Analysis of Boosting Algorithms

Penta Boosters

November 28, 2023

Contents

1 Group Members

SL No.	Name	Roll No.
1	Arijeet De	23M0742
2	A Asish	23M0759
3	Sarvam Maheshwari	23M2101
4	Ronak Upasham	23M0793
5	Yogesh Mandlik	23M0780

2 Problem Statement

This project is dedicated to a comprehensive exploration and hands-on implementation of boosting algorithms, including AdaBoost (Adaptive Boosting), Gradient Boosting, Decision Trees, and XGBoost. Our objective is to gain a deep understanding of the inner workings of these algorithms by coding them from scratch in Python, without relying on external libraries.

Our emphasis is on mastering the core concepts that underpin these boosting algorithms, encompassing essential aspects such as weak learner selection, weighted sample updates, and the construction of ensemble models. We will adopt a methodical, step-by-step approach, accompanied by practical code examples and detailed explanations.

To validate the effectiveness of our custom implementations, we will conduct benchmarking exercises against established boosting libraries. This entails a rigorous analysis of our models' performance across multiple datasets. Furthermore, we will engage in a comprehensive exploration of hyperparameter tuning strategies aimed at optimizing the results.

In summary, this project serves as an insightful and practical guide for building boosting algorithms from the ground up, empowering us to unlock the full potential of these techniques in our machine learning endeavors.

2.1 Tasks for Analysis

The following tasks are being used to evaluate our model and also to compare our model with the model from pre existing libraries.

- Accuracy (%)
- Training Time (in ms)

2.2 Datasets Used

1. **Iris Dataset**
   - Name: Iris Plants Database
   - Link: https://archive.ics.uci.edu/ml/datasets/iris
2. **Breast Cancer Dataset**
   - Name: Breast Cancer
   - Link: https://github.com/EpistasisLab/pmlb/blob/master/datasets/breast_cancer/metadata.yaml
3. **ann thyroid**
   - Name: Ann Thyroid
   - Link: https://github.com/EpistasisLab/pmlb/blob/master/datasets/ann_thyroid/metadata.yaml

3 Description of the Algorithms

3.1 Decision Tree

A Decision Tree is a versatile machine learning algorithm used for both classification and regression tasks. It constructs a tree-like model by recursively partitioning the data based on feature conditions.

At each node, the algorithm selects the feature that best separates the data. This process is repeated, creating a hierarchical structure of decisions leading to final predictions or outcomes. Decision Trees are intuitive, as they mimic human decision-making processes.

They excel in capturing non-linear relationships in data and handling both numerical and categorical features. However, they may be prone to overfitting, necessitating techniques like pruning.

3.2 Adaboost

AdaBoost, short for Adaptive Boosting, is an ensemble learning algorithm designed to enhance the performance of weak learners, often simple decision trees. It iteratively trains a sequence of models, assigning higher weights to misclassified instances in each iteration.

Subsequent models then prioritize learning from previously misclassified data, effectively adapting to the complex patterns in the dataset. The final prediction is a weighted combination of individual weak learner outputs.

AdaBoost is resilient against overfitting, and its iterative nature allows it to focus on challenging instances, improving overall predictive accuracy. The strength of AdaBoost lies in its ability to create a strong classifier from a collection of weak ones, making it a powerful tool for tackling intricate classification problems.

3.3 Gradient Boosting

Gradient Boosting is a machine learning algorithm that builds an ensemble of decision trees sequentially to create a robust predictive model. It focuses on minimizing the residual errors of the preceding trees in each iteration, with subsequent trees correcting the deficiencies of the ensemble.

The algorithm employs gradient descent optimization, adjusting the weights of data points based on the gradient of the loss function. By iteratively refining the model, Gradient Boosting excels in capturing intricate patterns and achieving high predictive accuracy.

It is widely used in regression and classification tasks, including applications in finance, healthcare, and natural language processing. Gradient Boosting variants like XGBoost and LightGBM further enhance performance with optimizations such as parallel processing and tree pruning.

3.4 XGBoost

XGBoost, or Extreme Gradient Boosting, is an advanced implementation of the Gradient Boosting algorithm designed for enhanced efficiency and predictive accuracy. It integrates regularization techniques, tree pruning, and parallel computing, making it exceptionally fast and effective. XGBoost builds an ensemble of decision trees sequentially, with each tree correcting errors of its predecessors.

The algorithm optimizes a user-defined objective function, employing gradient descent to minimize the residuals. XGBoost is versatile, handling regression, classification, and ranking tasks, and it excels in capturing complex relationships within data.

Its features include automatic handling of missing values, regularization to prevent overfitting, and the ability to work with sparse data. Due to its superior performance, XGBoost has become a popular choice in machine learning competitions and real-world applications. Its scalability and efficiency make it suitable for large datasets, and its robustness is further exemplified by its resistance to overfitting, contributing to its widespread adoption in diverse domains.

4 Results

We applied our algorithms and compared with pre existing libraries of the same algorithms on the datasets mentioned in 2.2

4.1 Iris Dataset

Model	Accuracy (%)	Training Time (ms)
Decision Tree (own)	97.78	39.29
Decision Tree (sklearn)	100	2.76
Adaboost (own)	96.67	95.54
Adaboost (sklearn)	100	23.86
Gradient Boosting (own)	93.33	13558.90
Gradient Boosting (sklearn)	95.55	534
XGBoost (own)	96	2148.69
XGBoost (sklearn)	100	7.39

4.2 Breast Cancer Dataset

Model	Accuracy (%)	Training Time (ms)
Decision Tree (own)	77.59	49.05
Decision Tree (sklearn)	63.79	4.65
Adaboost (own)	75.86	119.79
Adaboost (sklearn)	75.86	106.15
Gradient Boosting (own)	74.41	7170
Gradient Boosting (sklearn)	74.41	242
XGBoost (own)	83.33	48718
XGBoost (sklearn)	77.59	9.48

4.3 Ann Thyroid Dataset

Model	Accuracy (%)	Training Time (ms)
Decision Tree (own)	99.38	806.40
Decision Tree (sklearn)	99.72	10.54
Adaboost (own)	97.57	267.62
Adaboost (sklearn)	98.89	130.32
Gradient Boosting (own)	5	25435.84
Gradient Boosting (sklearn)	99.8	2716
XGBoost (own)	92.15	20805.68
XGBoost (sklearn)	99.10	24.34

5 Conclusion

In conclusion, our endeavor to implement boosting algorithms, namely AdaBoost, Gradient Boost, and XGBoost from scratch, has yielded promising results. Through meticulous coding and thoughtful design, we have achieved performance levels that closely rival those of standard libraries. Our implementations not only showcase comparable efficacy but also boast simplicity and transparency in their structure.

The process of coding these boosting algorithms has not only deepened our understanding of the underlying principles but has also empowered us to customize and fine-tune the models according to specific requirements. The transparency in our implementations facilitates a clearer comprehension of the algorithms' mechanics, making it accessible to a broader audience.

Furthermore, our project underscores the significance of hands-on implementation in fostering a profound understanding of machine learning algorithms. By delving into the intricacies of AdaBoost, Gradient Boost, and XGBoost, we have not only honed our coding skills but also gained insights into the nuances of boosting techniques.

While our implementations demonstrate competitive performance, it is crucial to acknowledge the continuous advancements in standard libraries and their optimization for various scenarios. Nonetheless, our project serves as a testament to the value of comprehending algorithms at their core, enabling practitioners to tailor solutions to unique challenges.

In essence, our journey in crafting boosting algorithms from scratch has been both educational and rewarding, emphasizing the importance of a balanced approach between theoretical understanding and practical implementation in the realm of machine learning.

6 References

Here are some important references related to the boosting algorithms that we're implementing:

1. AdaBoost Implementation from Scratch
   Link: https://www.kdnuggets.com/2020/12/implementing-adaboost-algorithm-from-scratch.html

2. SAMME: A Stagewise Additive Modeling Approach
   Link: https://hastie.su.domains/Papers/samme.pdf

3. Scikit-learn Decision Trees for Classification
   Link: https://scikit-learn.org/stable/modules/tree.html#classification

Additional references:

- GBM from Jerome Friedman
  - Link: https://jerryfriedman.su.domains/ftp/trebst.pdf
- "Gradient boosting machines, a tutorial" by Alexey Natekin and Alois Knoll
  - Link: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3885826/
- Chapter in "Elements of Statistical Learning" from Hastie, Tibshirani, Friedman (page 337)
  - Link: https://hastie.su.domains/ElemStatLearn/printings/ESLII_print10.pdf
- Wiki article about Gradient Boosting
  - Link: https://en.wikipedia.org/wiki/Gradient_boosting
- Frontiers tutorial article about GBM
  - Link: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3885826/
- Video lecture by Hastie about GBM at h2o.ai conference
  - Link: https://www.youtube.com/watch?v=wPqtzj5VZus
- "Benchmarking and Optimization of Gradient Boosting Decision Tree Algorithms, XGBoost: Scalable GPU Accelerated Learning"
  - Link: https://arxiv.org/pdf/1809.04559.pdf