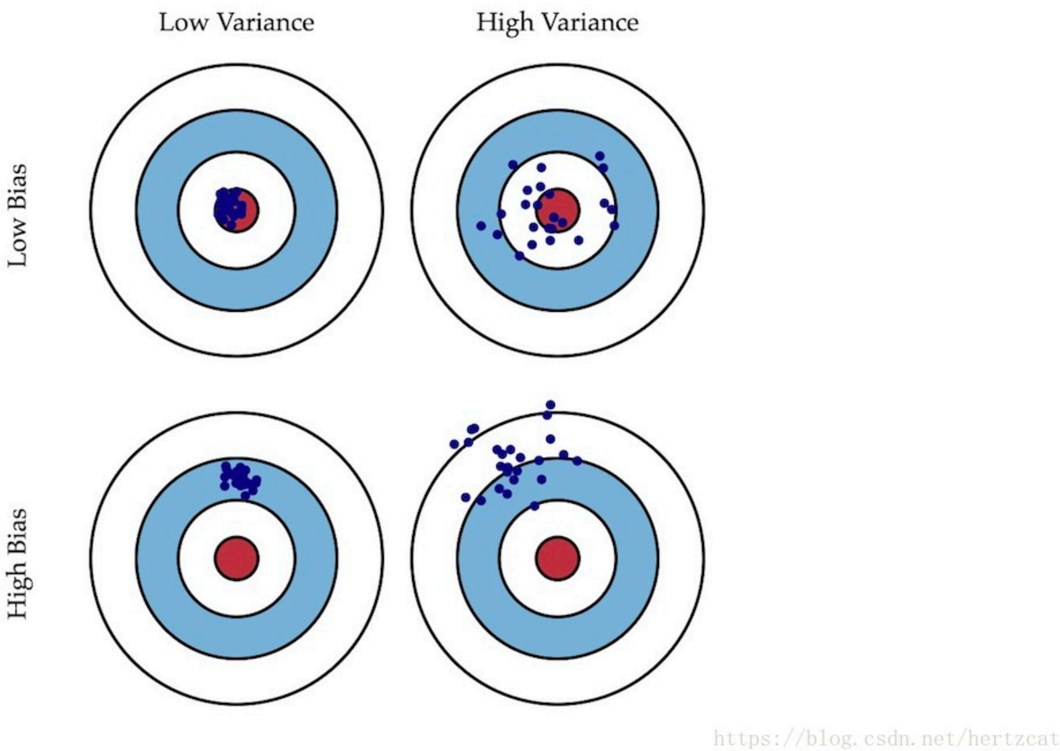
1. 如何防止过拟合?

过拟合-高方差， 欠拟合-高偏差



过拟合产生因素：

- 样本特征很多，样本数相对较少时，模型容易陷入过拟合
- 数据质量较差，噪声点太多
- 模型复杂度过高

为什么过拟合产生通常是因为系数比较大导致的?

答：过拟合，就是拟合函数需要顾忌每一个点，当存在噪声的时候，原本平滑的拟合曲线会变得波动很大。在某些很小的区间里，函数值的变化很剧烈，这就意味着函数在某些小区间里的导数值（绝对值）非常大，由于自变量值可大可小，所以只有系数足够大，才能保证导数值很大。

防止过拟合手段

- 数据增强、数据集扩增
  - 让模型接触到的知识更全面
- 开发集，提前终止训练 Early Stoping
- 正则化
  - 模型越复杂，正则化项的值越大。要使正则化项也很小，那么模型复杂程度受到限制，因此就能有效地防止过拟合。
- Dropout
  - 训练阶段将随机（p%）神经元置0
- Warm up
  - 模型在冷启动阶段容易学习到错误特征，所以开始阶段学习率设置很低

2.什么是梯度消失/爆炸？ 如何避免？

在梯度下降的过程中，如果某一层对激活函数求导>1，那么随着层数的增多，最终的求出的梯度更新将以指数形式增加，即发生**梯度爆炸**，如果此部分小于1，那么随着层数增多，求出的梯度更新信息将会以指数形式衰减，即发生了**梯度消失**。

> 1. Relu等非饱和激活函数
> 2. BN、 LN 等归一化操作
> 3. 残差操作

3. 神经网络适合采用交叉验证法吗？

不适合。交叉折叠的方差随着样本大小的增加而减小。

由于只有在成千上万的样本中才进行深度学习，因此交叉验证没有多大意义。

4. 什么是知识蒸馏？

> 神经网络用剩的logits不要扔，沾上鸡蛋液，裹上面包糠，喂给新网络。

本质：让小模型的logits近似大模型的logits。大模型已经学会将数据近似到一个未知的函数，那么，在大模型学会这个函数后，**只需要让小模型与大模型在给定输入下的logits的softmax分布匹配了**。

例子：经过训练后的原模型，其softmax分布包含有一定的知识——真实标签只能告诉我们，某个图像样本是一辆宝马，不是一辆垃圾车，也不是一颗萝卜；而经过训练的softmax可能会告诉我们，它最可能是一辆宝马，不大可能是一辆垃圾车，但绝不可能是一颗萝卜。

但是，经过softmax归一化，最终得到的分布是一个argmax的近似，其输出是一个接近one-hot的向量，其中一个值很大，其他的都很小。这种情况下，前面说到的「可能是垃圾车，但绝不是萝卜」这种知识的体现是非常有限的。相较类似one-hot这样的硬性输出，我们更希望输出更「软」一些。

所以会使用带有温控系数的softmax：

$$q_i = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)}$$

其中 T 趋向于0时，softmax输出将收敛为一个one-hot向量。

温度 T 趋向于无穷时，softmax的输出则更「软」。

在化学中，蒸馏是一个有效的分离沸点不同的组分的方法，大致步骤是先升温使低沸点的组分汽化，然后降温冷凝，达到分离出目标物质的目的。在前面提到的这个过程中，**我们先让温度 T 升高，然后在测试阶段恢复「低温」，从而将原模型中的知识提取出来，因此将其称为是蒸馏**，实在是妙。

因此，在训练新模型的时候，可以使用较高的 T 使得softmax产生的分布足够软，这时让新模型在同样的温度下的softmax输出近似原模型；在训练结束后用正常温度的$T=1$来预测。

5. 信息熵（香农熵）、交叉熵、相对熵（KL散度）

信息熵

> 对一个分布P进行编码的最短平均编码长度

某个随机事件发生的可能性是$P(x)$, 假设这件事发生所带来的信息量为$I(x)$

比如：摇骰子摇到5的概率是$\frac{1}{6}$，我们大概需要摇6次才能摇出一个5，这个6次就可以暂时当作信息量，可见，概率越大，信息量越少，成反比。

那么，我们如何确切的得知$I(x)$与 $P(x)$之间的具体关系呢？

再比如：我们摇到一次5和一次6这两个**条件独立**事件的概率是 $P(5,6) = P(5)P(6)$，但是，信息量是对事情**发生后**的表达，所以观察到两个事件同时发生的信息量，等于两个事件信息量的和$I(5,6) = I(5) + I(6)$。

由此，可以看出 $I(x)$和 $P(x)$呈对数关系， 可以转化为：

$I(x) = log(\frac{1}{P(x)}) = -log(P(x))$，所以$I(x) >= 0$

信息熵就是所有可能性的发生的**最小平均信息量（或者叫最短平均编码长度）**（数学期望）：

$H(p) = -\sum P(x)log(P(x))$，$H(p) >= 0$

交叉熵

> 用Q的分布对一个分布P进行编码的最短平均编码长度（Q==P 最短）

用$P(x)$的信息量（也叫编码长度）表示$P(x)$的信息熵（最短平均编码长度），也就是常规的信息熵公式

$H(p) = -\sum P(x)log(P(x))$

现在，我们使用一个虚拟的分布$Q(x)$去表示$P(x)$的信息熵

$H(p,q) = -\sum P(x)log(Q(x))$

当且仅当，$Q(x) == P(x)$时，交叉熵$H(p,q) =$ 信息熵$H(p)$**值最小 (数学推理省略，只有Q(x)分布和P(x)概率相同时，才能取到最小值)**

注意：$H(p,q) \neq H(q,p)$

相对熵（KL散度）

> 用Q的分布对一个分布P进行编码的**额外**编码长度（Q==P 是结果为0）
>
> **相对熵可以用来衡量两个概率分布之间的差异**

相对熵的定义是:

$H(p||q) = H(p,q) - H(p)$

$= -\sum P(x)log(Q(x)) - (-\sum P(x)log(P(x)))$

所以，当$Q == P$时，$H(p||q) == 0$

总结

信息量：$I(x)$ x发生所有需编码长度

信息熵：$H(p)$ 分布P的最短平均编码长度（信息量的数学期望）

交叉熵：$H(p,q)$ 分布Q对分布P的平均编码长度

相对熵：$H(p||q)$ 分布Q对分布P最短平均编码所需的额外编码长度