Naive bayes:

1. 设 $x = \{a_1, a_2, \dots a_m\}$ 为待分类项，而每个 a 为 x 的一个特征属性
2. 类别 集合 $C = \{y_1, \dots y_n\}$
3. 算 $p(y_1|x), p(y_2|x) \cdots p(y_n|x)$
4. 若 $p(y_k|x) = \max\{p(y_1|x), \dots p(y_n|x)\}$，则 $x \in y_k$

① 训练集：

② 求 $p(a_1|y_1), p(a_2|y_1) \cdots p(a_m|y_1)$; $p(a_1|y_2) \cdots, p(a_m|y_2) \cdots; p(a_1|y_n), \cdots p(a_m|y_n)$

③ 若各 $a_1, a_2 \cdots a_m$ 是独立的，则 $p(y_i|x) = \dfrac{p(x|y_i)p(y_i)}{p(x)}$

$\because p(x)$ 对 all 类别 为常数 $\therefore$ 只需 $p(x|y_i)p(y_i)$ 最大化

而 $p(x|y_i) = p(a_1|y_i) \cdots p(a_m|y_i) \cdot p(y_i) = p(y_i) \cdot \prod_{j=1}^{m} p(a_j|y_i)$

NB: 基础思想：对于待分类项，求解 ~~在这些~~ x_i 出现的条件下，各个类别 $y_1 \cdots y_n$ 出现的 x_i 概率，取最大概率的类别.

pros: ① 简单 ② 分类为新用率高，速度快，所需估计参数少，对缺失数据不敏感

cons: ① 假设各个属性互不相关，不对症 ② 需知先验prob

Decision Tree

基础思想：通过训练数据构建决策树，每个节点表示在某个属性上的测试，每个分枝代表该测试的一个输出，每个叶结点存放一个类标

ID3: info gain. ; C4.5: 信息增益比 ; CART: 基尼系数

pros: ① 无需参数假设 ② 适合高维数据.

③ 简单

cons: ① 易过拟合 ② 忽略属性间相关性

③ Imbalanced data, info gain 偏向于多样本的记

bagging 随机森林

特点：对 outliers 更鲁棒。dll 依赖个体分类器实力，把它们之间依赖性（不希望有相关性）。对每次划分属性很敏感

优点：①不易 overfit，②每次划分只取部分attr，∴在大型DB很有效

③即使 missing 很多，也能维持 high accuracy

④对 imbalanced data，可以平衡误差

缺点：①在某些噪声较大的分类，回归问题上会过拟合。

②级别划分较多的属性会对 rf 产生更大影响

boosting：对多分类器封装结合成强分类器。（串行）

优点①：dll 高，无需太多参数

优点②：对 outlier 敏感

缺点：

random forest 比一般决策树优点原因：

bagging，多树投票提高泛化能力且引入随机，避免单棵树过拟合，提高整体泛化能力

xgboost 与 gbdt

gbdt 以 CART 作为基分类器 ①xgboost 还支持线性分类器。

②有 L1 L2 的 regularization

控制模型复杂

③对 cost function 进行二阶 Taiyor 展开。

优化时一阶导信息

④支持列抽样

⑤自动学习 missing value 的分裂方向

⑥在特征粒度上并行

⑦在训练前，对 data 排序，保存为 block 结构，后面迭代反复用这个结构。