监督学习（上）

汪小圈

2025-03-10

内容安排

- 监督学习简介
- 回归问题
  - 线性回归模型
  - 岭回归与 Lasso 回归
- 分类问题
  - 逻辑回归
  - 线性判别分析
  - 支持向量机
  - 决策树
- 集成学习
  - Bagging 与随机森林
  - Boosting 与梯度提升树
  - Stacking

监督学习简介

- **监督学习**是指从**带有标签的数据**中自动学习规律和模式，并利用这些规律和模式对**新数据进行预测和决策**的过程
- 在监督学习中，我们拥有：
  - 输入特征 $\mathbf{x}$
  - 对应输出标签 y
  - 目标是学习一个从输入特征到输出标签的映射关系
- 主要任务类型：
  - **回归**：预测连续数值型输出
  - **分类**：预测离散类别标签

监督学习在金融领域的应用

- **风险评估**：根据客户的历史信用数据（特征）预测其信用风险等级（标签）
- **欺诈检测**：基于交易记录（特征）识别欺诈交易（标签）
- **量化交易**：预测股票价格走势（标签）以辅助交易决策（特征）
- **客户细分**：根据客户特征（特征）预测客户所属类别（标签），进行精准营销

回归问题描述

- 通过由 $K \times 1$ 维向量 $\mathbf{x}$ 表示的 K 个观测到的预测变量（特征）来预测**连续数值型**结果 y
- 由训练数据 $\{y_i, \mathbf{x}_i\}_1^N$ 找到如下关系中的未知函数 f

$$y_i = f(\mathbf{x}_i) + \epsilon_i$$

- 假设训练集中有 N 个观测，将观测值堆叠成
  - $N \times 1$ 维向量 $\mathbf{y} = (y_1, y_2, ..., y_N)'$
  - $N \times K$ 维矩阵 $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N)'$
  - $N \times 1$ 维向量 $\quad = (\epsilon_1, \epsilon_2, ..., \epsilon_N)'$
- 回归模型可以写为：$\mathbf{y} = f(\mathbf{X}) + \epsilon$

线性回归模型

- 线性模型：$\mathbf{y} = \mathbf{X} + \epsilon$，其中 β 是回归系数向量
- 对于单个样本 i，可以表示为：
  $y_i = \mathbf{x}_i^T \beta + \epsilon_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_K x_{iK} + \epsilon_i$
- 最小化误差平方和（最小二乘法）：

$$\min_{\beta} (\mathbf{y} - \mathbf{X}\beta)'(\mathbf{y} - \mathbf{X}\beta)$$

- 得到普通最小二乘 (OLS) 估计量 $\widehat{\beta} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$

高维环境下的过拟合问题

- 当 K 相对于 N 来说并不小甚至比 N 更大的情况下，基于 OLS 估计值的预测通常是不可靠的
- 协变量相对观测数量来说非常多时，OLS 会调整 $\hat{\beta}$ 来拟合噪音而非真实信号
- 虽然样本内 R^2 可能非常高，但样本外预测的 R^2 则往往非常低甚至小于零
- 解决方案：正则化 (Regularization)

岭回归 (Ridge Regression)

- 在最小化误差平方和的基础上，补充了 L^2 范数惩罚项 $'$：

$$\min_{\beta}\left[\frac{1}{N}(\mathbf{y}-\mathbf{X}\beta)'(\mathbf{y}-\mathbf{X}\beta)+\lambda\beta'\beta\right]$$

- 其中超参数 λ 用以控制惩罚的强度
- 估计结果为 $\hat{\beta}=(\mathbf{X}'\mathbf{X}+\lambda\mathbf{I}_K)^{-1}\mathbf{X}'\mathbf{y}$，其中 $\mathbf{I}_K$ 是 $K\times K$ 单位矩阵
- 通过向 $\mathbf{X}'\mathbf{X}$ 添加对角矩阵（即"岭"），求逆运算时 $\lambda\mathbf{I}_K$ 的存在将导致回归系数 $\hat{\beta}$ 向零收缩
- 若 $\mathbf{X}$ 是正交矩阵，即 $\mathbf{X}'\mathbf{X}=\mathbf{I}_K$，岭回归将 OLS 估计值中的每个回归系数向零等比例收缩，即 $\hat{\beta}_j=\hat{\beta}_{j,OLS}/(1+\lambda)$

Lasso 回归 (Least Absolute Shrinkage and Selection Operator)

- 在最小化误差平方和的基础上，补充了 L^1 范数惩罚项 $||\beta||_1 = \sum_{j=1}^{K} |\beta_j|$：

$$\min_{\beta} \left[\frac{1}{N} (\mathbf{y} - \mathbf{X}\beta)'(\mathbf{y} - \mathbf{X}\beta) + \gamma \sum_{j=1}^{K} |\beta_j| \right]$$

- 其中超参数 γ 用以控制惩罚的强度
- 无解析解，仅有数值解
- 若 $\mathbf{X}$ 是正交矩阵，Lasso 将 OLS 估计值向零移动一个固定量 γ，即 $\hat{\beta}_j = sgn(\hat{\beta}_{j,OLS})(|\hat{\beta}_{j,OLS}| - \gamma)_+$
- Lasso 的重要特性：可以将一些系数精确地缩减为零，实现自动特征选择

弹性网 (Elastic Net)

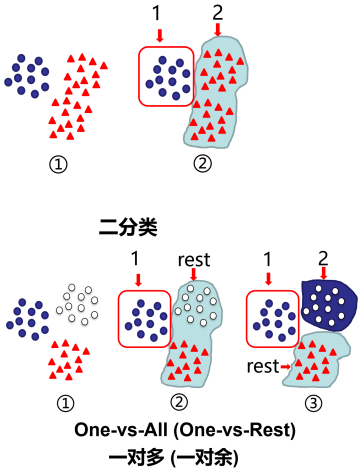
- 弹性网结合了岭回归和 Lasso 的惩罚项：

$$\min_{\beta} \left[\frac{1}{N}(\mathbf{y} - \mathbf{X}\beta)'(\mathbf{y} - \mathbf{X}\beta) + \gamma_1 \sum_{j=1}^{K} |\beta_j| + \gamma_2 \beta'\beta \right]$$

- 与 Lasso 一样，弹性网会将一些回归系数设为零，实现特征选择
- 同时也会像岭回归那样对回归系数进行收缩，减小模型方差
- 适用于特征数量多于样本数量的情况，尤其是当特征之间存在相关性时

分类问题描述

- 通过由 $K \times 1$ 维向量 $\mathbf{x}$ 表示的 K 个观测到的预测变量（特征）来预测结果 y
- 二分类：预测结果 $y \in \{0, 1\}$
- 多分类：预测结果 $y \in \{C_1, ..., C_L\}$
  - 有些二分类模型可以直接推广到多分类
  - 利用二分类模型来解决多分类问题，基本思路是”拆解法”，即将多分类任务拆为若干个二分类任务求解



类别不平衡问题

分类任务中不同类别的训练样本数量差别很大

- 类别平衡时，预测值 $y > 0.5$(即 $y/(1-y) > 1$) 判别为正例，否则为反例
- 类别不平衡时，假定正类样本数量 m^+ 较少，反类样本数量 m^- 较多，则 $\frac{y}{1-y} > \frac{m^+}{m^-}$ 时预测为正例
- “再缩放” 策略：当 $\frac{y'}{1-y'} = \frac{y}{1-y} \times \frac{m^-}{m^+} > 1$ 时预测为正例

解决方案:

- 直接对训练集里的反类样本进行” 欠抽样”(undersampling)
- 对训练集里的正类样本进行” 过抽样”(oversampling)
- 直接基于原始训练集进行学习，但在用训练好的分类器进行预测时，将再缩放的方法嵌入到其决策过程中，称为” 阈值移动”(threshold-moving)

逻辑回归 (Logistic Regression)

广义线性模型：使用 Sigmoid 函数 $g(.)$ 将分类任务的 y 与线性回归模型的预测值联系起来

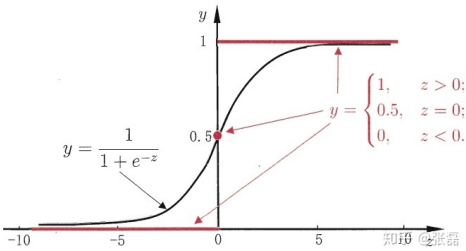
$$y_i = g(z_i) = g(\mathbf{x}_i'\beta + \epsilon_i)$$

逻辑函数 (logistic function)：

- $g(z) = \frac{1}{1+e^{-z}}$

对数几率回归 / 逻辑回归：

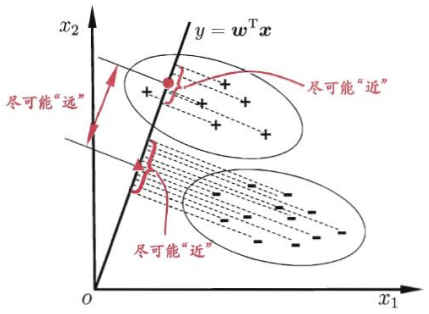
- 假设对数几率 (log odds) 是线性的
- $\ln \frac{y_i}{1-y_i} = \mathbf{x}_i'\beta + \epsilon_i$



线性判别分析 (Linear Discriminant Analysis, LDA)

思想：设法将样本投影到直线上，使得同类样本的投影点尽可能接近、异类样本的投影点尽可能远离

- 给定数据集 $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$，$y_i \in \{0, 1\}$
- 令 $\mathbf{X}_i$、μ_i、Σ_i 分别表示第 $i \in \{0, 1\}$ 类样本的集合、均值向量、协方差矩阵
- 若将样本投影到直线 β 上，两类样本的均值在直线的投影分别为 $\beta'\mu_0$ 和 $\beta'\mu_1$，两类样本的协方差分别为 $\beta'\Sigma_0\beta$ 和 $\beta'\Sigma_1\beta$



支持向量机图示 (Support Vector Machine, SVM)

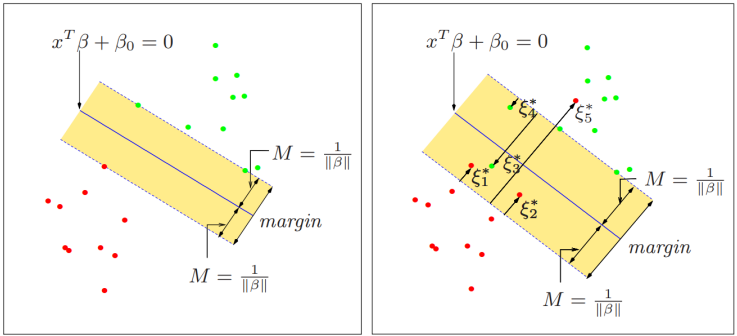


FIGURE 12.1. *Support vector classifiers. The left panel shows the separable case. The decision boundary is the solid line, while broken lines bound the shaded maximal margin of width* $2M = 2/\|\beta\|$. *The right panel shows the nonseparable (overlap) case. The points labeled* ξ_j^* *are on the wrong side of their margin by an amount* $\xi_j^* = M\xi_j$; *points on the correct side have* $\xi_j^* = 0$. *The margin is maximized subject to a total budget* $\sum \xi_i \leq$ *constant. Hence* $\sum \xi_j^*$ *is the total distance of points on the wrong side of their margin.*

支持向量机：线性可分

- 分类问题 $y_i \in \{-1, 1\}$
- 超平面定义为 $\{\mathbf{x} : f(\mathbf{x}) = \mathbf{x}'\beta + \beta_0 = 0\}$，其中 $||\beta|| = 1$
- 在样本类别线性可分的情况下，目标是找到最大间隔

$$\max_{\beta, \beta_0, ||\beta||=1} M$$

$$s.t. \;\; y_i(\mathbf{x}_i'\beta + \beta_0) \geq M, \;\; i = 1, ..., N$$

- 等同于如下最优问题

$$\min_{\beta, \beta_0} ||\beta||$$

$$s.t. \;\; y_i(\mathbf{x}_i'\beta + \beta_0) \geq 1, \;\; i = 1, ..., N$$

支持向量机：线性不可分

- 非线性映射：引入核函数 $h(\mathbf{x})$，最优化问题仅改变约束条件

$$\min_{\beta,\beta_0} ||\beta||$$
$$y_i(h(\mathbf{x}_i)'\beta + \beta_0) \geq 1,\ i = 1, ..., N$$

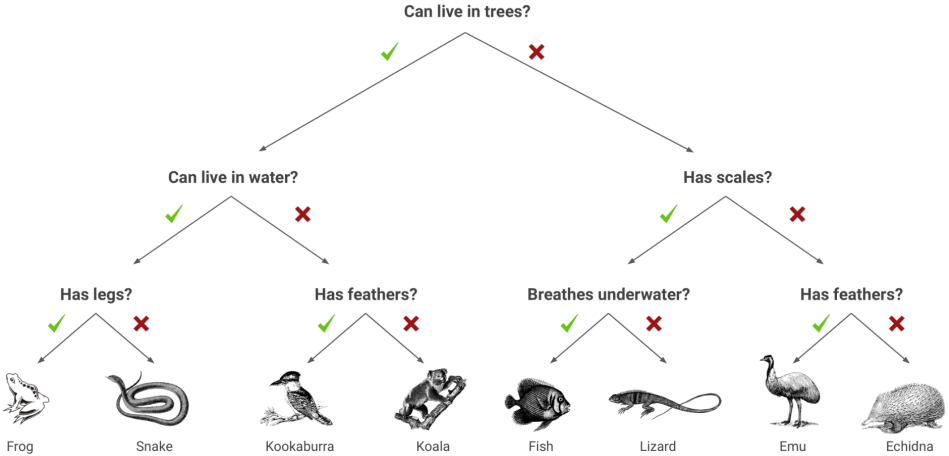
- 软间隔与正则化：假设类别间有重叠区域，允许划分超平面两侧有错误的分类，引入松弛变量 (slack variables) $= (\xi_1, ..., \xi_N)$，最优化问题仅改变约束条件

$$\min_{\beta,\beta_0} ||\beta||$$

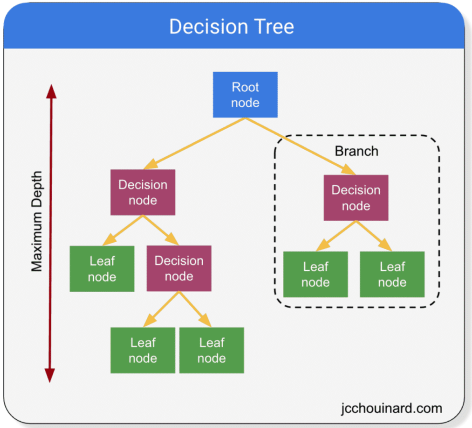
$$y_i(\mathbf{x}_i'\beta + \beta_0) \geq 1 - \xi_i$$

其中 $\xi_i \geq 0,\ \sum_{i=1}^{N} \xi_i \leq constant$

决策树示例 (Decision Tree)



决策树的基本概念



一棵决策树包含

- 一个根节点 (root node)
- 若干个叶节点 (leaf node)，包含决策结果
- 若干个非叶节点 (decision node)，根据属性进行分枝

超参数：

- 树的最大深度 (maximum depth)
- 叶节点包含的最小样本数量
- 分裂标准（信息增益、基尼指数等）

决策树的生长

递归的过程、贪心的算法

- 从根节点出发开始选择最优划分属性，确定分枝准则
- 在某枝再确定进一步最优划分属性和分枝准则
- 直至分枝至叶节点

达到叶节点的标准

- 当前节点包含的样本全属于同一类别，无需划分
- 当前节点所有样本在所有属性上取值相同，无法划分，将其类别设定为该节点所含样本最多的类别
- 当前节点包含的样本集合为空，不能划分，将其类别设定为其父节点所含样本最多的类别

集成学习简介 (Ensemble Learning)

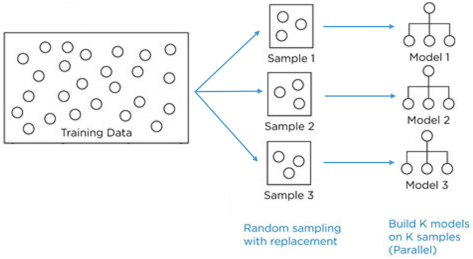
- **集成学习**是一种将**多个弱学习器 (Weak Learner) 组合成一个强学习器 (Strong Learner) 的技术**
- **核心思想：集思广益**
  - 组合多个弱学习器的预测结果，获得更全面、更鲁棒的预测能力
- **降低误差的方式**：
  - **降低方差**：通过并行训练多个基学习器，对结果平均或投票（如 Bagging）
  - **降低偏差**：通过串行训练基学习器，每个学习器纠正前一个的错误（如 Boosting）
  - **提高鲁棒性**：对异常值和噪声数据具有更强的抵抗力
- **主要方法**：
  - **Bagging (Bootstrap Aggregating)**
  - **Boosting (提升法)**
  - **Stacking (堆叠法)**

Bagging (Bootstrap Aggregating)

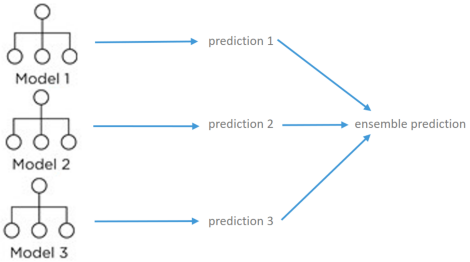
- **核心思想**：并行集成
  - 通过**自助采样**创建多个训练数据集
  - 在每个数据集上独立训练基学习器
  - 通过投票或平均合并预测结果
- **算法流程**：
  1. **自助采样**：从原始数据集有放回地随机抽样，构建多个子数据集
  2. **训练基学习器**：在每个子数据集上独立训练
  3. **集成预测**：分类问题用投票法，回归问题用平均法
- **优点**：
  - 有效降低方差（Variance）
  - 提高模型稳定性和泛化能力
  - 适用于容易过拟合的基学习器
  - 可以并行计算，提高效率

Bagging 图示

自助采样



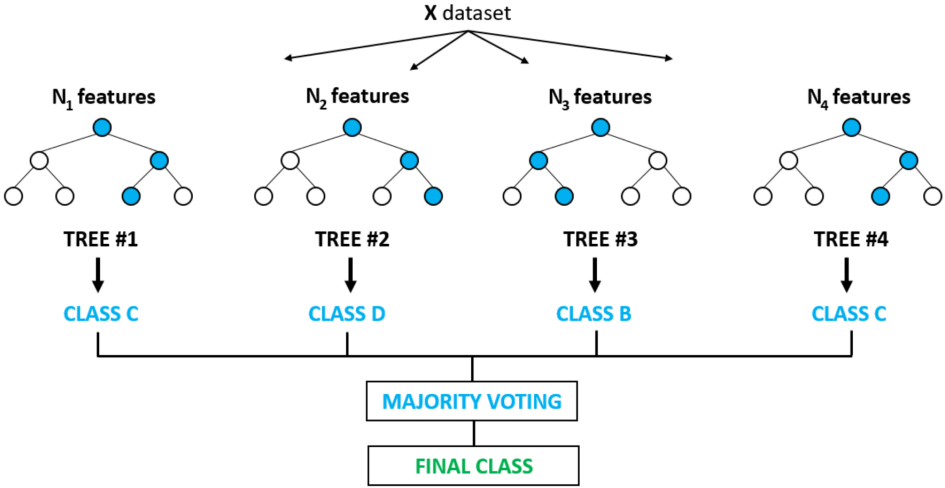
集成预测



随机森林 (Random Forest)

- **随机森林**是一种**基于 Bagging 思想的集成学习模型**，以**决策树为基学习器**
- 在 Bagging 的基础上，引入了**特征随机选择**:
  - 每个节点分裂时，随机选择一部分特征
  - 只在这部分特征中选择最优特征进行分裂
  - 进一步增加基学习器之间的差异性
- **优点**:
  - 高精度：集成多个决策树的预测结果
  - 鲁棒性强：对噪声和异常值不敏感
  - 不易过拟合：特征随机选择和样本随机选择降低了过拟合风险
  - 可以评估特征重要性

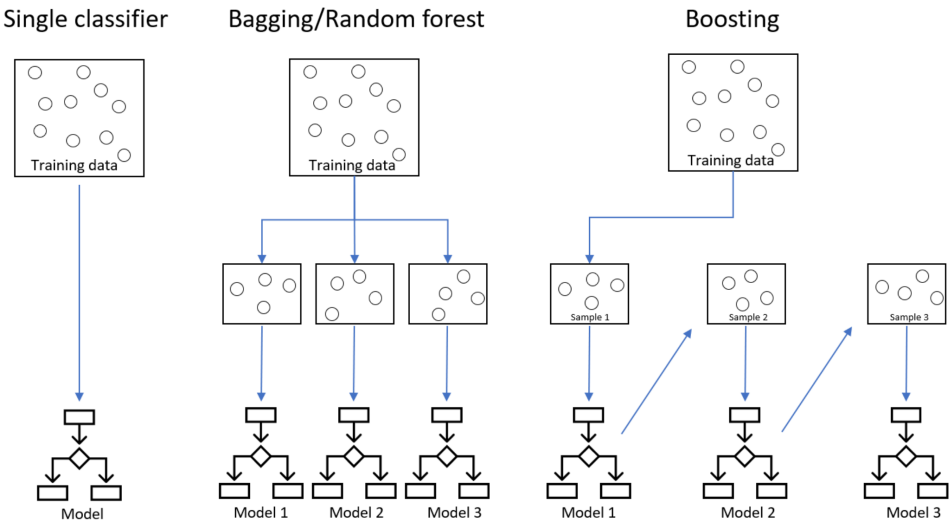
随机森林图示


X dataset
N1 features
N2 features
N3 features
N4 features
TREE #1
TREE #2
TREE #3
TREE #4
CLASS C
CLASS D
CLASS B
CLASS C
MAJORITY VOTING
FINAL CLASS

Boosting

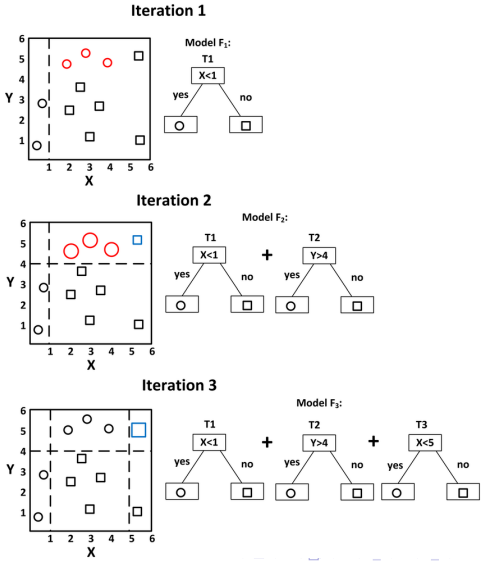
- **核心思想**：串行集成
  - 迭代训练多个基学习器
  - 每个新的基学习器都试图纠正前一个的错误
  - 通过加权组合基学习器的预测结果
- **与 Bagging 的区别**：
  - Bagging 中基学习器**相互独立、并行训练**
  - Boosting 中基学习器**序列依赖、串行训练**
- **主要 Boosting 算法**：
  - AdaBoost (Adaptive Boosting)
  - Gradient Boosting
  - XGBoost, LightGBM, CatBoost 等

Bagging 与 Boosting 区别



AdaBoost (Adaptive Boosting)

- **核心思想**：
  - 调整样本权重，增加被误分类样本的权重
  - 构建新的基学习器来纠正前一个的错误
  - 根据基学习器的性能分配权重，加权组合所有基学习器
- **算法流程**：
  1. 初始化每个样本权重相等
  2. 在加权数据上训练基学习器
  3. 计算基学习器的误差率和权重
  4. 更新样本权重（提高误分类样本的权重）
  5. 重复步骤 2-4，直到达到基学习器数量
  6. 加权组合所有基学习器



AdaBoost 算法

Algorithm 10.1 *AdaBoost.M1.*

1. Initialize the observation weights $w_i = 1/N,\ i = 1, 2, \ldots, N$.
2. For $m = 1$ to M:
   (a) Fit a classifier $G_m(x)$ to the training data using weights w_i.
   (b) Compute
   $$\text{err}_m = \frac{\sum_{i=1}^{N} w_i I(y_i \neq G_m(x_i))}{\sum_{i=1}^{N} w_i}.$$
   (c) Compute $\alpha_m = \log((1 - \text{err}_m)/\text{err}_m)$.
   (d) Set $w_i \leftarrow w_i \cdot \exp[\alpha_m \cdot I(y_i \neq G_m(x_i))],\ i = 1, 2, \ldots, N$.
3. Output $G(x) = \text{sign}\left[\sum_{m=1}^{M} \alpha_m G_m(x)\right]$.

梯度提升决策树 (Gradient Boosting Decision Tree, GBDT)

- **核心思想**:
  - 新树拟合的目标是上一棵树的损失函数的负梯度的值
  - 每棵新树都是在纠正之前所有树的残差
  - 提升模型预测能力
- **与 AdaBoost 的区别**:
  - AdaBoost 通过调整样本权重来学习新的基学习器
  - GBDT 通过拟合残差（负梯度）来学习新的基学习器
- **算法流程**:
  1. 初始化模型为一个常数
  2. 计算当前模型的残差（负梯度）
  3. 训练一个回归树来拟合残差
  4. 将新树添加到模型中
  5. 重复步骤 2-4，直到达到树的数量

梯度提升决策树算法

Algorithm 10.3 *Gradient Tree Boosting Algorithm.*

1. Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$.
2. For $m = 1$ to M:
   (a) For $i = 1, 2, \ldots, N$ compute

$$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}.$$

   (b) Fit a regression tree to the targets r_{im} giving terminal regions $R_{jm},\ j = 1, 2, \ldots, J_m$.
   (c) For $j = 1, 2, \ldots, J_m$ compute

$$\gamma_{jm} = \arg\min_\gamma \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma\right).$$

   (d) Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$.
3. Output $\hat{f}(x) = f_M(x)$.

高级 GBDT 实现对比

算法	核心优势	局限性
XGBoost	- 正则化防止过拟合- 二阶导数加速收敛- 并行计算效率高- 内置交叉验证	- 内存消耗较大- 对类别特征处理需要手动编码- 参数调优复杂度较高
LightGBM	- 直方图算法节省内存- 叶子生长策略提升精度- 支持类别特征- 大规模数据优化	- Leaf-wise 策略可能过拟合- 特征捆绑可能损失信息- 小数据集优势不明显
CatBoost	- 自动处理类别特征- 排序提升防过拟合- 默认参数鲁棒性强- GPU 加速支持	- 训练时间相对较长- 对非类别数据提升有限- 模型文件体积较大

Stacking（堆叠集成）

- **核心思想**：
  - 训练多个不同类型的基学习器
  - 使用一个元学习器 (meta-learner) 组合基学习器的输出
  - 形成多层堆叠的结构
- **算法流程**：
  1. 训练多个不同类型的基学习器 (如决策树、SVM、神经网络等)
  2. 基学习器对训练集进行预测，生成新特征
  3. 使用这些新特征训练元学习器
  4. 元学习器做出最终预测
- **优点**：
  - 利用不同算法的优势
  - 减少单一模型的局限性
  - 通常比单一模型性能更好
  - 适合处理复杂问题