Gradient Boosting

A Brief Introduction and Explanation

Wang Xinyu 1098648

Introduction

Gradient Boosting

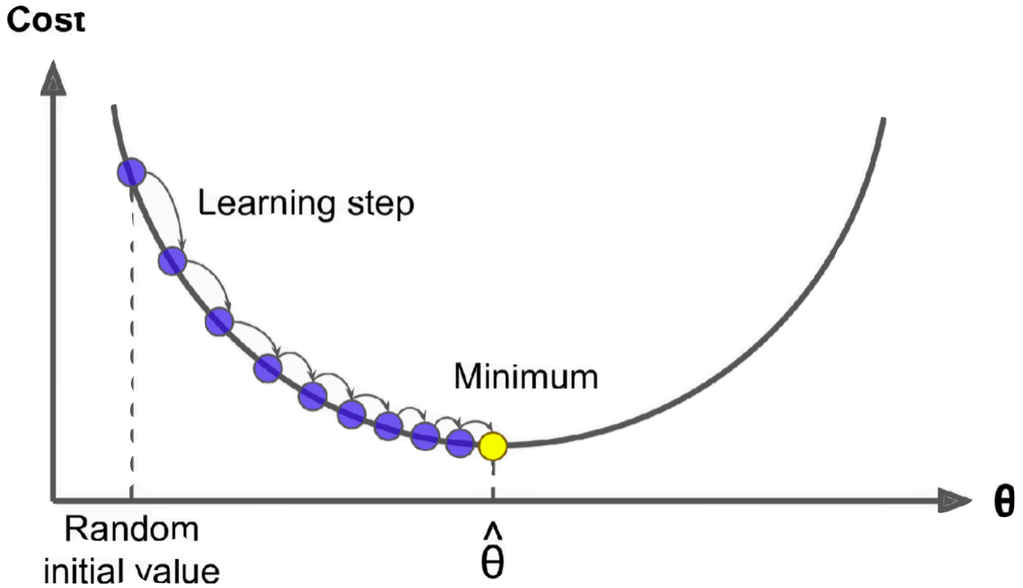
- Just like AdaBoost, Gradient Boosting works by sequentially adding predictors to an ensemble, each one correcting its predecessor.
- However, instead of tweaking the instance weights at every iteration like AdaBoost does, Gradient Boosting, which borrows its ideas from the **gradient descent** method, tries to fit the new predictor to the **residual errors** made by the previous predictor.

Gradient Descent

Gradient Descent

Brief Introduction or Review?

- Gradient Descent is a first-order iterative optimization algorithm for finding a local minimum of a differentiable function. The general idea of it is to tweak parameters iteratively in order to minimize a cost function.



The model parameters are initialized randomly and get tweaked repeatedly to minimize the cost function; the learning step size is proportional to the slope of the cost function, so the steps gradually get smaller as the parameters approach the minimum

Gradient Descent

Basic Steps

The basic steps for the gradient descent method are:

1. Given the gradient, calculate the change in the parameters with the learning rate
2. Re-calculate the new gradient with the new value of the parameter
3. Repeat step 1 until the gradient descent distance of θ_t meets the exceptions

In other words,

```
Repeat until convergence {
```

$$\theta_j \leftarrow \theta_{j-1} - \alpha \frac{\partial L(\theta)}{\partial \theta_j}$$

```
}
```

Gradient Boosting

Gradient Boosting

Brief Introduction

Suppose we have a regression problem with n samples, $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$, which needs to be fit by a function $F(x)$ to minimize the error.

To find an accurate F, we firstly find a weak function F_0 to fit. So, the difference between the prediction $F_0(x)$ and actual value y, or the residual error $r_0(x)$ is

$$r_0(x) = y - F_0(x)$$

Next, we find h_0 to fit r_0 , such that $F_1(x) = F_0(x) + h_0(x)$. However, the residual error still exists, which is

$$r_1(x) = y - F_1(x) \ldots$$

Gradient Boosting

Brief Introduction

During $0 \leq t \leq T$ steps of Gradient Boosting, suppose there are some imperfect model F_{t-1}

The algorithm does not change F_{t-1} directly, instead it try to add an estimator $r_{t-1}(x)$ to construct a new model $F_t(x) = F_{t-1}(x) + r_{t-1}(x)$ to improve the performance.

And the best estimator $r(x)$ should always satisfy:

$$F_t(x) = F_{t-1}(x) + r_{t-1}(x) = y \leftrightarrow r_{t-1}(x) = y - F_{t-1}(x)$$

where $r_{t-1}(x)$ is actually the residual error.

Gradient Boosting

Brief Introduction

The steps above could be repeat over and over again to make the function $F(x)$ close to the target value y, and we can express this process by the following equation:

$$\begin{cases} F_0 = \sum_{i=1}^{n} y_i \\ F_t(x) = F_{t-1}(x) + r_{t-1}(x) \end{cases}$$

Gradient Boosting

Gradient with square error

To fit function F , let the prediction value $\hat{y} = F(x)$, the loss function $L(y, \hat{y}) = (y - \hat{y})^2/2$, and total error of the whole dataset $J = \sum_{i=1}^{n} L(y_i, \hat{y}_i)$.

So,

$$\nabla J = (\frac{\partial J}{\partial \hat{y}_1} \dots \frac{\partial J}{\partial \hat{y}_i} \dots \frac{\partial J}{\partial \hat{y}_n})$$

$$\frac{\partial J}{\partial \hat{y}_i} = \frac{\partial \sum_i^n L(y_i, \hat{y}_i)}{\partial \hat{y}_i} = \frac{\partial L(y_i, \hat{y}_i)}{\partial \hat{y}_i}$$

$$= \frac{\partial [(y - \hat{y})^2/2]}{\partial \hat{y}_i} = -(y_i - \hat{y}_i)$$

$$y_i - \hat{y}_i = -\frac{\partial J}{\partial \hat{y}_i}$$

$$y - \hat{y} = -\nabla J$$

Residual Error is the Negative Gradient

Gradient Boosting

Gradient with other error

1. Absolute Error: $L(y, F) = |y - F|$

Negative Gradient: $$-\frac{\partial J}{\partial F(x_i)} = -\frac{\partial \sum_i^n L(y_i, F(x_i))}{\partial F(x_i)} = sign(y_i - F(x_i))$$

2. Huber Error $$L(y, F) = \begin{cases} (y - F)^2/2, & |y - F| \leq \delta \\ \delta(|y - F| - \delta/2, & |y - F| > \delta \end{cases}$$

Negative Gradient: $$-\frac{\partial J}{\partial F(x_i)} = -\frac{\partial \sum_i^n L(y_i, F(x_i))}{\partial F(x_i)} = \begin{cases} y - F(x_i), & |y_i - F(x_i)| \leq \delta \\ \delta sign(y_i - F(x_i)), & |y_i - F(x_i)| > \delta \end{cases}$$

Gradient Boosting

Steps of Gradient Boosting & Summary

Suppose we have a dataset with n samples $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ which needs to be fit by a function $F(x)$ to minimize the error. Then we perform Gradient Boosting:

1. Select a differentiable error function L
2. Construct an initial model such as $F = \sum_{i=1}^{n} y_i$
3. Perform M iterations
    1. Calculate negative gradient $-g(x_i) = -\dfrac{\partial L(y_i, F(x_i))}{\partial F(x_i)}$
    2. Construct function r to fit the negative gradient $-g(x_i)$
    3. Compute the weight coefficient γ such that the total error $\sum_{i+1}^{n} L(y_i, F + \gamma h)$ is least
    4. Update F to $F + \gamma h$

Thank You

Gradient Boosting

In the gradient descent method, we can see that for the final optimal solution θ^* is obtained after T iterations from the initial value θ_0 .

So if we set $\theta_0 = -\frac{\partial L(\theta)}{\partial \theta_0}$, then $\theta = \sum_{t=0}^{T} \alpha_t \cdot \left[-\frac{\partial L(\theta_{t-1})}{\partial \theta_{t-1}} \right]$.

For functions, we could use the same approaches as gradient descent. Let $L(y, F(x))$ be the loss function of our model, $F(x)$ is optimal function, the initial value $F_0(x) = r_0(x)$.

Hence $F(x) = \sum_{t=0}^{T} r_t(x)$ where $r_t(x) = \alpha_t \cdot \left[-\frac{\partial L(y, F_{t-1}(x))}{\partial F_{t-1}(x)} \right]$