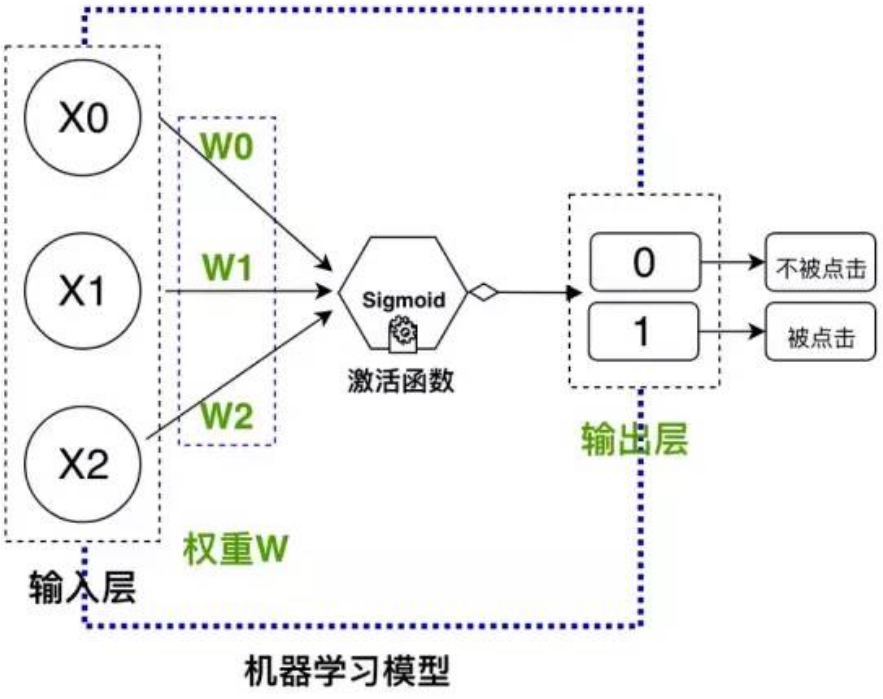
从零开始用Python搭建超级简单的点击率预估模型



点击率预估模型

本篇是一个基础机器学习入门篇文章，帮助我们熟悉机器学习中的神经网络结构与使用。

日常中习惯于使用Python各种成熟的机器学习工具包，例如sklearn、TensorFlow等等，来快速搭建各种各样的机器学习模型来解决各种业务问题。

本文将从零开始，仅仅利用基础的numpy库，使用Python实现一个最简单的神经网络(或者说是简易的LR，因为LR就是一个单层的神经网络)，解决一个点击率预估的问题。

声明：为了简单起见，下面的一切设定从简….

定义需要解决的问题：

老板：小李，这台机器上有一批微博的点击日志数据，你拿去分析一下，然后搞点击率预测啥的…

是的，就是预测一篇微博是否会被用户点击(被点击的概率)…..预测未来，貌似很神奇的样子！

热门微博

简单的介绍一下加深的业务数据

每一条微博数据有由三部分构成：{微博id, 微博特征X, 微博点击标志Y}

微博特征X有三个维度：

X={x0="该微博有娱乐明星”，x1="该微博有图”，x2="该微博有表情”}

微博是否被点击过的标志Y：

Y={y0=“点击”， y1=“未点击”}

数据有了，接下来需要设计一个模型，把数据输入进去进行训练之后，在预测阶段，只需要输入{微博id,微博特征X}，模型就会输出每一个微博id会被点击的概率。

这是一个有监督的机器学习任务

对于有监督的机器学习任务，可以简单的分为分类与回归问题，这里我们简单的想实现预测一条微博是否会被用户点击，预测目标是一个二值类别：点击，或者不点击，显然可以当做一个分类问题。

所以，我们需要搭建一个分类模型（点击率预测模型），这也就决定我们需要构建一个有监督学习的训练数据集。

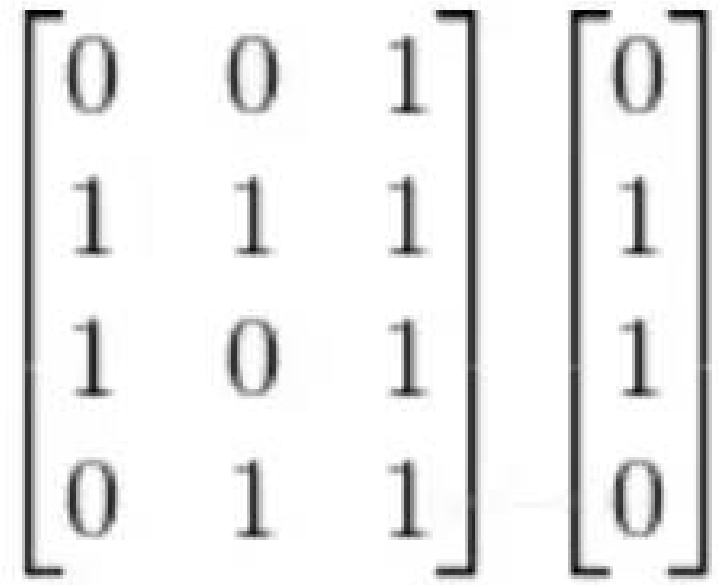
模型的选择

选择最简单神经网络模型，人工神经网络有几种不同类型的神经网络，比如前馈神经网络、卷积神经网络及递归神经网络等。本文将以简单的前馈或感知神经网络为例，这种类型的人工神经网络是直接从前到后传递数据的，简称前向传播过程。

整体的流程：

数据预处理(数值化编码)——>特征筛选——>选择模型(前馈神经网络)——>训练模型——>模型预测

假设，对4条微博的数据进行数值化编码，可以表示为如下的矩阵格式：



训练数据XY

解读一条样本数据：

第一条样本数据为：X0=[0 0 1]，分别对应着三维的特征，最后4x1的矩阵是Y，0表示无，1表示有，可知该特征对应的Y0是未点击。

所以，这条样本可以翻译为：[该微博没娱乐明星，没有图片，有表情]，最终y=0，代表该条微博没有被点击。

业务以及数据特征是不是很简单….简单有点看起来编的不太合理-！

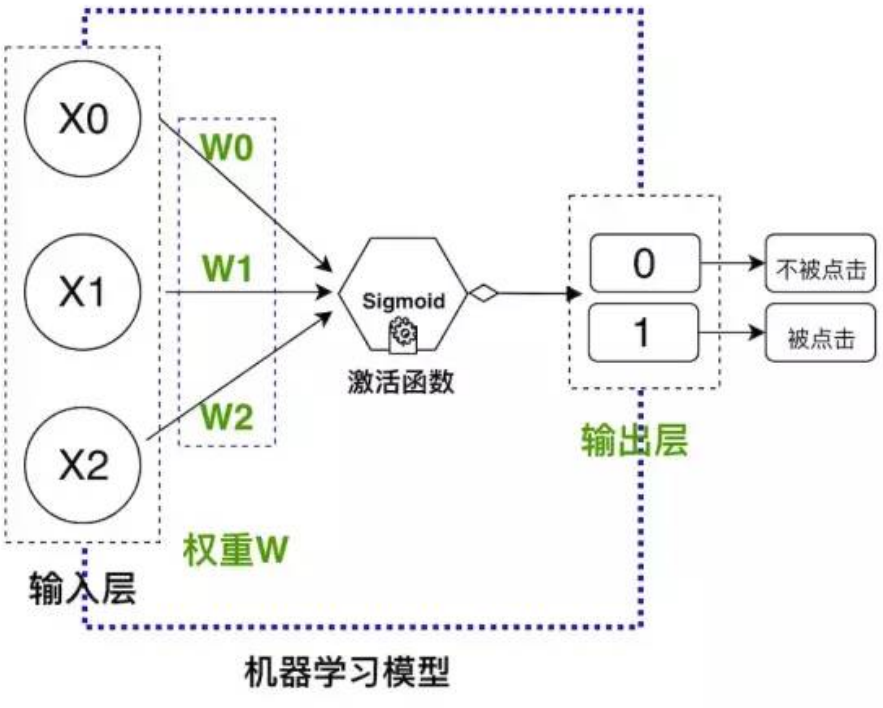
1.输入层：输入的业务特征数据

2.隐藏层：初始化权重参数

3.激活函数：选择激活函数

4.输出层：预测的目标，定义损失函数

我们即将使用的机器学习模型：



超级简单的前馈神经网络

机器学习模型类似一个黑盒子，输入历史点击的数据，进行训练，然后就可以对未来的额数据进行预测….我们上面设计的是一个超级简单的前馈神经网络，但是可以实现我们上面的目的。

关于激活函数：

通过引入激活函数，实现了非线性变换，增强了模型的拟合效果。

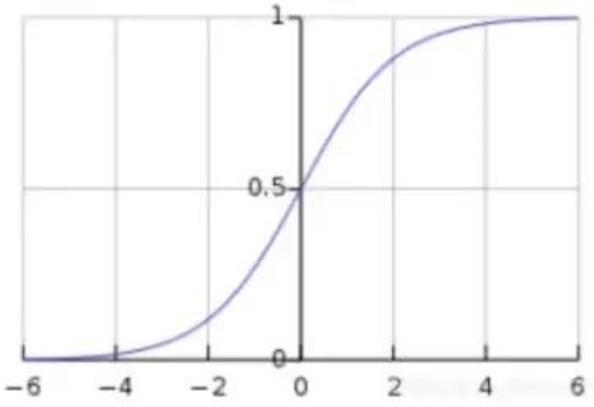
关乎激活函数，请看之前的文章吾爱NLP(2)--解析深度学习中的激活函数

在本文教程中，使用的是简单的Sigmoid激活函数，但注意一点，在深层神经网络模型中， sigmoid激活函数一般不作为首选，原因是其易发生梯度弥散现象。

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

sigmoid公式

此函数可以将任何值映射到0到1之间，并能帮助我们规范化输入的加权和。



sigmoid图像

对sigmoid激活函数求偏导：

$$
\begin{aligned}
g'(z) &= \frac{d}{dz}\frac{1}{1+e^{-z}} \\
&= \frac{1}{(1+e^{-z})^2}(e^{-z}) \\
&= \frac{1}{(1+e^{-z})}\cdot\left(1-\frac{1}{(1+e^{-z})}\right) \\
&= g(z)(1-g(z)).
\end{aligned}
$$

该偏导函数吗，等下写程序会用到，所以先放在这里！

训练阶段，模型的输入X已经确定，输出层的Y确定，机器学习模型确定，唯一需要求解的就是模型中的权重W，这就是训练阶段的目标。

主要由三个核心的流程构成：

前向计算—>计算损失函数—>反向传播

本文使用的模型是最简单的前馈神经网络，起始就是一个LR而已….所以整个过程这里就不继续介绍了，因为之前已经写过一篇关于LR的文章---逻辑回归（LR）个人学习总结篇，如果对其中的细节以及公式的推导有疑问，可以去LR文章里面去寻找答案。

这里再提一下权重参数W更新的公式：

$$\theta_j := \theta_j + \alpha \left(y^{(i)} - h_\theta(x^{(i)}) \right) x_j^{(i)}$$

至此，所有的写代码需要的细节都已经交代结束了，剩下的就是代码了。

```
1#  coding:utf-8
2import  numpy  as  np
3
4class  NeuralNetwork():
5#  随机初始化权重
6def  __init__(self):
7np.random.seed(1)
8self.synaptic_weights  =  2  *  np.random.random((3,  1))  -  1
9
10#  定义激活函数：这里使用sigmoid
11def  sigmoid(self,  x):
12return  1  /  (1  +  np.exp(-x))
13
14#计算Sigmoid函数的偏导数
15def  sigmoid_derivative(self,  x):
16return  x  *  (1  -  x)
17
18#  训练模型
```

```
19def train(self, training_inputs, training_outputs,learn_rate, training_iterations):
20#  迭代训练
21for iteration in range(training_iterations):
22        #前向计算
23        output = self.think(training_inputs)
24        #  计算误差
25        error = training_outputs - output
26        #  反向传播-BP-微调权重
27        adjustments = np.dot(training_inputs.T, error * self.sigmoid_derivative(output))
28        self.synaptic_weights += learn_rate*adjustments
29
30 def think(self, inputs):
31 #  输入通过网络得到输出
32 #  转化为浮点型数据类型
33 inputs = inputs.astype(float)
34 output = self.sigmoid(np.dot(inputs, self.synaptic_weights))
35return output
36
37if __name__ == "__main__":
38#  初始化前馈神经网络类
39neural_network = NeuralNetwork()
40print "随机初始化的权重矩阵W"
```

```
41print  neural_network.synaptic_weights
42#  模拟训练数据X
43train_data=[[0,0,1],  [1,1,1],  [1,0,1],  [0,1,1]]
44training_inputs  =  np.array(train_data)
45#  模拟训练数据Y
46training_outputs  =  np.array([[0,1,1,0]]).T
47#  定义模型的参数：
48#  参数学习率
49learn_rate=0.1
50#  模型迭代的次数
51epoch=150000
52neural_network.train(training_inputs,  training_outputs,  learn_rate,  epoch)
53print  "迭代计算之后权重矩阵W:  "
54print  neural_network.synaptic_weights
55#  模拟需要预测的数据X
56pre_data=[0,0,1]
57#  使用训练的模型预测该微博被点击的概率
58print  "该微博被点击的概率："
59print  neural_network.think(np.array(pre_data))
60
61"""
62终端输出的结果：
```

```
63随机初始化的权重矩阵W
64[[-0.16595599]
65[ 0.44064899]
66[-0.99977125]]
67迭代计算之后权重矩阵W:
68[[12.41691302]
69[-0.20410552]
70[-6.00463275]]
71该微博被点击的概率：
72[0.00246122]
73[Finished in 20.2s]
74"""
```

根据终端输出的模型训练以及预测的结果，针对预测数据pre_data=[0,0,1]，模型输出该微博被点击的概率为0.00246，很显然被点击的概率比较小，可以认为简单认为该微博不会被点击！

是的，我们的业务目标初步实现了----输入任意一条微博的样本数据到我们的机器学习模型中，既可以输出该样本被点击的概率。

上面的就是我们设计的一个超级简单的模型，假设了一个超级简单的业务场景，并随机设定了超简单的训练数据，如果有 编 的不合理地方多多包涵！！！该例子虽然可能并不能帮你解决实际的业务问题，但是对于机器学习的新手理解神经网络，或许会有一点点帮助吧！

转自：　　人工智能LeadAI