

机器学习与深度学习面试系列二（模型评估与过拟合）

在模型评估过程中，有哪些主要的验证方法，它们的优缺点是什么？

- Holdout检验法。最简单也是最直接的验证方法，它将原始的样本集合随机划分成训练集和验证集两部分。例如：70%的样本用于模型训练，30%的样本用于模型验证，包括绘制ROC曲线、计算精确率和召回率等指标来评估模型性能。缺点很明显，即在验证集上计算出来的最后评估指标与原始划分有很大关系。为了消除随机性，引入了“交叉检验”的思想。
- 交叉检验法。

k-fold交叉验证: 首先将全部样本划分成k个大小相等的样本子集;依次遍历 这k个子集，每次把当前子集作为验证集，其余所有子集作为训练集，进行模型的 训练和评估;最后把k次评估指标的平均值作为最终的评估指标。在实际实验 中，k经常取10。

留一验证:每次留下1个样本作为验证集，其余所有样本作为测试集。样本总数为n，依次对n个样本进行遍历，进行n次验证，再将评估指标求平均值得到最终 的评估指标。在样本总数较多的情况下，留一验证法的时间开销极大。事实上，留一验证是留p验证的特例。留p验证是每次留下p个样本作为验证集，而从n个元素中选择p个元素有 C_n^p 种可能，因此它的时间开销更是远远高于留一验证，故而很少在实际工程中被应用。

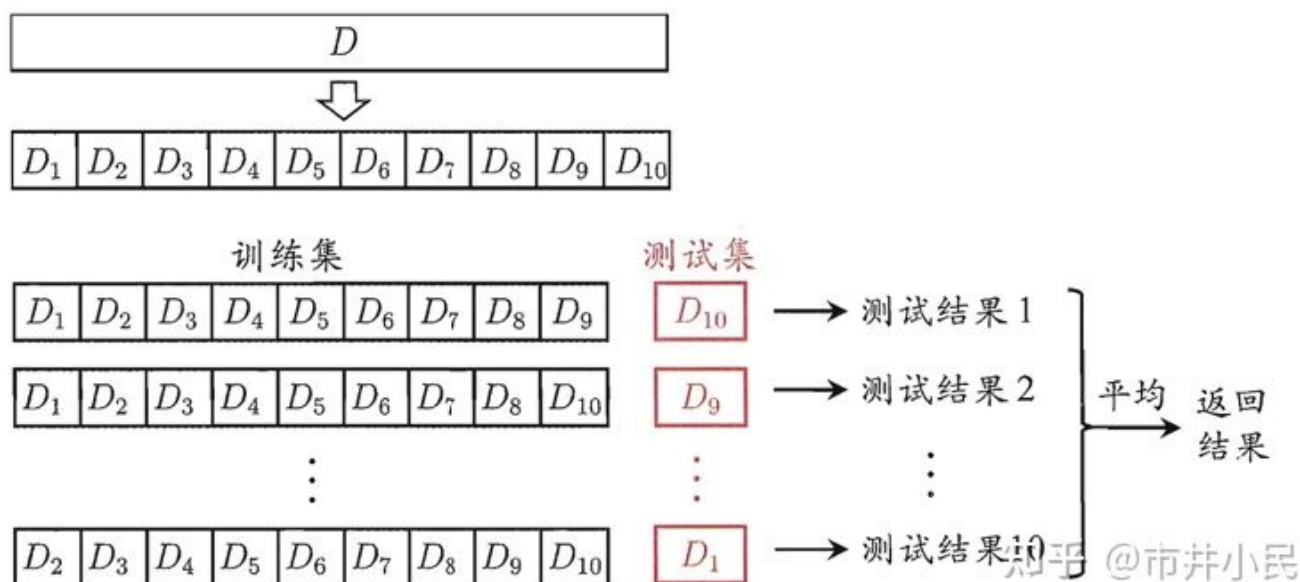
- 自助（bootstrapping）法。当样本规模比较小时，Holdout检验法和交叉检验法将样本集进行划分会让训练集进一步减小，这可能会影响模型训练效果。自助法是基于自助采样法的检验方法。对于总数为n的样本集合，进行n次有 放回的随机抽样，得到大小为n的训练集。n次采样过程中，有的样本会被重复采 样，有的样本没有被抽出过，将这些没有被抽出的样本作为验证集，进行模型验证。

在自助法的采样过程中，对n个样本进行n次自助抽样，当n趋于无穷大时， 最终有多少数据从未被选择过？

样本在m次采样中始终不被采到的概率是 $(1 - \frac{1}{m})^m$ ，取极限得到：

$\lim_{m \rightarrow \infty} (1 - \frac{1}{m})^m = \frac{1}{e} \approx 0.368$ 。即通过自助采样，初始数据集 D 中约有 36.8% 的样本未出现在采样数据集 D' 中。于是我们可将 D' 用作训练集，剩下的用作测试集。这样实际评估的模型与期望评估的模型都使用 m 个训练样本，而我们仍有数据总量约36.8%的、没在训练集中出现的样本用于测试。这样的测试结果，亦称“包外估计” (out-of-bag estimate)。

介绍一下交叉验证数据划分？



我们在训练集上进行模型的训练，用在测试集上的判别效果来估计模型在实际使用时的泛化能力。有时我们还把训练数据另外划分为训练集和验证集，基于验证集上的性能来进行模型选择和调参。注意区分验证集和测试集。

你知道哪些性能度量指标？

对学习器的泛化性能进行评估，不仅需要有效可行的实验估计方法，还需要有衡量模型泛化能力的评价标准。性能度量反映了任务需求，在对比不同模型的能力时，使用不同的性能度量往往会导致不同的评判结果。这意味着模型的"好坏"是相对的，什么样的模型是好的？不仅取决于算法和数据，还决定于任务需求。

1. 回归任务最常用的性能度量是"均方误差" (mean squared error)。

$$E(f; D) = \frac{1}{m} \sum_{i=1}^m (f(x_i) - y_i)^2$$

2. 错误率与准确率(精度)。分类任务中最常用的两种性能度量，错误率是分类错误的样本数占样本总数的比例，准确率(精度)则是分类正确的样本数占样本总数的比例。准确率(精度)：

$$Acc = \frac{n_{correct}}{n_{total}} \quad \text{错误率: } Err = \frac{n_{error}}{n_{total}} = 1 - Acc$$

3. 精确率（查准率）、召回率（查全率）与F1。精确率是指分类正确的正样本个数占分类器判定为正样本的样本个数的比例。召回率是指分类正确的正样本个数占真正的正样本个数的比例。

$$F1 \text{ score 是精准率和召回率的调和平均值, } F1 = \frac{2 * precision * recall}{precision + recall}$$

准确率指标有什么局限性？

首先要区分一下准确率和精确率这两个指标，具体见上一个问题。

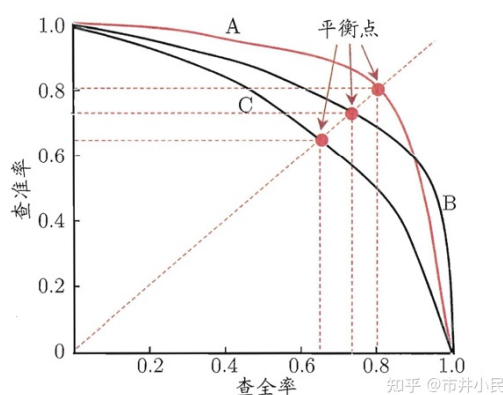


比如：当负样本占99%时，分类器把所有样本都预测为负样本也可以获得99%的准确率。所以，当不同类别的样本比例非常不平衡时，占比大的类别往往成为影响准确率的最主要因素。为了解决这个问题，可以使用更为有效的平均准确率(每个类别下的样本准确率的算术平均)作为模型评估的指标。

再如：在信息检索中，我们经常会关心“检索出的信息中有多少比例是用户感兴趣的”和“用户感兴趣的信息中有多少被检索出来了”。这时候准确率和错误率就不够用了，可以使用精确率和召回率来做此类需求的性能度量。

精确率与召回率的权衡？

精确率和召回率是既矛盾又统一的两个指标，为了提高精确率，分类器需要尽量在“更有把握”时才把样本预测为正样本，但此时往往会因为过于保守而漏掉很多“没有把握”的正样本，导致召回率降低。为了综合评估一个排序模型的好坏，不仅要看模型在不同Top N下的精确率和召回率，而且最好绘制出模型的P-R(Precision- Recall)曲线。



P-R曲线与平衡点示意图

除此之外，F1 score和ROC曲线也能综合地反映一个排序模型的性能。

混淆矩阵是什么？

对于二分类问题，可将样例根据其真实类别与学习器预测类别的组合划分为真正例 (true positive)、假正例 (false positive)、真反例 (true negative)、假反例 (false negative)四种情形，令 TP、FP、TN、FN 分别表示其对应的样例数，则显然有 $TP+FP+TN+FN=\text{样例总数}$ 。



真实情况	预测结果	
	正例	反例
正例	TP (真正例)	FN (假反例)
反例	FP (假正例)	TN (真反例)

知乎 @市井小民

混淆矩阵 (confusion matrix)

$$\text{精确率} = \frac{TP}{TP + FP}$$

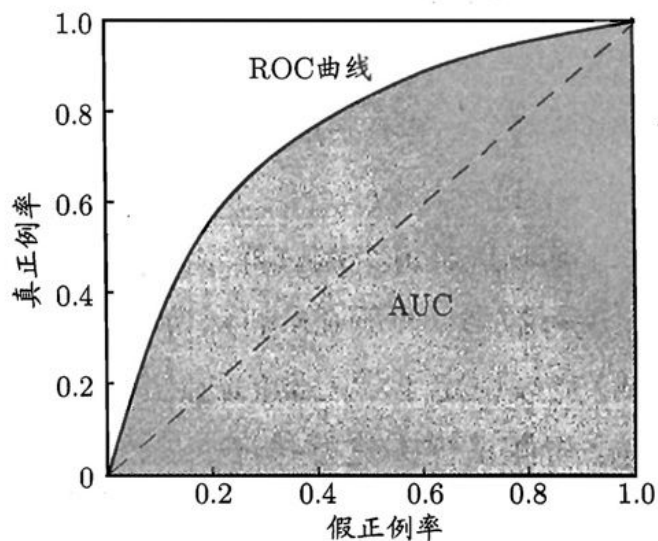
$$\text{召回率} = \frac{TP}{TP + FN}$$

ROC 与 AUC?

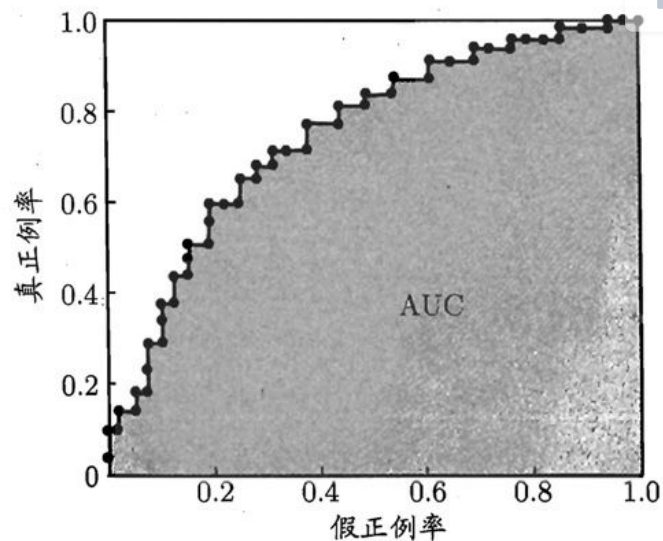
ROC 全称是"受试者工作特征" (Receiver Operating Characteristic)曲线。我们根据学习器的预测结果对样例进行排序, 按此顺序逐个把样本作为正例进行预测, 每次计算出两个重要量的值, 分别以它们为横、纵坐标作图'就得到了 "ROC 曲线"。ROC 曲线的纵轴是"真正例率" (True Positive Rate, 简称 TPR), 横轴是"假正例率" (False Positive Rate, 简称 FPR)。

$$TPR = \frac{TP}{TP + FN}$$

$$FPR = \frac{FP}{TN + FP}$$



(a) ROC 曲线与 AUC



(b) 基于有限样例绘制的 ROC 曲线与 AUC

ROC曲线与AUC示意图

现实任务中通常是利用有限个测试样例来绘制 ROC 图，此时仅能获得有限个(真正例率，假正例率)坐标对，无法产生图(a)中的光滑 ROC 曲线，只能绘制出如图(b)所示的近似 ROC 曲线。对角线对应于 "随机猜测" 模型，而距离点 (0,1) 最近的点对应于"理想模型"。

若一个学习器的 ROC 曲线被另一个学习器的曲线完全"包住"，则可断言后者的性能优于前者；若两个学习器的 ROC 曲线发生交叉，则难以一般性地断言两者孰优孰劣。此时如果一定要进行比较，则较为合理的判据是比较 ROC 曲线下的面积，即 AUC。计算 AUC 值只需要沿着 ROC 横轴做积分就可以了。由于 ROC 曲线一般都处于 $y=x$ 这条直线的上方(如果不是的话，只要把模型预测的概率反转成 $1-p$ 就可以得到一个更好的分类器)，所以 AUC 的取值一般在 0.5~1 之间。AUC 越大，说明分类器越可能把真正的正样本排在前面，分类性能越好。

从 AUC 判断分类器（预测模型）优劣的标准：

- $AUC = 1$ ，是完美分类器，采用这个预测模型时，存在至少一个阈值能得出完美预测。绝大多数预测的场合，不存在完美分类器。
- $0.5 < AUC < 1$ ，优于随机猜测。这个分类器（模型）妥善设定阈值的话，能有预测价值。
- $AUC = 0.5$ ，跟随机猜测一样（例：丢铜板），模型没有预测价值。
- $AUC < 0.5$ ，比随机猜测还差；但只要总是反预测而行，就优于随机猜测。

一句话来说，AUC 值越大的分类器，正确率越高。

ROC 曲线相比 P-R 曲线有什么特点？

相比 P-R 曲线，ROC 曲线有一个特点，当正负样本的分布发生变化时，ROC 曲线的形状能够基本保持不变，而 P-R 曲线的形状一般会发生较剧烈的变化。所以，ROC 曲线的适用场景更多，被广泛用

超参数有哪些调优方法？

1. 网格搜索。通过查找搜索范围内的所有的点来确定最优值。如果采用较大的搜索范围以及较小的步长，网格搜索有很大概率找到全局最优值。然而，这种搜索方案十分消耗计算资源和时间，特别是需要调优的超参数比较多的时候。因此在实际应用中，网格搜索法一般会先使用较广的搜索范围和较大的步长，来寻找全局最优值可能的位置。然后会逐渐缩小搜索范围和步长，来寻找更精确的最优值。这种操作方案可以降低所需的时间和计算量，但由于目标函数一般是非凸的，所以很可能会错过全局最优值。
2. 随机搜索。随机搜索的思想与网格搜索比较相似，只是不再测试上界和下界之间的所有值，而是在搜索范围中随机选取样本点。它的理论依据是，如果样本点集足够大，那么通过随机采样也能大概率地找到全局最优值，或其近似值。随机搜索一般会比网格搜索要快一些，但它的结果也是没法保证的。
3. 贝叶斯优化算法。网格搜索和随机搜索在测试一个新点时，会忽略前一个点的信息；而贝叶斯优化算法则充分利用了之前的信息。贝叶斯优化算法通过对目标函数形状进行学习，找到使目标函数向全局最优值提升的参数。具体来说，它学习目标函数形状的方法是，首先根据先验分布，假设一个搜集函数；然后，每一次使用新的采样点来测试目标函数时，利用这个信息来更新目标函数的先验分布；最后，算法测试由后验分布给出的全局最值最可能出现的位置的点。对于贝叶斯优化算法，有一个需要注意的地方，一旦找到了一个局部最优值，它会在该区域不断采样，所以很容易陷入局部最优值。为了弥补这个缺陷，贝叶斯优化算法会在探索和利用之间找到一个平衡点，“探索”就是在还未取样的区域获取采样点；而“利用”则是根据后验分布在最可能出现全局最值的区域进行采样。（具体可参考：高斯过程）

什么是经验误差与泛化误差？


经验误差（训练误差）：模型在训练集上的误差称为“经验误差”（empirical error）或者“训练误差”“training error”。

泛化误差：模型在新样本集（测试集）上的误差称为“泛化误差”（generalization error）。

对于一个机器学习算法，我们是想最小化泛化误差，但实际上我们在最小化训练误差。这就是导致我们模型出现偏差的原因。

偏置-方差困境？

首先来看一个定义：泛化期望。如果从 X 中，以 $p(x)$ 独立的取出所有大小为 m 的训练集 D ，并在不同的训练集上得到不同的模型。这些不同的模型的平均泛化就是泛化期望。但实际上，我们往往只有一个训练集 D ，我们可能得到一个比泛化期望更好的结果，也可能得到一个更差的结果。

定义上述多个训练集产生的平均模型为： $\hat{f}_m(x) = E[\hat{f}(x|D)]$ ，真实的函数定义为： $f(x)$ 

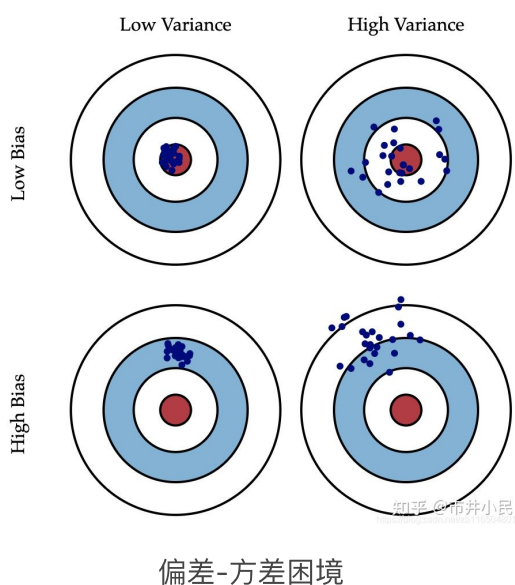
定义泛化期望为：

$$\begin{aligned}\hat{E}_G &= E_D[E_G(D)] = E_D\left[\sum_{x \in X} p(x)(\hat{f}(x|D) - f(x))^2\right] \\&= \sum_{x \in X} p(x) E_D[(\hat{f}(x|D) - f(x))^2] \\&= \sum_{x \in X} p(x) E_D[(\hat{f}(x|D) - \hat{f}_m(x)) + (\hat{f}_m(x) - f(x))]^2 \\&= \sum_{x \in X} p(x) (E_D[(\hat{f}(x|D) - \hat{f}_m(x))^2 + (\hat{f}_m(x) - f(x))^2] + 2E_D[(\hat{f}(x|D) - \hat{f}_m(x))(\hat{f}_m(x) - f(x))]) \\&= \sum_{x \in X} p(x) E_D[(\hat{f}(x|D) - \hat{f}_m(x))^2 + (\hat{f}_m(x) - f(x))^2] \\&= \sum_{x \in X} p(x) E_D[(\hat{f}(x|D) - \hat{f}_m(x))^2] + \sum_{x \in X} p(x) (\hat{f}_m(x) - f(x))^2\end{aligned}$$

定义：

$Variance = \sum_{x \in X} p(x) E_D[(\hat{f}(x|D) - \hat{f}_m(x))^2]$ ，它表示各个训练集D训练出来的模型的方差。它越大，说明各个训练集训练出来的模型不稳定，过拟合。

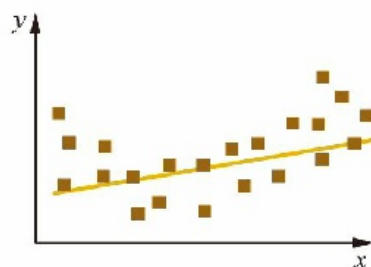
$Bias = \sum_{x \in X} p(x) (\hat{f}_m(x) - f(x))^2$ ，它衡量的是平均模型的泛化误差，其越大，表示该模型越简单，没有学习到数据的潜在结构，与真实函数偏差较大。



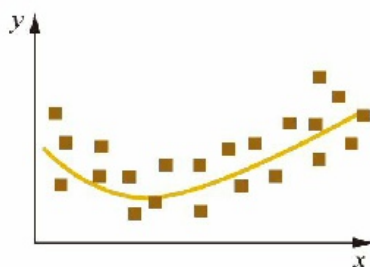
所谓偏差-方差困境，就是说没有办法同时降低偏差和方差。你只能在它们之间取得均衡。对应到模型就是，你想降低偏差，所以你会增加模型的复杂度，防止欠拟合；但是你又不能让模型太复杂而导致方差增加，造成过拟合。在模型的复杂度上，需要找到一个平衡点。

什么是过拟合？

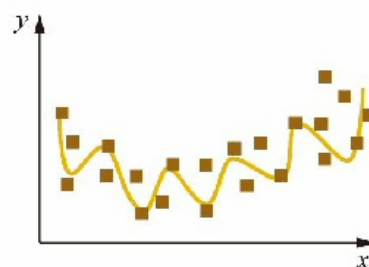
所谓过拟合，指的是模型在训练集上表现的很好，但是在交叉验证集合测试集上表现一般，也就是说模型对未知样本的预测表现一般，泛化（generalization）能力较差。



(a) 欠拟合



(b) 正常模型

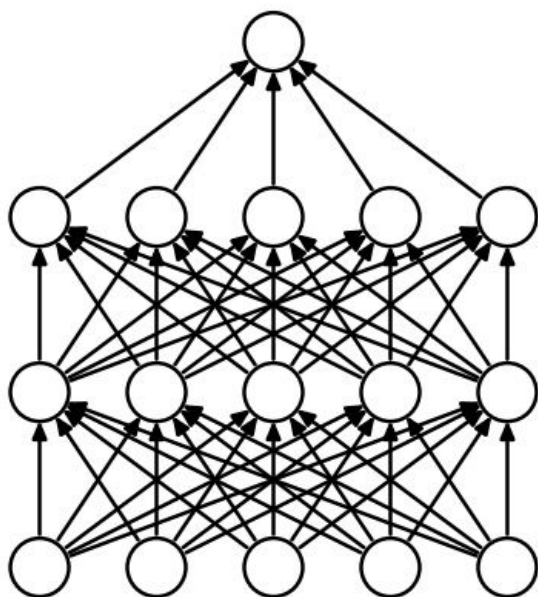


(c) 过拟合

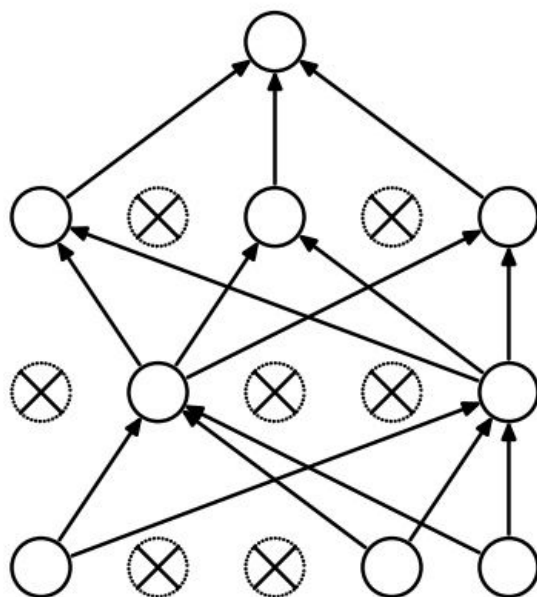
解决过拟合有哪些方法？

1. 获得更多的数据（数据增强）。例如：直接获取更大的数据集（通常做不到）、图像平移、旋转等（注意识别6和9，b和p这种不能旋转）、语音识别任务加入噪声、迁移学习使用预训练模型潜在的使用了更大的数据集。
2. 降低模型复杂度。例如：减少神经网络层数和神经元个数、决策树中降低树的深度、剪枝等。
3. 正则化方法。例如：L1(稀疏权重)，L2(权重衰减)
4. 噪声鲁棒性。例如：对输入数据添加方差极小的噪声、权重添加噪声（dropout）、输出目标添加噪声（标签平滑）
5. 半监督学习。
6. 多任务学习。通过合并几个任务中的样例(可以视为对参数 施加的软约束)来提高泛化的一种方式。正如额外的训练样本能够将模型参数推向 具有更好泛化能力的值一样，当模型的一部分被多个额外的任务共享时，这部分将 被约束为良好的值(如果共享合理)，通常会带来更好的泛化能力。
7. 提前终止（early stop）。当验证集上的误差在事先指定的循环次数内没有进一步改善时，算法提前终止。
8. 参数绑定和参数共享（CNN）。
9. Bagging和其他继承学习方法。
10. 对抗训练。

如何理解Dropout？



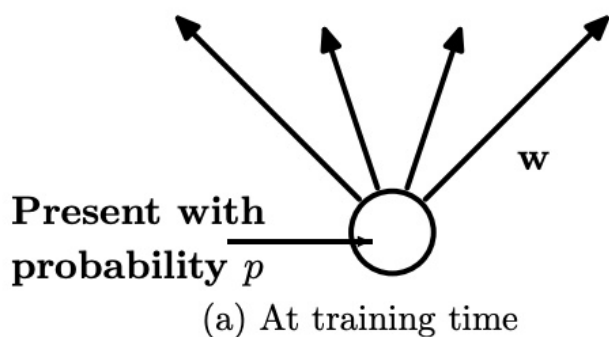
(a) Standard Neural Net



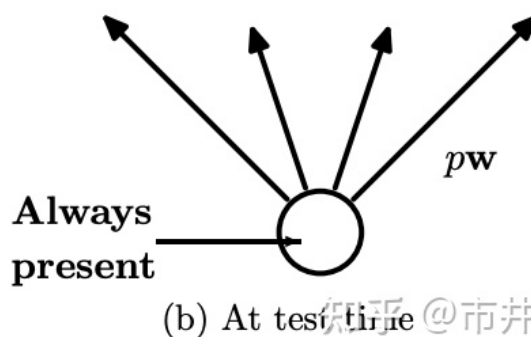
(b) After applying dropout

Dropout 示意图

简单的说，dropout就是在训练阶段，以一定的概率将中间层节点隐藏起来，每次迭代换一批节点隐藏。



(a) At training time



(b) At test time

对于任何一个神经元，在训练阶段，他有 p 的概率存在，并且以 w 权重连接下一层。在测试阶段，神经元总是存在的，权重 w 需要乘以 p 。

Dropout能减少过拟合的原因：

1. 类似于继承学习取平均。训练了 N 个子网络，然后取平均
2. 减少了神经元之间复杂的共适关系，迫使网络去学习更加鲁棒的特征
3. dropout可以理解为是对隐藏节点的权重添加噪声，使整个网络更具鲁棒性