GBDT算法理解

GBDT算法结构

GBDT是boosting算法的一种。它是以决策树为基函数的提升方法。因为是最终求解是基函数的结果的累加，所以决策树是回归树。

GBDT可以看做是由K颗树组成的加法模型：

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), f_k \in F$$

以回归任务为例，回归树可以看作为一个把特征向量映射为某个score的函数。于一般的机器学习算法不同的是，加法模型不是学习d维空间中的权重，而是直接学习函数（决策树）集合。

上述加法模型的目标函数定义为：

$$Obj = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k)$$

其中Ω表示决策树的复杂度。

GBDT优化方法

解这一优化问题，可以用前向分布算法（forward stagewise algorithm）。因为学习的是加法模型，如果能够从前往后，每一步只学习一个基函数及其系数（结构），逐步逼近优化目标函数。具体地，我们从一个常量预测开始，每次学习一个新的函数，过程如下：

$$\begin{gathered}\hat{y}_i^0 = 0 \\ \hat{y}_i^1 = f_1(x_i) = \hat{y}_i^0 + f_1(x_i) \\ \hat{y}_i^2 = f_1(x_i) + f_2(x_i) = \hat{y}_i^1 + f_2(x_i) \\ \dots \\ \hat{y}_i^t = \sum_{k=1}^{t} f_k(x_i) = \hat{y}_i^{t-1} + f_t(x_i)\end{gathered}$$

在第t步，模型对x_i的预测为：$\hat{y}_i^t = \hat{y}_i^{t-1} + f_t(x_i)$，其中$f_t(x_i)$为这一轮我们要学习的函数（决策树）。这个时候目标函数可写为：

$$Obj = \sum_{i=1}^{n} l(y_i, \hat{y}_i^t) + \sum_{i=1}^{t} \Omega(f_i)$$

$$Obj = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{t-1} + f_t(x_i)) + \Omega(f_t) + constant$$

损失函数为平方损失（square loss），则目标函数为：

$$Obj = \sum_{i=1}^{n} (y_i - (\hat{y}_i^{t-1} + f_t(x_i)))^2 + \Omega(f_t) + constant$$

$$Obj = \sum_{i=1}^{n} [2(\hat{y}_i^{t-1} - y_i) f_t(x_i) + f_t(x_i)^2] + \Omega(f_t) + constant$$

此时为拟合残差。 对于一般损失函数，利用损失函数的负梯度在当前模型的值，拟合一个回归树。

GBDT目标函数的选择

在sklearn中，对于分类模型，有对数似然损失函数"deviance"和指数损失函数"exponential"两者输入选择。默认是对数似然损失函数"deviance"。它对二元分离和多元分类各自都有比较好的优化。