Neural Network

倒数第一层Softmax函数
倒数第二层输出Logit值

Mengdie Huang

November 8, 2019

Content

https://blog.csdn.net/lyl771857509/article/details/79428475

Introduction

Loss Function

损失函数：定义在single样本上，计算单个样本误差

Cost Function

代价函数：定义在the whole训练集上，计算Loss function的期望

Object Function

目标函数：最优化问题（maximize/minimize）对应的函数。

$Object\ Function$

$= Empirical\ Risk + Structural\ Risk$ (经验风险+结构风险)

$= Cost\ Function + Regularization$ (代价函数+正则化)

目标：用$f_1(x)$或$f_2(x)$或$f_3(x)$来拟合Price

分析：

经验风险：训练集all样本的平均Loss →

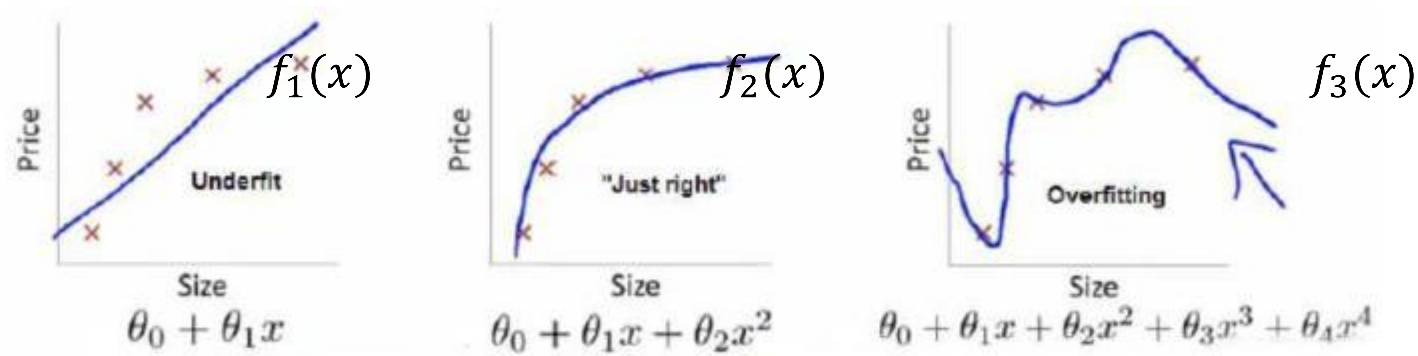
$$\min Empirical\ Risk = \min \frac{1}{N}\left(\sum_{i=1}^{N} L(y_i, f(x_i))\right)$$

$$f_1(x) > f_2(x) > f_3(x)$$

结构风险：度量$f(x)$的复杂度（即正则化）→

$$J(f) = L_1\ norm \text{ 或 } J(f) = L_2\ norm$$

$$f_1(x) < f_2(x) < f_3(x)$$



Content

https://www.cnblogs.com/zongfa/p/8971213.html
https://www.cnblogs.com/eczhou/p/7860483.html

Softmax Function计算公式

Function： Output of Neurons →map Probability in [0,1]

Vector： $Z=(Z_1,Z_2,\ldots,Z_i)$ 称 Network原始输出Z为置信度

$$Softmax(Z_i)=S_i=\frac{e^i}{\sum_j e^j} \qquad (e\approx 2.718)$$

Eg.

$$Z=(3,1,-3)$$

$$(e^{Z_1},e^{Z_2},e^{Z_3})=(e^3,e^1,e^{-3})\approx(20,2.7,0.05)$$

$$\sum_j e^j=\sum_{j=1}^{3}e^{Z_j}=e^{Z_1}+e^{Z_2}+e^{Z_3}=22.75$$

$$(S_1,S_2,S_3)=\left(\frac{e^{Z_1}}{\sum_j e^j},\frac{e^{Z_2}}{\sum_j e^j},\frac{e^{Z_3}}{\sum_j e^j}\right)$$

$$=\left(\frac{e^{Z_1}}{e^{Z_1}+e^{Z_2}+e^{Z_3}},\frac{e^{Z_2}}{e^{Z_1}+e^{Z_2}+e^{Z_3}},\frac{e^{Z_3}}{e^{Z_1}+e^{Z_2}+e^{Z_3}}\right)$$

$$=\left(\frac{20}{22.75},\frac{2.7}{22.75},\frac{0.05}{22.75}\right)$$

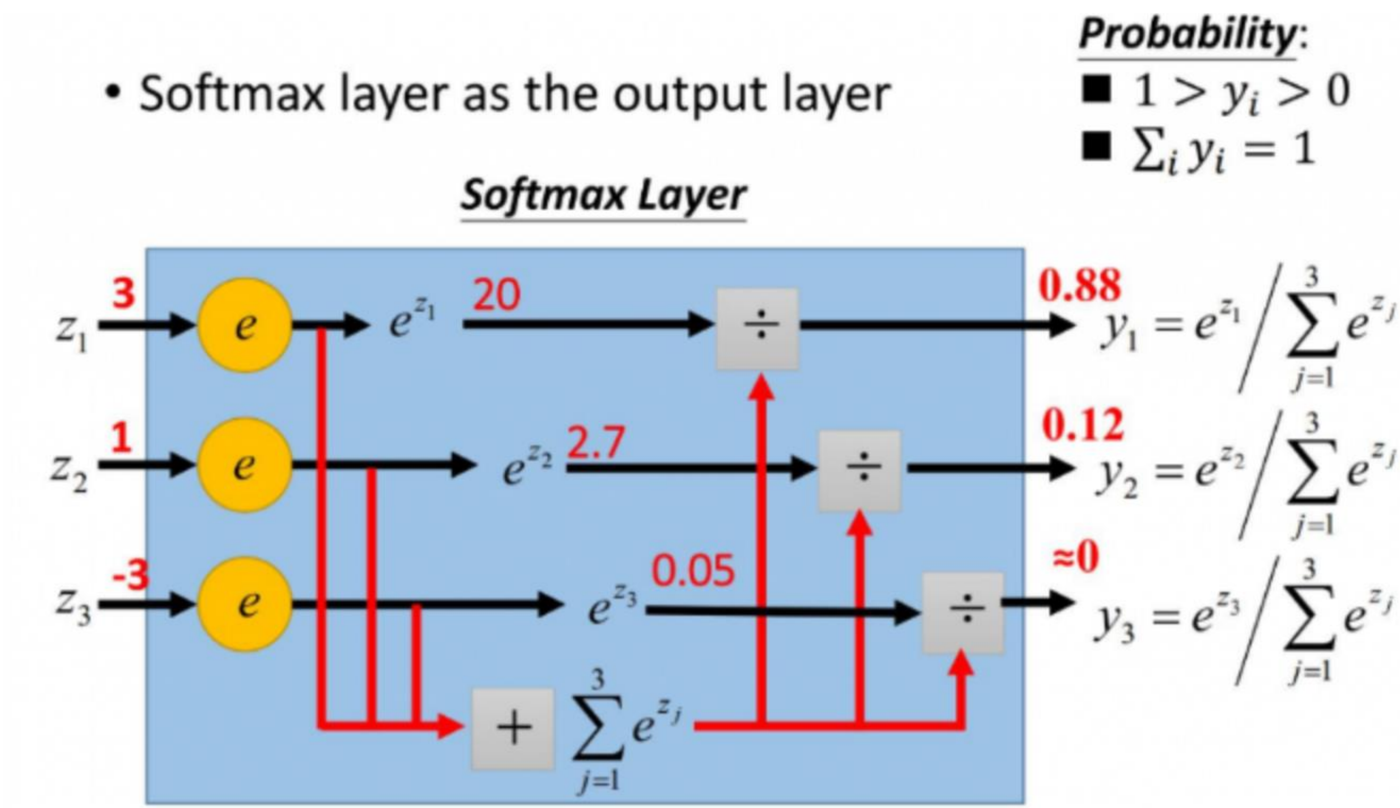
$$\approx(0.88,0.12,0.00)$$

Neural Network：最后一层选取输出预测label时，就选取概率最大的节点作为预测值输出。如果是多分类问题，需要选取n个节点输出时，就找概率最大的前n个值输出。

Softmax Function计算实例

Eg.

$$V = (3, 1, -3)$$
$$(S_1, S_2, S_3) \approx (0.88, 0.12, 0.00)$$



Softmax Function 求导

（1）Softmax作为神经网络的last layer，输出一组概率值，而Softmax层实际上不是神经元层，它不具有网络参数。

（2）Neural Network每一层的权重矩阵W的更新过程：

首先明确，权重更新是一个倒推链式过程！last layer → first layer，逐层用Loss Function对各节点的权重求偏导，执行更新。

Step 1　倒数第二层weight update

用整个模型的Loss Function对倒数第二层权重矩阵的每一个节点权重求偏导数，

$$w_k{}' = w_k - \lambda \cdot \frac{\partial(Loss)}{\partial(w_k)}$$

Step 2　链式法则 $\frac{\partial(Loss)}{\partial(w_k)} = \frac{\partial(Loss)}{\partial(y)} \cdot \frac{\partial(y)}{\partial(w_k)}$

由倒数第二层的权重矩阵倒推更新倒数第三层的权重矩阵。

（3）为什么要用loss对每个权重矩阵求偏导数？

在用梯度下降法对Loss进行改善时，每次优化一个步长的梯度，就需要进行梯度求导。

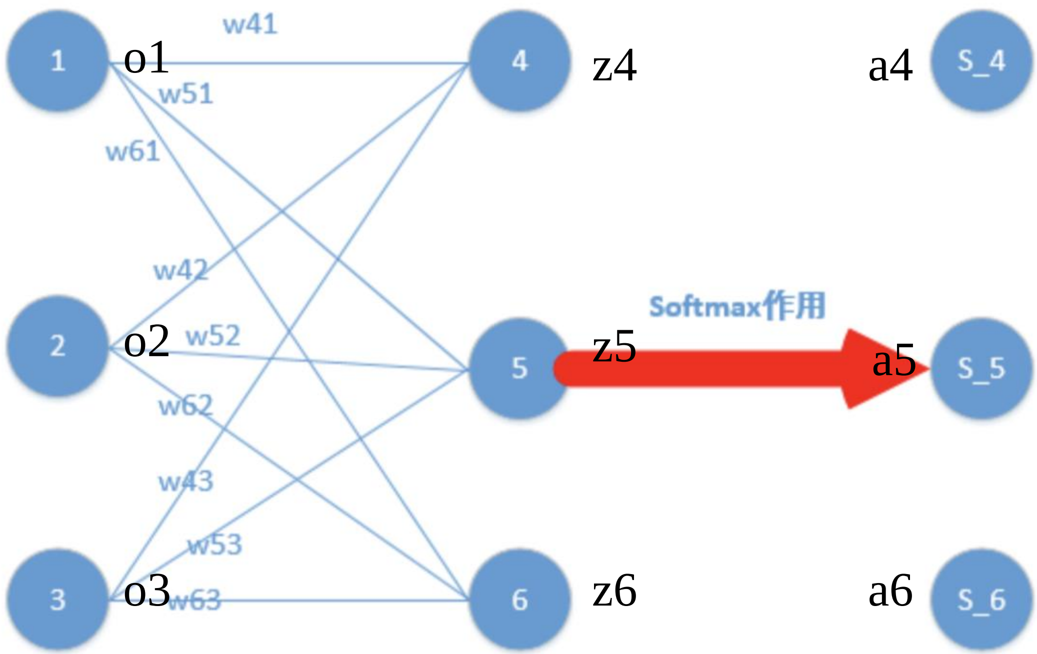
Softmax Function 相关求导案例

Step 1　倒数第三层→倒数第二层

已知节点输出o1、o2、o3，模型参数：

$w_{41}, w_{42}, w_{43}, w_{51}, w_{52}, w_{53}, w_{61}, w_{62}, w_{63}$

求z4、z5、z6？



z4 = w41*o1+w42*o2+w43*o3

z5 = w51*o1+w52*o2+w53*o3

z6 = w61*o1+w62*o2+w63*o3

$$a_4 = \frac{e^{z4}}{z^{z4} + z^{z5} + z^{z6}}$$

$$a_5 = \frac{e^{z5}}{z^{z4} + z^{z5} + z^{z6}}$$

$$a_6 = \frac{e^{z6}}{z^{z4} + z^{z5} + z^{z6}}$$

Step 2　倒数第二层→倒数第一层(Softmax)

$$Softmax(z_4, z_5, z_6) = (a_4, a_5, a_6)$$

Softmax Function 相关求导案例

Step 3　使用交叉熵作为Loss Function

计算预测的概率分布和真实答案的概率分布之间的距离

i：节点标号。　y_i：真实值。　a_i：Softmax值。

$$Softmax(z_4, z_5, z_6) = (a_4, a_5, a_6)$$

$$Loss = -\sum_i y_i ln a_i = -(y_4 ln a_4 + y_5 ln a_5 + y_6 ln a_6)$$

Step 4　二分类问题，只预测一个结果，(a_4, a_5, a_6)中只有一个元素a_j真实值y_j为1，其余都为0

设　$a_j = 1, j = 4,5,6$中的一个

则　$Loss = -y_j ln a_j = -ln a_j$

Step 5　设目标：对权重w_{41}求偏导数

a.损失函数求偏导数传到节点$i = 4$：　$\frac{\partial(Loss)}{\partial(a_4)}$

b.链式法则对权重w_{41}求偏导数：

$$\frac{\partial(Loss)}{\partial(w_{41})} = \frac{\partial(Loss)}{\partial(a_j)} \cdot \frac{\partial(a_j)}{\partial(z_4)} \cdot \frac{\partial(z_4)}{\partial(w_{41})} = -(\frac{1}{a_j}) \cdot \frac{\partial(a_j)}{\partial(z_4)} \cdot o1$$

$o1$已知，关键在求$\frac{\partial(a_j)}{\partial(z_4)}$！

Softmax Function 相关求导案例

Step 6

目标：求 $\frac{\partial(a_j)}{\partial(z_4)}$

解答：分情况讨论

（1）$j=i=4$，表明a_4对应真实值

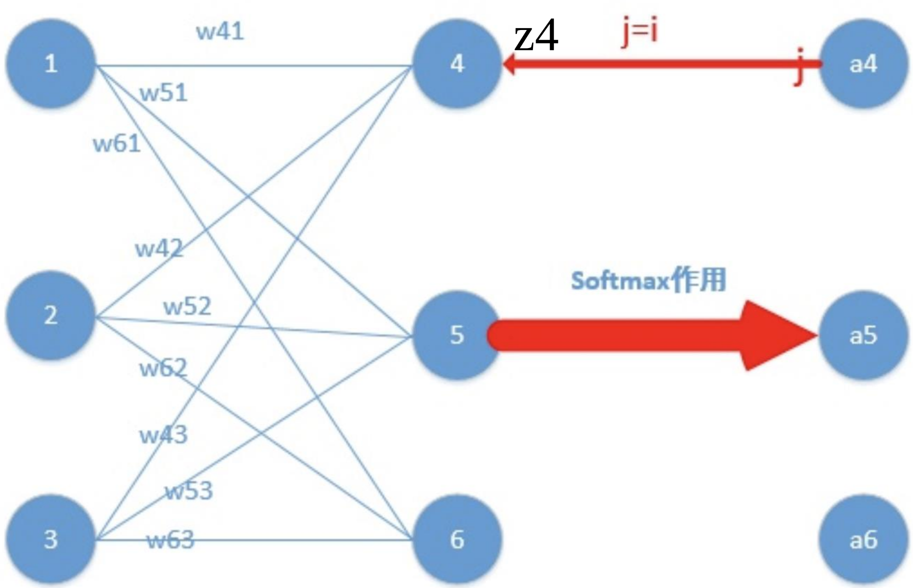
则 $\frac{\partial(a_j)}{\partial(z_i)}=a_j(1-a_j)$

if $j=i$:

$$\frac{\partial a_j}{\partial z_i}=\frac{\partial}{\partial z_i}\left(\frac{e^{z_j}}{\sum_k e^{z_k}}\right)$$

$$=\frac{(e^{z_j})'\cdot\sum_k e^{z_k}-e^{z_j}\cdot e^{z_j}}{\left(\sum_k e^{z_k}\right)^2}$$

$$=\frac{e^{z_j}}{\sum_k e^{z_k}}-\frac{e^{z_j}}{\sum_k e^{z_k}}\cdot\frac{e^{z_j}}{\sum_k e^{z_k}}=a_j(1-a_j)$$



Step 7　权重梯度

$$\frac{\partial(Loss)}{\partial(w_{41})}=-\left(\frac{1}{a_j}\right)\cdot\frac{\partial(a_j)}{\partial(z_i)}\cdot o1$$

$$=-\left(\frac{1}{a_j}\right)\cdot\frac{\partial(a_j)}{\partial(z_4)}\cdot o1$$

$$=-\left(\frac{1}{a_j}\right)\cdot a_j(1-a_j)\cdot o1$$

$$=(a_j-1)\cdot o1=(a_4-1)\cdot o1$$

Softmax Function 相关求导案例

Step 6

目标：求 $\frac{\partial(a_j)}{\partial(z_4)}$

解答：分情况讨论

（2）$j \neq i = 4$，表明a_4不对应真实值

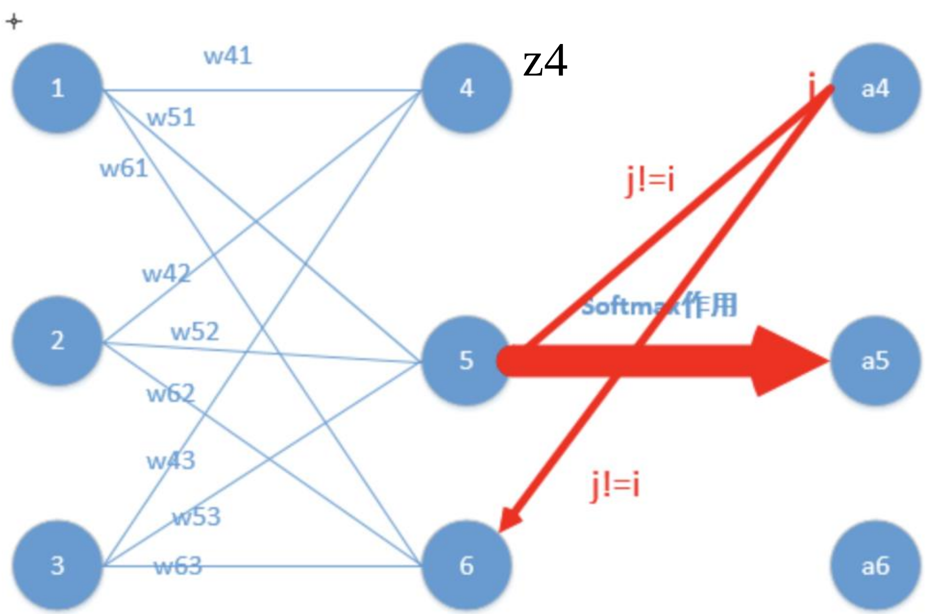
则 $\frac{\partial(a_j)}{\partial(z_i)} = -a_j a_i$

if $j \neq i$:

$$\frac{\partial a_j}{\partial z_i} = \frac{\partial}{\partial z_i}\left(\frac{e^{z_j}}{\sum_k e^{z_k}}\right)$$

$$= \frac{0 \cdot \sum_k e^{z_k} - e^{z_j} \cdot e^{z_i}}{\left(\sum_k e^{z_k}\right)^2}$$

$$= -\frac{e^{z_j}}{\sum_k e^{z_k}} \cdot \frac{e^{z_i}}{\sum_k e^{z_k}} = -a_j a_i$$



Step 7　　权重梯度

$$\frac{\partial(Loss)}{\partial(w_{41})} = -\left(\frac{1}{a_j}\right) \cdot \frac{\partial(a_j)}{\partial(z_i)} \cdot o1$$

$$= -\left(\frac{1}{a_j}\right) \cdot \frac{\partial(a_j)}{\partial(z_4)} \cdot o1$$

$$= -\left(\frac{1}{a_j}\right) \cdot (-a_j a_i) \cdot o1$$

$$= -\left(\frac{1}{a_j}\right) \cdot (-a_j a_4) \cdot o1 = a_4 \cdot o1$$

Softmax Function 求导计算举例

已知：某个训练样本的分类分数向量，该样本的真实分类是第二个

$$V = (z_1, z_2, z_3) = (2,3,4)$$

求偏导数？

解：

$$\begin{aligned} Softmax(z_1, z_2, z_3) \\ &= (a_1, a_2, a_3) \\ &= \left(\frac{e^{z_1}}{e^{z_1}+e^{z_2}+e^{z_3}}, \frac{e^{z_2}}{e^{z_1}+e^{z_2}+e^{z_3}}, \frac{e^{z_3}}{e^{z_1}+e^{z_2}+e^{z_3}}\right) \\ &= (0.0903,0.2447,0.665) \end{aligned}$$

则偏导数可快速求得：

$$\begin{aligned} \left(\frac{\partial(Loss)}{\partial(a_1)}, \frac{\partial(Loss)}{\partial(a_2)}, \frac{\partial(Loss)}{\partial(a_3)}\right) \\ &= (a_1, a_2 - 1, a_3) \\ &= (0.0903,0.2447 - 1,0.665) \\ &= (0.0903, -0.7553,0.665) \end{aligned}$$

Softmax

Apply Softmax Function in Neural Network

解决多分类问题方法：设置n个（类别数目）输出节点

Neural Network Input: 样例

Neural Network output: n维数组（每一维度表示一个类别）

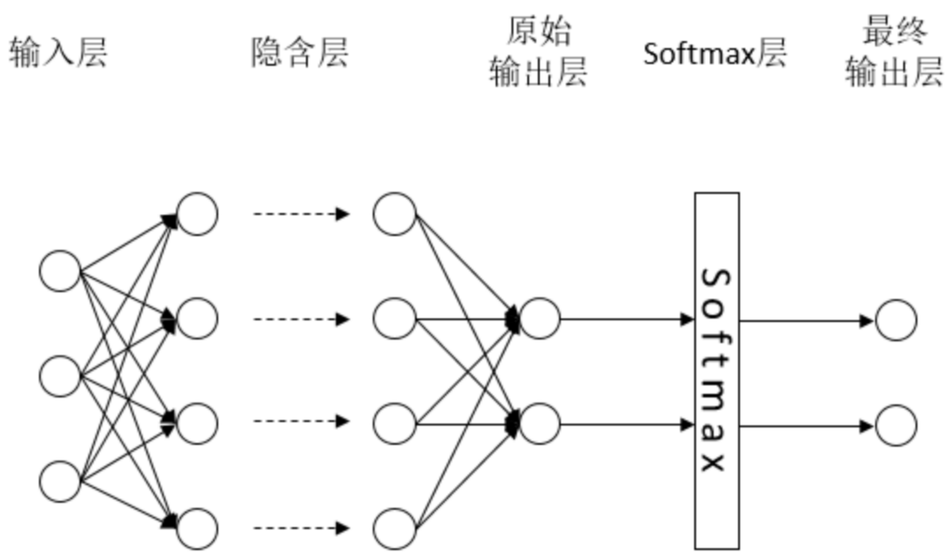
(1) 前向传播算法得到每个维度值

维度值含义：样例属于该类别的概率大小。

$\sum_n$ 维度值=1

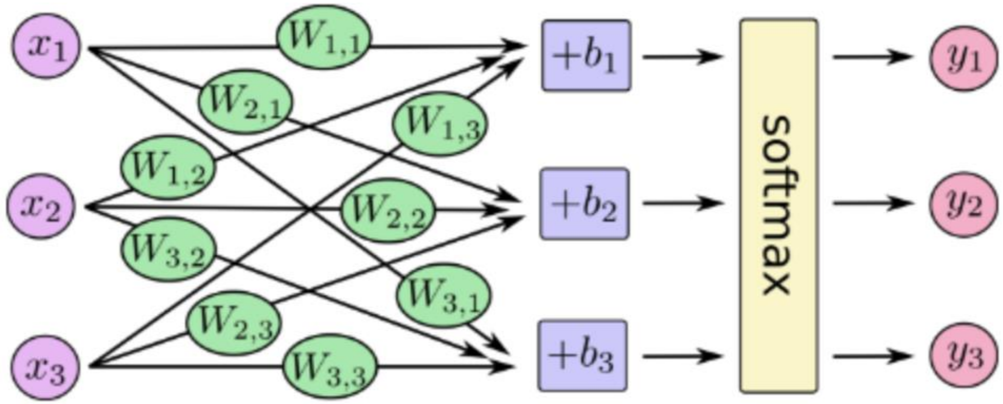
(2) 概率事件：一个样例属于某一个类别，则样例的分类符合一个概率分布。

(3) n维分类→Softmax回归→概率分布



Softmax回归的图解

Step 1　Softmax在Network中的功能位置



Step 2　过程抽象成等式

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \text{softmax}\begin{pmatrix} W_{1,1}x_1 + W_{1,2}x_2 + W_{1,3}x_3 + b_1 \\ W_{2,1}x_1 + W_{2,2}x_2 + W_{2,3}x_3 + b_2 \\ W_{3,1}x_1 + W_{3,2}x_2 + W_{3,3}x_3 + b_3 \end{pmatrix}$$

Step 3　等式向量化，加速计算

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \text{softmax}\left(\begin{bmatrix} W_{1,1} & W_{1,2} & W_{1,3} \\ W_{2,1} & W_{2,2} & W_{2,3} \\ W_{3,1} & W_{3,2} & W_{3,3} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}\right)$$

Content

https://www.jianshu.com/p/96f762d317b9

https://www.jianshu.com/p/498f7bf488a2

https://www.jianshu.com/p/d93dec4f4a46

Logit Model

一、线性回归Model的局限性

Requirement：因变量是定量变量，如定距vector、定比vector；不能是定性变量，如定序vector、定类vector。

二、实际情况

Real World：经常出现因变量是定性变量，比如分类变量。

三、一元线性回归

假设神经元数目：1　　　　　　　　神经元output：　0或1

神经元learning problem → Binary Logistic Regression（二值逻辑回归）

三、一元线性回归

一元线性回归的数学模型抽象：

Step 1　Number of data point：m

Step 2　Data point：$\left(x^{(1)},y^{(1)}\right)\left(x^{(2)},y^{(2)}\right)\ldots\left(x^{(m)},y^{(m)}\right)$　|即$\left(x^{(i)},y^{(i)}\right)$,

整数$i\in[1,m]$

Step 3　目标：找出一条直线$y=\theta x+b$，即求得一对(θ,b)，

满足all data points到该线的距离和最小。

三、一元线性回归

一元线性回归的数学模型抽象：

Step 4　代价函数(Cost Function)：

$$J(\theta,b)=\frac{1}{2}\sum_{1}^{m}(\text{第}i\text{个预测值}-\text{第}i\text{个真实值})^2$$

$$=\frac{1}{2}\sum_{1}^{m}\left(\left(\theta x^{(i)}+b\right)-y^{(i)}\right)^2$$

$$=\frac{\left(\left(\theta x^{(1)}+b\right)-y^{(1)}\right)^2+\left(\left(\theta x^{(2)}+b\right)-y^{(2)}\right)^2+\cdots+\left(\left(\theta x^{(m)}+b\right)-y^{(m)}\right)^2}{2}$$

一元线性回归→最优化问题（找到一条最优直线）

Step 5　求达到 $\min J(\theta,b)$的θ和b值

方法（1）：建立$J(\theta,b)$关于变量θ和b的方程组

方法（2）：梯度下降法求最佳θ和b

$$\begin{cases}\dfrac{\partial J(\theta,b)}{\partial\theta}=0\\[2ex]\dfrac{\partial J(\theta,b)}{\partial b}=0\end{cases}$$

三、一元线性回归

一元线性回归的数学模型抽象：

梯度下降法求最佳θ和b

Step 1　设定初始值 θ_0，b_0，学习率 α

Step 2　开始递归

$$\begin{cases} \theta_{\text{update}} = \theta - \alpha \cdot \dfrac{\partial J(\theta, b)}{\partial \theta} \\ b_{\text{update}} = b - \alpha \cdot \dfrac{\partial J(\theta, b)}{\partial b} \end{cases}$$

.

.

.

Step n　递归结束，return 最佳点(θ, b)

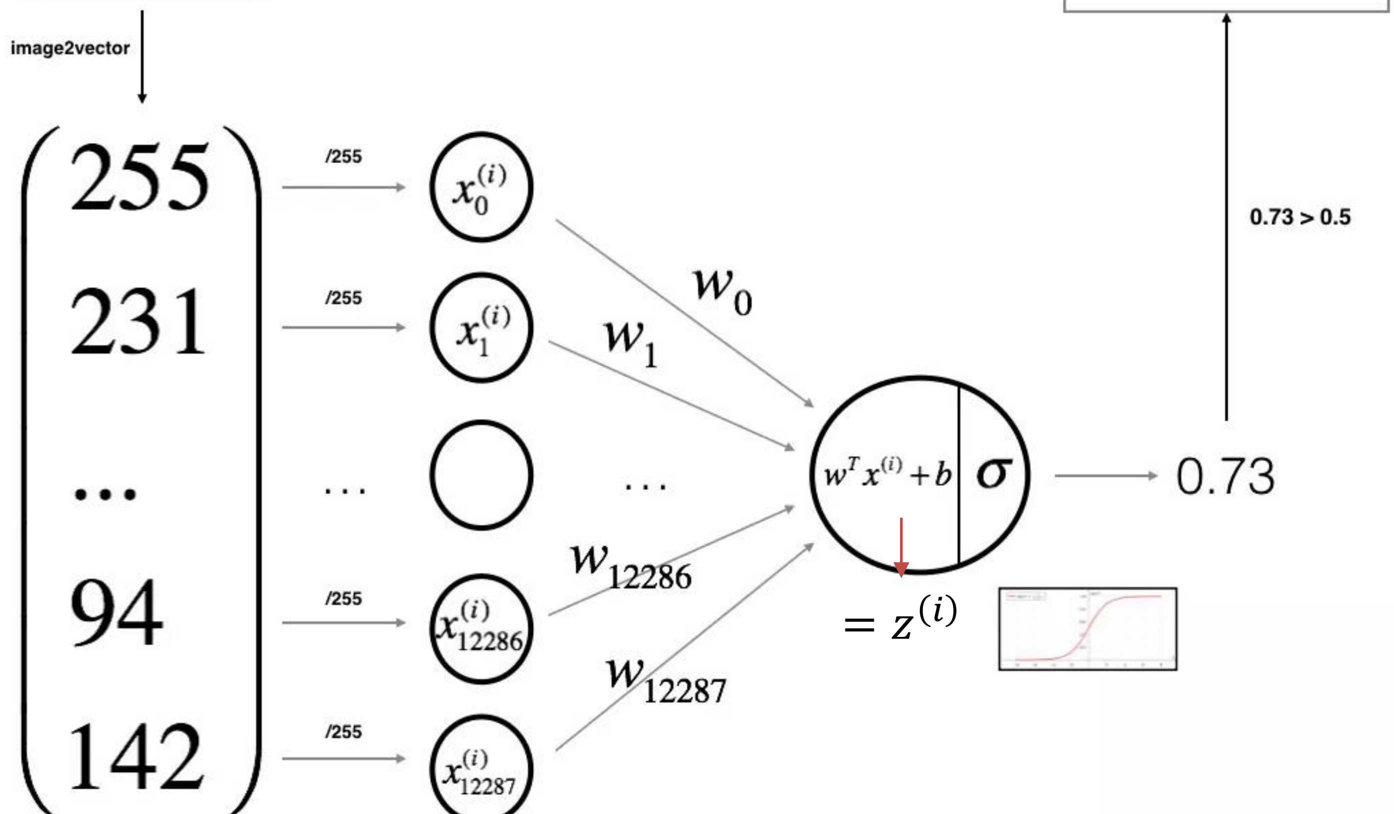
四、反向传播推导

A Neuron Input: 像素值

A Neuron Output: 0←判断是猫

1←判断不是猫

“it’s a cat”



四、反向传播推导

前向传播公式：How to compute cost function

Input: $x^{(i)} = (x_0^{(i)}, x_1^{(i)}, \ldots, x_{12887}^{(i)})^T$ |有12888 pixels

目标： 找到训练参数$w = (w_0, w_1, \ldots, w_{12887})^T$（多维）和$b$

多元线性回归→最优化问题（找到一个最优超平面）

Output:

Step 1 多元线性回归（神经网络节点的前半段计算）

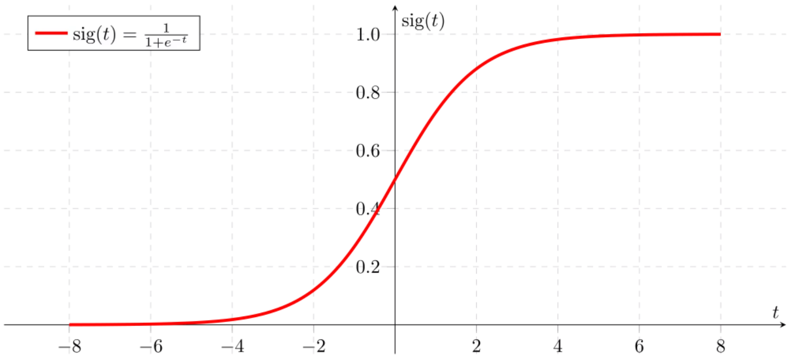
$$z^{(i)} = w^T x^{(i)} + b$$

$$= \begin{bmatrix} w_0 & w_1 & \ldots & w_{12887} \end{bmatrix} \begin{bmatrix} x_0^{(i)} \\ x_1^{(i)} \\ \ldots \\ x_{12887}^{(i)} \end{bmatrix} + b$$

$$= w_0 x_0^{(i)} + w_1 x_1^{(i)} + \cdots + w_{12887} x_{12887}^{(i)} + b$$

四、反向传播推导

前向传播公式：How to compute cost function



Output:

Step 2　将$z^{(i)}$变换到（0，1）

$$\hat{y}^{(i)} = a^{(i)} = sigmoid(z^{(i)})$$

注：$\hat{y}$(cap)表示y上有一条折线，是回归估计的预测值。

$$a^{(i)} = sigmoid\left(z^{(i)}\right) = \frac{1}{1+e^{-z^{(i)}}} = \frac{1}{1+e^{-\left(w_0x_0^{(i)}+w_1x_1^{(i)}+\cdots+w_{12887}x_{12887}^{(i)}+b\right)}}$$

Step 3　计算m个样本的总代价函数(即 $J(w,b)$)

求平均

Loss函数：$L\left(a^{(i)},y^{(i)}\right) = -\left[y^{(i)}\log a^{(i)}\right] - \left[\left(1-y^{(i)}\right)\log\left(1-a^{(i)}\right)\right]$

Cost函数：$J(w,b) = \frac{1}{m}\sum_1^m L(\text{预测值向量,真实值向量}) = \frac{1}{m}\sum_1^m L\left(a^{(i)},y^{(i)}\right)$

四、反向传播推导

反向传播公式：即对cost function 的变量求偏导，用链式求导。

已知：

$$J(w,b)=\frac{1}{m}\sum_{1}^{m}L(a^{(i)},y^{(i)})$$

求导数= $\frac{1}{a^{(i)}}$ ？

$$L\left(a^{(i)},y^{(i)}\right)=-(y^{(i)}\log a^{(i)})-\left(1-y^{(i)}\right)\log\left(1-a^{(i)}\right)$$

$$\hat{y}^{(i)}=a^{(i)}=sigmoid\left(z^{(i)}\right)=\frac{1}{1+e^{-z^{(i)}}}$$

$$z^{(i)}=w^{T}x^{(i)}+b$$

求解：

$$\begin{cases}\frac{\partial J}{\partial w}=\frac{\partial J}{\partial z}\cdot\frac{\partial z}{\partial w}\\ \frac{\partial J}{\partial b}=\frac{\partial J}{\partial z}\cdot\frac{\partial z}{\partial w}\end{cases}$$

四、反向传播推导

求解：

Step 1 $\frac{\partial z}{\partial w} = x^{(i)}$ $\frac{\partial z}{\partial b} = 1$

Step 2 $\frac{\partial J}{\partial z} = \frac{\partial J}{\partial L} \cdot \frac{\partial L}{\partial a} \cdot \frac{\partial a}{\partial z} = (\frac{1}{m}\sum_1^m \frac{\partial L}{\partial a}) \cdot \frac{\partial a}{\partial z}$

$$\frac{\partial L}{\partial a} = ? \quad = -\frac{y^{(i)}}{a^{(i)}} + \frac{1-y^{(i)}}{1-a^{(i)}}$$

$$\frac{\partial a}{\partial z} = ? \quad = \frac{1}{1+e^{-z^{(i)}}} \cdot \left(1 - \frac{1}{1+e^{-z^{(i)}}}\right)$$

sigmod function：

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

$$\frac{\partial f(z)}{\partial z} = \frac{1}{1+e^{-z}} \cdot \left(1 - \frac{1}{1+e^{-z}}\right) = f(z)(1 - f(z))$$

四、反向传播推导

求解：

Step 1 $\frac{\partial z}{\partial w} = x^{(i)}$ $\frac{\partial z}{\partial b} = 1$

Step 2 $\frac{\partial J}{\partial z} = \frac{\partial J}{\partial L} \cdot \frac{\partial L}{\partial a} \cdot \frac{\partial a}{\partial z} = (\frac{1}{m}\sum_1^m \frac{\partial L}{\partial a}) \cdot \frac{\partial a}{\partial z}$

$$\frac{\partial J}{\partial z} = \left(\frac{1}{m}\sum_1^m -\frac{y^{(i)}}{a^{(i)}} + \frac{1-y^{(i)}}{1-a^{(i)}}\right) \cdot \frac{1}{1+e^{-z^{(i)}}} \cdot \left(1 - \frac{1}{1+e^{-z^{(i)}}}\right)$$

$$= \left(\frac{1}{m}\sum_1^m -\frac{y^{(i)}}{a^{(i)}} + \frac{1-y^{(i)}}{1-a^{(i)}}\right) \cdot \frac{1}{1+e^{-z^{(i)}}} \cdot \left(1 - \frac{1}{1+e^{-z^{(i)}}}\right)$$

$$= \left(\frac{1}{m}\sum_1^m -\frac{y^{(i)}}{a^{(i)}} + \frac{1-y^{(i)}}{1-a^{(i)}}\right) \cdot a^{(i)} \cdot (1 - a^{(i)})$$

$$= \frac{1}{m}\sum_1^m \frac{1-y^{(i)}}{a^{(i)}} - \frac{y^{(i)}}{1-a^{(i)}} = \frac{1}{m}\sum_1^m (a^{(i)} - y^{(i)})\,?$$

四、反向传播推导

求解：

Step 1 $\frac{\partial z}{\partial w} = x^{(i)}$ $\frac{\partial z}{\partial b} = 1$

Step 2 $\frac{\partial J}{\partial z} = \frac{1}{m}\sum_1^m (a^{(i)} - y^{(i)})$

Step 3 $\frac{\partial J}{\partial w} = \frac{\partial J}{\partial z} \cdot \frac{\partial z}{\partial w} = \frac{1}{m}\sum_1^m \left(a^{(i)} - y^{(i)}\right) \cdot x^{(i)} = \frac{1}{m} X (A - Y)^T ?$

$$\frac{\partial J}{\partial b} = \frac{\partial J}{\partial z} \cdot \frac{\partial z}{\partial w} = \frac{1}{m}\sum_1^m (a^{(i)} - y^{(i)})$$

可以看出，$J(w, b)$对w, b求的偏导也是各样本点平均值的形式

五、损失函数推导

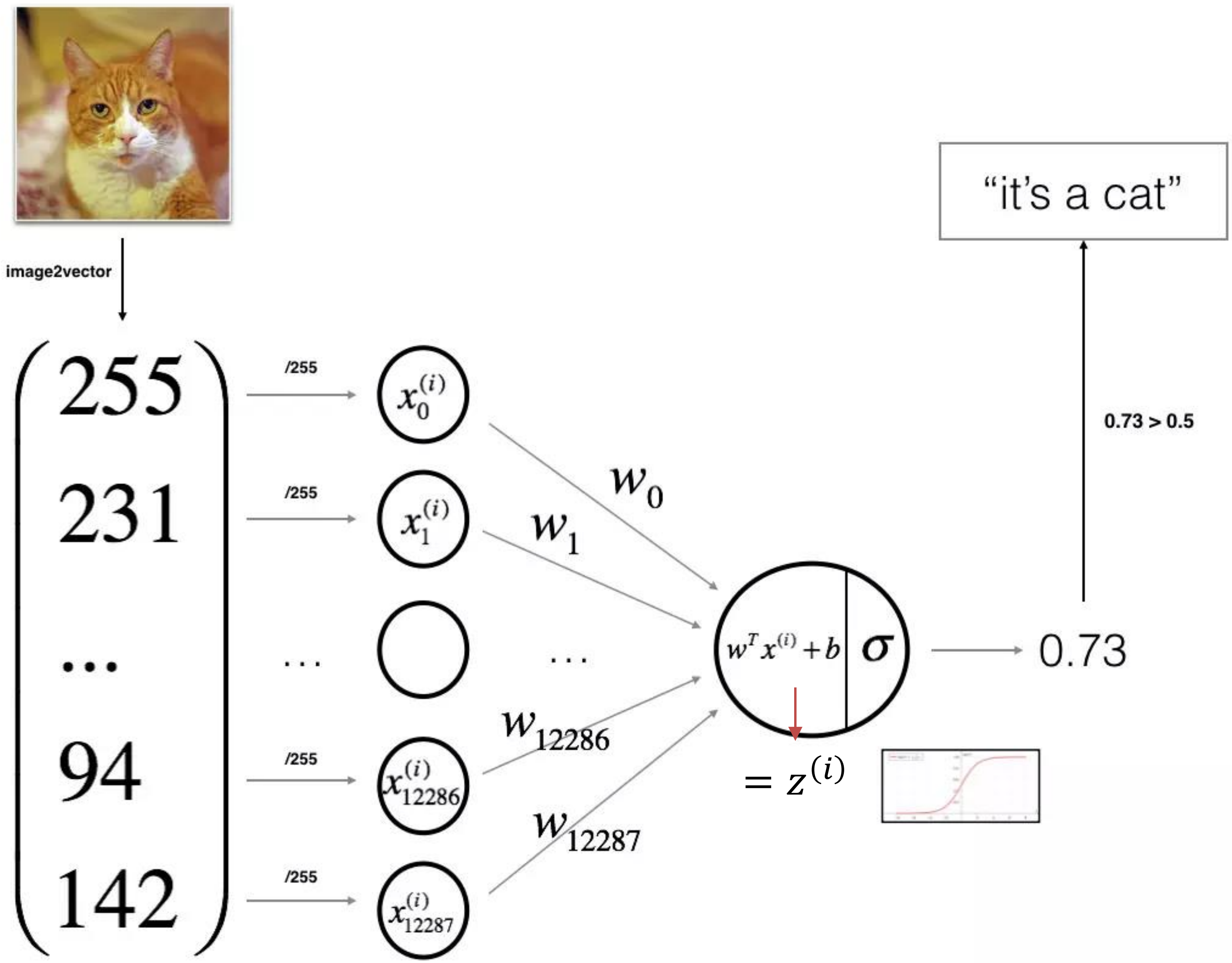


image2vector
255
231
...
94
142
/255
/255
/255
/255
$x_0^{(i)}$
$x_1^{(i)}$
$x_{12286}^{(i)}$
$x_{12287}^{(i)}$
w_0
w_1
w_{12286}
w_{12287}
$w^T x^{(i)} + b$
σ
$= z^{(i)}$
0.73
0.73 > 0.5
"it's a cat"

五、损失函数推导

Sigmod Function 值域： $\hat{y}^{(i)} = sigmoid(z^{(i)}) \in (0,1)$

构建伯努利概率分布

Logistic Regression output：0或1

如何构建伯努利概率分布：

Step 1　简写$\hat{y}^{(i)}$

$$z^{(i)} = w^T x^{(i)} + b$$

$$\hat{y}^{(i)} = a^{(i)} = sigmoid(z^{(i)})$$

$$\longrightarrow \quad \hat{y} = \emptyset(w^T x + b)$$

Step 2　约定$\hat{y}$的概率含义

$\hat{y} = p(y = 1|x)$　　|给定样本x时，y属于类别1的概率

$1 - \hat{y} = p(y = 0|x)$　　|给定样本x时，y属于类别0的概率

五、损失函数推导

Sigmod Function 值域： $\hat{y}^{(i)} = sigmoid(z^{(i)}) \in (0,1)$

构建伯努利概率分布

Logistic Regression output：0或1

如何构建伯努利概率分布：

Step 3 整合成条件概率公式$\Pr(y|x)$

$$\begin{cases} \Pr(y|x) = \hat{y} & , If\ y = 1 \\ \Pr(y|x) = 1 - \hat{y}, & If\ y = 0 \end{cases}$$

Step 4 合并

$$\Pr(y|x) = \hat{y}^{y}(1 - \hat{y})^{1-y}$$

六、支持处理分类变量的统计分析法

（1）判别分析

（2）Probit分析（离散选择模型）这是什么？

（3）Logistic回归分析：1）二元Logistic回归分析：vector只有1/0两种取值

2）多元Logistic回归分析：vector有多种取值 (后文说 Logistic Regression要求output只有1或0。Which right？)

（4）对数线性Model

七、 Logit Model性质

（1）首个离散选择模型 （1959 Luce推导出）

（2）Logit模型与最大效用理论一致 （1960 Marschark证明）

（3）极值分布→可以推出Logit形式的模型 （1965 Marley证明）

（4） Logit形式的模型效用非确定项→服从极值分布（1974 McFadden证明）

（5）亮点：选择枝的减少/增加不影响各选择枝之间被选概率的比值！

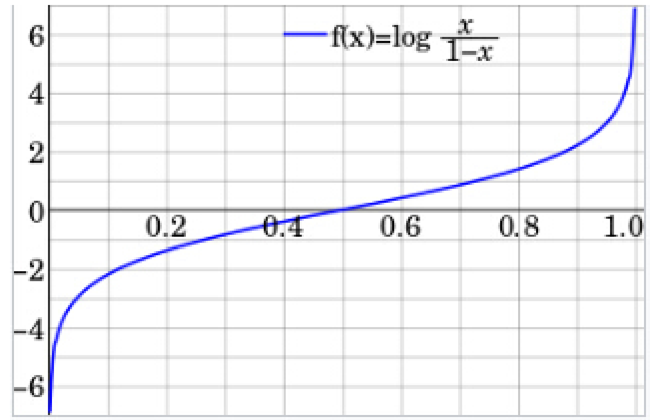
Logit Model（分类评定模型） = Logistic Regression（逻辑回归）

求Logit值的相关参数定义：

（1）优势比： $odds = \frac{P(y=1)}{P(y=0)}$

（2）效用值： $logit = \log(odds)$

$$logit(p) = \log\left(\frac{p}{1-p}\right), p \in (0,1)$$



Plot of logit(p) in the domain of 0 to 1, where the base of logarithm is e

$= \log(p) - \log(1-p) = -\log()$

The logit function is the inverse of the sigmoidal "logistic" function. When the function's variable represents a probability p, the logit function gives the log-odds, or the logarithm of the odds $p/(1 - p)$.

Thanks