一：XGBoost算法的优点

（1） 正则化

Xgboost在代价函数里加入了正则项，用于控制模型的复杂度。正则项里包含了树的叶子节点个数，每个叶子节点上输出的score的L2模的平方和。正则项降低了模型的variance，使学习出来的模型更加简单，防止过拟合，这也是XGBoostt优于传统GBDT的一个特征

（2） 并行处理

XGBoostt工具支持并行。XGBoostt的并行不是tree粒度的并行。Xgboost也是一次迭代完才能进行下一次迭代的（第t次迭代的代价函数里包含）。XGBoost的并行式在特征粒度上的，也就是说每一颗树的构造都依赖于前一颗树。

决策树的学习最耗时的一个步骤就是对特征的值进行排序（因为要确定最佳分割点），XGBoost在训练之前，预先对数据进行了排序，然后保存为block结构，后面的迭代中重复使用这个结构，大大减小计算量。这个block结构也使得并行成为了可能，在进行节点的分类时，需要计算每个特征的增益，大大减少计算量。这个block结构也使得并行成为了可能，在进行节点的分裂的时候，需要计算每个特征的增益，最终选增益最大的那个特征去做分裂，那么各个特征的增益计算就可以开多线程进行。

（3） 灵活性

XGBoost支持用户自定义目标函数和评估函数，只要目标函数二阶可导就行。它对模型增加了一个全新的维度，所以我们的处理不会受到任何限制。

（4）  缺失值处理

对于特征的值有缺失的样本，XGBoost可以自动学习出他的分裂方向。XGBoost内置处理缺失值的规则。用户需要提供一个和其他样本不同的值，然后把它作为一个参数穿进去，以此来作为缺失值的取值。XGBoost在不同节点遇到缺失值时采用不同的处理方法，并且会学习未来遇到缺失值时的处理方法。

（5）  剪枝

XGBoost先从顶到底建立所有可以建立的子树，再从底到顶反向机芯剪枝，比起GBM，这样不容易陷入局部最优解

（6）  内置交叉验证

XGBoost允许在每一轮Boosting迭代中使用交叉验证。因此可以方便的获得最优Boosting迭代次数，而GBM使用网格搜索，只能检测有限个值。

二：XGBoost的算法主流程，基于决策树弱分类器。

不涉及运行效率的优化和健壮性优化的内容。

输入是训练集样本I={(x,y1),(x2,y2),...(xm,ym)}I={(x,y1),(x2,y2),...(xm,ym)}， 最大迭代次数T, 损失函数L， 正则化系数λ,γλ,γ。

输出是强学习器f(x)

对迭代轮数t=1,2,...T有：

（1）计算第i个样本(i-1,2,..m)在当前轮损失函数L基于ft−1(xi)ft−1(xi)的一阶导数gtigti，二阶导数htihti,计算所有样本的一阶导数和Gt=∑i=1mgtiGt=∑i=1mgti,二阶导数和Ht=∑i=1mhtiHt=∑i=1mhti

（2） 基于当前节点尝试分裂决策树，默认分数score=0，G和H为当前需要分裂的节点的一阶二阶导数之和。

　　　　对特征序号 k=1,2...K:

　　　　a) GL=0,HL=0GL=0,HL=0

　　　 b.1) 将样本按特征k从小到大排列，依次取出第i个样本，依次计算当前样本放入左子树后，左右子树一阶和二阶导数和：

GL=GL+gti,GR=G−GLGL=GL+gti,GR=G−GL

HL=HL+hti,HR=H−HLHL=HL+hti,HR=H−HL

　　　　b.2) 尝试更新最大的分数：

score=max(score,1/2\*GL^2/(HL+λ)+1/2GR^2\*(HR+λ)−1/2(GL+GR)^2(HL+HR+λ)−γ)

（3） 基于最大score对应的划分特征和特征值分裂子树。

（4）如果最大score为0，则当前决策树建立完毕，计算所有叶子区域的wtjwtj, 得到弱学习器ht(x)ht(x)，更新强学习器ft(x)ft(x),进入下一轮弱学习器迭代.如果最大score不是0，则转到第2)步继续尝试分裂决策树。