1. **方差是什么，方差代表的意义**

**https://gss2.bdstatic.com/9fo3dSag_xI4khGkpoWK1HF6hhy/baike/s%3D113/sign=c388d5738013632711edc632a28ea056/023b5bb5c9ea15cee484a9a6bc003af33a87b233.jpg**

**方差用来计算每一个变量（观察值）与总体均数之间的差异。**

1. **pca是什么**

**特征中心化，即每一维的数据都减去该维的均值；**

**计算协方差矩阵；**

**计算协方差矩阵的特征向量和特征值；**

**选取大的特征值对应的特征向量，得到新的数据集；**

1. **防止过拟合的方法**
2. **获取更多的数据，比如从源头获取更多的数据量、数据增强等；**
3. **选用合适的模型，比如避免选择过复杂的模型；**
4. **选用更少的特征**
5. **正则化；**
6. **Early stopping，在训练的过程中，记录到目前为止最好的validation accuracy，当连续10次Epoch（或者更多次）没达到最佳accuracy时，则可以认为accuracy不再提高了。此时便可以停止迭代了（Early Stopping）；**
7. **dropout；**
8. **结合多种模型，诸如Bagging、Boosting等；**
9. **加入BN层**
10. **dropout的意义**

**取平均的作用： 先回到正常的模型（没有dropout），我们用相同的训练数据去训练5个不同的神经网络，一般会得到5个不同的结果，此时我们可以采用 “5个结果取均值”或者“多数取胜的投票策略”去决定最终结果。（例如 3个网络判断结果为数字9,那么很有可能真正的结果就是数字9，其它两个网络给出了错误结果）。这种“综合起来取平均”的策略通常可以有效防止过拟合问题。因为不同的网络可能产生不同的过拟合，取平均则有可能让一些“相反的”拟合互相抵消。dropout掉不同的隐藏神经元就类似在训练不同的网络（随机删掉一半隐藏神经元导致网络结构已经不同)，整个dropout过程就相当于对很多个不同的神经网络取平均。而不同的网络产生不同的过拟合，一些互为“反向”的拟合相互抵消就可以达到整体上减少过拟合；**

**减少神经元之间复杂的共适应关系： 因为dropout程序导致两个神经元不一定每次都在一个dropout网络中出现。（这样权值的更新不再依赖于有固定关系的隐含节点的共同作用，阻止了某些特征仅仅在其它特定特征下才有效果的情况）。 迫使网络去学习更加鲁棒的特征 （这些特征在其它的神经元的随机子集中也存在）。换句话说假如我们的神经网络是在做出某种预测，它不应该对一些特定的线索片段太过敏感，即使丢失特定的线索，它也应该可以从众多其它线索中学习一些共同的模式（鲁棒性）。（这个角度看 dropout就有点像L1，L2正则，减少权重使得网络对丢失特定神经元连接的鲁棒性提高）；**

1. **L1和L2的区别**

**L1是稀疏**

**L2则可以使权重变小**

1. **高通滤波低通滤波**

**高通滤波：提取图像边缘，进而锐化图像；**

**将图像转换到频率域后，频率图像具有较高的‘规范性’，依据能量分布可以有效去除噪声，低频部分主要是灰度平滑区域，高频部分可能是噪声或者图像边缘特征，主要是灰度变化较大区域。通过低通滤波器，滤除高频部分，保留低频部分，可有效去除噪声；**

1. **给一幅灰度图，如何快速求出任意给定窗口的数值和**
2. **提取图像特征的方法，比如边缘信息用啥提取**

[**https://blog.csdn.net/arag2009/article/details/64439221**](https://blog.csdn.net/arag2009/article/details/64439221)

1. **牛顿法**

[**https://blog.csdn.net/suixinsuiyuan33/article/details/69525186**](https://blog.csdn.net/suixinsuiyuan33/article/details/69525186)

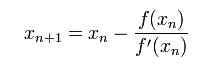
[**https://blog.csdn.net/u012759136/article/details/52302426**](https://blog.csdn.net/u012759136/article/details/52302426)

[**https://www.cnblogs.com/qniguoym/p/8058186.html**](https://www.cnblogs.com/qniguoym/p/8058186.html)

**首先，选择一个接近函数*f*(*x*)零点的*x*0，计算相应的*f*(*x*0)和切线斜率*f '*(*x*0)（这里*f '*表示函数*f* 的导数）。然后我们计算穿过点(*x*0, *f* (*x*0))并且斜率为*f*'(*x*0)的直线和*x*轴的交点的*x*坐标，也就是求如下方程的解：**

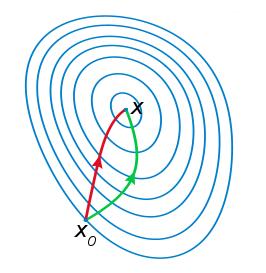
**https://images0.cnblogs.com/blog2015/764050/201508/222309088311820.png**

**我们将新求得的点的*x*坐标命名为*x*1，通常*x*1会比*x*0更接近方程*f* (*x*) = 0的解。因此我们现在可以利用*x*1开始下一轮迭代。迭代公式可化简为如下所示：**

****

**从本质上去看，牛顿法是二阶收敛，梯度下降是一阶收敛，所以牛顿法就更快。如果更通俗地说的话，比如你想找一条最短的路径走到一个盆地的最底部，梯度下降法每次只从你当前所处位置选一个坡度最大的方向走一步，牛顿法在选择方向时，不仅会考虑坡度是否够大，还会考虑你走了一步之后，坡度是否会变得更大。所以，可以说牛顿法比梯度下降法看得更远一点，能更快地走到最底部。（牛顿法目光更加长远，所以少走弯路；相对而言，梯度下降法只考虑了局部的最优，没有全局思想。）**

**根据wiki上的解释，从几何上说，牛顿法就是用一个二次曲面去拟合你当前所处位置的局部曲面，而梯度下降法是用一个平面去拟合当前的局部曲面，通常情况下，二次曲面的拟合会比平面更好，所以牛顿法选择的下降路径会更符合真实的最优下降路径。**

****

**注：红色的牛顿法的迭代路径，绿色的是梯度下降法的迭代路径。**

**牛顿法的优缺点总结：**

**优点：二阶收敛，收敛速度快；**

**缺点：牛顿法是一种迭代算法，每一步都需要求解目标函数的Hessian矩阵的逆矩阵，计算比较复杂。**