 训练样本标准化后，网络训练可以得到结果，但是预测结果有问题；
3. 还原参数值后，预测结果正确，但是此还原方法并不能普遍适用；
4. 标准化测试样本，而不需要还原参数值，可以保证普遍适用；
5. 标准化标签值，可以使得网络训练收敛快，但是在预测时需要把结果反标准化，以便得到真实值。