蒋豆芽

深度学习——损失函数 试读

来自 【机器学习面试题汇总与解析（蒋豆芽面试题总结）】 | 168 浏览 | 1 回复 | 2021-04-18

蒋豆芽

+关注

机器学习面试题汇总与解析——损失函数

1. **说一下你了解的损失函数?** ☆☆☆☆☆
2. **说说你平时都用过什么损失函数，各自什么特点?** ☆☆☆☆☆
3. **交叉熵函数与最大似然函数的联系和区别?** ☆☆☆☆☆
4. **在用sigmoid作为激活函数的时候，为什么要用交叉熵损失函数，而不用均方误差损失函数?** ☆☆☆☆☆
5. **关于交叉熵损失函数（Cross-entropy）和 平方损失（MSE）的区别?** ☆☆☆☆☆
6. **推导交叉熵损失函数?** ☆☆☆☆☆
7. **为什么交叉熵损失函数有log项?** ☆☆☆☆☆
8. **说说adaboost损失函数**☆☆☆☆
9. **说说SVM损失函数**☆☆☆☆
10. **简单的深度神经网络（DNN）的损失函数是什么?** ☆☆☆☆
11. **说说KL散度**☆☆☆☆
12. **说说Yolo的损失函数**☆☆☆☆☆
13. **交叉熵的设计思想是什么**☆☆☆☆☆
14. **说说iou计算**☆☆☆☆☆
15. **手写miou计算**☆☆☆☆☆

==========

- 本专栏适合于**Python已经入门**的学生或人士，有一定的编程基础。
- 本专栏适合于**算法工程师、机器学习、图像处理求职**的学生或人士。
- 本专栏针对面试题答案进行了**优化，尽量做到好记、言简意赅。这才是一份面试题总结的正确打开方式。这样才方便背诵**
- 如专栏内容有错漏，欢迎在评论区指出或私聊我更改，一起学习，共同进步。
- **相信大家都有着高尚的灵魂，请尊重我的知识产权，未经允许严禁各类机构和个人转载、传阅本专栏的内容。**

==========

蒋豆芽

参考回答

1. 用于**回归的损失函数**：
    1. **绝对值损失函数**
    2. **平方损失函数（squared loss)**：常用于线性回归。
2. 用于**分类的损失函数**：
    1. **0-1损失函数（zero-one loss）**：在感知机中，使用的该损失函数
    2. **对数损失函数（log loss）**：用于最大似然估计，等价于交叉熵损失函数
    3. **指数损失函数（exponential loss）**：在adaboost中使用的该损失函数
    4. **合页损失函数（hinge loss）**：在SVM中使用的该损失函数
    5. **交叉熵损失函数（cross-entropy loss）**：用于分类任务
3. 用于**分割的损失函数**：
    1. **IOU loss**：就是交集和并集的比值，常用于分割任务、回归任务。
    2. **Dice loss**
    3. **Tversky Loss**：引入α和β来控制$Recall$和$Precision$
4. 用于**检测的损失函数**：
    1. **Smooth L1 Loss**：融合绝对值损失函数和平方损失函数，faster RCNN用的该损失函数
    2. **Focal loss**：用于单阶段目标检测方法中样本极端不平衡的情况。

答案解析

1. **为什么需要损失函数?**

    损失函数用来评价模型的**预测值**和**真实值**不一样的程度，在模型正常拟合的情况下，损失函数值越低，模型的性能越好。不同的模型用的损失函数一般也不一样。

    损失函数分为**经验风险损失函数**和**结构风险损失函数**。经验风险损失函数指**预测结果**和**实际结果**的差值，结构风险损失函数是指**经验风险损失函数**加上**正则项**。

    好，我们一个个分析：

2. 用于**回归的损失函数**：

    在回归问题中，|**真实值-预测值**|表示偏差，偏差越大，模型效果越差。

    1. **绝对值损失函数**

    $$L(Y, f(x)) = |Y - f(x)| \tag{.}$$

    这个不难，就是计算**预测结果**$f(x)$与**真实结果**Y的**绝对差值**。

蒋豆芽

针对一个数据求平方损失，而针对n个数据求平方损失后再**加和求平均**$\frac{1}{N}\sum_n(Y-f(x))^2$，即**均方误差**（**MSE**，mean squared error)。

平方损失常用于线性回归，即“最小二乘法”。

3. 用于**分类的损失函数**：

1. 首先是**0-1损失函数（zero-one loss)**

$$L(Y,f(x))=\begin{cases}1, & Y\neq f(x)\\ 0, & Y=f(x)\end{cases} \tag{.}$$

意义很清楚，预测结果等于真实结果，那么损失为0（损失可以理解为偏差）；而如果不等于，损失就为1。

很明显，非黑即白，所以这个损失函数是比较严格的，用的很少。比如在**感知机**用的这个损失函数。

实际上我们可以放宽条件，如下公式：

$$L(Y,f(x))=\begin{cases}1, & |Y-f(x)|\geq T\\ 0, & |Y-f(x)|<T\end{cases} \tag{.}$$

这里我们可以设置一个阈值T，这样就没那么严格了。

2. **对数损失函数（log loss)**

$$L(Y,P(Y|X))=-logP(Y|X) \tag{.}$$

对于样本X，其分类为Y。$P(Y|X)$是**条件概率**，意思是在样本X的条件下，正确分类为Y的概率。那我们当然希望$P(Y|X)$越大越好啊，因为这样模型预测越准，求得的模型参数就越符合我们**样本的分布。**

因为我们习惯损失函数求最小，这就是为什么要加个**负号“-”**。$P(Y|X)$越大，$L(Y,P(Y|X))$越小。

那为什么要使用对数呢? 这和最大（极大）似然估计有关，对数损失是用于最大似然估计的。**最大似然估计**，通俗理解来说，**就是利用已知的样本结果信息，反推最具有可能（最大概率）导致这些样本结果出现的模型参数值。**

我们假定一组参数在一堆数据（**样本结果**）下的**似然值**为P(θ|x1,x2,...,xn)=P(x1|θ)*P(x2|θ)*...*P(xn|θ)，可以看出来，似然值等于每一条数据在这组参数下的条件概率**之积。**求概率是**乘性**，而求损失是**加性**，所以才需要借助log（对数）来**转积为和**，另一方面也是为了简化运算。

对数损失在**逻辑回归**和**多分类任务**上广泛使用。交叉熵损失函数的标准型就是对数损失函数，本质没有区别。

3. **指数损失函数（exponential loss)**

AdaBoost算法就是使用的指数损失函数。

那么为什么AdaBoost算法使用指数损失函数，而不使用其他损失函数呢?

这是因为，当**前向分步算法的损失函数是指数损失函数**时，其学习的具体操作等价于AdaBoost算法的学习过程。证明过程比较复杂，感兴趣的同学可以查查资料。

4. **合页损失函数（hinge loss)**

$$L(Y, f(x)) = max(0, \quad 1 - Y \cdot f(x)) \tag{.}$$

SVM就是使用的合页损失，还加上了正则项。公式意义是，当样本被正确分类且函数间隔大于1时，合页损失是0，否则损失是$1 - Y \cdot f(x)$。

5. **交叉熵损失函数（cross-entropy loss)**

$$L(Y, f(x)) = -[Ylnf(x) + (1-Y)ln(1-f(x))] \tag{.}$$

本质上是一种**对数似然函数**，可用于二分类和多分类任务中。

二分类问题中的loss函数（配合**sigmoid**）：

$$L(Y, f(x)) = -[Ylnf(x) + (1-Y)ln(1-f(x))] \tag{.}$$

多分类问题中的loss函数（配合**softmax**）：

$$L(Y, f(x)) = -Ylnf(x) \tag{.}$$

注：以上损失函数都是标准型，针对单个数据。**针对所有数据还应该加和求平均。**

4. 用于**分割的损失函数**:

分割任务其实就是像素级的分类，所以也可以采用**交叉熵损失函数**，当然分割任务也有其他损失函数。

1. **Jaccard coefficient**（Jaccard系数）

就是IOU（Intersection over Union），交集占并集的大小。

$$Jaccard = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \tag{.}$$

公式其实很简单，就是交集占并集的大小。

那么我们的$IOUloss = 1 - Jaccard$

2. **Dice coefficient**（Dice系数）

dice coefficient 源于二分类，本质上是衡量两个样本(A和B)的重叠部分。

$$Dice = \frac{2 \cdot |A \cap B|}{|A| + |B|} \tag{.}$$

那么我们的$Diceloss = 1 - Dice$

3. **Tversky Loss**

其实就是增加权重，控制$Recall$和$Precision$

5. 用于**检测的损失函数**：

1. **Smooth L1 Loss**

Smooth L1 Loss结合了L1和L2两种损失函数。公式如下：

$$L_2(x) = x^2$$

$$L_1(x) = |x|$$

$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases}$$

其中x为预测框与 groud truth 之间 elementwise 的差异。

在Faster R-CNN中，使用的是smooth L1 loss。smooth L1 loss可以理解为，当|x|<1时，为L2损失（即假设样本服从标准高斯分布），当|x|>1时，为L1损失（即假设样本服从拉普拉斯分布），这样的好处在于训练时|x|>1时**快速下降**，|x|<1时**精细调整**。

2. **Focal loss**

$$FL = \begin{cases} -\alpha(1-p)^{\gamma} log(p), & if \quad y = 1 \\ -(1-\alpha)p^{\gamma} log(1-p), & if \quad y = 0 \end{cases} \tag{.}$$

实验表明γ取2，α取0.25的时候效果最佳。

Focal loss针对单阶段目标检测方法中样本极端不平衡的问题提出的。思路很清晰，大量的样本其实贡献的loss很有限（大部分都是易学习的负样本），导致少量的正样本的loss难以贡献。故增加这部分少量样本的loss的权重（即y=1），惩罚大量负样本的loss的权重。

类似的问题还有：

2. **说说你平时都用过什么损失函数，各自什么特点？** ☆☆☆☆☆

参考回答

回答参考上面。

答案解析

无。

3. **交叉熵函数与最大似然函数的联系和区别？** ☆☆☆☆☆

参考回答

蒋豆芽

联系：**交叉熵函数**可以由**最大似然函数**在**伯努利分布**的条件下推导出来，或者说**最小化交叉熵函数**的本质就是**对数似然函数的最大化。**

答案解析

这里我们来推导一下：

设一个随机变量X满足**伯努利分布**，

$$P(X=1)=p, P(X=0)=1-p \tag{.}$$

则X的**概率密度函数**为：

$$P(X)=p^X(1-p)^{1-X} \tag{.}$$

我们通过一组数据D，可以统计得到X和$1-X$的值，但是p的概率未知，接下来我们就用极大似然估计的方法来估计这个p值。

对于数据集合D，其**对数似然函数**为：

$$\begin{aligned} logP(D) &= log\prod_i^N P(D_i) \\ &= \sum_i logp(D_i) \\ &= \sum_i (D_i logp + (1-D_i)log(1-p)) \end{aligned}$$

可以看到上式和**交叉熵函数**的形式几乎相同，**极大似然估计**就是要求这个式子的最大值。而由于上面函数的值总是小于0，一般像神经网络等对于损失函数会用最小化的方法进行优化，所以一般会在前面加一个负号，得到**交叉熵函数**（或**交叉熵损失函数**）：

$$Loss = -\sum_i [D_i logp + (1-D_i)ln(1-p)] \tag{.}$$

这里我们就可以看出**交叉熵函数**与**极大似然估计**的联系，**最小化交叉熵函数**的本质就是**对数似然函数的最大化。**

现在我们可以用求导得到极大值点的方法来求其**极大似然估计**，首先将对数似然函数对p进行求导，并令导数为0，得到

$$\sum_i [D_i \frac{1}{p} + (1-D_i)\frac{1}{1-p}] = 0 \tag{.}$$

消去分母，得：

$$p = \frac{1}{N}\sum_i D_i \tag{.}$$

这就是伯努利分布下**最大似然估计**求出的概率p

蒋豆芽

参考回答

1. 因为**交叉熵损失函数**可以**完美解决平方损失函数权重更新过慢**的问题，具有“误差大的时候，权重更新快；误差小的时候，权重更新慢”的良好性质。
2. **sigmoid**作为激活函数的时候，如果采用**均方误差损失函数**，那么这是一个**非凸优化**问题，不宜求解。而采用**交叉熵损失函数**依然是一个**凸优化**问题，更容易优化求解。

答案解析

我们来分析一下参数更新的过程：

对于**均方误差损失函数**，常常定义为：

$$C=\frac{1}{2N}\sum_x(a-y)^2 \tag{.}$$

其中y是我们期望的输出，a是神经元实际输出：

$$a=\sigma(x), z=wx+b \tag{.}$$

在训练神经网络的时候我们使用梯度下降的方法来更新w和b，求导：

$$\frac{\partial C}{\partial w}=(a-y)\sigma^{'}(z)x \tag{.}$$
$$\frac{\partial C}{\partial b}=(a-y)\sigma^{'}(z)$$

然后更新参数w和b：

$$w=w^*-\eta\frac{\partial C}{\partial w}=w^*-\eta(a-y)\sigma^{'}(z)x \tag{.}$$
$$b=b^*-\eta\frac{\partial C}{\partial b}=b^*-\eta(a-y)\sigma^{'}(z)$$

η代表学习率。可以看出，参数更新和$\sigma^{'}(z)$有关。如果这里的$\sigma(z)$使用的是sigmoid函数，而我们之前讲激活函数就提到，sigmoid函数两端梯度很小，导致参数更新缓慢。

同样的对于**交叉熵损失函数**，计算一下参数更新的梯度公式就会发现原因。**交叉熵损失函数**一般定义为：

$$C=-\frac{1}{N}\sum_x[ylna+(1-y)ln(1-a)] \tag{.}$$

同样可以看看它的导数，$a=\sigma(x)$：

$$\begin{aligned}\frac{\partial C}{\partial a}&=-\frac{1}{n}\sum_x[y\frac{1}{a}+(y-1)\frac{1}{1-a}]\\&=-\frac{1}{n}\sum_x[\frac{1}{a(1-a)}y-\frac{1}{1-a}]\\&=-\frac{1}{n}\sum_x[\frac{1}{\sigma(x)(1-\sigma(x))}y-\frac{1}{1-\sigma(x)}]\end{aligned}$$

蒋豆芽

$$\frac{\partial z}{\partial a}\frac{\partial a}{\partial z}$$
$$= -\frac{1}{n}\sum_x[\frac{1}{\sigma(x)(1-\sigma(x))}y - \frac{1}{1-\sigma(x)}] \bullet \sigma'(x)$$
$$= -\frac{1}{n}\sum_x[\frac{1}{\sigma(x)(1-\sigma(x))}y - \frac{1}{1-\sigma(x)}] \bullet \sigma(x)(1-\sigma(x))$$
$$= -\frac{1}{n}\sum_x(y-a)$$

所以有：

$$\frac{\partial C}{\partial w} = \frac{\partial C}{\partial z}\frac{\partial z}{\partial w} = (a-y)x$$

$$\frac{\partial C}{\partial b} = \frac{\partial C}{\partial z}\frac{\partial z}{\partial b} = (a-y)$$

所以参数更新公式为：

$$w = w^* - \eta\frac{\partial C}{\partial w} = w^* - \eta(a-y)x \quad (.)$$
$$b = b^* - \eta\frac{\partial C}{\partial b} = b^* - \eta(a-y)$$

可以看到参数更新公式中没有$\sigma^{'}(z)$这一项，权重的更新受$(a-y)$影响，受到误差的影响，所以**当误差大的时候，权重更新快；当误差小的时候，权重更新慢。**这是一个很好的性质。

这就是为什么当使用sigmoid作为激活函数的时候，常用**交叉熵损失函数**而不用**均方误差损失函数。**

5. **关于交叉熵损失函数（Cross-entropy）和 平方损失（MSE）的区别？** ☆☆☆☆☆

参考回答

1. **概念不一样**

均方差损失函数（MSE）是求n个样本的n个输出与期望输出的差的平方的平均值。

$$L(Y, f(x)) = \frac{1}{N}\sum_n(Y - f(x))^2 \quad (.)$$

交叉熵损失函数（cross-entropy）描述模型预测值和真实值的差距大小，越大代表越不相近。

$$C = -\frac{1}{N}\sum_x[ylna + (1-y)ln(1-a)] \quad (.)$$

2. **参数更新速度不一样**

均方差损失函数受sigmoid函数影响，导数更新缓慢。

交叉熵损失函数参数更新只和误差有关，**当误差大的时候，权重更新快；当误差小的时候，权重更新慢。**这是一个很好的性质。

4. **使用场景不一样**

   MSE更适合**回归问题**，**交叉熵损失函数**更适合**分类问题**。

答案解析

无。

6. **推导交叉熵损失函数?** ☆☆☆☆☆

参考回答

我们以偏置b的梯度计算为例，推导出交叉熵代价函数。

我们假设损失函数为C，那么损失函数对b求偏导：

$$\begin{aligned}\frac{\partial C}{\partial b} &= \frac{\partial C}{\partial a}\cdot\frac{\partial a}{\partial z}\cdot\frac{\partial z}{\partial b} \\ &= \frac{\partial C}{\partial a}\cdot\sigma^{'}(z)\cdot\frac{\partial(wx+b)}{\partial b} \\ &= \frac{\partial C}{\partial a}\cdot\sigma^{'}(z) = \frac{\partial C}{\partial a}\cdot a(1-a)\end{aligned} \tag{.}$$

在上面，由二次代价函数推导出来的b的梯度公式为：

$$\frac{\partial C}{\partial b} = (a-y)\sigma^{'}(z) \tag{.}$$

为了消掉该公式中的$\sigma^{'}(z)$，我们想找到一个代价函数使得：

$$\frac{\partial C}{\partial b} = (a-y) \tag{.}$$

即：

$$\frac{\partial C}{\partial a}\cdot a(1-a) = (a-y) \tag{.}$$

对两侧求积分，可得：

$$C = -[ylna + (1-y)ln(1-a)] + constant \tag{.}$$

而这就是前面介绍的交叉熵代价函数。

答案解析

注：sigmoid函数求导为：$\sigma^{'}(z) = \sigma(z)(1-\sigma(z))$

上面是一种方式。还有一种方式是通过最大似然估计的方式求得交叉熵公式，这里不再赘述。

蒋豆芽

这里有两种回答方式。

第一种：因为是公式推导出来的，比如第六题的推导，推导出来的有log项。

第二种：通过最大似然估计的方式求得交叉熵公式，这个时候引入log项。这是因为似然函数（概率）是乘性的，而loss函数是加性的，所以需要引入log项“**转积为和**”。而且也是为了**简化运算**。

答案解析

无。

8. **说说adaboost损失函数**☆☆☆☆

参考回答

回答参考上面。

答案解析

无。

9. **说说SVM损失函数**☆☆☆☆

参考回答

回答参考上面。

答案解析

无。

10. **简单的深度神经网络（DNN）的损失函数是什么？**☆☆☆☆

参考回答

深度神经网络（DNN）涉及到梯度消失的问题，如果使用**均方误差**作为损失函数配合**sigmoid激活函数**，那么参数更新缓慢。这个时候应该考虑使用**交叉熵**作为损失函数，可以避免参数更新缓慢的问题。

答案解析

参考上面的回答。

蒋豆芽

参考回答

相对熵（relative entropy），又被称为Kullback-Leibler散度（Kullback-Leibler divergence）或信息散度（information divergence），是**两个概率分布（probability distribution）间差异的非对称性度量** 。在信息理论中，**相对熵等价于两个概率分布的信息熵（Shannon entropy）的差值。**

设$P(x)$，$Q(x)$是随机变量X上的两个概率分布，则在离散和连续随机变量的情形下，相对熵的定义分别为：

$$KL(P||Q) = \sum P(x) log \frac{P(x)}{Q(x)} \tag{.}$$

$$KL(P||Q) = \int P(x) log \frac{P(x)}{Q(x)} dx$$

$Q(x)$为**理论概率分布**，$P(x)$为模型**预测概率分布**，而KL就是度量这两个分布的差异性，当然差异越小越好，所以KL也可以用作损失函数。

答案解析

类似的还有一个JS散度，基于KL散度的变体，**解决了KL散度非对称的问题。GAN网络**用的JS散度作为损失函数。

12. **说说Yolo的损失函数**☆☆☆☆☆

参考回答

Yolo的损失函数由四部分组成：

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2$$

1. 对预测的中心坐标做损失

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

2. 对预测边界框的宽高做损失

蒋豆芽

3. 对预测的类别做损失

$$\sum_{i=0}^{S^2} \mathbb{1}_i^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2$$

4. 对预测的置信度做损失

$$\sum_{i=0}^{s^2} \sum_{j=0}^{B} \ell_{ij}^{obj} [(C_i - \hat{C}_I)^2] + \lambda_{noobj} \sum_{i=0}^{s^2} \sum_{j=0}^{B} \ell_{ij}^{noobj} [(C_i - \hat{C}_I)^2]$$

我们发现每一项loss的计算都是L2 loss，即使是分类问题也是。所以说yolo是把**分类**问题转为了**回归**问题。

答案解析

Yolo v2的损失函数延续了yolo v1，而yolo v3的损失函数，其中**分类**部分替换成了**交叉熵损失函数**。

13. **交叉熵的设计思想是什么**☆☆☆☆☆

参考回答

交叉熵函数的本质是对数函数。

交叉熵函数使用来描述模型预测值和真实值的差距大小，越大代表越不相近。

交叉熵损失函数可以**完美解决平方损失函数权重更新过慢**的问题，具有“误差大的时候，权重更新快；误差小的时候，权重更新慢”的良好性质。

对数损失在**逻辑回归**和**多分类任务**上广泛使用。交叉熵损失函数的标准型就是对数损失函数，本质没有区别。

答案解析

无。

14. **说说iou计算**☆☆☆☆☆

参考回答

答案参考上面。

答案解析

```
A:
左下角坐标(left_x,left_y)
右上角坐标(right_x,right_y)
```

蒋豆芽

```
7
8  def IOU(rectangle A, rectangleB):
9      W = min(A.right_x, B.right_x) - max(A.left_x, B.left_x)
10     H = min(A.right_y, B.right_y) - max(A.left_y, B.left_y)
11     if W <= 0 or H <= 0:
12         return 0;
13     SA = (A.right_x - A.left_x) * (A.right_y - A.left_y)
14     SB = (B.right_x - B.left_x) * (B.right_y - B.left_y)
15     cross = W * H
16
17     return cross / (SA + SB - cross)
```

15. **手写miou计算** ☆ ☆ ☆ ☆ ☆

参考回答

mIOU一般都是基于类进行计算的，将每一类的IOU计算之后累加，再进行平均，得到的就是mIOU。

答案解析

无。

资源分享 python 机器学习 算法工程师 春秋招 面试题 面经 举报

收藏 赞

相关专栏

机器学习面试题汇总与解析（蒋豆芽面试题总结）

27篇文章 | 90订阅 已订阅

1条评论 默认排序

蒋豆芽 N 0 0 1#

2021-07-10 11:59:35

550681844，大家可以加入这个群，里面有学习资料下载，希望能帮到大家。问题答案：牛客网

请留下你的观点吧~

蒋豆芽

牛客博客，记录你的成长

关于博客 | 意见反馈 | 免责声明 | 牛客网首页