What is Gradient Boosting?

Gradient boosting is a versatile and powerful machine learning technique that excels at capturing complex data relationships and is widely used for various tasks.

As its name suggests, gradient boosting combines principles from gradient descent and boosting, a concept first popularized by AdaBoost, to build flexible and highly accurate predictive models (Friedman).

What is it Used For?

Gradient boosting is very versatile and can be applied to:

- Regression
- Classification
- Ranking
- And various other machine learning tasks

An example of gradient boosting's utility is shown in this paper, which details gradient boosting's effectiveness when evaluating reservoir quality in tight sandstone: https://www.degruyter.com/document/doi/10.1515/geo-2022-0354/html

How Does It Work?

Gradient boosting works by iteratively combining "weak learners" to form a single, strong model. Its key components are:

Weak Learners:

Weak learners are simple/underpowered models that won't capture all aspects of the data. Gradient boosting sequentially improves upon these models. Common weak learners include:

- Simple Linear regression
- Simple Logistic regression
- Decision trees (most common)
- Random forests

Loss Function:

Gradient boosting uses a loss function to measure how well the model performs. Ideally, the loss function should be easily differentiable as the "gradient" part of gradient boosting relies on gradient descent optimization. Some common loss functions include:

- Mean Squared Error (MSE) for regression tasks
- Log Loss/Binary Cross-Entropy for classification tasks
- Custom loss functions for specific applications

Steps of the Generic Gradient Boosting Algorithm

After selecting a weak learner and loss function, here are the general steps of the algorithm:

1. **Start with an initial weak learner**
    a. This can be as simple as a vector/array of 0's for a regression or an average probability for a classification.
2. **Compute pseudo-residuals:**
    a. For each data point, calculate the negative gradient of the loss function with respect to the model's predictions. These pseudo-residuals represent the errors the current model needs to correct.
3. **Fit a new weak learner to the pseudo-residuals:**
    a. Train a new weak learner using the same input data, but with the pseudo-residuals as the target variable.
4. **Optimize multiplier size**
    a. Perform an optimization step to find the best value for γ (the value to be multiplied by the new weak learner).
    b. This involves finding the minimum value from the loss function with the true values/labels and the predicted values/labels of the updated model (current model + γ * new weak learner) as input.
5. **Update the model:**
    a. Create the updated model by adding the new weak learner (multiplied by γ) to the current model.
6. **Repeat steps 2-5:**
    a. Continue iterating until the desired level of accuracy is achieved or until the model stops improving.

To further visualize the algorithm, here is a pseudo-code representation of Gradient Boosting using trees (Note that this algorithm features improvements suggested by Friedman specifically for Gradient Boosting when applied with decision trees):

10.11 Right-Sized Trees for Boosting 361

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 $\hat{f}(x) = f_M(x)$.

(Hastie, pg. 361)

Applying Gradient Boosting

To put the information I learned into practice, I implemented a basic version of Gradient Boosting, using a decision tree classifier as the weak learner to classify emails as spam or non-spam. Additionally, to gain familiarity and practical experience, I used the popular open-source machine learning library XGBoost (Also known as eXtreme Gradient Boosting) to classify spam emails using the same dataset.

- https://github.com/bennettbDEV/GradientBoostingv2

Sources

Web Sources

"What is Gradient Boosting?" Snowflake, www.snowflake.com/guides/what-gradient-boosting/.

Li, Cheng. Gradient Boosting: A Tutorial. chengli.io/tutorials/gradient_boosting.pdf.

Usman, M. Boosted Tree. web.njit.edu/~usman/courses/cs675_fall16/BoostedTree.pdf.

Articles and Papers

Breiman, Leo. "Prediction Games and Arcing Algorithms." Neural Computation, vol. 11, no. 7, 1999, pp. 1493–1517. https://doi.org/10.1162/089976699300016131.

Breiman, Leo, et al. "Arcing Classifier (with Discussion and a Rejoinder by the Author)." The Annals of Statistics, vol. 26, no. 3, 1998, pp. 801–849. https://doi.org/10.1214/aos/1024691079.

Friedman, Jerome H. "Greedy Function Approximation: A Gradient Boosting Machine." IMS 1999 Reitz Lecture, 1999.

Ma, Longfei, Xiao, Hanmin, Tao, Jingwei, Zheng, Taiyi and Zhang, Haiqin. "An intelligent approach for reservoir quality evaluation in tight sandstone reservoir using gradient boosting decision tree algorithm" Open Geosciences, vol. 14, no. 1, 2022, pp. 629-645. https://doi.org/10.1515/geo-2022-0354

Books

Hastie, Trevor, Robert Tibshirani, and Jerome H. Friedman. The Elements of Statistical Learning. 2nd ed., Springer, 2009, pp. 337–384. ISBN 978-0-387-84857-0.