Gradient Boosted Decision Tree Classifier

Rohitha Ravindra Myla , Dongwon Ha, Stephen Marriott

DATA2060

2024/12/13

github

Motivation

What is Gradient Boosting?

- **Ensemble learning** – a strong learner constructed from multiple weak learners
- Each successive weak learner **corrects the previous** weak learner's prediction
- Weak learners perform slightly better than random guess: decision trees, linear models, k-NN, etc.

Why use Gradient Boosting?

Advantages	Disadvantages
• Model is **intuitive** • **Versatile** – works on both classification and regression problems • **Robust to outliers**, due to ensemble approach • Incredibly **accurate** for structured tabular data – tops many of the Kaggle competitions	• Not the most **interpretable** • Can **overfit** to training data – improper hyperparameter tuning • Computationally **costly**, particularly with big data

Our Approach and Intuition

General Architecture - Representation

- Weak learner: shallow decision tree

- Final model F(x) is sequence of N decision trees
  - $F_N(x) = F_0(x) + \eta\Sigma^N h_i(x)$
  - $F_0(x)$ is the initialized prediction
  - η is the learning rate (0 to 1)
  - $h_i(x)$ is the prediction made by decision tree *i,* trained on residuals from previous trees

- Not directly predicting class labels, but pseudo-residuals
  - Pseudo-residuals = true label – probability prediction

General Architecture - Loss

- Binary classification: use the **Cross Entropy Loss**
    - $L(y, p(x)) = -(y\log(p(x)) + (1-y)\log(1-p(x)))$

    - y are the true labels, 1 & 0
    - p(x) is the probability predictor for the positive class 1

General Architecture - Optimizer

- Gradient descent on the pseudo-residuals

- Pseudo-residuals calculated as the negative gradient of the loss
  - $r = -dL(y,p(x))/dF(x) = y - p(x)$

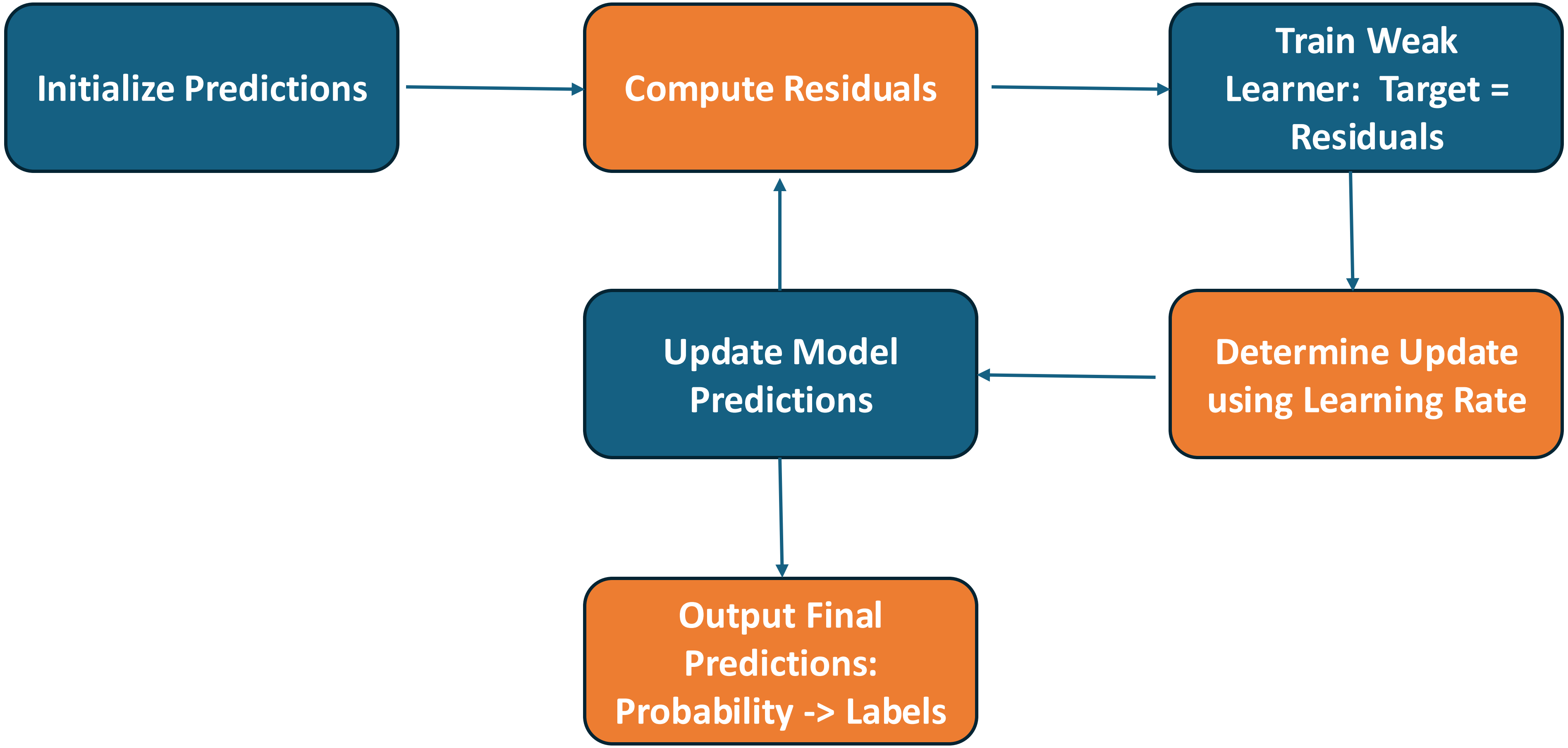
- Decision trees trained on the previous iteration's residuals, which are then used to update the probability prediction

- $F_{i+1}(x) = F_0(x) + \eta h_i(X)$

Why does this work?

- Principle of gradient descent
  - For a datapoint with true label 1, the smaller p(x) is, the larger the residual becomes, and the larger the correction to the next p(x) prediction becomes
  - The inverse is true for true label 0

How does the model work?



Model Pseudocode

- **Inputs:**

  Training set: S = $(x_1, y_1), \ldots, (x_m, y_m)$

  Weak Learner: Decision tree DT

  Number of trees: N

  Learning rate: η

- **Initialize:**

  Set initial predictions as log-odds of the positive class: $F_0(x) = \log(p_{y=1}/1\text{-}p_{y=1})$

  For *i* = 0, 1, …, N-1:

      Compute the residuals: $\mathbf{r_i} = dL(y,p(x))/dF(x) = y - p(x)$

      Train a weak learner with residuals as targets: $h_i(x) = DT(F_i(x), S)$

      Update the model: $F_{i+1}(x) = F_i(x) + \eta h_i(x)$

- **Outputs:**

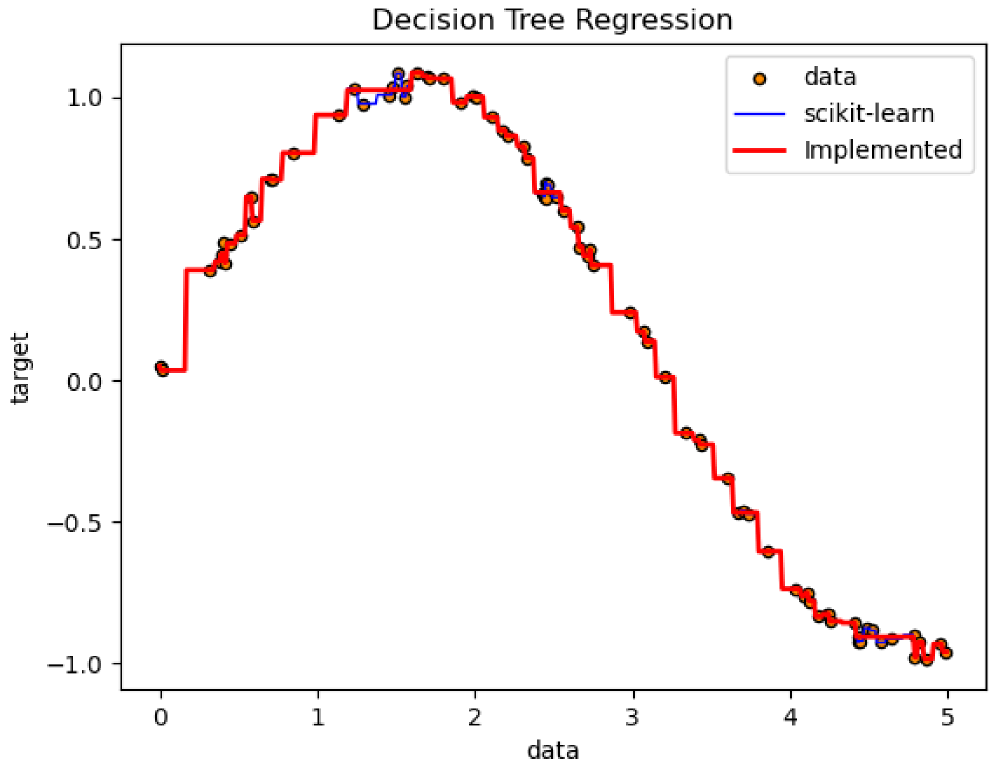
  Predictions: $y = \text{argmax}(F_N(x))$

Implementation

Implementation of DTR (Decision Tree Regressor)

- Dataset: y=sin(x)+noisy data
- Hyperparameters: max_depth = 16

Metric	DTR from sklearn	Implemented
MSE	0.0030	0.0029



Implementation of GBC (Gradient Boosting Classification)

- **Dataset:** sklearn.datasets.make_hastie_10_2
  - Dataset used for binary classification in Hastie et al. 2009, Example 10.2
  - 12000 samples
  - 10 features

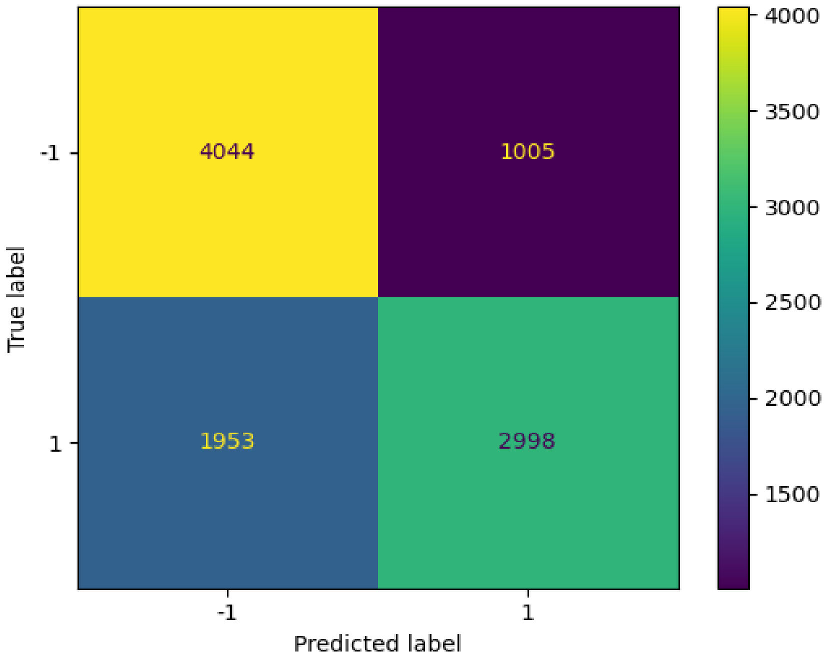
- **Metric:** accuracy / confusion matrix

- **Reference:** T. Hastie, R. Tibshirani and J. Friedman, “Elements of Statistical Learning Ed. 2”, Springer, 2009.

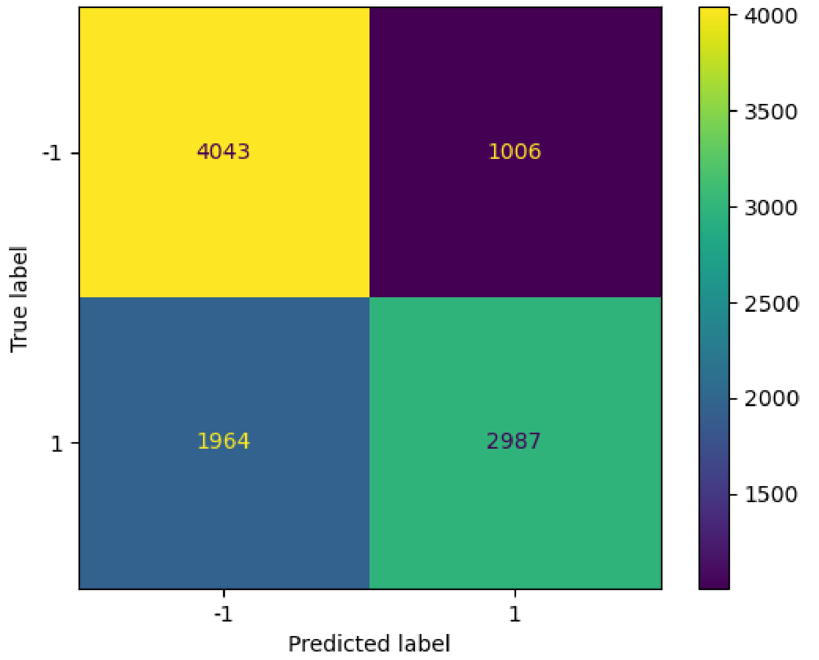
Implementation of GBC (Gradient Boosting Classification)

- Hyperparameters:
  - N_estimators (number of weak learners) = 4
  - Learning rate = 0.1
  - Max_depth = 4

Metric	GBC from sklearn	Implemented
Accuracy	0.7042	0.703



Reference



Implemented

Previous Work

- Gradient Boosted Decision Trees for High Dimensional Sparse Output

(By Si Si ;Huan Zhang ;S. Sathiya Keerthi ;Dhruv Mahajan ;Inderjit S. Dhillon ;Cho-Jui Hsieh 2)

- Stochastic gradient boosted distributed decision trees

(By Jerry Ye, Jyh Herng Chow, Jiang Chen, Zhaohui Zheng)

- Efficient Gradient Boosted Decision Tree Training on GPUs

(By Zeyi Wen; Bingsheng He; Ramamohanarao Kotagiri; Shengliang Lu; Jiashuai Shi)

Summary

- Implemented **Gradient Boosting Classifier** using **Decision Tree Regressor** as a weak learner

- **Interesting parts:**
  - **Residuals:** Use multiple decision tree regressor to fit residuals, improve the accuracy of the model
  - **Learning rate:** Learning rate acts as a regularization mechanism, controlling how aggressively the model fits to the residuals.

- **Challenges:**
  - **Decision tree regressor:** Find the optimal split for the regression problem
  - **Computational cost:** Should implement efficient algorithm for the scalability

Questions?

Thank you!