* CART

类别型：计算Gini(D, a)，越小越好

连续型：

1. 值从小到大排序
2. 枚举值进行分割，<值，在左，>值，在右
3. 计算左右子树中y\_hat的均值
4. 计算左右子树中y与y\_hat的距离和，最小的距离和处即为最佳切分点

* Boosting

目标函数F(x)，由很多若分类器组成

优化：min (y-F(x))\*\*2，无法直接使用梯度下降

解决：对残差进行优化

F(x) 🡺 f1(x)、f2(x)…、fn(x)

构造f1(x)，希望(y-f1(x))\*\*2尽可能小

构造f2(x)，希望(y-f1(x)-f2(x))\*\*2尽可能小

构建第t+1个分类器时，前t个已经固定了

令残差

从损失函数的角度看，之间的差值越大，代表优化的越好 🡺 目标：

，当时，

梯度提升树GBDT=CART+Boosting

每一次更新模型，都是在上一次模型残差的基础上，求导，更新

* XGBoost = GBDT+极致的工程设计（并发、贪心排序）+二阶导+改进正则项（除GBDT中队对叶子个数加惩罚，还对权重惩罚）

，当时，

必 🡺牛顿提升树

* LightGBM

将原来使用排序算法来加速决策树学习的方式改为使用直方图中bins的方式

将level层面分割的方式改为leaf层面

并行层面的优化

* CatBoost

1. 对categorical的处理进行了优化，能自动将类别特征转换为数值特征
2. 自动寻找特征的组合，丰富特征的维度（每次分割会将之前使用的类别特征与其组合，贪心的进行类别特征的交叉组合）
3. Ordered boosting算法解决预测漂移问题

Categorical中值少，直接one-hot编码，然后train树模型

值很多，若one-hot编码，一是维度灾难，二是，树模型切分遍历很慢。对这种类别特征进行预编码。比如频次编码（count encoding，将特征包含的N个特征值作为编码数，对离群值敏感，可采用对数变换）、频次排序编码（Label Count Encoding，出现次数最多、少的类别，编码为0）、均值编码（去类别对应的y的均值来代替类别标签）等