深度学习六十问！一位算法工程师经历30+场CV面试后总结的常见问题合集下篇（含答案）

Original CV开发者都爱看的 极市平台 Today

收录于话题
#CV面经

4个

作者丨灯会
来源丨极市平台
编辑丨极市平台

极市导读

本篇主要包含数据类问题、正则化、激活函数与梯度以及回归、SVM支持向量机、K-Means均值以及机器学习相关常考内容等相关面试经验。 >>加入极市CV技术交流群，走在计算机视觉的最前沿

系列文章：

深度学习三十问！一位算法工程师经历30+场CV面试后总结的常见问题合集（含答案）

一位算法工程师从30+场秋招面试中总结出的超强面经—语义分割篇（含答案）

一位算法工程师从30+场秋招面试中总结出的超强面经——目标检测篇（含答案）

作者灯会为21届中部985研究生，凭借自己整理的面经，去年在腾讯优图暑期实习，七月份将入职百度cv算法工程师。在去年灰飞烟灭的算法求职季中，经过30+场不同公司以及不同部门的面试中积累出了CV总复习系列，此为深度学习基础篇（下）。

数据类问题

1.样本不平衡的处理方法

①欠采样 - 随机删除观测数量足够多的类，使得两个类别间的相对比例是显著的。虽然这种方法使用起来非常简单，但很有可能被我们删除了的数据包含着预测类的重要信息。

②过采样 - 对于不平衡的类别，我们使用拷贝现有样本的方法随机增加观测数量。理想情况下这种方法给了我们足够的样本数，但过采样可能导致过拟合训练数据。

③合成采样（ SMOTE ）-该技术要求我们用合成方法得到不平衡类别的观测，该技术与现有的使用最近邻分类方法很类似。问题在于当一个类别的观测数量极度稀少时该怎么做。比如说，我们想用图片分类问题确定一个稀有物种，但我们可能只有一幅这个稀有物种的图片。

④在loss方面，采用focal loss等loss进行控制不平衡样本。

不平衡类别会造成问题有两个主要原因：1.对于不平衡类别，我们不能得到实时的最优结果，因为模型/算法从来没有充分地考察隐含类。2.它对验证和测试样本的获取造成了一个问题，因为在一些类观测极少的情况下，很难在类中有代表性。

2.讲下数据增强有哪些方法

翻转，旋转，缩放,裁剪，平移，添加噪声，有监督裁剪，mixup，上下采样，增加不同惩罚

解决图像细节不足问题（增强特征提取骨干网络的表达能力）

3.过拟合的解决办法

数据扩充/数据增强/更换小网络/正则化/dropout/batch normalization

增加训练数据、减小模型复杂度、正则化,L1/L2正则化、集成学习、早期停止

什么是过拟合

过拟合（overfitting）是指在模型参数拟合过程中的问题，由于训练数据包含抽样误差，训练时，复杂的模型将抽样误差也考虑在内，将抽样误差也进行了很好的拟合。

产生过拟合根本原因:

观察值与真实值存在偏差, 训练数据不足，数据太少，导致无法描述问题的真实分布, 数据有噪声, 训练模型过度，导致模型非常复杂

什么是欠拟合：训练的模型在训练集上面的表现很差，在验证集上面的表现也很差

原因：训练的模型太简单，最通用的特征模型都没有学习到

正则化

正则化的原理：在损失函数上加上某些规则（限制），缩小解空间，从而减少求出过拟合解的可能性。

机器学习中几乎都可以看到损失函数后面会添加一个额外项，常用的额外项一般有两种，一般英文称作 l1-norm 和l2-norm，中文称作 L1正则化 和 L2正则化，或者 L1范数 和 L2范数。

1. L0、L1、L2正则化

L0 范数：向量中非0元素的个数。

L1 范数 (Lasso Regularization)：向量中各个元素绝对值的和。

L2 范数(Ridge Regression)：向量中各元素平方和再求平方根。

2. L1、L2正则化区别，为什么稀疏的解好?

L1会趋向于产生少量的特征，而其他的特征都是0，而L2会选择更多的特征，这些特征都会接近于0。

实现参数的稀疏有什么好处吗?

一个好处是可以简化模型，避免过拟合。另一个好处是参数变少可以使整个模型获得更好的可解释性。

3.L1正则化和L2正则化的作用

L1正则化可以产生稀疏权值矩阵，即产生一个稀疏模型，可以用于特征选择。

L2正则化可以防止模型过拟合（overfitting）；一定程度上，L1也可以防止过拟合。

4.正则化有哪几种，分别有什么作用?

L0 范数和 L1 范数都能够达到使参数稀疏的目的，但 L0 范数更难优化求解，L1 范数是 L0 范数的最优凸近似，而且它比 L0 范数要容易优化求解。

L2 范数不但可以防止过拟合，提高模型的泛化能力，还可以让我们的优化求解变得稳定和快速。L2 范数对大数和 outlier 更敏感。

L1、L2范数，L1趋向于0，但L2不会，为什么?

L1范数更容易产生稀疏的权重，L2范数更容易产生分散的权重

激活函数与梯度

在多层神经网络中，上层节点的输出和下层节点的输入之间具有一个函数关系，这个函数称为激活函数（又称激励函数）。

1.激活函数的意义如下：

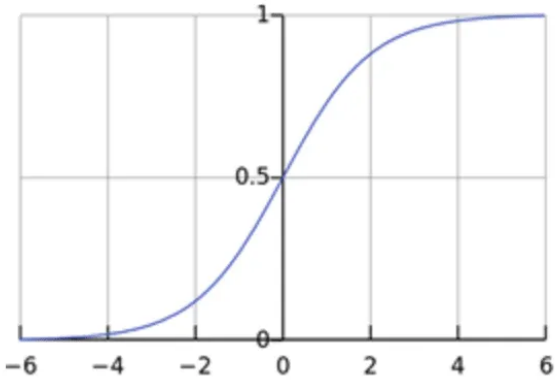
①模拟生物神经元特性，接受输入后通过一个阈值模拟神经元的激活和兴奋并产生输出；②为神经网络引入非线性，增强神经网络的表达能力；③导出神经网络最后的结果(在输出层时)。

常用的激活函数？sigmoid，tanh，ReLU, leaky ReLU, PReLU, ELU，random ReLU等。

①sigmoid

我们通常就用其中最常用的logistic函数来代指sigmoid函数：

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



sigmoid函数和阶跃函数非常相似，但是解决了光滑和连续的问题，同时它还成功引入了非线性。由于其值域处在0~1，所以往往被用到二分类任务的输出层做概率预测。

当输入值大于3或者小于-3时，梯度就非常接近0了，在深层网络中，这非常容易造成“梯度消失”（也就是反向传播时误差难以传递到前面一层）而使得网络很难训练。此外，sigmoid函数的均值是0.5，但是不符合我们对神经网络内数值期望为0的设想。

特点： 它能够把输入的连续实值变换为0和1之间的输出，特别的，如果是非常大的负数，那么输出就是0；如果是非常大的正数，输出就是1.

缺点： 缺点1：在深度神经网络中梯度反向传递时导致梯度爆炸和梯度消失，其中梯度爆炸发生的概率非常小，而梯度消失发生的概率比较大。缺点2：Sigmoid 的 output 不是0均值（即zero-centered）。缺点3：其解析式中含有幂运算，计算机求解时相对来讲比较耗时。

②tanh函数

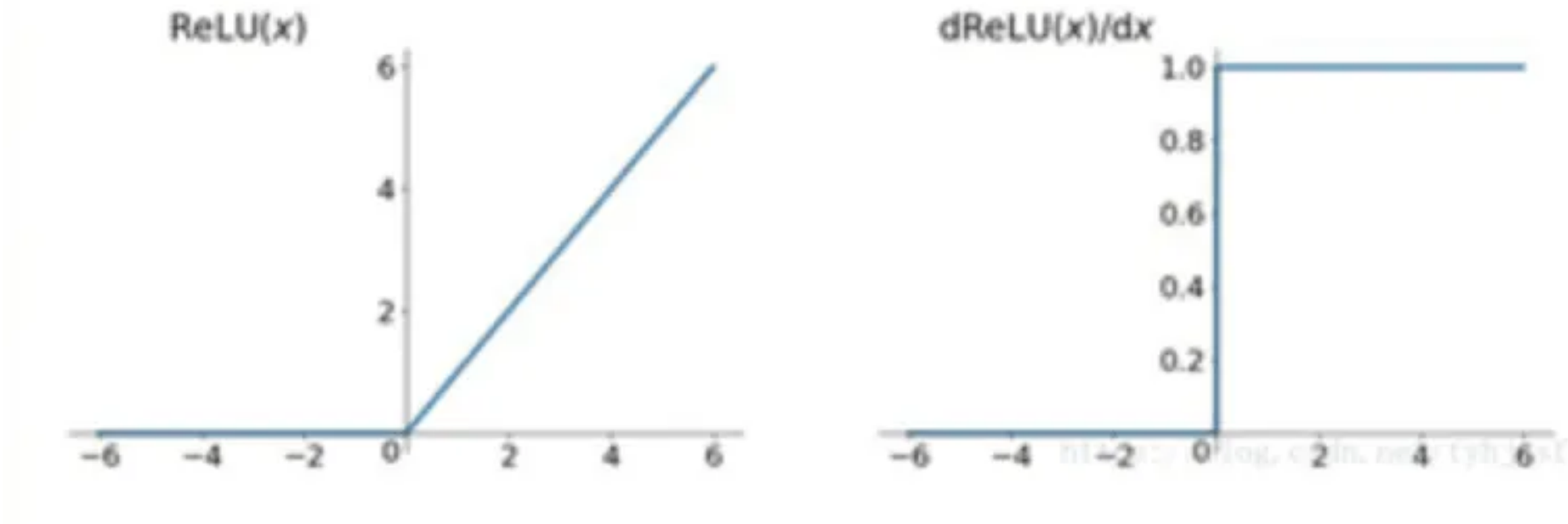
$$f(x)=\tanh(x)=\frac{2}{1+e^{-2x}}-1=2\,\mathrm{sigmoid}(2x)-1 \quad \tanh x=\frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$$

这个tanh函数又被称作双曲正切函数，可以看出它的函数范围是（-1，1）而均值为0，解决了上面sigmoid的一个问题。但是不难发现，该函数依旧没有解决梯度消失的问题。

③Relu函数

ReLu函数又叫线性整流单元，应该说是当前最常用的一个激活函数了，尤其是在卷积神经网络和层次较深的神经网络中。

$$\mathrm{Relu}\ =\max(0,x)$$



ReLu将x=0处的光滑曲线替换为了折线，这使得它的计算也相对更加简单，而且有助于随机梯度下降算法收敛（有研究指出收敛加速可达6倍）。当然ReLu函数也能够缓解梯度消失的问题。

当然，ReLu还是有一些缺陷的，对于小于0的这部分值，梯度会迅速降为0而不再影响网络训练。这会造成部分神经元从来不被激活，也称作“死区”。这也给了ReLu函数的变种很多发挥空间。

优点：

1）解决了gradient vanishing问题 (在正区间)

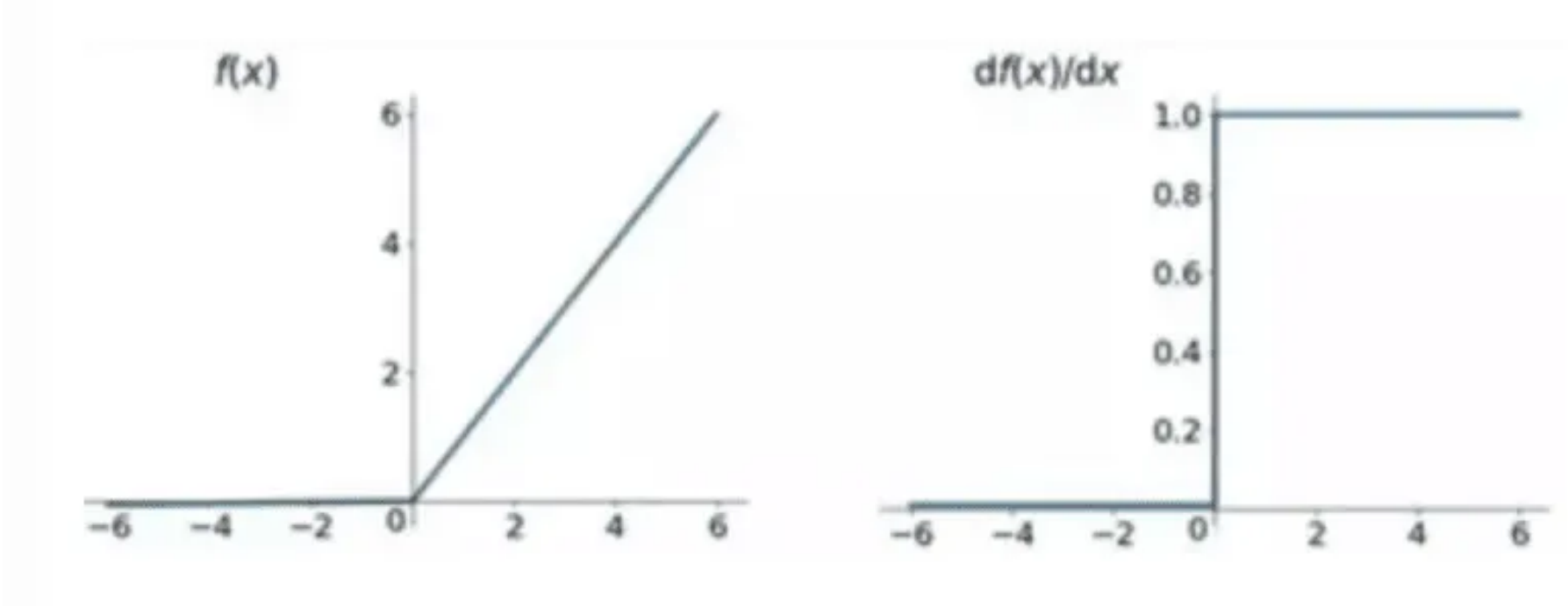
2）计算速度非常快，只需要判断输入是否大于0

3）收敛速度远快于sigmoid和tanh

④Leaky ReLu

$$f(x) = \max(0.01x, x) \quad f(y_i) = \begin{cases} y_i & if, \quad y_i > 0 \\ 0.01 * y_i & \text{if}, \quad y_i \leq 0 \end{cases}$$

将x<=0部分调整为f(x)=αx,其中α一般设为一个较小的正数如0.01或0.001。



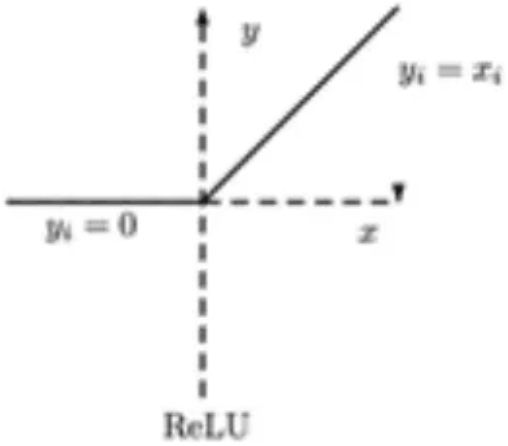
这样就将小于0部分的梯度从零提高到α，给了这些被抑制部分一定参与网络训练的可能。

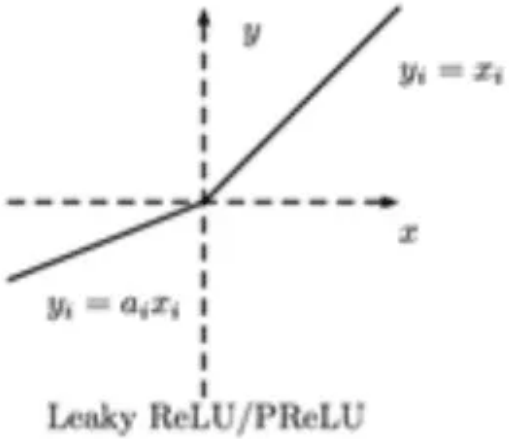
⑤PReLu

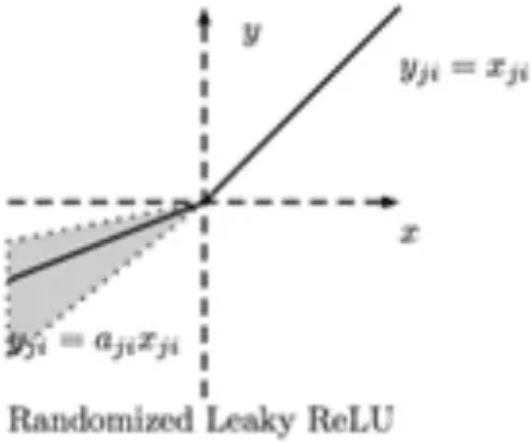
参数化ReLu（Parameterised ReLu,PReLu）的形式和Leaky ReLu一样，唯一地不同是它将α视作一个可训练的参数而不是人为设定的超参数。这样，就避免了Leaky ReLu中的选定α值的问题。

$$f(y_i) = \begin{cases} y_i & \text{if}, \quad y_i > 0 \\ a_i * y_i & \text{if}, \quad y_i \leq 0 \end{cases}$$

⑥andomized Leaky ReLu

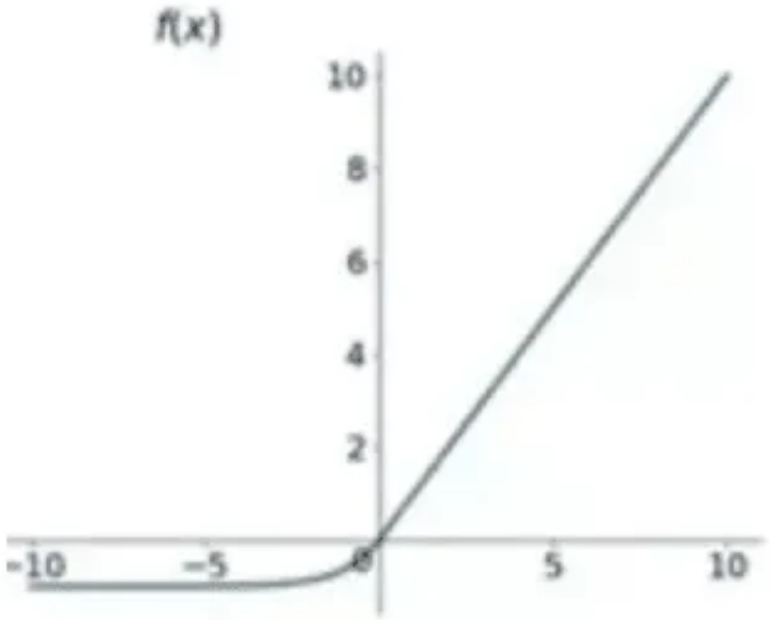


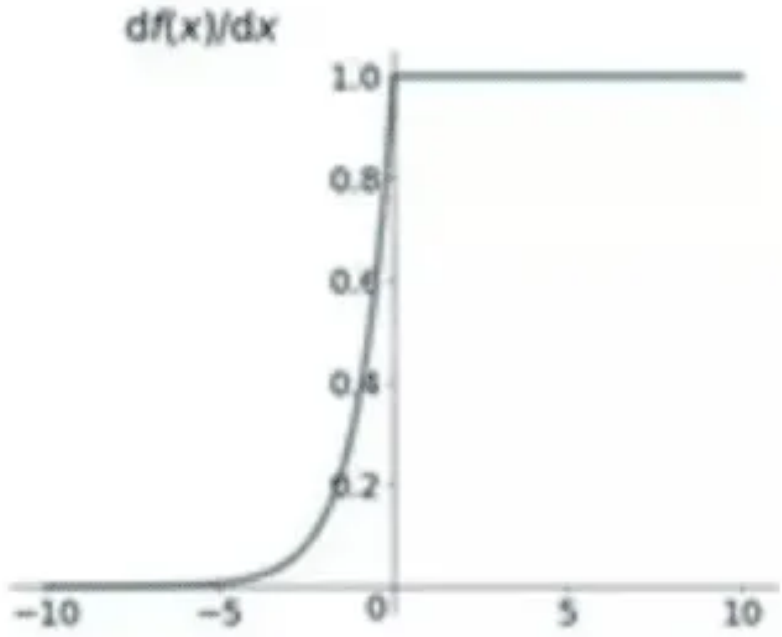




⑦ELU

$$f(x) = \begin{cases} x, & \text{if } x > 0 \\ \alpha(e^x - 1), & \text{otherwise} \end{cases}$$





ELU也是为解决ReLU存在的问题而提出，显然，ELU有ReLU的基本所有优点，以及：不会有Deal ReLU问题；输出的均值接近0，zero-centered。

它的一个小问题在于计算量稍大。类似于Leaky ReLU，理论上虽然好于ReLU

2.写出Sigmoid、Sigmoid的导数

Sigmoid函数：

$$\mathrm{f}(\mathrm{x}) = \frac{1}{1+e^{-x}} = \left(1+e^{-x}\right)^{-1}$$

导数：

$$f(x)' = f(x) \cdot [1 - f(x)]$$

3. sigmoid和relu的优缺点

Relu优点：（1）relu函数在大于0的部分梯度为常数，所以不会产生梯度弥散现象.。而对于sigmod函数，在正负饱和区的梯度都接近于0，可能会导致梯度消失现象。（2）Relu函数的导数计算更快，所以使用梯度下降时比Sigmod收敛起来要快很多。

Relu缺点：Relu死亡问题。当 x 是小于 0 的时候，那么从此所以流过这个神经元的梯度将都变成 0；这个时候这个 ReLU 单元在训练中将死亡（也就是参数无法更新），这也导致了数据多样化的丢失（因为数据一旦使得梯度为 0，也就说明这些数据已不起作用）。

Sigmod优点：具有很好的解释性，将线性函数的组合输出为0，1之间的概率。

Sigmodu缺点：（1）激活函数计算量大，反向传播求梯度时，求导涉及除法。（2）反向传播时，在饱和区两边导数容易为0，即容易出现梯度消失的情况，从而无法完成深层网络的训练。

4. softmax和sigmoid在多分类任务中的优劣

多个sigmoid与一个softmax都可以进行多分类.如果多个类别之间是互斥的，就应该使用softmax，即这个东西只可能是几个类别中的一种。如果多个类别之间不是互斥的，使用多个sigmoid。

5.用softmax做分类函数，假如现在要对1w甚至10w类做分类会出现什么问题?

过拟合，怎么解决，面试官让自己想(不能使用softmax，使用三元组损失)

6.梯度爆炸，梯度消失，梯度弥散是什么，为什么会出现这种情况以及处理办法

梯度弥散（梯度消失）：通常神经网络所用的激活函数是sigmoid函数，sigmod函数容易引起梯度弥散。这个函数能将负无穷到正无穷的数映射到0和1之间，并且对这个函数求导的结果是$f'(x)=f(x)(1-f(x))$表示两个0到1之间的数相乘，得到的结果就会变得很小了。神经网络的反向传播是逐层对函数偏导相乘，因此当神经网络层数非常深的时候，最后一层产生的偏差就因为乘了很多的小于1的数而越来越小，最终就会变为0，从而导致层数比较浅的权重没有更新，这就是梯度消失。

梯度爆炸：就是由于初始化权值过大，前面层会比后面层变化的更快，就会导致权值越来越大，梯度爆炸的现象就发生了。

梯度消失/爆炸是什么？（反向传播中由于链式求导法则的连乘，如果乘数都比较小趋于0，最终传递到网络输入层的梯度会变得很小（梯度消失），如果乘数都很大，最终的梯度也会变得很大（梯度爆炸），其实二者都是因为网络太深导致权值更新不稳定，本质上是因为梯度反向传播中的连乘效应）

梯度消失与梯度爆炸的产生原因

梯度消失：（1）隐藏层的层数过多；（2）采用了不合适的激活函数(更容易产生梯度消失，但是也有可能产生梯度爆炸)

梯度爆炸：（1）隐藏层的层数过多；（2）权重的初始化值过大

梯度消失与梯度爆炸的解决方案

（1）用ReLU、Leaky-ReLU、P-ReLU、R-ReLU、Maxout等替代sigmoid函数。

（2）用Batch Normalization。（3）LSTM的结构设计也可以改善RNN中的梯度消失问题。（4）预训练+微调（5）使用残差网络

回归

1.分类和回归的区别，各举例3个模型

定量输出称为回归，或者说是连续变量预测；定性输出称为分类，或者说是离散变量预测。

常见分类模型有感知机、朴素贝叶斯、逻辑回归(LR)、支持向量机(SVM)、神经网络等；

常见回归模型有线性回归、多项式回归、岭回归（L2正则化）、Lasso回归（L1正则化）等

2.线性回归和逻辑回归的区别

线性回归：利用数理统计中回归分析，来确定两种或两种以上变量间相互依赖的定量关系的一种统计分析方法。一元线性回归分析：y=ax+by=ax+b，只包括一个自变量和一个因变量，且二者的关系可用一条直线近似表示。多元线性回归分析：hθ(x)=θ0+θ1x1+...

+θnxn，包括两个或两个以上的自变量，并且因变量和自变量是线性关系。

逻辑回归：逻辑回归（Logistic Regression）是用于处理因变量为分类变量的回归问题，常见的是二分类或二项分布问题，也可以处理多分类问题，它实际上是属于一种分类方法。

逻辑回归的数学表达模型：

$$h_\theta(x)=g\left(\theta^T x\right)=\frac{1}{1+e^{-\theta^T x}}(\text{ sigmoid 函数 })$$

区别：LR通常用于二分类，使用的是交叉熵损失函数；线性回归用于回归，使用的是均方误差损失函数

3.怎么优化LR？就是求解LR

梯度下降、极大似然法。

SVM—支持向量机

支持向量机（support vector machines, SVM）是一种二分类模型，它的基本模型是定义在特征空间上的间隔最大的线性分类器，间隔最大使它有别于感知机；SVM还包括核技巧，这使它成为实质上的非线性分类器。SVM的学习策略就是间隔最大化，可形式化为一个求解凸二次规划的问题，也等价于正则化的合页损失函数的最小化问题。SVM的的学习算法就是求解凸二次规划的最优化算法。

1.SVM的原理

SVM学习的基本想法是求解能够正确划分训练数据集并且几何间隔最大的分离超平面。

$$\boldsymbol{w} \cdot x+b=0$$

即为分离超平面，对于线性可分的数据集来说，这样的超平面有无穷多个（即感知机），但是几何间隔最大的分离超平面却是唯一的。

2. SVM的核函数了解哪些？为什么要用核函数?

当样本在原始空间线性不可分时，可将样本从原始空间映射到一个更高维的特征空间，使得样本在这个特征空间内线性可分。

①线性核函数

$$\kappa(x, x_i) = x \cdot x_i$$

线性核，主要用于线性可分的情况，我们可以看到特征空间到输入空间的维度是一样的，其参数少速度快，对于线性可分数据，其分类效果很理想

②多项式核函数

$$\kappa(x, x_i) = ((x \cdot x_i) + 1)^d$$

多项式核函数可以实现将低维的输入空间映射到高纬的特征空间，但是多项式核函数的参数多，当多项式的阶数比较高的时候，核矩阵的元素值将趋于无穷大或者无穷小，计算复杂度会大到无法计算。

③高斯（RBF）核函数

$$\kappa(x, x_i) = \exp\left(-\frac{||x - x_i||^2}{\delta^2}\right)$$

高斯径向基函数是一种局部性强的核函数，其可以将一个样本映射到一个更高维的空间内，该核函数是应用最广的一个，无论大样本还是小样本都有比较好的性能，而且其相对于多项式核函数参数要少，因此大多数情况下在不知道用什么核函数的时候，优先使用高斯核函数。

④sigmoid核函数

$$\kappa\left(x, x_i\right) = \tanh(\eta < x, x_i > + \theta)$$

采用sigmoid核函数，支持向量机实现的就是一种多层神经网络。

如果特征的数量大到和样本数量差不多，则选用LR或者线性核的SVM；

如果特征的数量小，样本的数量正常，则选用SVM+高斯核函数；

如果特征的数量小，而样本的数量很大，则需要手工添加一些特征从而变成第一种情况。

3.SVM如何解决线性不可分问题?

间隔最大化，通过引入软间隔、核函数解决线性不可分问题。

4. SVM优化的目标是啥？SVM的损失函数，SVM的适用场景

凸优化问题，

$$\begin{aligned} &\min_{v,b} \frac{1}{2}\|w\rceil^q \\ &\text{s.t.} \quad y_i\left(w^T x_i+b\right) \geq 1 \end{aligned}$$

SVM的损失函数就是合页损失函数加上正则化项；

$$\sum_{i}^{N}\left[1-y_i(w \cdot x_i+b)\right]_{+}+\lambda\|w\|^2$$

5.SVM为什么要对偶(优化复杂度转变，核化)

①首先是我们有不等式约束方程，这就需要我们写成min max的形式来得到最优解。而这种写成这种形式对x不能求导，所以我们需要转换成max min的形式，这时候，x就在里面了，这样就能对x求导了。而为了满足这种对偶变换成立，就需要满足KKT条件（KKT条件是原问题与对偶问题等价的必要条件，当原问题是凸优化问题时，变为充要条件）。

②对偶问题将原始问题中的约束转为了对偶问题中的等式约束

③方便核函数的引入

④改变了问题的复杂度。由求特征向量w转化为求比例系数a，在原始问题下，求解的复杂度与样本的维度有关，即w的维度。在对偶问题下，只与样本数量有关。

6.LR和SVM介绍+区别，什么场景用SVM比较好

相同点：第一，LR和SVM都是分类算法；第二，如果不考虑核函数，LR和SVM都是线性分类算法，也就是说他们的分类决策面都是线性的。第三，LR和SVM都是监督学习算法。第四，LR和SVM都是判别模型。

不同点： 第一，本质上是其损失函数（loss function）不同。注：lr的损失函数是 cross entropy loss，adaboost的损失函数是exp potional loss ,svm是hinge loss，常见的回归模型通常用 均方误差 loss。第二，支持向量机只考虑局部的边界线附近的点，而逻辑回归考虑全局（远离的点对边界线的确定也起作用）。第三，在解决非线性问题时，支持向量机采用核函数的机制，而LR通常不采用核函数的方法。第四，线性SVM依赖数据表达的距离测度，所以需要对数据先做normalization，LR不受其影响。第五，SVM的损失函数就自带正则！而LR必须另外在损失函数上添加正则项。

SVM(支持向量机)主要用于分类问题,主要的应用场景有字符识别、面部识别、行人检测、文本分类等领域。

7.支持向量回归原理(SVR)

SVR（支持向量回归）是SVM（支持向量机）中的一个重要的应用分支。SVR回归与SVM分类的区别在于，SVR的样本点最终只有一类，它所寻求的最优超平面不是SVM那样使两类或多类样本点分的“最开”，而是使所有的样本点离着超平面的总偏差最小。

K-Means(K均值)

K-Means是聚类算法中的最常用的一种，算法最大的特点是简单，好理解，运算速度快，但是只能应用于连续型的数据，并且一定要在聚类前需要手工指定要分成几类。

算法思想：

```
1 [选择K个点作为初始质心
```

```
repeat
    将每个点指派到最近的质心，形成K个簇
    重新计算每个簇的质心
until 簇不发生变化或达到最大迭代次数 ]
```

1.K-means聚类的原理以及过程?

K-means算法是很典型的基于距离的聚类算法，采用距离作为相似性的评价指标，即认为两个对象的距离越近，其相似度就越大。该算法认为簇是由距离靠近的对象组成的，因此把得到紧凑且独立的簇作为最终目标。

2.K-means聚类怎么衡量相似度的?

欧式距离

3.K值怎么来进行确定?(轮廓系数法和手肘法)

4.简要阐述一下KNN算法和K-Means算法的区别

①KNN算法是分类算法，分类算法肯定是需要有学习语料，然后通过学习语料的学习之后的模板来匹配我们的测试语料集，将测试语料集合进行按照预先学习的语料模板来分类；

②Kmeans算法是聚类算法，聚类算法与分类算法最大的区别是聚类算法没有学习语料集合。

机器学习相关常考内容

1.什么是特征归一化

数据的标准化（normalization）是将数据按比例缩放，使之落入一个小的特定区间。在某些比较和评价的指标处理中经常会用到，去除数据的单位限制，将其转化为无量纲的纯数值，便于不同单位或量级的指标能够进行比较和加权其中最典型的就是数据的归一化处理，即将数据统一映射到[0,1]区间上。

特征归一化的作用：数据归一化后， 更容易正确的收敛到最优解、提升模型的精度归一化的另一好处是提高精度、深度学习中数据归一化可以防止模型梯度爆炸

2.为什么要用1*1卷积?

增加网络的深度（加入非线性）、升维或者是降维、跨通道信息交互（channal 的变换）

3.padding的作用

①保持边界信息，如果没有加padding的话，输入图片最边缘的像素点信息只会被卷积核操作一次，但是图像中间的像素点会被扫描到很多遍，那么就会在一定程度上降低边界信息的参考程度，但是在加入padding之后，在实际处理过程中就会从新的边界进行操作，就从一定程度上解决了这个问题。②可以利用padding对输入尺寸有差异图片进行补齐，使得输入图片尺寸一致。③在卷积神经网络的卷积层加入Padding，可以使得卷积层的输入维度和输出维度一致。④卷积神经网络的池化层加入Padding，一般都是保持边界信息和①所述一样。

4. pooling如何反向传播

max pooling: 下一层的梯度会原封不动地传到上一层最大值所在位置的神经元，其他位置的梯度为0；average pooling: 下一层的梯度会平均地分配到上一层的对应相连区块的所有神经元。

Pooling的作用和缺点

增大感受野、平移不变性、降低优化难度和参数。缺点:造成梯度稀疏，丢失信息

感受野的理解：一个卷积核可以映射原始输入图的区域大小。

感受野的计算公式?

$$l_k = l_{k-1} + \left[(f_k - 1) * \prod_{i=1}^{k-1} s_i \right]$$

其中lk−1为第k−1层对应的感受野大小，fk为第k层的卷积核大小，或者是池化层的池化尺寸大小。

5.反向传播的原理： 它的主要思想是由后一级的误差计算前一级的误差，从而极大减少运算量。

6.各种数据的channel是指什么意思 每个卷积层中卷积核的数量

7.卷积层和全连接层的区别

全连接层的权重矩阵是固定的，即每一次feature map的输入过来必须都得是一定的大小，所以网络最开始的输入图像尺寸必须固定，才能保证传送到全连接层的feature map的大小跟全连接层的权重矩阵匹配。

卷积层就不需要固定大小了，因为它只是对局部区域进行窗口滑动，所以用卷积层取代全连接层成为了可能。

8.网络权重初始化

把w初始化为0、对w随机初始化、Xavier initialization、He initialization

9.讲下Attention的原理

减少处理高维输入数据的计算负担,结构化的选取输入的子集,从而降低数据的维度。让系统更加容易的找到输入的数据中与当前输出信息相关的有用信息,从而提高输出的质量。帮助类似于decoder这样的模型框架更好的学到多种内容模态之间的相互关系。

Attention有什么缺点

Attention模块的参数都是通过label和预测值的loss反向传播进行更新，没有引入其他监督信息，因而其受到的监督有局限，容易对label过拟合。

10. AuC，RoC，mAP，Recall，Precision，F1-score

召回率(Recall) = 预测为真实正例 / 所有真实正例样本的个数。

准确率(Precision) =预测为真实正例 / 所有被预测为正例样本的个数。

Precision：P=TP/(TP+FP) 精准率（查准率），Recall：R=TP/(TP+FN) 召回率（查全率 ）

mAP: mean Average Precision, 即各类别AP的平均值，AP: PR曲线下面积，后文会详细讲解，PR曲线: Precision-Recall曲线。

ROC：全称Receiver Operating Characteristic曲线，常用于评价二分类的优劣。

AUC：全称Area Under Curve，被定义为ROC曲线下的面积，取值范围在0.5到1之间。

F1-score：F1值，又称调和平均数，公式(2)和(3)中反应的precision和recall是相互矛盾的，当recall越大时，预测的覆盖率越高，这样precision就会越小，反之亦然，通常，使用F1-score来调和precision和recall。

$$F1 = \frac{2 * \text{ precision } * \text{ recall}}{\text{precision } + \text{ recall}}$$

11. dropout的原理 在进行传播的时候删除一些结点，降低网络的复杂性

dropout训练和测试有什么区别吗?

Dropout 在训练时采用，是为了减少神经元对部分上层神经元的依赖，类似将多个不同网络结构的模型集成起来，减少过拟合的风险。而在测试时，应该用整个训练好的模型，因此不需要dropout。

原文： 在训练过程中，从不同数量的“稀疏”网络中删除样本。在测试时，仅使用权重较小的单个未精简网络，就很容易估算出所有这些精简网络的预测结果的平均值。

12.是否了解free anchor

FreeAnchor基于先进的单级探测器RetinaNet。通过用自由锚帧匹配损失替换RetinaNet的损失。

13. pytorch 多gpu训练机制的原理

Pytorch 的多 GPU 处理接口是 torch.nn.DataParallel(module, device_ids)，其中 module 参数是所要执行的模型，而 device_ids 则是指定并行的 GPU id 列表。

并行处理机制是，首先将模型加载到主 GPU 上，然后再将模型复制到各个指定的从 GPU 中，然后将输入数据按 batch 维度进行划分，具体来说就是每个 GPU 分配到的数据 batch 数量是总输入数据的 batch 除以指定 GPU 个数。每个 GPU 将针对各自的输入数据独立进行 forward 计算，最后将各个 GPU 的 loss 进行求和，再用反向传播更新单个 GPU 上的模型参数，再将更新后的模型参数复制到剩余指定的 GPU 中，这样就完成了一次迭代计算。

14.PyTorch里增加张量维度和减少张量维度的函数

扩大张量：

```
torch.Tensor.expand(*sizes) → Tensor
```

压缩张量：

```
torch.squeeze(input, dim=None, out=None) → Tensor
```

15.nn.torch.conv2d()的参数

```
class torch.nn.Conv2d(in_channels,out_channels, kernel_size, stride=1, padding=0, dilation=1, groups=1,bias=True)
```

in_channels：输入的通道数目 out_channels：输出的通道数目

kernel_size：卷积核的大小，类型为int 或者元组，当卷积是方形的时候，只需要一个整数边长即可，卷积不是方形，要输入一个元组表示 高和宽。

stride：卷积每次滑动的步长为多少，默认是 1

padding：设置在所有边界增加 值为 0 的边距的大小（也就是在feature map 外围增加几圈 0 ），例如当 padding =1 的时候，如果原来大小为 3 × 3 ，那么之后的大小为 5 × 5 。即在外围加了一圈 0 。dilation：控制卷积核之间的间距

16. tensorflow搭建网络和训练的流程

①训练的数据②定义节点准备接收数据③定义神经层：隐藏层和预测层

④定义loss表达式⑤选择optimizer使loss达到最小

17.TensorFlow的参数初始化机制

tf中使用tf.constant_initializer(value)类生成一个初始值为常量value的tensor对象。tf中使用 tf.random_normal_initializer() 类来生成一组符合标准正太分布的tensor。

18.TensorFlow 怎么在网络结构实现一个 if 判断 布尔类型

19.Tensorflow中scope的作用

在tensorflow中使用tf.name_scope()和tf.variable_scope()函数主要是为了变量共享。

20.还有什么办法可以加速python代码吗?

简要：我补充说可以用GPU、batchsize。然后面试官继续追问还有没有，最后他说了cpu加载数据和gpu训练数据的差异，如果只用cpu加载，那发挥不出gpu的优势，可以用异步来加速，即先加载一部分数据到缓存。

21.图像的特征提取有哪些算法

1、SIFT：尺度不变特征变换(Scale-invariant features transform)。SIFT是一种检测局部特征的算法，该算法通过求一幅图中的特征点（interest points,or corner points）及其有关scale 和 orientation 的描述子得到特征并进行图像特征点匹配，获得了良好效果。SIFT特征不只具有尺度不变性，即使改变旋转角度，图像亮度或拍摄视角，仍然能够得到好的检测效果2、SURF:加速稳健特征（Speeded Up Robust Features）。SURF是对SIFT算法的改进，其基本结构、步骤与SIFT相近，但具体实现的过程有所不同。SURF算法的优点是速度远快于SIFT且稳定性好。3、HOG:方向梯度直方图（Histogram of Oriented Gradient）。4、DOG：高斯函数的差分(Difference of Gaussian)5、LBP特征，Haar特征等

22.极大似然估计和最大后验估计的区别是什么?

贝叶斯公式:

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

极大似然估计（MLE）：在已经得到试验结果（即样本）的情况下，估计满足这个样本分布的参数，将使这个样本出现的概率最大的那个参数Θ作为真参数Θ的估计。在样本固定的情况下，样本出现的概率与参数Θ之间的函数，称为似然函数。

最大后验概率（MAP）：最大后验估计是根据经验数据，获得对难以观察的量的点估计。与最大似然估计不同的是，最大后验估计融入了被估计量的先验分布，即模型参数本身的概率分布。

最大后验概率估计其实就是多了一个参数的先验概率，也可以认为最大似然估计就是把先验概率认为是一个定值；后验概率 := 似然 * 先验概率

23. EM算法---最大期望算法

在概率模型中寻找参数最大似然估计或者最大后验估计的算法，其中概率模型依赖于无法观测的隐性变量。

最大期望算法经过两个步骤交替进行计算：第一步是计算期望（E），利用对隐藏变量的现有估计值，计算其最大似然估计值；第二步是最大化（M），最大化在E步上求得的最大似然值来计算参数的值。M步上找到的参数估计值被用于下一个E步计算中，这个过程不断交替进行。

24.降维方法：主成分分析（PCA）、线性判别分析（LDA）、局部线性嵌入（LLE）、LE、SVD

PCA原理和执行步骤：主成分分析(PCA) 是最常用的线性降维方法，它的目标是通过某种线性投影，将高维的数据映射到低维的空间中表示，并期望在所投影的维度上数据的方差最大，以此使用较少的数据维度，同时保留住较多的原数据点的特性。是将原空间变换到特征向量空间内，数学表示为AX = γX。

LDA算法：LDA是一种有监督的（supervised）线性降维算法。与PCA保持数据信息不同，核心思想：往线性判别超平面的法向量上投影，是的区分度最大（高内聚，低耦合）。LDA是为了使得降维后的数据点尽可能地容易被区分！

25.条件随机场

CRF即条件随机场（Conditional Random Fields），是在给定一组输入随机变量条件下另外一组输出随机变量的条件概率分布模型，它是一种判别式的概率无向图模型，既然是判别式，那就是对条件概率分布建模。

26.隐马尔科夫模型（HMM）

隐马尔可夫模型（Hidden Markov Model，HMM）是统计模型，它用来描述一个含有隐含未知参数的马尔可夫过程。其难点是从可观察的参数中确定该过程的隐含参数。然后利用这些参数来作进一步的分析，例如模式识别。

27.伯努利分布

伯努利分布(Bernoulli distribution)又名两点分布或0-1分布。

28.余弦相似度距离和欧氏距离的区别?

欧式距离：如果是平面上的两个点 A(x1,y1) 和 B(x2,y2) ，那么 A 与 B 的欧式距离就是

$$\sqrt{(x_1-x_2)^2+(y_1-y_2)^2}$$

余弦相似度距离： 余弦相似度用向量空间中两个向量夹角的余弦值作为衡量两个个体间差异的大小。相比距离度量，余弦相似度更加注重两个向量在方向上的差异，而非距离或长度上。

29.知道决策树算法吗？ID3，C4.5和CART树

决策树呈树形结构，在分类问题中，表示基于特征对实例进行分类的过程。学习时，利用训练数据，根据损失函数最小化的原则建立决策树模型；预测时，对新的数据，利用决策模型进行分类。

决策树的分类：离散性决策树、连续性决策树。

离散性决策树：离散性决策树，其目标变量是离散的，如性别：男或女等；

连续性决策树：连续性决策树，其目标变量是连续的，如工资、价格、年龄等；

决策树的优点：（1）具有可读性，如果给定一个模型，那么过呢据所产生的决策树很容易推理出相应的逻辑表达。（2）分类速度快，能在相对短的时间内能够对大型数据源做出可行且效果良好的结果。

决策树的缺点：（1）对未知的测试数据未必有好的分类、泛化能力，即可能发生过拟合现象，此时可采用剪枝或随机森林。

①ID3 ---- ID3算法最核心的思想是**采用信息增益来选择特征**

②C4.5采用信息增益比，用于减少ID3算法的局限（在训练集中，某个属性所取的不同值的个数越多，那么越有可能拿它来作为分裂属性，而这样做有时候是没有意义的）

③CART算法采用gini系数，不仅可以用来分类，也可以解决回归问题。

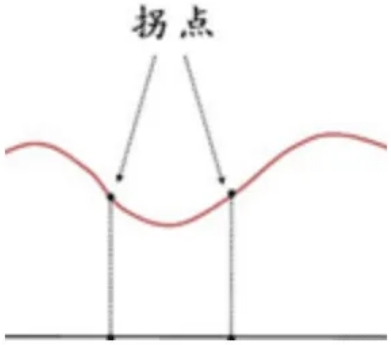
30. K折交叉验证（k-fold cross validation）具体是怎么做的

K折交叉验证用于模型调优，所有的数据都被用来训练，会导致过拟合，K折交叉验证可以缓解过拟合。将数据分为k组，每次从训练集中，抽取出k份中的一份数据作为验证集，剩余数据作为训练集。测试结果采用k组数据的平均值。

31.拐点怎么求?

拐点，又称反曲点，在数学上指改变曲线向上或向下方向的点，直观地说拐点是使切线穿越曲线的点（即连续曲线的凹弧与凸弧的分界点）。

若函数y=f(x)在c点可导，且在点c一侧是凸，另一侧是凹，则称c是函数y=f(x)的拐点。



32.讲一下偏差和方差

模型误差 = 偏差(Bias) + 方差(Variance) + 不可避免的误差

偏差：描述的是预测值（估计值）的期望与真实值之间的差距。偏差越大，越偏离真实数据。

方差：描述的是预测值的变化范围，离散程度，也就是离其期望值的距离。方差越大，数据的分布越分散。

33.鞍点的定义：目标函数在此点上的梯度（一阶导数）值为 0， 但从改点出发的一个方向是函数的极大值点，而在另一个方向是函数的极小值点。

34.假设检验的基本思想

假设检验的基本思想是小概率反证法思想。小概率思想是指小概率事件(P<0．01或P<0．05)在一次试验中基本上不会发生。反证法思想是先提出假设(检验假设Ho)，再用适当的统计方法确定假设成立的可能性大小，如可能性小，则认为假设不成立，若可能性大，则还不能认为假设不成立。

35.熵是什么意思，写出熵的计算公式

熵定义为：信息的数学期望。

$$H = \sum_{i=0}^{n} p(x_i) l(x_i) = -\sum_{i=0}^{n} p(x_i) \log_2 p(x_i)$$

36. KL散度的公式

KL散度（Kullback-Leibler Divergence）也叫做相对熵，用于度量两个概率分布之间的差异程度。

$$D(p||q) = H(p,q) - H(p) = \sum_{x} P(x) \log \frac{p(x)}{q(x)}$$

37.集成学习（bagging和boosting）bagging和boosting的联系和区别

Bagging和Boosting都是将已有的分类或回归算法通过一定方式组合起来，形成一个性能更加强大的分类器，更准确的说这是一种分类算法的组装方法。即将弱分类器组装成强分类器的方法。

boosting（提升法）：Boosting是一族可将弱学习器提升为强学习器的算法。其工作机制为：先从初始训练集训练出一个基学习器，再根据基学习器的表现对训练样本分布进行调整，使得先前基学习器做错的训练样本在后续受到更多关注，然后基于调整后的样本分布来训练下一个基学习器；如此重复进行，直至基学习器数目达到事先指定的值T，最终将这T个基学习器进行加权结合。

Bagging（套袋法）：Bagging是指采用Bootstrap（有放回的均匀抽样）的方式从训练数据中抽取部分数据训练多个分类器，每个分类器的权重是一致的，然后通过投票的方式取票数最高的分类结果最为最终结果。

区别：1）样本选择上：Bagging：训练集是在原始集中有放回选取的，从原始集中选出的各轮训练集之间是独立的.Boosting：每一轮的训练集不变(个人觉得这里说的训练集不变是说的总的训练集，对于每个分类器的训练集还是在变化的，毕竟每次都是抽样)，只是训练集中每个样例在分类器中的权重发生变化.而权值是根据上一轮的分类结果进行调整.

2）样例权重：Bagging：使用均匀取样，每个样例的权重相等Boosting：根据错误率不断调整样例的权值，错误率越大则权重越大.

3）预测函数：Bagging：所有预测函数的权重相等.Boosting：每个弱分类器都有相应的权重，对于分类误差小的分类器会有更大的权重.

4）并行计算：Bagging：各个预测函数可以并行生成Boosting：各个预测函数只能顺序生成，因为后一个模型参数需要前一轮模型的结果.

1）Bagging + 决策树 = 随机森林

2）AdaBoost + 决策树 = 提升树

3）Gradient Boosting + 决策树 = GBDT

38.随机森林的原理

随机森林属于集成学习（Ensemble Learning）中的bagging算法。在集成学习中，主要分为bagging算法和boosting算法。我们先看看这两种方法的特点和区别。

Bagging（套袋法）

bagging的算法过程如下：

从原始样本集中使用Bootstraping方法随机抽取n个训练样本，共进行k轮抽取，得到k个训练集。（k个训练集之间相互独立，元素可以有重复）

对于k个训练集，我们训练k个模型（这k个模型可以根据具体问题而定，比如决策树，knn等）

对于分类问题：由投票表决产生分类结果；对于回归问题：由k个模型预测结果的均值作为最后预测结果。（所有模型的重要性相同）

Boosting（提升法）

boosting的算法过程如下：

对于训练集中的每个样本建立权值wi，表示对每个样本的关注度。当某个样本被误分类的概率很高时，需要加大对该样本的权值。

进行迭代的过程中，每一步迭代都是一个弱分类器。我们需要用某种策略将其组合，作为最终模型。（例如AdaBoost给每个弱分类器一个权值，将其线性组合最为最终分类器。误差越小的弱分类器，权值越大）

下面是将决策树与这些算法框架进行结合所得到的新的算法：

1）Bagging + 决策树 = 随机森林

2）AdaBoost + 决策树 = 提升树

3）Gradient Boosting + 决策树 = GBDT

随机森林的随机体现在哪里?

随机森林的随机性体现在每颗树的训练样本是随机的，树中每个节点的分裂属性集合也是随机选择确定的。有了这2个随机的保证，随机森林就不会产生过拟合的现象了。

调参：一般采用网格搜索法优化超参数组合。这里将调参方法简单归纳为三条：1、分块调参（不同框架参数分开调参）；2、一次调参不超过三个参数；3、逐步缩小参数范围。

39.树模型（RF, GBDT, XGBOOST）

Adaboost与GBDT两者boosting的不同策略是两者的本质区别。

Adaboost强调Adaptive（自适应），通过不断修改样本权重（增大分错样本权重，降低分对样本权重），不断加入弱分类器进行boosting。

而GBDT则是旨在不断减少残差（回归），通过不断加入新的树旨在在残差减少（负梯度）的方向上建立一个新的模型。——即损失函数是旨在最快速度降低残差。

而XGBoost的boosting策略则与GBDT类似，区别在于GBDT旨在通过不断加入新的树最快速度降低残差，而XGBoost则可以人为定义损失函数（可以是最小平方差、logistic loss function、hinge loss function或者人为定义的loss function），只需要知道该loss function对参数的一阶、二阶导数便可以进行boosting，其进一步增大了模型的泛华能力，其贪婪法寻找添加树的结构以及loss function中的损失函数与正则项等一系列策略也使得XGBoost预测更准确。

XGBoost的具体策略可参考本专栏的XGBoost详述。GBDT每一次的计算是都为了减少上一次的残差，进而在残差减少（负梯度）的方向上建立一个新的模型。

XGBoost则可以自定义一套损失函数，借助泰勒展开（只需知道损失函数的一阶、二阶导数即可求出损失函数）转换为一元二次函数，得到极值点与对应极值即为所求。

参考链接

- https://jishuin.proginn.com/p/763bfbd56c95
- https://zhuanlan.zhihu.com/p/80714877
- https://blog.csdn.net/u012426298/article/details/81813911
- https://zhuanlan.zhihu.com/p/61725100
- https://www.huaweicloud.com/articles/9571645efd7d469652b55c47f5115f34.html
- https://zhuanlan.zhihu.com/p/344399453

本文亮点总结

1.梯度爆炸：就是由于初始化权值过大，前面层会比后面层变化的更快，就会导致权值越来越大，梯度爆炸的现象就发生了。

2.决策树呈树形结构，在分类问题中，表示基于特征对实例进行分类的过程。学习时，利用训练数据，根据损失函数最小化的原则建立决策树模型；预测时，对新的数据，利用决策模型进行分类。

如果觉得有用，就请分享到朋友圈吧！

极市干货

YOLO教程：一文读懂YOLO V5 与 YOLO V4｜大盘点｜YOLO 系目标检测算法总览｜全面解析YOLO V4网络结构

实操教程：PyTorch vs LibTorch：网络推理速度谁更快？｜只用两行代码，我让Transformer推理加速了50倍｜PyTorch AutoGrad C++层实现

算法技巧（trick）：深度学习训练tricks总结（有实验支撑）｜深度强化学习调参Tricks合集｜长尾识别中的Tricks汇总（AAAI2021）

最新CV竞赛：2021 高通人工智能应用创新大赛｜CVPR 2021｜Short-video Face Parsing Challenge｜3D人体目标检测与行为分析竞赛开赛，奖池7万+，数据集达16671张！

极市原创作者激励计划

极市平台深耕CV开发者领域近5年，拥有一大批优质CV开发者受众，覆盖微信、知乎、B站、微博等多个渠道。通过极市平台，您的文章的观点和看法能分享至更多CV开发者，既能体现文章的价值，又能让文章在视觉圈内得到更大程度上的推广。

对于优质内容开发者，极市可推荐至国内优秀出版社合作出书，同时为开发者引荐行业大牛，组织个人分享交流会，推荐名企就业机会，打造个人品牌 IP。

投稿须知：

1.作者保证投稿作品为自己的原创作品。

2.极市平台尊重原作者署名权，并支付相应稿费。文章发布后，版权仍属于原作者。

3.原作者可以将文章发在其他平台的个人账号，但需要在文章顶部标明首发于极市平台

投稿方式：

添加小编微信Fengcall（微信号：fengcall19），备注：**姓名-投稿**

△**长按添加极市平台小编**

觉得有用麻烦给个在看啦~

收录于话题 #CV面经·4个

下一篇 · 深度学习三十问！一位算法工程师经历30+场CV面试后总结的常见问题合集（含答案）

Read more

People who liked this content also liked

我们用Windows官方跑了跑Linux GUI应用程序，不愧是“胶水操作系统”

量子位

无需深度学习即可提取图像特征

小白学视觉

3分钟，了解3题必问算法题(NMS/分割/分类)

AIWalker