机器学习和深度学习基本概念简介

机器学习

机器学习=寻找函数

不同的函数：regression（输出scale），classification（选择），Structured learning

如何寻找函数：

- 1、写出一个带有未知数的函数（base on domain knowledge）（也许是个猜测）

  y=b+x1，这里带有未知数的函数为model，x1为Feature，w为权重（weight）和b为偏置（bias）

- 2、定义Loss，L（b,w)。

  用于评价w和b的设置是好还是不好，label是指真实值。

  损失的计算方式有MAE（绝对差），MSE（相减平方）。

  调整参数画出等高线图，是Error Surface。

- 3、最佳化。

  找合适的未知参数的值使Loss最小。Gradient Descent（梯度下降）。

  梯度下降：

  假设只有一个参数w，w不同数值得到不同Loss，获得曲线。

  随机选取w的初始值w0，然后计算w0的微分。 $\frac{\partial L}{\partial w}\Big|_{w=w^0}$

  向Loss下降的方向移动，移动距离由$\eta \frac{\partial L}{\partial w}\Big|_{w=w^0}$决定，这里的η是learning rate（学习速率），是自己设定的。自己决定的参数是hyperparameters。

  移动结束有俩个结果，一个是达到运动次数停止，理想停下是微分为0时。但有时候无法找到全局最优，大部分为局部最优（这是假命题）。

  因此由一个参数w推广为俩个参数w，b，步骤为：

  1. 初始化：随机选取w0，b0

2. 求微分偏导：$\frac{\partial L}{\partial w}\Big|_{w=w^0,b=b^0}$，$\frac{\partial L}{\partial b}\Big|_{w=w^0,b=b^0}$

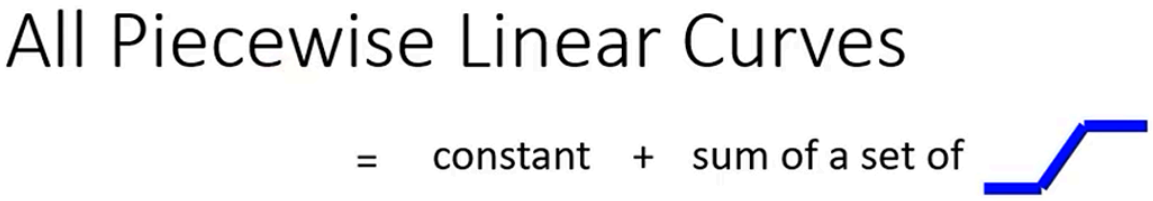
3. 更新w，b：$w^1 \leftarrow w^0 - \eta \frac{\partial L}{\partial w}\Big|_{w=w^0,b=b^0}$，$b^1 \leftarrow b^0 - \eta \frac{\partial L}{\partial b}\Big|_{w=w^0,b=b^0}$

以上步骤可以归结为训练

深度学习

Linear Model太过于简单，无法模拟折线曲线，也叫Model bias

所有的peacewise Linear curves（由多个锯齿状的线段组成的）可



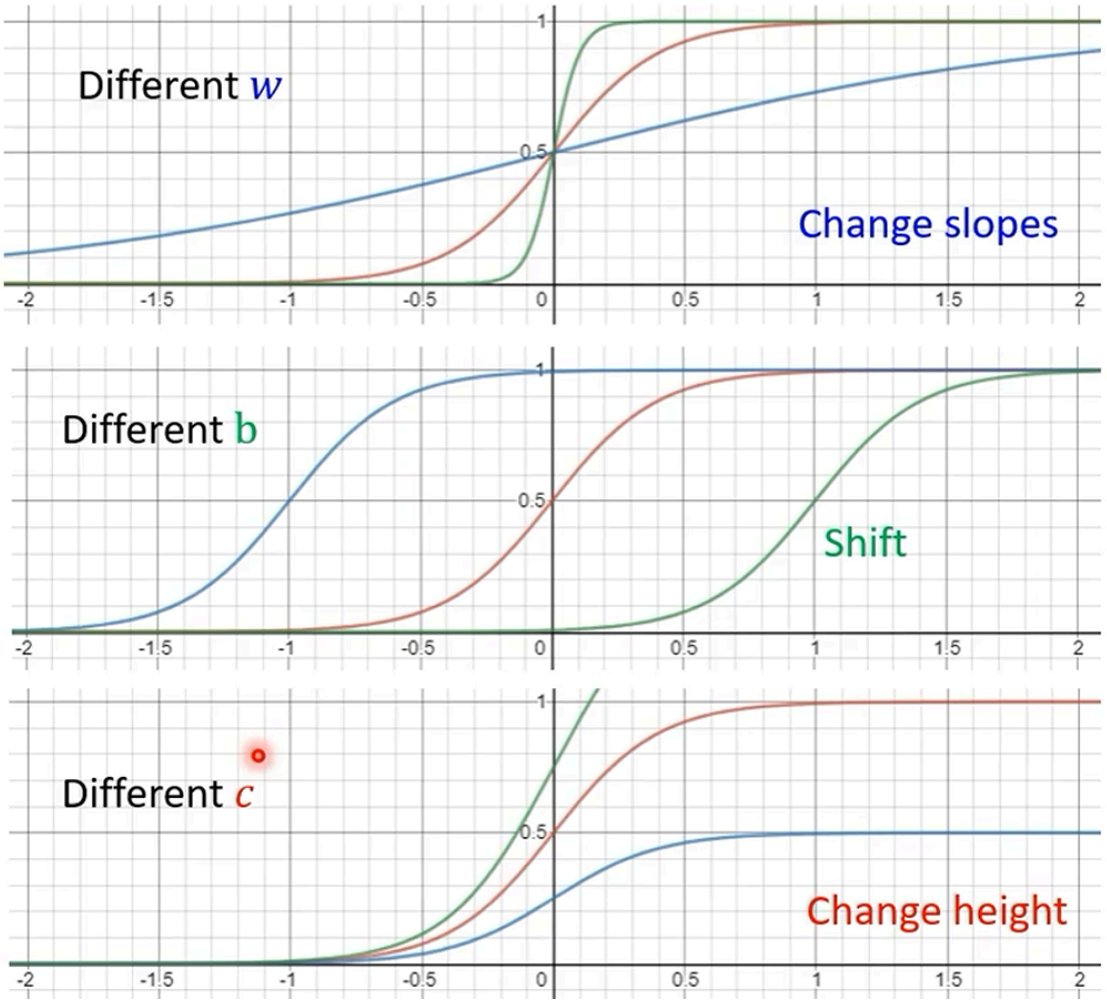
对于曲线，可在曲线上取一些点再连接起来，因此可以用peacewise Linear curves去逼近任意一条曲线，而peacewise Linear curves 可以用足够多的蓝色函数去表示。

▼ 如何写出蓝色function？→用sigmoid function 去逼近它

sigmoid函数：$y = c \cdot \frac{1}{1+e^{-(b+wx_1)}} = c \cdot \mathrm{sigmoid}(b + wx_1)$

当x1趋于无穷大，则y趋于c，x1趋于无穷小，y趋于0

由此上面蓝色函数为hard sigmoid



因此，我们现在就有了如下的公式

$$y = b + \sum_i c_i \, sigmoid\left(b_i + \sum_j w_{ij} x_j\right) \qquad i: 1,2,3 \quad j: 1,2,3$$

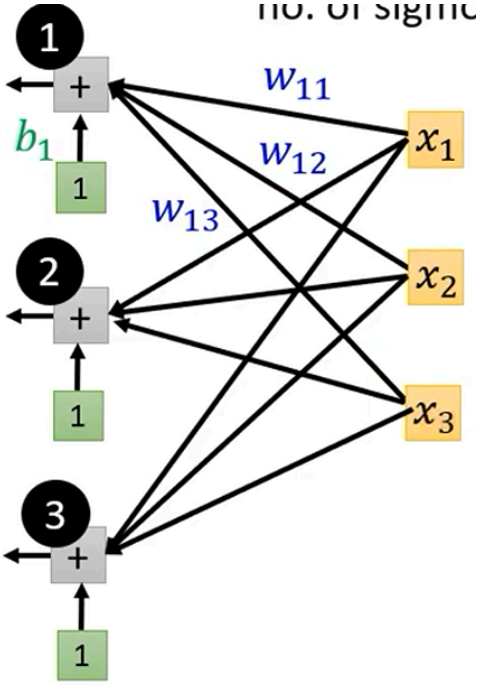
这里的i是代表第i段sigmoid函数，j代表不同的Feature，就是例子中的考虑前三天的播放量的前三天数值。

$r_1 = b_1 + w_{11}x_1 + w_{12}x_2 + w_{13}x_3$

w_{ij}: weight for x_j for i-th sigmoid

$r_2 = b_2 + w_{21}x_1 + w_{22}x_2 + w_{23}x_3$

$r_3 = b_3 + w_{31}x_1 + w_{32}x_2 + w_{33}x_3$

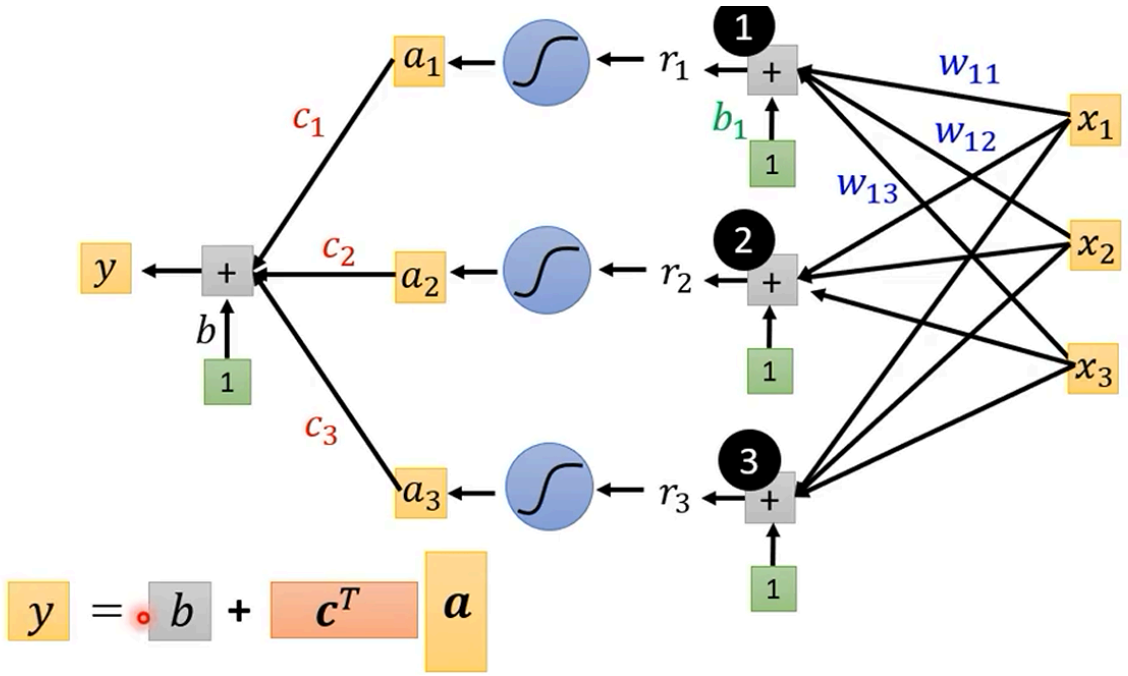


可以写为矩阵的形式：

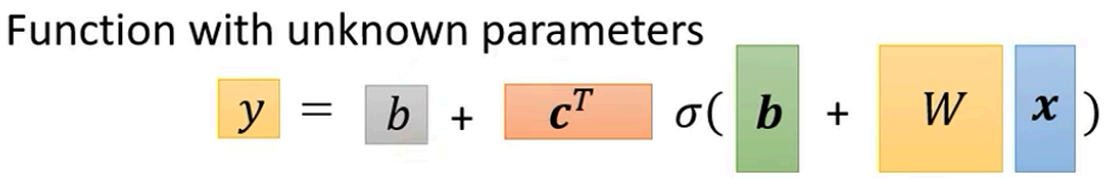
$$r_1 = b_1 + w_{11}x_1 + w_{12}x_2 + w_{13}x_3$$
$$r_2 = b_2 + w_{21}x_1 + w_{22}x_2 + w_{23}x_3$$
$$r_3 = b_3 + w_{31}x_1 + w_{32}x_2 + w_{33}x_3$$

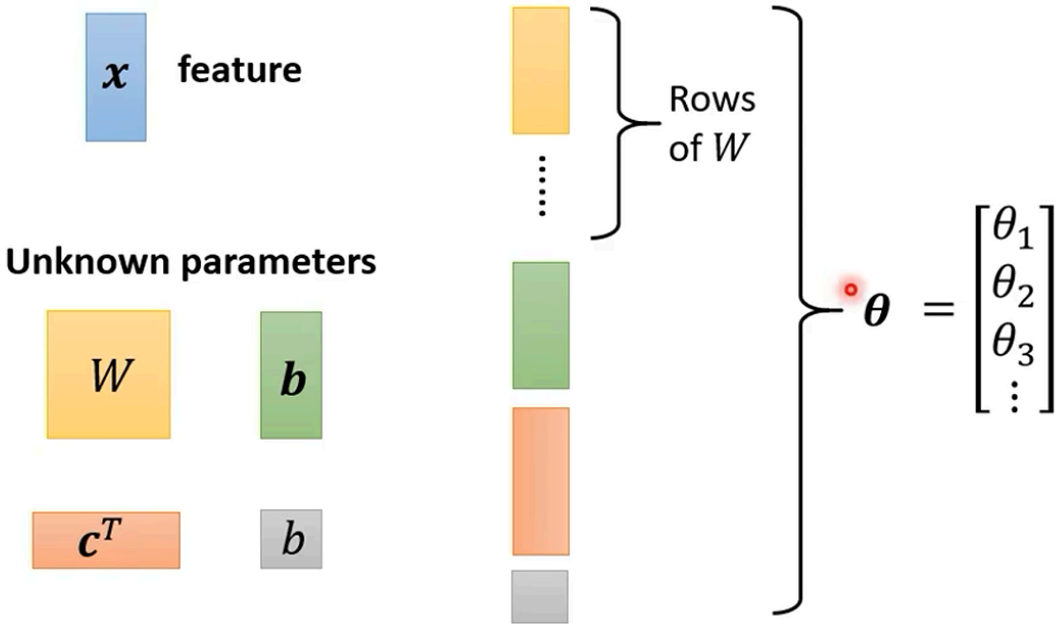
$$\begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} + \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

$$\boldsymbol{r} = \boldsymbol{b} + W\ \boldsymbol{x}$$



其实下图中的W展开可以按row，也可以按column





由此上面就改写了机器学习的第一步，就是定义一个有未知数的函数

具体用几个sigmoid函数，是一个hyperparameters。

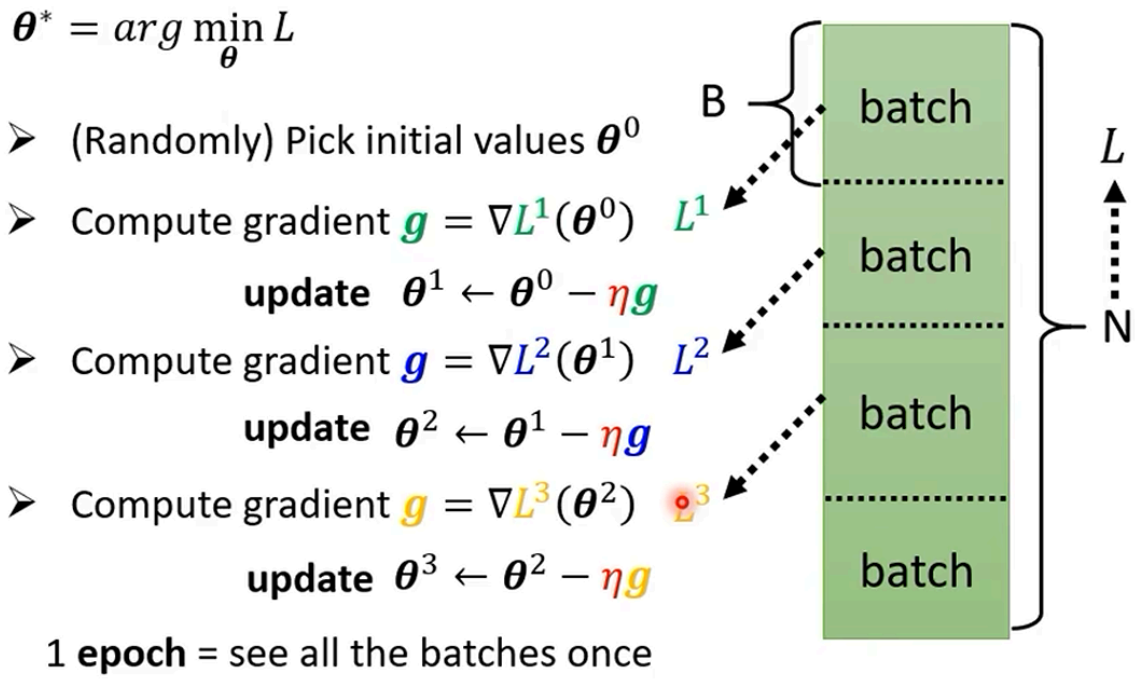
接下来定义Loss，L（θ），跟之前没什么区别

然后是优化，还是梯度下降

$$\boldsymbol{g}=\begin{bmatrix}\frac{\partial L}{\partial \theta_1}\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^0}\\ \frac{\partial L}{\partial \theta_2}\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^0}\\ \vdots\end{bmatrix} \qquad \boldsymbol{g}=\nabla L(\boldsymbol{\theta}^0)$$

$$\begin{bmatrix}\theta_1^1\\ \theta_2^1\\ \vdots\end{bmatrix} \leftarrow \begin{bmatrix}\theta_1^0\\ \theta_2^0\\ \vdots\end{bmatrix} - \begin{bmatrix}\eta\frac{\partial L}{\partial \theta_1}\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^0}\\ \eta\frac{\partial L}{\partial \theta_2}\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^0}\\ \vdots\end{bmatrix}$$

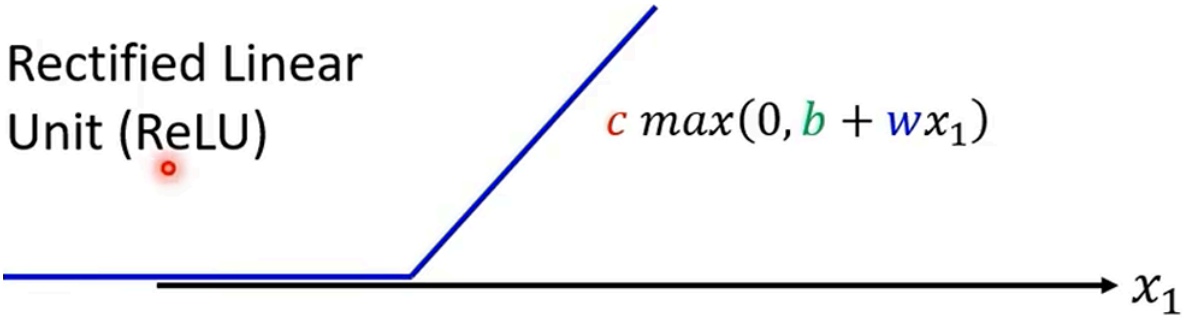
对于有大量数据data，总数为N，我们将其分为N/B个batch组，每一个batch中有B个数据，原本用所有数据计算Loss，现在只选取一个B组中的数据进行计算



每一次更新参数为update，并且batch的大小也是自己决定的。

对于多个数据，损失函数可以是所有的数据损失加起来，也可以取平均

hard sigmoid可以由俩个rectified Linear Unit（ReLU）组成



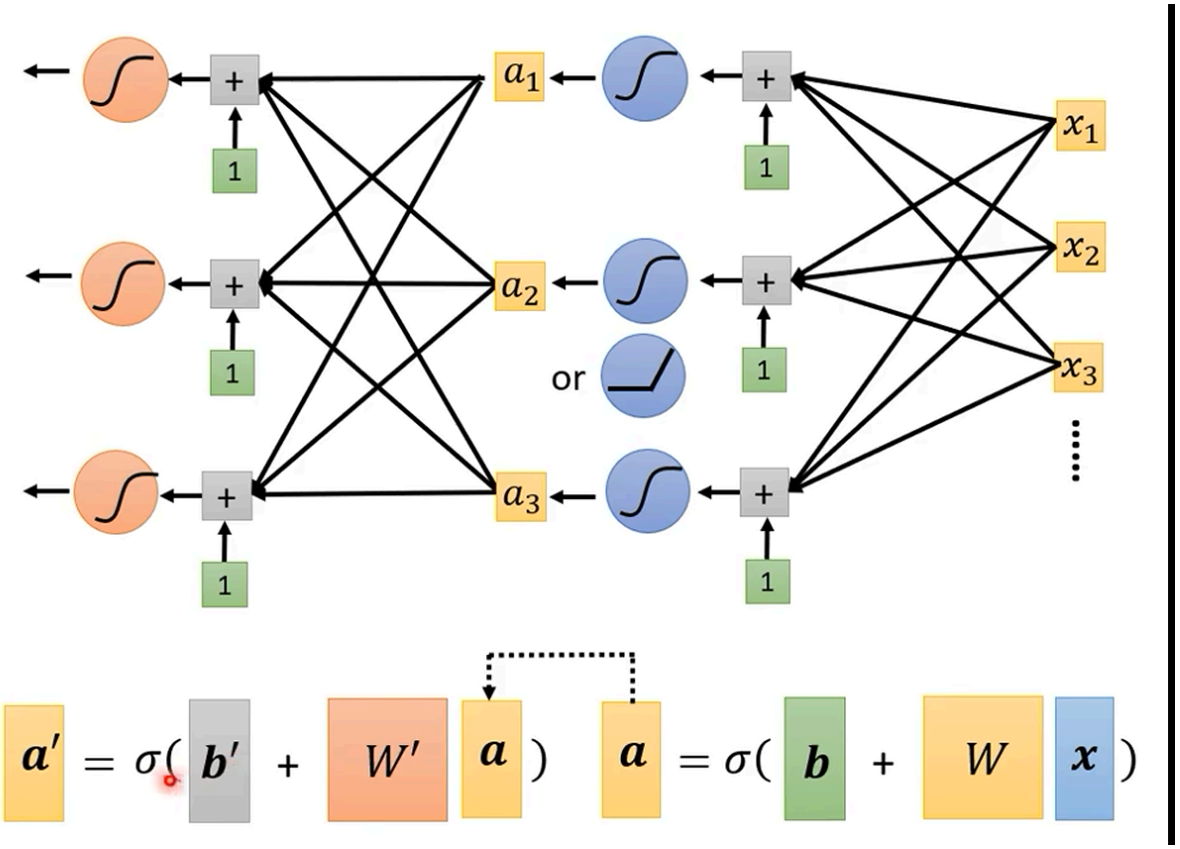
Sigmoid → ReLU

$$y = b + \sum_i c_i\ sigmoid\left(b_i + \sum_j w_{ij}x_j\right)$$

$$y = b + \sum_{2i} c_i\ max\left(0, b_i + \sum_j w_{ij}x_j\right)$$

注意由sigmoid变为ReLU需要将i变为2倍，这俩个函数统称为activation function（激活函数）。

进一步改进模型可以采用多层：



sigmoid或ReLU叫Neuron，很多Neuron就是Neuron Network，后面将每一层叫为hidden layer，很多layers就是 Deep learning

对于训练数据和测试数据机器表现不一样的情况为overfitting，指在训练数据上变好，但是测试数据上变差